

SEGGER

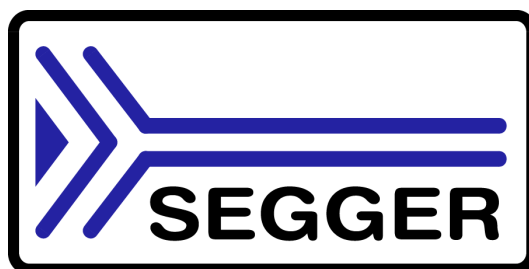
Eval Software

Getting Started with
SEGGER Eval Software

Document: AN00020

Revision: 10

Date: April 15, 2016



A product of SEGGER Microcontroller GmbH & Co. KG

www.segger.com

Disclaimer

Specifications written in this document are believed to be accurate, but are not guaranteed to be entirely free of error. The information in this manual is subject to change for functional or performance improvements without notice. Please make sure your manual is the latest edition. While the information herein is assumed to be accurate, SEGGER Microcontroller GmbH & Co. KG (SEGGER) assumes no responsibility for any errors or omissions. SEGGER makes and you receive no warranties or conditions, express, implied, statutory or in any communication with you. SEGGER specifically disclaims any implied warranty of merchantability or fitness for a particular purpose.

Copyright notice

You may not extract portions of this manual or modify the PDF file in any way without the prior written permission of SEGGER. The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such a license.

© 2015 - 2016 SEGGER Microcontroller GmbH & Co. KG, Hilden / Germany

Trademarks

Names mentioned in this manual may be trademarks of their respective companies.

Brand and product names are trademarks or registered trademarks of their respective holders.

Contact address

SEGGER Microcontroller GmbH & Co. KG

In den Weiden 11

D-40721 Hilden

Germany

Tel. +49 2103-2878-0

Fax. +49 2103-2878-28

E-mail: support@segger.com

Internet: <http://www.segger.com>

Manual versions

This manual describes the current software version. If any error occurs, inform us and we will try to assist you as soon as possible.

Contact us for further information on topics or routines not yet specified.

Print date: April 15, 2016

Revision	Date	By	Description
10	160415	RH	Minor issues fixed.
9	160121	RH	Chapter "How to start" updated, wording & spelling improved.
8	151130	RH	Chapter "SEGGER Embedded Studio" updated.
7	151119	RH	emModbus and new sample descriptions added.
6	150825	MC	Added the Keil MDK chapter.
5	150701	MC	Fixed erroneous descriptions, improved wording & spelling.
4	150611	MC	Added the emIDE and SES chapters.
3	150515	SC	Added the Microchip MPLAB X chapter.
2	150319	YR	Added the Renesas e2 studio chapter.
1	150316	SC	Fixed some minor issues.
0	141208	SC	Initial version.

About this document

Assumptions

This document assumes that you already have a solid knowledge of the following:

- The software tools used for building your application (assembler, linker, C compiler)
- The C programming language
- The target processor
- DOS command line

If you feel that your knowledge of C is not sufficient, we recommend *The C Programming Language* by Kernighan and Richie (ISBN 0-13-1103628), which describes the standard in C-programming and, in newer editions, also covers the ANSI C standard.

How to use this manual

This manual explains all the functions and macros that the product offers. It assumes you have a working knowledge of the C language. Knowledge of assembly programming is not required.

Typographic conventions for syntax

This manual uses the following typographic conventions:

Style	Used for
Body	Body text.
Keyword	Text that you enter at the command-prompt or that appears on the display (that is system functions, file- or pathnames).
Parameter	Parameters in API functions.
Sample	Sample code in program examples.
Sample comment	Comments in program examples.
Reference	Reference to chapters, sections, tables and figures or other documents.
GUIElement	Buttons, dialog boxes, menu names, menu commands.
Emphasis	Very important sections.

Table 1.1: Typographic conventions



SEGGER Microcontroller GmbH & Co. KG develops and distributes software development tools and ANSI C software components (middleware) for embedded systems in several industries such as telecom, medical technology, consumer electronics, automotive industry and industrial automation.

SEGGER's intention is to cut software development time for embedded applications by offering compact flexible and easy to use middleware, allowing developers to concentrate on their application.

Our most popular products are emWin, a universal graphic software package for embedded applications, and embOS, a small yet efficient real-time kernel. emWin, written entirely in ANSI C, can easily be used on any CPU and most any display. It is complemented by the available PC tools: Bitmap Converter, Font Converter, Simulator and Viewer. embOS supports most 8/16/32-bit CPUs. Its small memory footprint makes it suitable for single-chip applications.

Apart from its main focus on software tools, SEGGER develops and produces programming tools for flash micro controllers, as well as J-Link, a JTAG emulator to assist in development, debugging and production, which has rapidly become the industry standard for debug access to ARM cores.

Corporate Office:

<http://www.segger.com>

United States Office:

<http://www.segger-us.com>

EMBEDDED SOFTWARE (Middleware)



emWin

Graphics software and GUI

emWin is designed to provide an efficient, processor- and display controller-independent graphical user interface (GUI) for any application that operates with a graphical display.



embOS

Real Time Operating System

embOS is an RTOS designed to offer the benefits of a complete multitasking system for hard real time applications with minimal resources.



embOS/IP

TCP/IP stack

embOS/IP a high-performance TCP/IP stack that has been optimized for speed, versatility and a small memory footprint.



emFile

File system

emFile is an embedded file system with FAT12, FAT16 and FAT32 support. Various Device drivers, e.g. for NAND and NOR flashes, SD/MMC and Compact-Flash cards, are available.



USB-Stack

USB device/host stack

A USB stack designed to work on any embedded system with a USB controller. Bulk communication and most standard device classes are supported.

SEGGER TOOLS

Flasher

Flash programmer

Flash Programming tool primarily for micro controllers.

J-Link

JTAG emulator for ARM cores

USB driven JTAG interface for ARM cores.

J-Trace

JTAG emulator with trace

USB driven JTAG interface for ARM cores with Trace memory. supporting the ARM ETM (Embedded Trace Macrocell).

J-Link / J-Trace Related Software

Add-on software to be used with SEGGER's industry standard JTAG emulator, this includes flash programming software and flash breakpoints.



Table of Contents

1	Introduction	9
1.1	About this manual	10
1.2	What is the purpose of this eval package?	10
1.3	Software components in the package	11
2	Licensing	13
2.1	License terms	14
3	Software components	15
3.1	Setup	16
3.2	Project structure	17
3.3	Eval limitations	18
4	How to start	19
4.1	Running a sample application	20
4.2	Evaluating the SeggerDemo	21
5	SEGGER Embedded Studio	27
5.1	Changing the sample application	28
5.2	Build, download and run your application	29
6	emIDE	31
6.1	Changing the sample application	32
6.2	Build, download and run your application	32
7	IAR	33
7.1	Changing the sample application	34
7.2	Build, download and run your application	34
8	CrossWorks	35
8.1	Changing the sample application	36
8.2	Build, download and run your application	36
9	Renesas e ² studio	39
9.1	Changing the sample application	40
9.2	Build, download and run your application	40
10	Microchip MPLAB X	43
10.1	Changing the sample application	44
10.2	Build, download and run your application	44
11	Keil MDK	47
11.1	Changing the sample application	48
11.2	Build, download and run your application	49
12	Prebuild sample application	51
12.1	Using the J-Link Commander	52

13	Sample applications.....	53
13.1	List of sample applications.....	54
13.2	SeggerDemo	59
13.3	emFile samples	60
13.4	emWin samples	63
13.5	embOS/IP samples.....	66
13.6	emModbus samples	68
13.7	embOS samples.....	69
13.8	emUSB samples.....	70
13.9	emUSB Host samples.....	72
14	Literature and references.....	73

Chapter 1

Introduction

This chapter provides basic information about the purpose of this manual and the entire software package, which consists of different middleware components.

1.1 About this manual

This document describes the different SEGGER eval software components and the sample applications that are delivered with them.

1.2 What is the purpose of this eval package?

SEGGER eval packages are designed to provide customers and potential customers (you) with a complete and easy to use software package for the specified target hardware and several IDEs (e.g. Segger Embedded Studio, IAR Embedded Workbench, and Rowley CrossWorks).

It allows straightforward evaluation of the target hardware, the target compiler and SEGGER's middleware components.

Various eval packages can be retrieved from the following location:

<https://www.segger.com/evalboards.html>

1.2.1 What are the components of the software package?

The SEGGER eval software components are provided in library form, whereas the sample applications are provided as source code.

In addition, most SEGGER eval software packages include a "Prebuild"-folder containing prebuilt executables. These may simply be downloaded to the target hardware, allowing users to completely omit the compilation of the source code.

1.2.2 Can I recompile the supplied applications?

Yes, you may modify and recompile any of the application samples that are included in this software package. While an appropriate target compiler is required to do so, even an evaluation version of the compiler can be used in most cases. Please refer to "ReadMe.txt" in the "Start"-folder for more information on the respective target compiler.

1.2.3 Can I write my own applications with this eval package?

Yes, you may also write your own applications. However, the purpose of this eval package is to test the soft- and hardware for fitness. You must not use the resulting application in a product.

1.3 Software components in the package

1.3.1 emFile

emFile is SEGGER's embedded file system that can be used on any media for which you can provide basic hardware access functions.

emFile is a high-performance library that has been optimized for minimum memory consumption in RAM and ROM, high speed and versatility.

The emFile documentation may be found at "Doc\UM02001_emFile.pdf".

1.3.2 emModbus

emModBus is SEGGER's implementation of the Modbus protocol, supports communication via UART (ASCII, RTU) and Ethernet (Modbus/TCP and Modbus/UDP) and is capable to communicate with any Modbus compliant device.

The emModbus documentation may be found at "Doc\UM14001_emModbus.pdf".

1.3.3 emWin

emWin is SEGGER's embedded Graphical User Interface (GUI) using a feature rich API and providing an efficient, processor- and LCD-controller-independent GUI for any application that operates with a graphical LCD. It may be adapted to any size physical and virtual display with any LCD controller and CPU.

The emWin documentation may be found at "Doc\UM03001_emWin5.pdf".

1.3.4 embOS

embOS is SEGGER's embedded priority-controlled multitasking system. It is designed to be used as an embedded operating system for the development of real-time applications and has been optimized for minimum memory consumption in both RAM and ROM, as well as high speed and versatility.

The embOS documentation may be found at "Doc\UM01001_embOS.pdf" and "Doc\UM010xx_embOS_<cpu>_<compiler>.pdf".

1.3.5 embOS/IP

embOS/IP is SEGGER's embedded TCP/IP stack. It is a CPU independent, high-performance TCP/IP stack that has been optimized for speed, versatility and small footprint.

The embOS/IP documentation may be found at "Doc\UM07001_embOSIP.pdf".

1.3.6 emUSB Device

emUSB Device is SEGGER's embedded USB Device stack. It features bulk communication as well as device classes such as MSD, CDC or HID.

The emUSB Device documentation may be found at "Doc\UM09001_emUSB.pdf".

1.3.7 emUSB Host

emUSB Host is SEGGER's embedded USB Host stack. It features bulk communication as well as device classes such as MSD or HID, and also supports external hubs.

The emUSB Host documentation may be found at "Doc\UM10001_emUSBH.pdf".

Chapter 2

Licensing

This chapter describes the licensing terms under which the SEGGER Eval Software is published.

2.1 License terms

This SEGGER Eval Software package may only be used when fully agreeing to the terms mentioned in this chapter and agreeing to the license terms mentioned in "License.txt".

If this description contradicts the terms of the "License.txt", the terms of the license file should supersede this description.

In case of doubt, please contact us: info@segger.com

2.1.1 What you may do

You may use the software contained in this eval package to evaluate SEGGER software on the target hardware.

You may recompile and modify the sample programs provided as part of the package.

2.1.2 What you are not allowed to do

You are not allowed to use the software in this eval package for something other than evaluating the SEGGER software. You are not allowed to use this software package in a product.

Chapter 3

Software components

This chapter explains the folder structure used for the software package, the project structure, and the limitations of the eval version.

3.1 Setup

Requirements

Compilation of the provided application samples requires an appropriate target compiler (or eval version of the target compiler) as indicated in `ReadMe.txt`.

Furthermore, if emWin was included in the software package, it also contains a ready-to-go project for Microsoft Visual Studio 2010 that allows a recompilation of the emWin simulation. To use the project, either Microsoft Visual Studio 2010 or Microsoft Visual Studio .Net are required.

Installation

All SEGGER Eval Packages are supplied as ZIP-files. The most recent version of the software can be retrieved from the following location:

<https://www.segger.com/evalboards.html>

Extract the ZIP-file to a folder of your choice while retaining the folder structure of the ZIP-file. In general, no further installation steps are required. You may simply use the provided sample applications or start to modify them in order to write your application.

SEGGER Eval Packages include evaluation builds of some or all of the following products:

Component	Description
emFile	SEGGER's file system
emModbus	SEGGER's Modbus stack
embOS	SEGGER's real time operating system
embOS/IP	SEGGER's TCP/IP stack
emUSB Device	SEGGER's USB Device stack
emUSB Host	SEGGER's USB Host stack
emWin	SEGGER's graphic library and GUI

Each component may be purchased separately and can be used in any combination. For ordering information, please contact us: info@segger.com

3.2 Project structure

Each component of the eval package is provided in library form with additional header files; sample applications are provided as source code.

The root directory of the eval package includes three folders: "Doc", "Prebuild", and "Start". Furthermore, the License.txt file and this application note are located at the root directory.

The "Prebuild"-folder is included in many SEGGER Eval Software packages and contains prebuilt executables. These may simply be downloaded to the target hardware without a need for a compiler. For additional information please refer to *Chapter 12*.

The "Doc"-folder contains the User & Reference guides for the included middleware components.

The "Start"-folder contains all files that are required for the SEGGER Eval Software itself. Therefore, it includes some or all of the directories described below:

Directory	Description
Application	Includes various application samples.
Config	Includes the configuration files of the included middleware components.
FS	Includes the emFile header files and libraries.
GUI	Includes the emWin header files and libraries.
Inc	Includes several header files that are not directly related to one of the included middleware components.
IP	Includes the embOS/IP header files and libraries.
MB	Includes the emModbus header files and libraries.
OS	Includes the embOS header files and libraries as well as two source files: <code>Main.c</code> and <code>OS_Error.c</code> .
SEGGER	Includes utility functions that are not directly related to one of the included middleware components.
Setup	Includes the Board Support Package (BSP). The BSP consists of all files that are required to initialize the target hardware, or to build an executable for the target hardware.
Shared	Includes shared files.
USB	Includes the emUSB Device header files and libraries.
USBH	Includes the emUSB Host header files and libraries.
Windows	Includes example applications for Microsoft Windows and the drivers required for USB functionality. Refer to the User and reference guide for emUSB [UM09001] for more information about the installation of these drivers.

3.3 Eval limitations

The included versions of the different components of the eval package have the following limitations:

Component	Description
emFile	The eval version of the emFile libraries can only handle one open file at any given time.
embOS	The eval version of the embOS libraries runs without a time limit with a maximum of three tasks. If your application creates more than three tasks, embOS stops after a defined time limit. For embOS versions before v3.82t, the time limit is 12 minutes. For versions including v3.82t and later, the time limit is 12 hours.
embOS/IP	The eval version of the embOS/IP libraries have a time limit of 12 hours to the connection.
emModbus	The eval version of the emModbus libraries have a time limit of 12 hours to the connection.
emUSB Device	The eval version of the emUSB Device libraries have a time limit of 12 hours to the connection.
emUSB Host	The eval version of the emUSB Host libraries have a time limit of 12 hours to the connection.
emWin	The eval version of the emWin library shows an evaluation notification before the actual application starts.

Any usage of the eval package or any of its part indicates your acknowledgment and agreement to the SEGGER eval software license. A `License.txt` is located in the root directory of the eval package.

Chapter 4

How to start

This chapter provides information on how to start working with the sample applications.

4.1 Running a sample application

The folder "Application" includes a default application sample, which itself consists of either a single file or a subfolder, and a subfolder "Excluded", which contains all other application samples and is excluded from build. The default application sample may vary depending on the included middleware components: If emWin was included in the package, the SEGGERDEMO application sample is the default application. If emWin was not included, however, the default is either OS_StartLEDBlink or OS_Start2Tasks.

Each available sample contains a Task `MainTask()` and possibly other tasks. The `MainTask()` is always created from within the `main()` routine, which itself is located at `Main.c` in the folder "OS". `Main.c` is required by all supplied sample applications.

```
#include "RTOS.h"
#include "BSP.h"

/*****
 *
 *      Prototypes
 *
 *****/

#ifdef __cplusplus
extern "C" { /* Make sure we have C-declarations in C++ programs */
#endif
void MainTask(void);
#ifdef __cplusplus
}
#endif

/*****
 *
 *      Static data
 *
 *****/

static OS_STACKPTR int Stack0[768]; /* Task stack */
static OS_TASK      TCB0;          /* Task-control-block */

/*****
 *
 *      main()
 *
 * Function description
 * Application entry point
 */
int main(void) {
    OS_IncDI(); /* Initially disable interrupts */
    OS_InitKern(); /* Initialize OS */
    OS_InitHW(); /* Initialize Hardware for OS */
    BSP_Init(); /* Initialize BSP module */
    BSP_SetLED(0); /* Initially set LED */
    /* You need to create at least one task before calling OS_Start() */
    OS_CREATETASK(&TCB0, "MainTask", MainTask, 100, Stack0);
    OS_Start(); /* Start multitasking */
    return 0;
}
```

To switch to another application sample than the default application, you may simply exclude that default application and include any other application instead.

Some embOS/IP and emUSB application samples are client/server applications and consist of one application for the embedded target and a second application for personal computers running Microsoft Windows. The latter are provided both in source code and as executable within the "Windows" directory. To recompile the executable, an appropriate compiler is required: For example, suitable compilers are delivered with Microsoft Visual C++ 6.0 or Microsoft Visual Studio .Net.

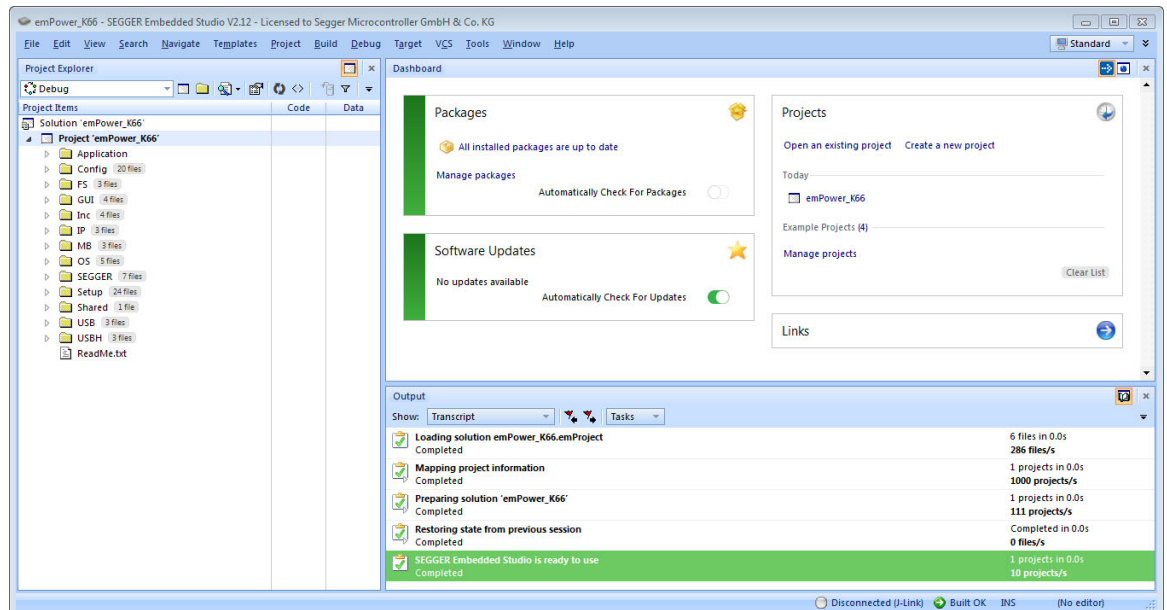
4.2 Evaluating the SeggerDemo

The following is a generic step by step description to fully evaluate the SeggerDemo application sample. All pictures in this chapter are exemplary; the actual display will differ when using a different IDE. For IDE-specific descriptions, please refer to the following chapters.

Furthermore, the steps described below may differ from hardware to hardware, but, generally, will be similar.

4.2.1 Step 1: Open the project

- Get a fresh copy of the project and files.
- Open the project file with your IDE. Your screen should look similar to the screenshot below.



4.2.2 Step 2: Establish connections

- Make sure that your eval board is powered.
- Make sure your eval board and your PC are connected via your debug probe (e.g. SEGGER J-Link or an onboard debugger).

Optional:

- Connect your eval board with a RS232 cable to your PC.¹
- Connect your target to a network with an ethernet cable. Ensure that your PC is connected to the same network.²
- Establish an USB connection between your eval board and your PC via USB cable.³
- Plug in a Human Interface Device to your eval board.⁴

1. Only applies if UART is supported by the BSP

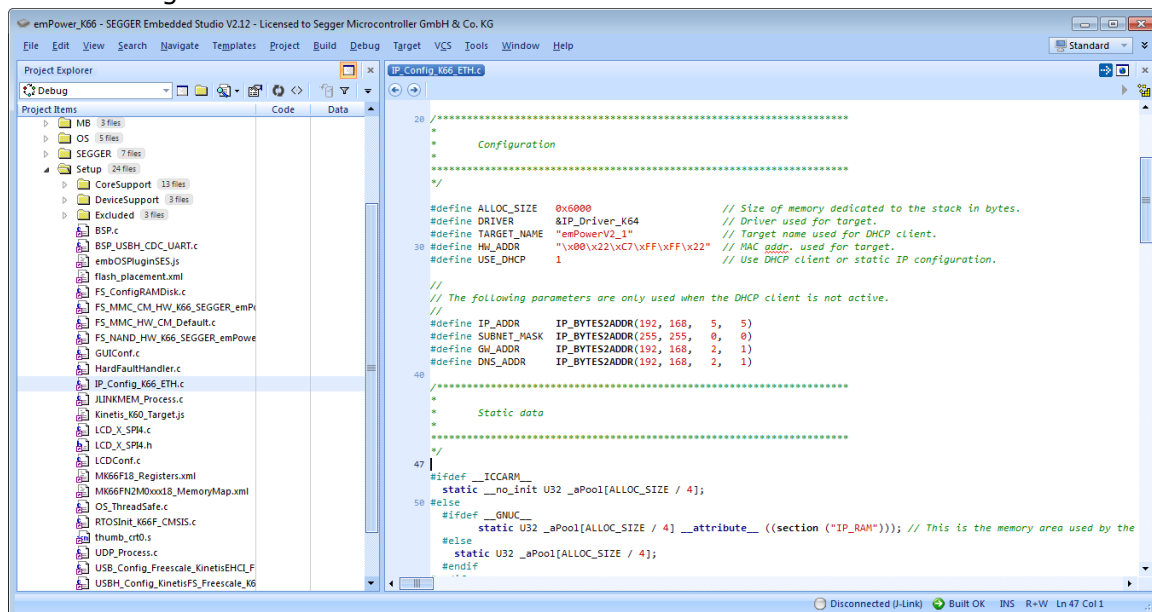
2. Only applies if the BSP contains embOS/IP

3. Only applies if the BSP contains emUSB Device

4. Only applies if the BSP contains emUSB Host

4.2.3 Step3: Configuration

- If you are not using a DHCP server in your network you will need to manually configure the IP address and the subnetmask for your target in the file **Start\Setup\IP_Config_*.c**. The screenshot below shows an example configuration using the IP address 192.168.5.5 with subnetmask 255.255.0.0.



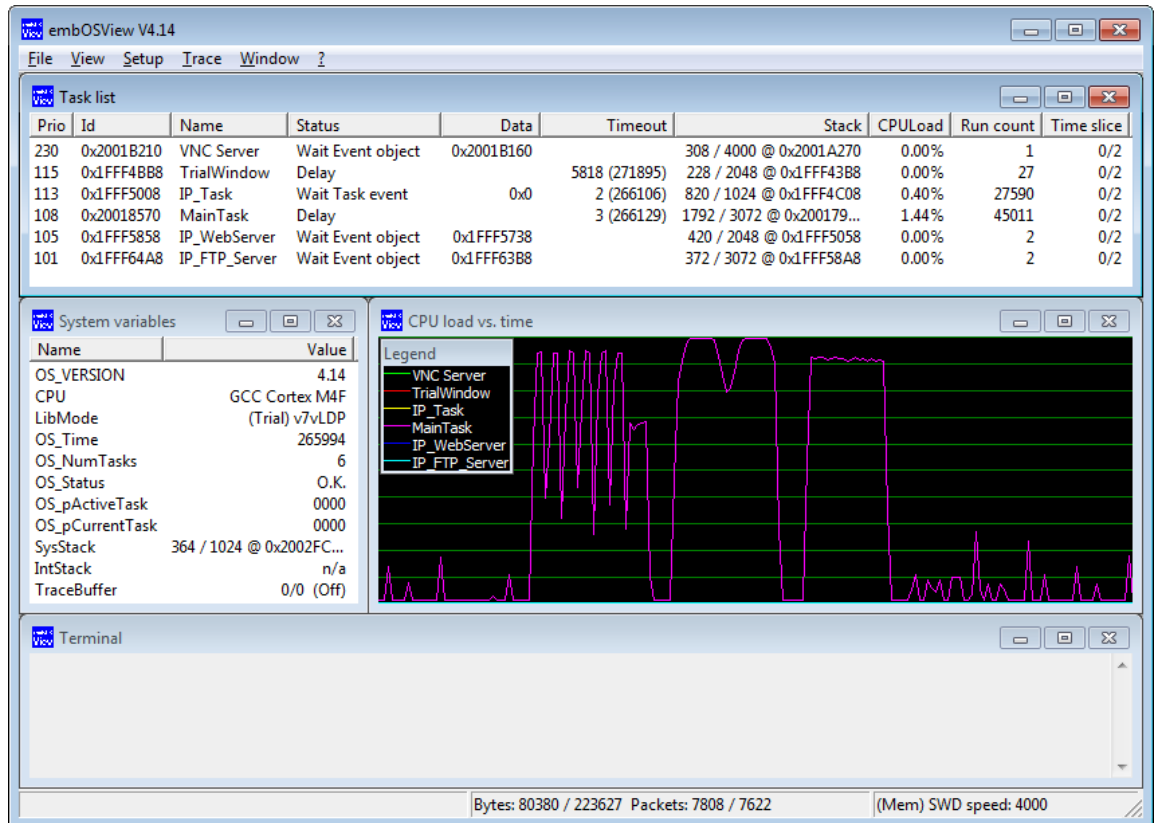
4.2.4 Step 4: Start the application

- Build the project. The build log should report no errors or warnings.
- Download the application and start the debug session. Your application should halt at `main()`.
- Upon continuation of the application, the SeggerDemo should be displayed on the LCD.¹

1. Only applies if the BSP contains emWin

4.2.5 Step5: Evaluate the middleware components

- Start **Windows\OS\embOSView.exe** to verify that the UART is sending OS specific data.¹ Your screen should look similar to the screenshot below. Alternatively, if you are using the SEGGER J-Link, instead of communicating via UART you may also use the J-Link interface or Ethernet to communicate with embOSView. In case of deviant results, please refer to the embOS Generic Documentation (Doc\UM01001_embOS.pdf) for troubleshooting



- In the lower left corner of the SeggerDemo slideshow, you will see the target's IP address.² Open a command prompt and try to ping your target. Your screen should look similar to the screenshot below.³

```

Administrator: Command Prompt

c:\>ping 192.168.11.204

Pinging 192.168.11.204 with 32 bytes of data:
Reply from 192.168.11.204: bytes=32 time<1ms TTL=64
Reply from 192.168.11.204: bytes=32 time<1ms TTL=64
Reply from 192.168.11.204: bytes=32 time<1ms TTL=64
Reply from 192.168.11.204: bytes=32 time<1ms TTL=64

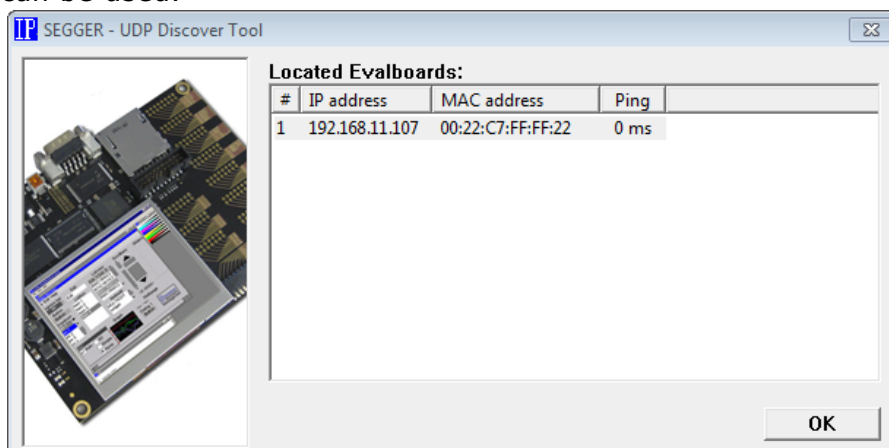
Ping statistics for 192.168.11.204:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms

c:\>

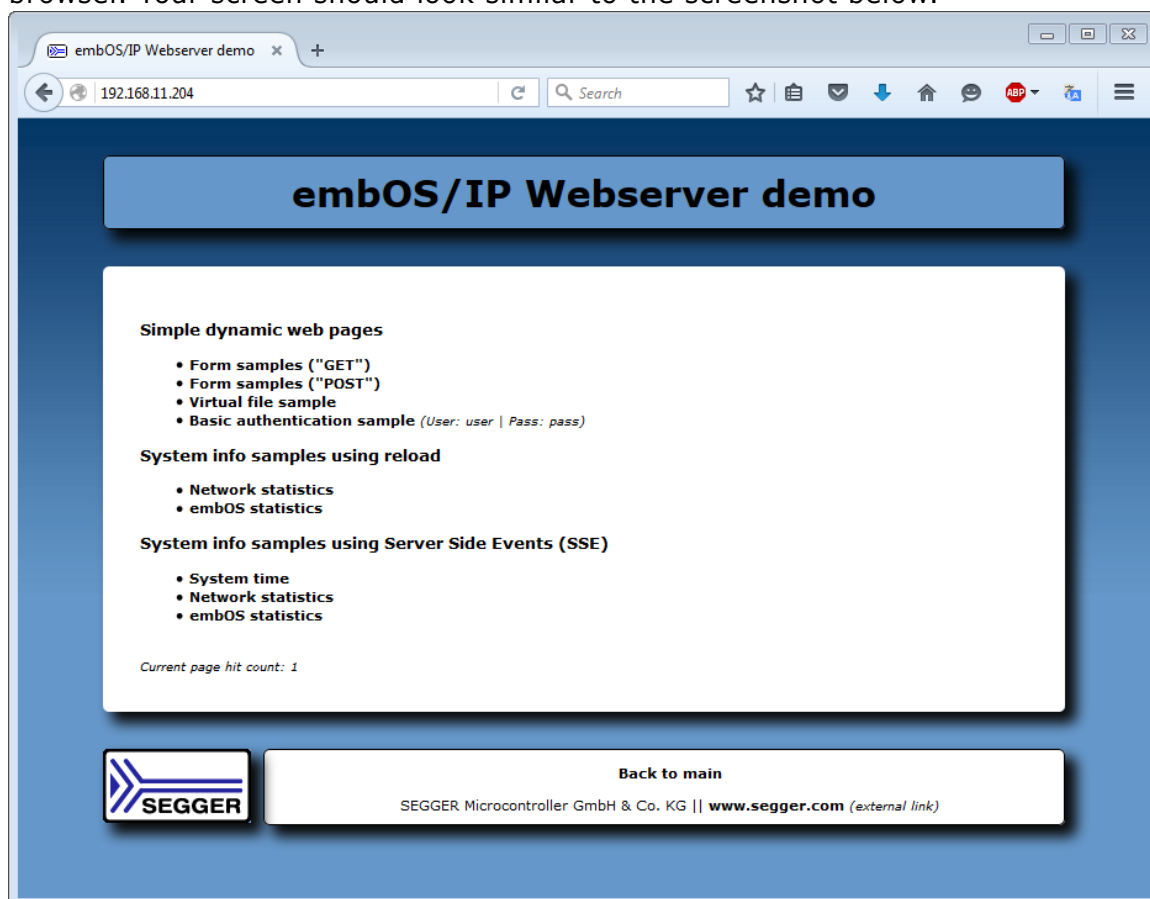
```

1. Only applies if UART is supported by the BSP
 2. Depends on screen size
 3. Only applies if the BSP contains embOS/IP

- The IP address of your board can also be identified via UDP. For this purpose the sample UDPDiscover, located at **Start\Windows\IP\UDPDiscoverGUI.exe**, can be used.¹



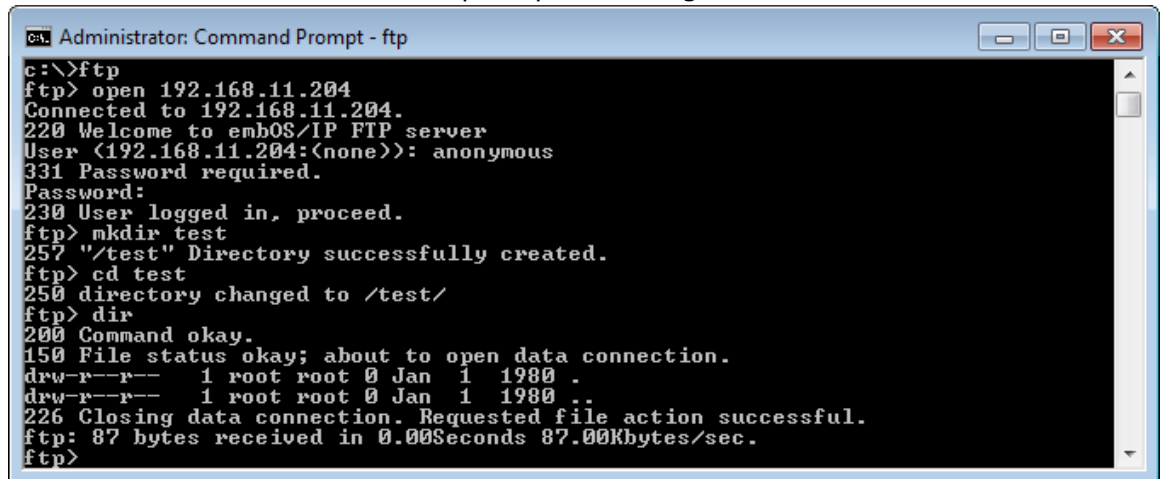
- Connect to the webserver by entering **http://<target_ip>** into your web-browser. Your screen should look similar to the screenshot below.²



1. Only applies if the BSP contains embOS/IP

2. Only applies if the BSP contains embOS/IP and a webserver is included

- You can test the FTP server by connecting a FTP client to the target on port 21. You can do this via a command prompt following to the screenshot below.¹



```

Administrator: Command Prompt - ftp
c:\>ftp
ftp> open 192.168.11.204
Connected to 192.168.11.204.
220 Welcome to embOS/IP FTP server
User (192.168.11.204:(none)): anonymous
331 Password required.
Password:
230 User logged in, proceed.
ftp> mkdir test
257 "/test" Directory successfully created.
ftp> cd test
250 directory changed to /test/
ftp> dir
200 Command okay.
150 File status okay; about to open data connection.
drw-r--r--  1 root root 0 Jan  1 1980 .
drw-r--r--  1 root root 0 Jan  1 1980 ..
226 Closing data connection. Requested file action successful.
ftp: 87 bytes received in 0.00Seconds 87.00Kbytes/sec.
ftp>

```

- A new mass storage device should be visible in your explorer.² If you want to change the memory configuration (e.g. RAMDisk, SD-Card, NAND-Flash...) include the correspondent config file (**Start\Setup\Exclude\FS_Config_*.c**) and exclude the current one.
- A VNC server is running in the background.³ To connect to the server a VNC client, located at **Start\Windows\GUI\emVNC.exe**, can be used.



- After evaluating all parts of the SeggerDemo, you may want to evaluate the other samples located in the **Application** folder. Please refer to the chapter *Sample applications* on page 53 for more information.

1. Only applies if the BSP contains embOS/IP and a ftp server is included

2. Only applies if the BSP contains emUSB Device and emFile

3. Only applies if the BSP contains embOS/IP and a VNC Server is included

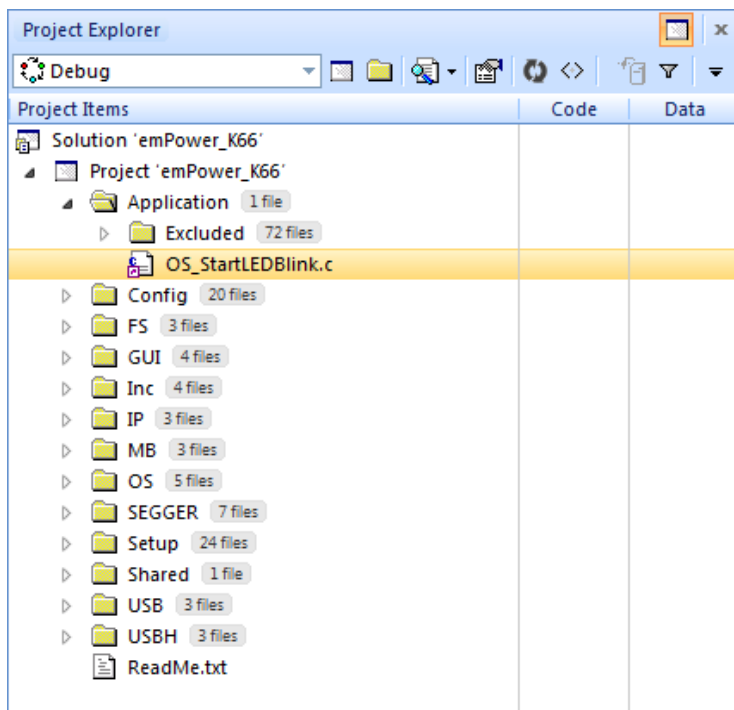
Chapter 5

SEGGER Embedded Studio

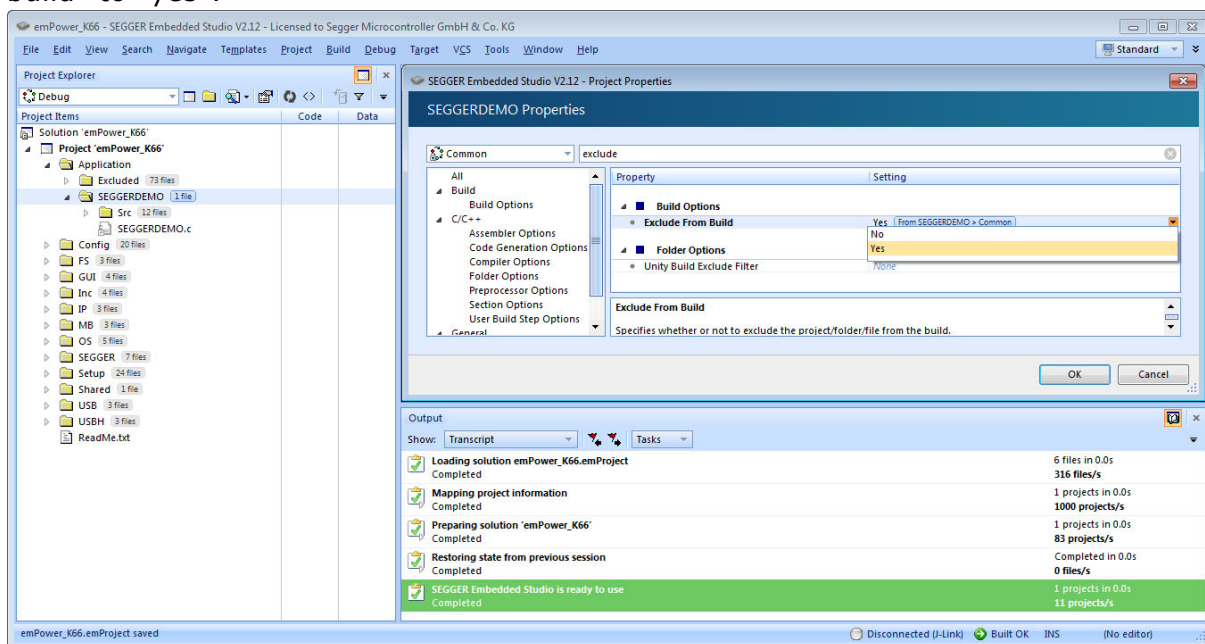
This chapter contains additional information to start working with the sample applications and SEGGER Embedded Studio.

5.1 Changing the sample application

Open the **Application** folder and use "drag and drop" in the **Project Explorer** window to change the sample application. Move the included sample application (e.g. OS_StartLEDBlink.c) to the folder **Excluded** and move the favored sample application from the folder **Excluded** to the folder **Application**. This works also with a whole folder e.g. the SEGGERDEMO folder.



To exclude a whole folder (e.g. SEGGERDEMO) right click on it and select **Edit Properties** to open the window **Properties**. Type "exclude" into the properties search field (located at the top of the properties window) and set the Build option "Exclude from build" to "yes".

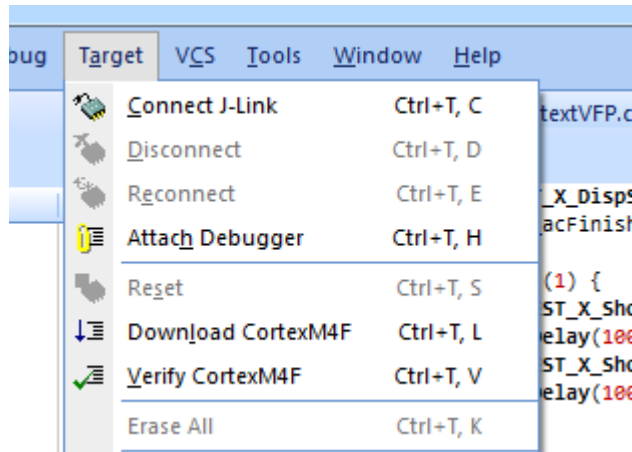


5.2 Build, download and run your application

Press [F7] to compile the selected application. The build log should report no errors or warnings.

Before you download the application, make sure you are connected to the target. To connect to a target, open the tab **Target** and click on **Connect J-Link**.

The tab will look similar to the picture below.



If you are connected to your target, you may download the application by pressing [F5]. As soon as your application halts at main(), press [F5] again to run the application.

Chapter 6

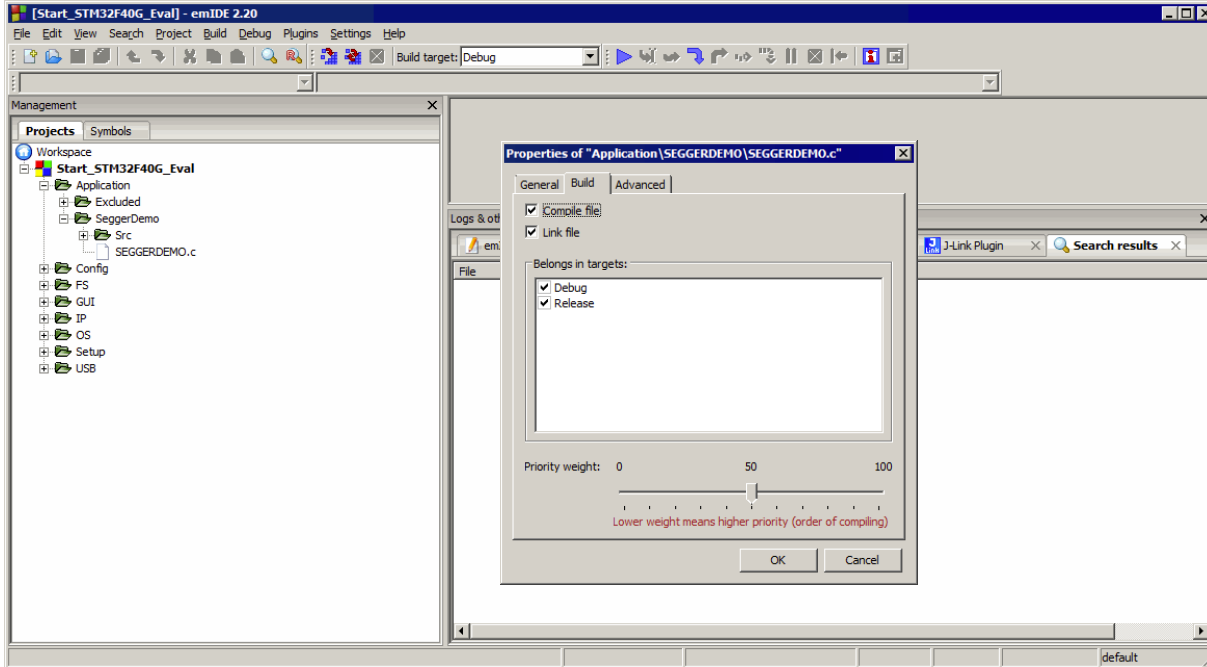
emIDE

This chapter contains additional information to start working with the sample applications and emIDE.

6.1 Changing the sample application

To change the sample application, open the **Application** folder, open the context menu of the currently selected application with a right click and choose "Properties". Then select the tab "Build" in the properties window and un-check the upper check boxes, which are labeled "Compile file" and "Link file".

To include a sample application, go to **Application/Excluded** and repeat the previous steps on the file you wish to be included.



6.2 Build, download and run your application

Press [F7] to compile the selected application. The build log should report no errors or warnings.

You may download the application by pressing [F5]. When your application halts at main(), press [F5] once more to resume the application's execution.

Chapter 7

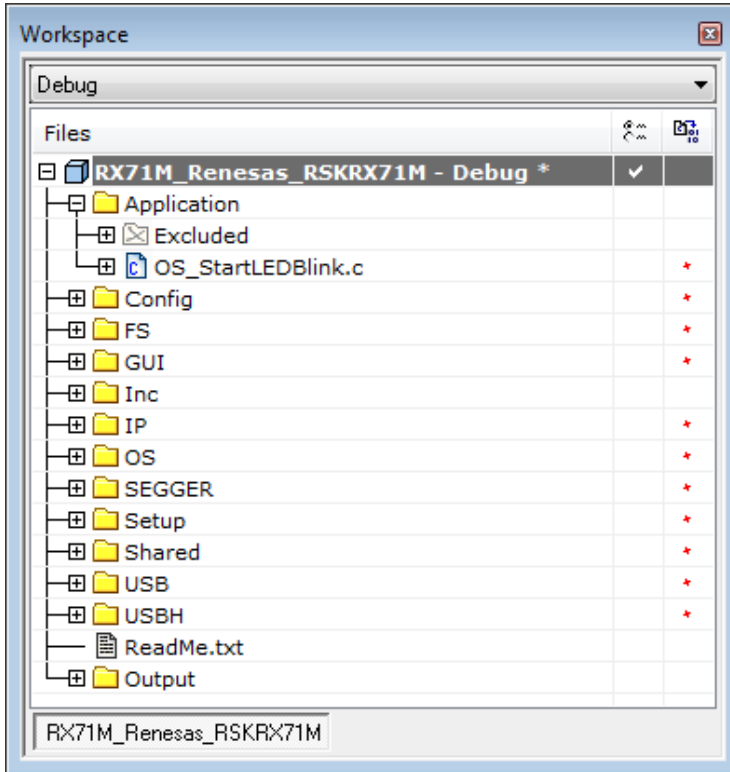
IAR

This chapter contains additional information to start working with the sample applications and IAR Embedded Workbench.

7.1 Changing the sample application

Open the **Application** folder and use "drag and drop" in the **Workspace** window of the IAR Embedded Workbench to change the sample application. Move the included sample application (e.g. `OS_StartLEDBlink.c`) to the folder **Excluded** and move the favored sample application from the folder **Excluded** to the folder **Application**. This works also with a whole folder e.g. the `SEGGERDEMO` folder.

Changing an application is only possible if no debug session is running.



7.2 Build, download and run your application

Press [F7] to compile the included application. The build log should report no errors and no warnings.

After a successful build you can download the application by pressing [ctrl + D].

When the application halts at `main()` press [F5] to run your application.

Chapter 8

CrossWorks

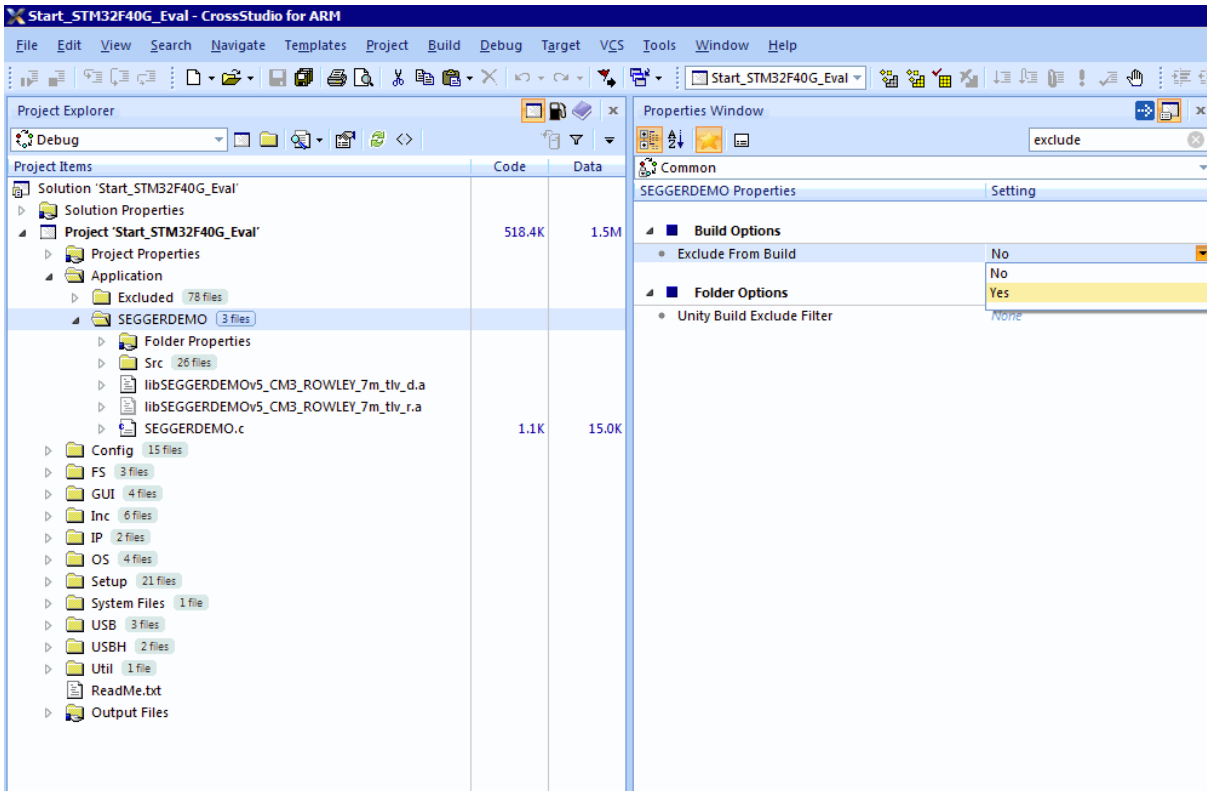
This chapter contains additional information to start working with the sample applications and Rowley's CrossWorks.

8.1 Changing the sample application

To change the sample application open the **Application** folder, open the context menu of the current application (e.g. `OS_StartLEDBlink.c`) with a right click and choose "Exclude from build" to exclude it.

To include a sample application go to **Application/Excluded** and repeat the previous steps on the file you wish to be included.

To exclude a whole folder (e.g. the `SEGGERDEMO`) right click on it and open the properties window. Type into the properties search field (located at the top of the properties window) "exclude" and set the Build option "Exclude from build" to "yes".

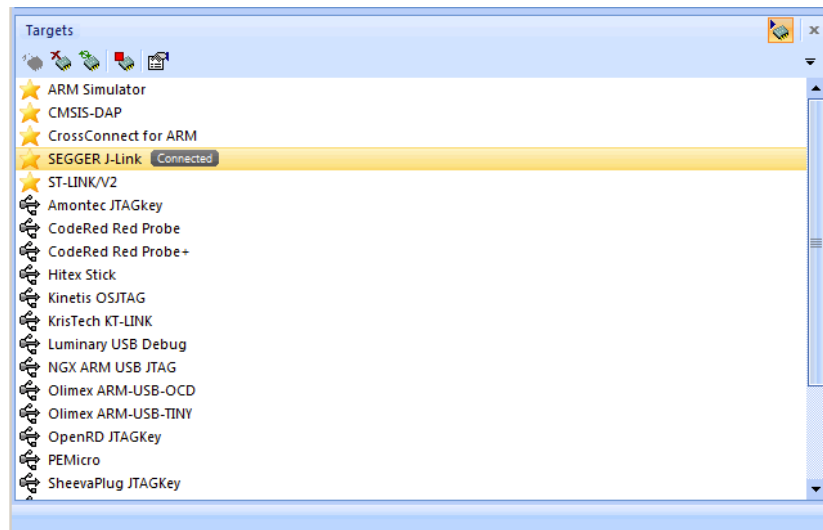


8.2 Build, download and run your application

Press [F7] to compile the included application. The build log should report no errors and no warnings.

Before you can download the application make sure you are connected to the target. To connect to a target double click on an entry in the "Targets" window or use the appropriate entry in the context menu.

The targets window of CrossWorks should look similar to the picture below.



If you are connected to your target you can download the application by pressing [F5].

As soon as your application halts at main() press [F5] again to run the application.

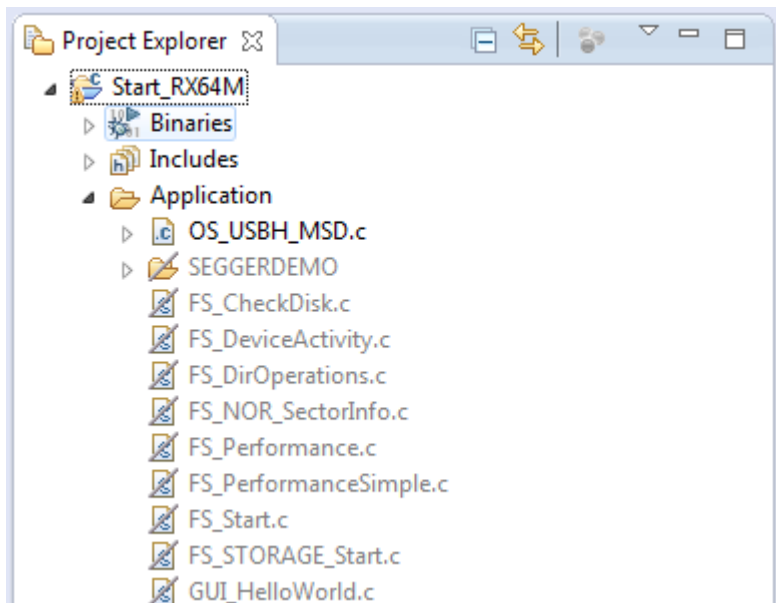
Chapter 9

Renesas e² studio

This chapter contains additional information to start working with the sample applications and Renesas e² studio.

9.1 Changing the sample application

Open the **Application** folder in the **Project Explorer** window of the e² studio to change the sample application.



Right-click the already included application and select **Properties**, in the properties menu check the **Exclude from build** checkbox.

Right-click the application you wish to evaluate, select **Exclude from build...**, deselect all available configurations and click **OK**.

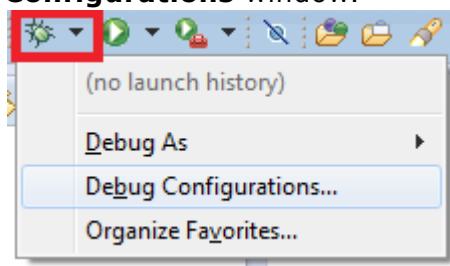
This works also with a whole folder e.g. the `SEGGERDEMO` folder.

Changing an application is only possible if no debug session is running.

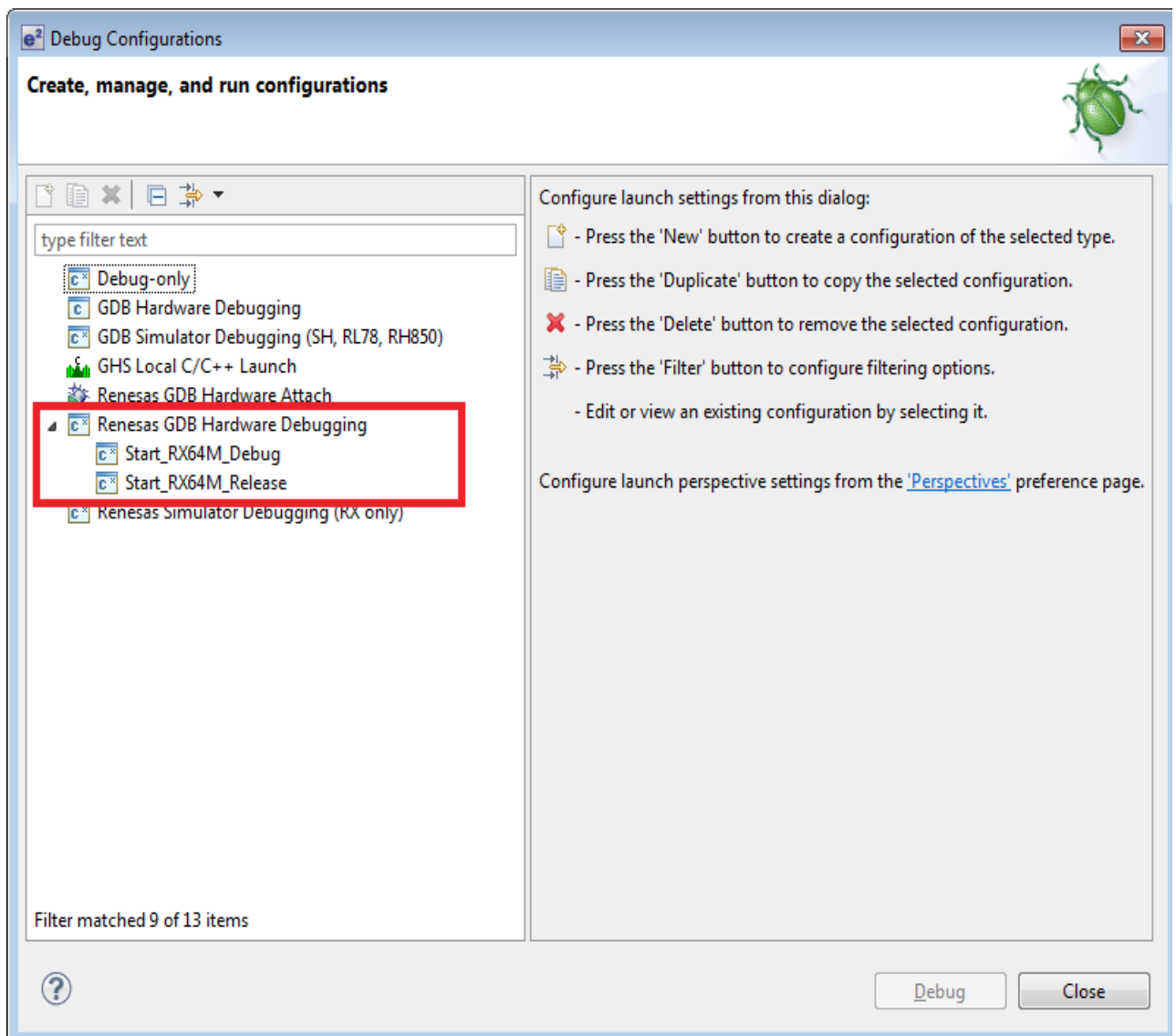
9.2 Build, download and run your application

Press [ctrl + b] to compile the included application. The build log should report no errors or warnings.

After a successful build you can download the application by opening the **Debug Configurations** window.



In the **Debug Configurations** window select the appropriate debug configuration and double-click it:



As soon as your application halts at `main()` press `[F8]` to run the application.

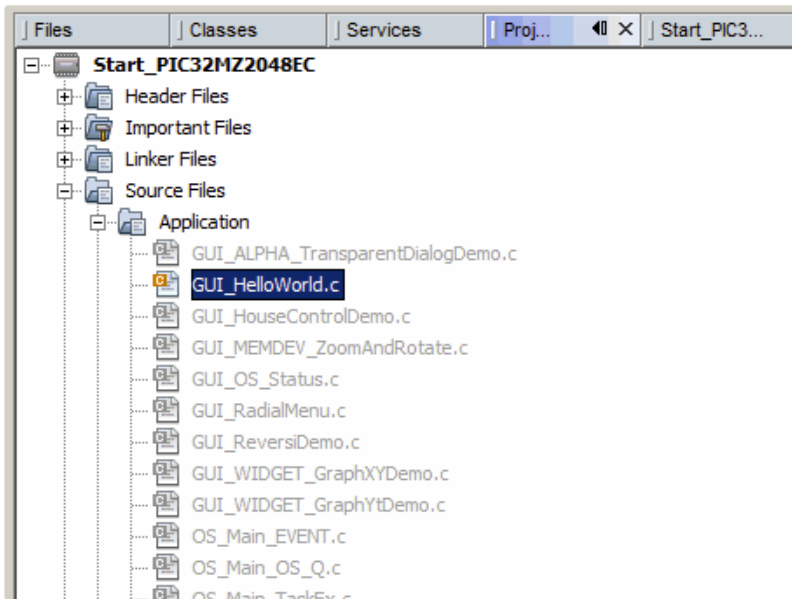
Chapter 10

Microchip MPLAB X

This chapter contains additional information to start working with the sample applications and Microchip MPLAB X.

10.1 Changing the sample application

Open the **Application** folder in the **Project Explorer** window of MPLAB X to change the sample application.



Right-click the already included application and select **Exclude from build...**, select all available configurations and click **OK**.

Right-click the application you wish to evaluate, select **Exclude from build...**, deselect all available configurations and click **OK**.

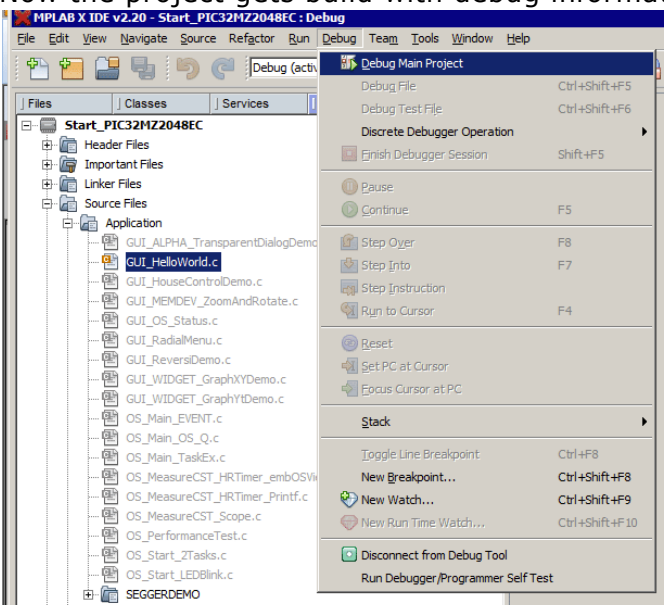
This works also with a whole folder e.g. the `SEGGERDEMO` folder.

Changing an application is only possible if no debug session is running.

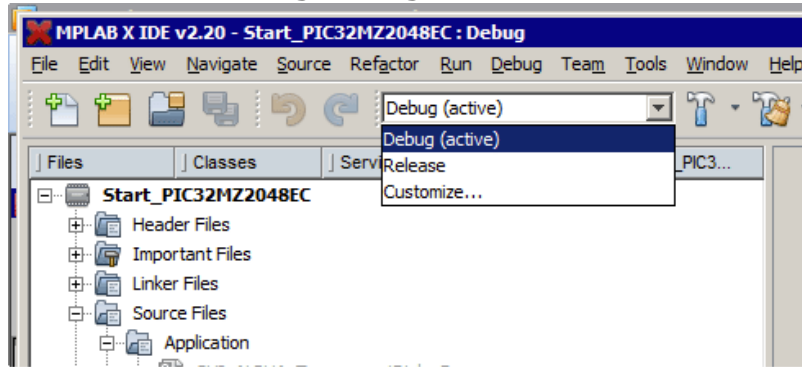
10.2 Build, download and run your application

To download and run the included application press [F6]. Then the project will get compiled and downloaded. The build log should report no errors or warnings.

To run a debug session open the **Debug** menu and choose **Debug Main Project**. Now the project gets build with debug information and downloaded to the target.



To choose another **Debug Configuration** click in the configuration drop down menu and choose a **Debug Configuration**:



These are some useful shortcut keys:

- [Ctrl + F8] - Toggle line breakpoint
- [F5] - Run the application
- [F7] - Step into
- [F8] - Step over

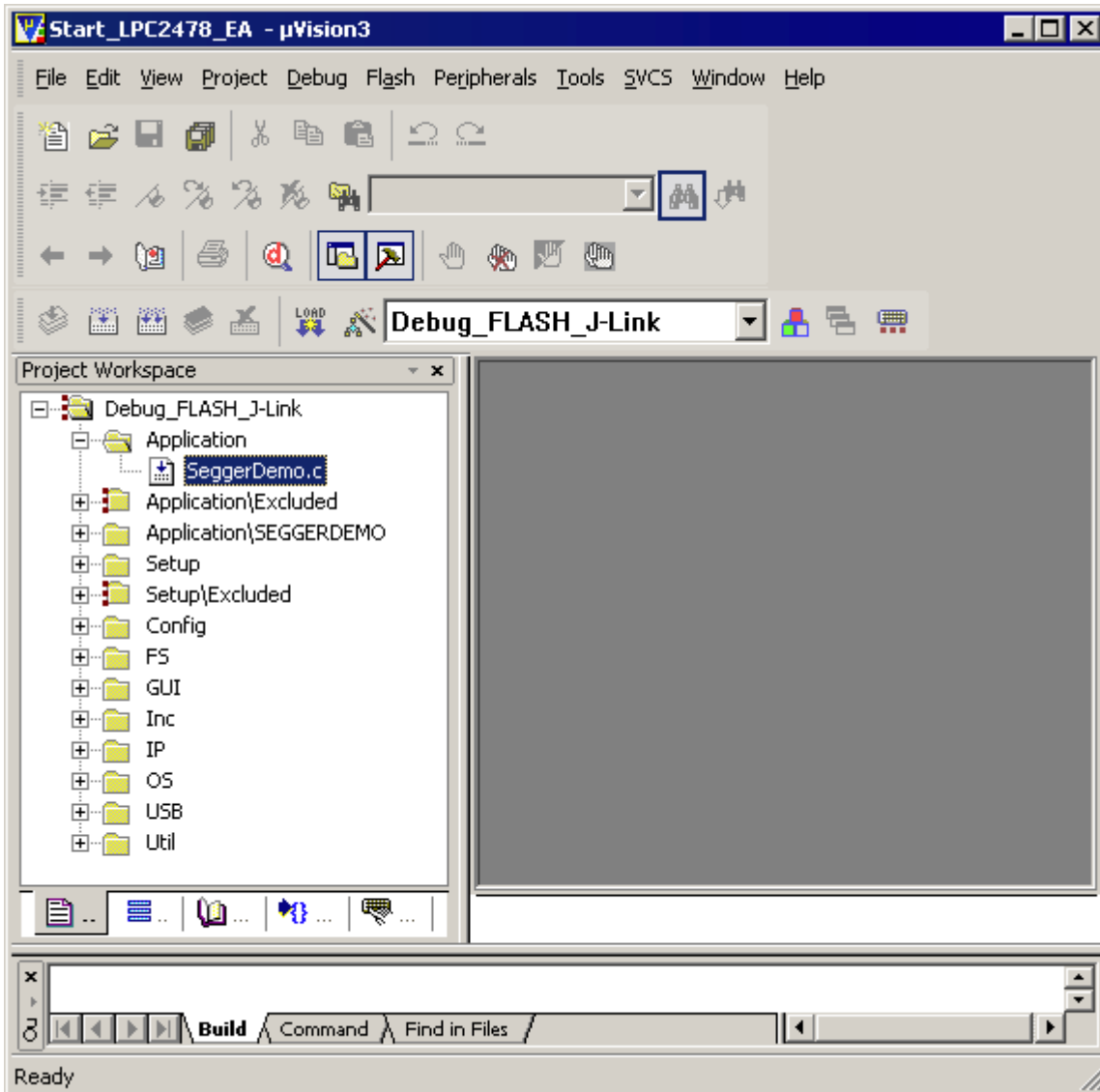
Chapter 11

Keil MDK

This chapter contains additional information to start working with the sample applications and Keil MDK.

11.1 Changing the sample application

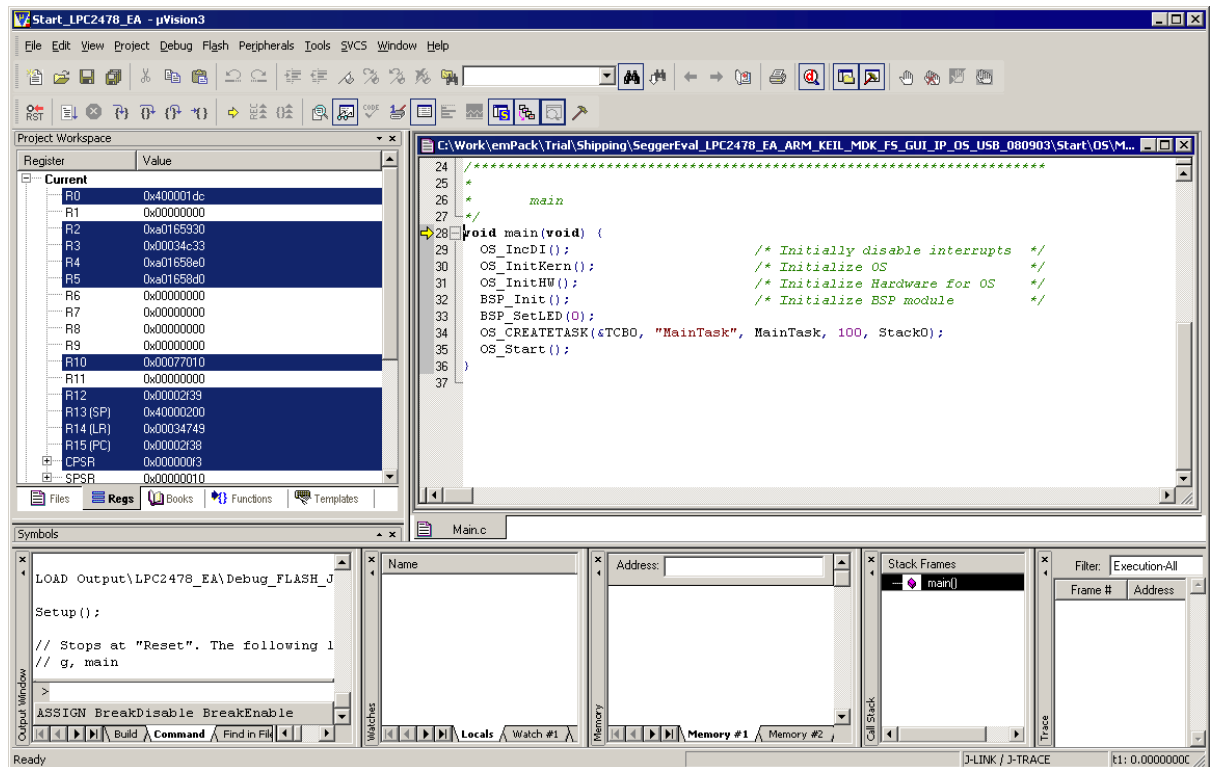
To change the sample application, open the **Application** folder and use "drag and drop" in the **Project** window of the KEIL MDK IDE. Move the included sample application (e.g. `SeggerDemo.c`) in the **Application\Excluded** folder and move the favored sample application from the **ApplicationExcluded** folder to the **Application** folder.



11.2 Build, download and run your application

Press [F7] to compile the selected sample application. The build log should report no errors or warnings.

Press CTRL+[F5] to download the application and start a debug session. After download your screen should be similar to the screenshot below, halting at `main()`.



As soon as your application halts at `main()`, press [F5] again to run the application.

Chapter 12

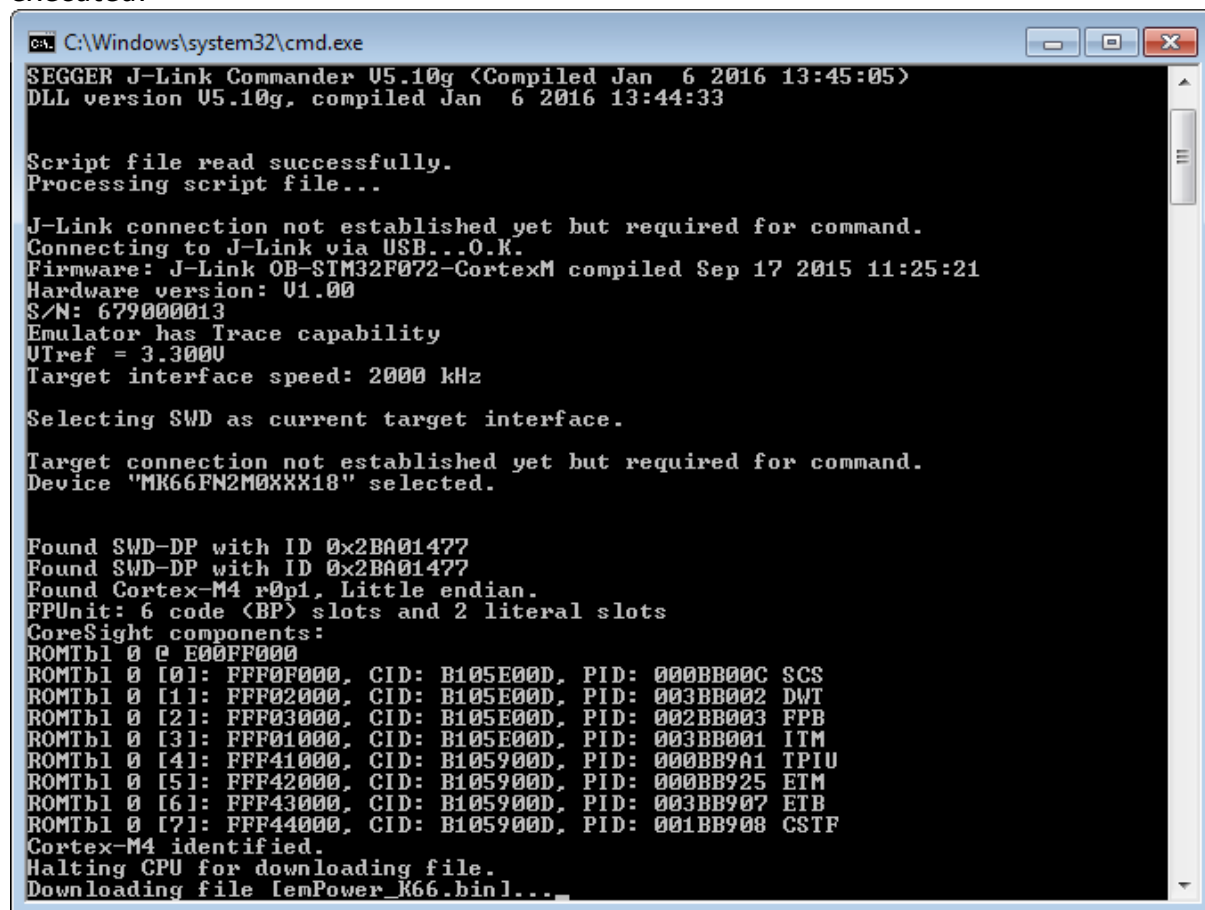
Prebuild sample application

This chapter explains how to download the prebuilt application samples into your target using the `SEGGER J-Link` and the `J-Link Commander`.

12.1 Using the J-Link Commander

If you are using a SEGGER J-Link as debug probe, it is possible to download the pre-built sample application via the J-Link Commander by following the steps described in this chapter.

Connect your target to your host PC via J-Link, ensure your board is powered and start the batch file "StartBinary.bat". The download of the application will now be executed.



```

C:\Windows\system32\cmd.exe
SEGGER J-Link Commander V5.10g <Compiled Jan  6 2016 13:45:05>
DLL version V5.10g, compiled Jan  6 2016 13:44:33

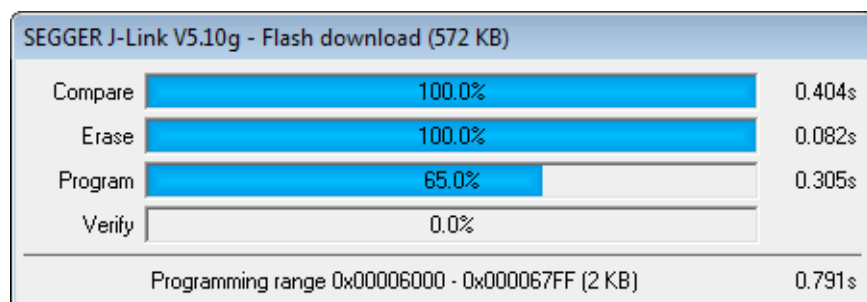
Script file read successfully.
Processing script file...

J-Link connection not established yet but required for command.
Connecting to J-Link via USB...O.K.
Firmware: J-Link OB-STIM32F072-CortexM compiled Sep 17 2015 11:25:21
Hardware version: V1.00
S/N: 679000013
Emulator has Trace capability
UTref = 3.300V
Target interface speed: 2000 kHz

Selecting SWD as current target interface.

Target connection not established yet but required for command.
Device "MK66FN2M0XXX18" selected.

Found SWD-DP with ID 0x2BA01477
Found SWD-DP with ID 0x2BA01477
Found Cortex-M4 r0pl, Little endian.
FPUnit: 6 code <BP> slots and 2 literal slots
CoreSight components:
ROMTbl 0 @ E00FF000
ROMTbl 0 [0]: FFF0F000, CID: B105E00D, PID: 000BB00C SCS
ROMTbl 0 [1]: FFF02000, CID: B105E00D, PID: 003BB002 DWT
ROMTbl 0 [2]: FFF03000, CID: B105E00D, PID: 002BB003 FPB
ROMTbl 0 [3]: FFF01000, CID: B105E00D, PID: 003BB001 ITM
ROMTbl 0 [4]: FFF41000, CID: B105900D, PID: 000BB9A1 TPIU
ROMTbl 0 [5]: FFF42000, CID: B105900D, PID: 000BB925 ETM
ROMTbl 0 [6]: FFF43000, CID: B105900D, PID: 003BB907 ETB
ROMTbl 0 [7]: FFF44000, CID: B105900D, PID: 001BB908 CSTF
Cortex-M4 identified.
Halting CPU for downloading file.
Downloading file [emPower_K66.bin]...
  
```



After the download has been finished, the prebuild application will start automatically.

For additional information on how to use the J-Link or the J-Link Commander, please refer to the J-Link User & Reference guide (UM08001_JLink.pdf).

The J-Link User & Reference guide and the J-Link Commander are part of any J-Link software package, which may be downloaded from www.segger.com.

Chapter 13

Sample applications

This chapter describes the usage of most sample applications.

13.1 List of sample applications

The list below contains a short description of the available samples. All samples are well documented and should have a sample description in the header. In addition, the most important sample applications are described in detail within the following sections. All application samples available with this specific SEGGER Eval Software package are located in the **Start\Application** folder.

File(s)	Description
SEGGERDEMO*.*	SEGGER middleware product demo. It demonstrates the usage of emWin by showing product features, the usage of embOS/IP by running a webserver and an FTP server, the usage of emFile by providing a filesystem, the usage of emUSB by providing mass storage access to the filesystem and the usage of embOS as RTOS.
emFile samples	
FS_CheckDisk.c	Application sample demonstrating disk checking functionality.
FS_DeviceActivity	Application sample demonstrating the usage of the callback invoked on each device operation.
FS_DirOperations.c	Application sample demonstrating the creation of directories and files.
FS_NAND_Dump.c	Application sample that dumps the contents of a NAND flash to an emFile volume.
FS_NOR_SectorInfo.c	Application sample that outputs sector information of NOR flash devices.
FS_Performance.c	Application sample demonstrating performance measurements. Outputs detailed results.
FS_PerformanceSimple.c	Application sample demonstrating performance measurements. Outputs limited results.
FS_Start.c	Application sample demonstrating the start-up of the filesystem.
emWin samples	
GUI_ALPHA_TransparentDialogDemo.c	Application sample demonstrating alpha blending.
GUI_HelloWorld.c	Application sample that displays 'Hello World' on the LCD.
GUI_HouseControlDemo.c	Application sample showing a house control unit.
GUI_IP_ALPHA_TransparentDialogDemo.c	Application sample demonstrating alpha blending. A VNC server is running in the background.
GUI_IP_emExplorer.c	Application sample demonstrating the usage of emWin as file explorer to show the contents of an SD-card. A VNC server is running in the background.
GUI_IP_HouseControlDemo.c	Application sample showing a house control unit. A VNC server is running in the background.

File(s)	Description
GUI_IP_ReversiDemo.c	Application sample that allows to play the game 'Reversi'. A VNC server is running in the background.
GUI_WIDGET_GraphXYDemo.c	Application sample demonstrating the usage of the Graph widget. A VNC server is running in the background.
GUI_WIDGET_GraphYtDemo.c	Application sample demonstrating the usage of the Graph widget. A VNC server is running in the background.
GUI_MEMDEV_ImageFlow.c	Application sample demonstrating the usage of memory devices by showing image flows.
GUI_MEMDEV_ZoomAndRotate.c	Application sample demonstrating the usage of memory devices by zooming and rotating pictures.
GUI_OS_Status.c	Application sample showing the current OS and task status on the display.
GUI_RadialMenu	Application sample showing the creation of a radial menu with motion support.
GUI_ReversiDemo.c	Application sample that allows to play the game 'Reversi'.
GUI_USB_MSD_FS_Start.c	Application sample demonstrating the usage of the USB MSD component and the GUI to display the current connection status.
GUI_USB_MSD_FS_StartNew.c	Application sample demonstrating the usage of the USB MSD component and the GUI to display the current connection status with images.
GUI_WIDGET_GraphXYDemo.c	Application sample demonstrating the usage of the Graph widget.
GUI_WIDGET_GraphYtDemo.c	Application sample demonstrating the usage of the Graph widget.
embOS/IP samples	
OS_IP_DNSClient.c	Application sample demonstrating the usage of the Domain Name System by resolving a domain name to IP an address.
OS_IP_FTPClient.c	Application sample demonstrating FTP client usage.
OS_IP_FTPServer.c	Application sample demonstrating FTP server usage. Utilizes the filesystem as storage space.
OS_IP_NonBlockingConnect.c	Application sample demonstrating the connection to a server using non-blocking sockets.
OS_IP_Ping.c	Application sample that outputs information about incoming and outgoing ICMP communication.
OS_IP_SendMail.c	Application sample demonstrating the sending of an email using the Simple Mail Transfer Protocol.
OS_IP_Shell.c	Application sample demonstrating a simple Telnet-Client, allowing output of information about the target system.

File(s)	Description
OS_IP_SimpleServer.c	Application sample demonstrating a simple Telnet-Client that returns the current OS time on keypress.
OS_IP_SpeedClient_TCP.c	Application sample demonstrating TCP/IP speed measurements.
OS_IP_SpeedClient_TCP_ZeroCopy.c	Application sample demonstrating TCP/IP speed measurements using the zero copy API.
OS_IP_Start.c	Application sample demonstrating the start-up of the TCP/IP stack.
OS_IP_Start_ACD.c	Application sample demonstrating the usage of Automatic Collision Detection functionality.
OS_IP_Start_AutoIP.c	Application sample demonstrating AutoIP functionality.
OS_IP_Start_NetBIOS.c	Application sample demonstrating the usage of NetBIOS functionality.
OS_IP_Start_SNTPClient.c	Application sample demonstrating the usage of the IP stack to retrieve a timestamp from a NTP server.
OS_IP_Start_VLAN.c	Application sample demonstrating the usage of embOS & embOS/IP without any server or client program.
OS_IP_TFTPClient.c	Application sample demonstrating Trivial FTP client usage.
OS_IP_TFTPServer.c	Application sample demonstrating Trivial FTP server usage.
OS_IP_UDPDiscover.c	Application sample demonstrating the detection of target devices by using UDP broadcasts.
OS_IP_UDPDiscoverZeroCopy.c	Application sample demonstrating the detection of target devices by using UDP broadcasts through the zero copy API.
OS_IP_Webserver.c	Webserver example application.
OS_IP_Webserver_Lcd.c	Webserver example using LCD.
OS_IP_Webserver_UPnP.c	Webserver example using UPnP.
emModbus samples	
OS_IP_MB_MasterTCP	Application sample demonstrating how to implement a Modbus master using the Modbus/TCP protocol.
OS_IP_MB_SlaveTCP	Application sample demonstrating how to implement a Modbus slave using the Modbus/TCP protocol
embOS samples	
OS_EventObject.c	Application sample demonstrating the usage of EVENT objects.
OS_ExtendedTask.c	Application sample demonstrating the extension of tasks.
OS_ExtendedTaskContext.c	Application sample demonstrating the dynamic extension of task contexts.
OS_MeasureCPU_Performance.c	Application sample that checks the performance of the entire system.
OS_MeasureCST_HRTimer_embOSView.c	Application sample measuring the context switch time, using embOSView for outputs on a Windows PC.

File(s)	Description
OS_MeasureCST_HRTimer_Printf.c	Application sample measuring the context switch time, using printf() for outputs to the debug terminal/console.
OS_MeasureCST_HRTimer_Scope.c	Application sample measuring the context switch time, designed to visualize results by usage of an oscilloscope.
OS_Queue.c	Sample program demonstrating the usage of queues.
OS_Start2Tasks.c	Application sample demonstrating the creation of tasks.
OS_Start2TasksEx.c	Application sample demonstrating the creation of extended tasks.
OS_StartLEDBlink.c	Application sample demonstrating the usage of two tasks that toggle LEDs.
emUSB Device samples	
USB_BULK_Echo1.c	Application sample demonstrating a simple 1-byte echo server.
USB_BULK_Echo1_Timed.c	Application sample demonstrating a 1-byte echo server with simple timing.
USB_BULK_EchoFast.c	Application sample demonstrating a fast echo server.
USB_BULK_Performance.c	Application sample demonstrating performance measurements.
USB_BULK_ShowDeviceState.c	Application sample showing the status of the current USB state.
USB_BULK_Test.c	Application sample to test the stack.
USB_CDC_Echo.c	Application sample demonstrating the usage of virtual COM port.
USB_CDC_Start.c	Application sample demonstrating the usage of virtual COM port.
USB_CompositeDevice_CDC_Bulk.c	Application sample demonstrating the usage of a composite device of CDC and BULK.
USB_CompositeDevice_CDC_CDC.c	Application sample demonstrating the usage of a composite device of CDC and CDC.
USB_CompositeDevice_CDC_HID.c	Application sample demonstrating the usage of a composite device of CDC and HID.
USB_CompositeDevice_CDC_MSD.c	Application sample demonstrating the usage of a composite device of CDC and MSD.
USB_HID_Echo1.c	Application sample demonstrating the usage of HID for data transfers without the need for drivers.
USB_HID_Keyboard.c	Application sample demonstrating a keyboard sample input.
USB_HID_Keyboard_Mouse.c	Application sample demonstrating a keyboard sample input and an USB mouse moving.
USB_HID_Mouse.c	Application sample demonstrating an USB mouse moving.
USB_HID_Mouse_Echo1.c	Application sample demonstrating an USB mouse moving while an echo server is running in the background.

File(s)	Description
USB_HID_MouseByJoystick.c	Application sample controlling an USB mouse by using joystick controls.
USB_MSD_CDROM_FS_Start.c	Application sample demonstrating the usage of a CDROM image on the file system as storage.
USB_MSD_CDROM_Start.c	Application sample demonstrating the usage of the file system driver as MSC storage driver.
USB_MSD_FS_DisconnectOnWrite.c	Application sample demonstrating the <code>DisconnectOnWrite</code> hook.
USB_MSD_FS_MT_Start.c	Application sample demonstrating multi-tasking and the usage of the filesystem driver as MSC storage driver.
USB_MSD_FS_Start.c	Application sample demonstrating the usage of the filesystem driver as MSC storage driver.
USB_MSD_FS_WriteOnDisconnect.c	Application sample demonstrating the <code>WriteOnDisconnect</code> hook.
USB_MSD_Start_StorageByName.c	Application sample demonstrating device access by name.
USB_MSD_StartStorageRAM.c	Application sample demonstrating MSD using a RAMdisk.
USB_MTP_Start.c	Application sample demonstrating the usage of the Media Transfer Protocol.
USB_OTG_Start.c	Application sample demonstrating USB On-The-Go functionality.
USB_Printer.c	Application sample demonstrating the usage of the USB printer device class.
emUSB Host samples	
OS_USBH_CDC.c	Application sample demonstrating the usage of CDC.
OS_USBH_CreateInterfaceList.c	Application sample demonstrating the usage of the <code>USBH_CreateInterfaceList</code> function.
OS_USBH_FT232.c	Application sample demonstrating the usage of FT232 USB to serial UART interface.
OS_USBH_HID.c	Application sample demonstrating the enumeration of Human Interface Devices with input data handling.
OS_USBH_MSD.c	Application sample demonstrating the enumeration of Mass Storage Devices with file system operations.
OS_USBH_MSD_EnumErrorCheck.c	Application sample showing connection information when an USB stick gets plugged in.
OS_USBH_Printer.c	Application sample demonstrating the usage of the USB printer device class.

13.2 SeggerDemo

This application sample demonstrates the straightforward combination of all SEGGER software products in one single application. It can feature one or more of the the following components:

- embOS as RTOS coordinating all components
- embOS/IP providing TCP/IP support
- emFile providing a filesystem for server applications
- emUSB-Device providing MSD access to the server filesystem
- emUSB-Host providing Human Interface Device support
- emWin showing pictures and emWin widget samples

The application consists of a Webserver sample, an FTP server sample, and a USB MSD sample running in the background, while various GUI samples are shown on the LCD.¹

The folder "Start\Application\SEGGERDEMO" includes the SEGGERDEMO's source code, which may be modified and recompiled. However, in case the SEGGER eval software was designed for use with IAR's Embedded Workbench, the folder will also contain prebuilt libraries for the SEGGERDEMO. If present, these libraries will be used in the project by default.

For further information, please refer to *Evaluating the SeggerDemo* on page 21.

1. The included samples can vary depending on the hardware components of the eval board.

13.3 emFile samples

These samples use SEGGER emFile to provide a filesystem on the target.

13.3.1 FS_CheckDisk.c

This sample shows a simple checkdisk program.

This sample has no function when using a RAMDisk as filesystem.

13.3.2 FS_DirOperations.c

This sample creates 3 directories. In each directory 32 files are created. After creating the directories and files, the content of each directory is shown.

Console output

```
High level formatting:
.....Ok
.....Ok
.....Ok
Contents of
DIR00 (Dir) Attributes: ---- Size: 0
Contents of \DIR00
. (Dir) Attributes: ---- Size: 0
.. (Dir) Attributes: ---- Size: 0
FILE0000.TXT      Attributes: A--- Size: 19
FILE0001.TXT      Attributes: A--- Size: 19
FILE0002.TXT      Attributes: A--- Size: 19
FILE0003.TXT      Attributes: A--- Size: 19
FILE0004.TXT      Attributes: A--- Size: 19
FILE0005.TXT      Attributes: A--- Size: 19
FILE0006.TXT      Attributes: A--- Size: 19
FILE0007.TXT      Attributes: A--- Size: 19
FILE0008.TXT      Attributes: A--- Size: 19
FILE0009.TXT      Attributes: A--- Size: 19
FILE0010.TXT      Attributes: A--- Size: 19
FILE0011.TXT      Attributes: A--- Size: 19
FILE0012.TXT      Attributes: A--- Size: 19
FILE0013.TXT      Attributes: A--- Size: 19
FILE0014.TXT      Attributes: A--- Size: 19
FILE0015.TXT      Attributes: A--- Size: 19
FILE0016.TXT      Attributes: A--- Size: 19
FILE0017.TXT      Attributes: A--- Size: 19
FILE0018.TXT      Attributes: A--- Size: 19
FILE0019.TXT      Attributes: A--- Size: 19
FILE0020.TXT      Attributes: A--- Size: 19
FILE0021.TXT      Attributes: A--- Size: 19
FILE0022.TXT      Attributes: A--- Size: 19
FILE0023.TXT      Attributes: A--- Size: 19
FILE0024.TXT      Attributes: A--- Size: 19
FILE0025.TXT      Attributes: A--- Size: 19
FILE0026.TXT      Attributes: A--- Size: 19
FILE0027.TXT      Attributes: A--- Size: 19
FILE0028.TXT      Attributes: A--- Size: 19
FILE0029.TXT      Attributes: A--- Size: 19
FILE0030.TXT      Attributes: A--- Size: 19
FILE0031.TXT      Attributes: A--- Size: 19

DIR01 (Dir) Attributes: ---- Size: 0
Contents of \DIR01
. (Dir) Attributes: ---- Size: 0
.. (Dir) Attributes: ---- Size: 0
FILE0000.TXT      Attributes: A--- Size: 19
FILE0001.TXT      Attributes: A--- Size: 19
FILE0002.TXT      Attributes: A--- Size: 19
```

```

FILE0003.TXT      Attributes: A--- Size: 19
FILE0004.TXT      Attributes: A--- Size: 19
FILE0005.TXT      Attributes: A--- Size: 19
FILE0006.TXT      Attributes: A--- Size: 19
FILE0007.TXT      Attributes: A--- Size: 19
FILE0008.TXT      Attributes: A--- Size: 19
FILE0009.TXT      Attributes: A--- Size: 19
FILE0010.TXT      Attributes: A--- Size: 19
FILE0011.TXT      Attributes: A--- Size: 19
FILE0012.TXT      Attributes: A--- Size: 19
FILE0013.TXT      Attributes: A--- Size: 19
FILE0014.TXT      Attributes: A--- Size: 19
FILE0015.TXT      Attributes: A--- Size: 19
FILE0016.TXT      Attributes: A--- Size: 19
FILE0017.TXT      Attributes: A--- Size: 19
FILE0018.TXT      Attributes: A--- Size: 19
FILE0019.TXT      Attributes: A--- Size: 19
FILE0020.TXT      Attributes: A--- Size: 19
FILE0021.TXT      Attributes: A--- Size: 19
FILE0022.TXT      Attributes: A--- Size: 19
FILE0023.TXT      Attributes: A--- Size: 19
FILE0024.TXT      Attributes: A--- Size: 19
FILE0025.TXT      Attributes: A--- Size: 19
FILE0026.TXT      Attributes: A--- Size: 19
FILE0027.TXT      Attributes: A--- Size: 19
FILE0028.TXT      Attributes: A--- Size: 19
FILE0029.TXT      Attributes: A--- Size: 19
FILE0030.TXT      Attributes: A--- Size: 19
FILE0031.TXT      Attributes: A--- Size: 19

```

```

DIR02 (Dir) Attributes: ---- Size: 0

```

```

Contents of \DIR02

```

```

. (Dir) Attributes: ---- Size: 0
.. (Dir) Attributes: ---- Size: 0
FILE0000.TXT      Attributes: A--- Size: 19
FILE0001.TXT      Attributes: A--- Size: 19
FILE0002.TXT      Attributes: A--- Size: 19
FILE0003.TXT      Attributes: A--- Size: 19
FILE0004.TXT      Attributes: A--- Size: 19
FILE0005.TXT      Attributes: A--- Size: 19
FILE0006.TXT      Attributes: A--- Size: 19
FILE0007.TXT      Attributes: A--- Size: 19
FILE0008.TXT      Attributes: A--- Size: 19
FILE0009.TXT      Attributes: A--- Size: 19
FILE0010.TXT      Attributes: A--- Size: 19
FILE0011.TXT      Attributes: A--- Size: 19
FILE0012.TXT      Attributes: A--- Size: 19
FILE0013.TXT      Attributes: A--- Size: 19
FILE0014.TXT      Attributes: A--- Size: 19
FILE0015.TXT      Attributes: A--- Size: 19
FILE0016.TXT      Attributes: A--- Size: 19
FILE0017.TXT      Attributes: A--- Size: 19
FILE0018.TXT      Attributes: A--- Size: 19
FILE0019.TXT      Attributes: A--- Size: 19
FILE0020.TXT      Attributes: A--- Size: 19
FILE0021.TXT      Attributes: A--- Size: 19
FILE0022.TXT      Attributes: A--- Size: 19
FILE0023.TXT      Attributes: A--- Size: 19
FILE0024.TXT      Attributes: A--- Size: 19
FILE0025.TXT      Attributes: A--- Size: 19
FILE0026.TXT      Attributes: A--- Size: 19
FILE0027.TXT      Attributes: A--- Size: 19
FILE0028.TXT      Attributes: A--- Size: 19
FILE0029.TXT      Attributes: A--- Size: 19
FILE0030.TXT      Attributes: A--- Size: 19
FILE0031.TXT      Attributes: A--- Size: 19

```

13.3.3 FS_Performance.c

Sample program for performance measurement.

Console output

```
High level formatting
W0 Writing chunks of 524288 Bytes (Clusters/file size preallocated):
.....OK
Second time writing chunks of 524288 Bytes (Dynamic allocation of clusters):
.....OK
R0 Reading chunks of 524288 Bytes (80% fill)
.....OK
Test: 0 (Min/Max/Av): 4/5/4; (First write (Clusters/file size preallocated))
Speed:128000.00 kByte/s
Test: 1 (Min/Max/Av): 6/6/6; (W1 Second write (Dynamic allocation of clusters))
Speed: 85333.34 kByte/s
Test: 2 (Min/Max/Av): 8/8/8; (Read) Speed: 64000.00 kByte/s
Test 0 Speed: 128000.00 kByte/s
Test 1 Speed: 85333.34 kByte/s
Test 2 Speed: 64000.00 kByte/s
Finished...
```

13.3.4 FS_Start.c

Start application for the file system.

Console output

```
High level formatting
Running sample on
  Free space: 4172800 bytes
  Write test data to file \File.txt
  Free space: 4171776 bytes
  Finished
```

13.4 emWin samples

These samples use SEGGER emWin to show graphical features on a LCD display.

13.4.1 GUI_HelloWorld.c

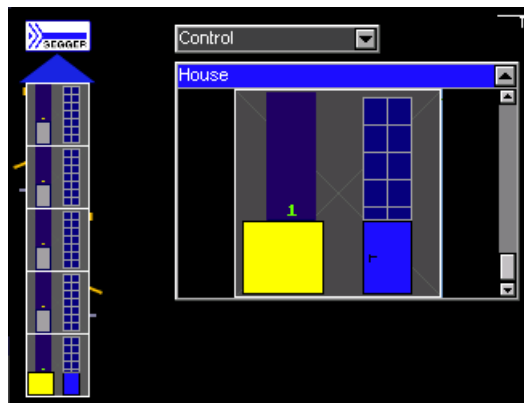
This sample shows a simple text output on the LCD. Your display should show something similar to the screenshot below.



13.4.2 GUI_HouseControlDemo.c

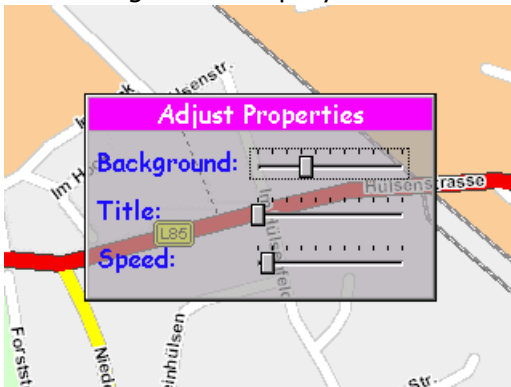
This sample shows how a possible house control unit may look like.

You can control several stations of the house. The actual state of the stations can be monitored in real time. Your display should show something similar to the screenshot below.



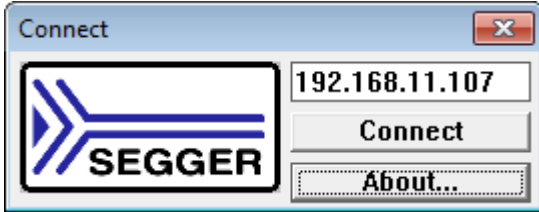
13.4.3 GUI_IP_ALPHA_TransparentDialogDemo.c

This sample shows a navigation-like sample using alpha blending to show a transparent dialog. Your display should show something similar to the screenshot below.



A VNC server is running in the background.

You can connect to the VNC server with the VNC client located at **Start\Windows\GUI\emVNC.exe**. Start emVNC and enter the IP address of your eval board.



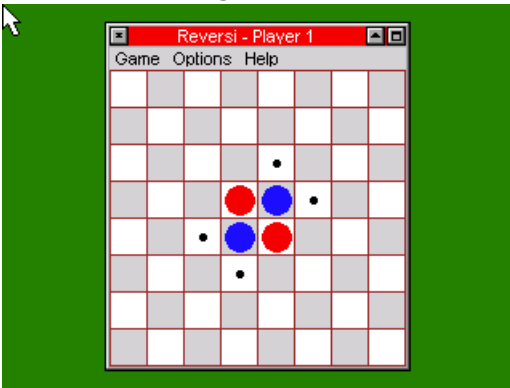
13.4.4 GUI_IP_emExplorer.c

This sample uses emWin widgets to show the content of a connected SD-card in an explorer-like window.

A VNC server is running in the background. For further information please refer to *GUI_IP_ALPHA_TransparentDialogDemo.c* on page 63.

13.4.5 GUI_IP_ReversiDemo.c

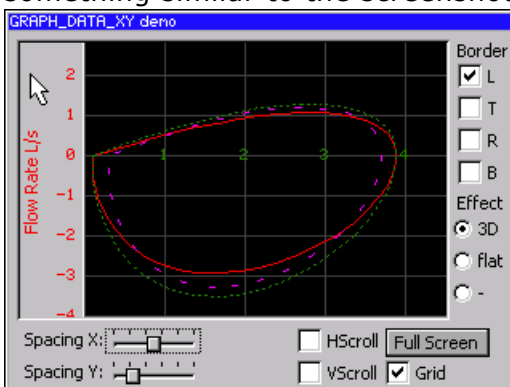
This sample shows a reversi game created with emWin widgets. Your display should show something similar to the screenshot below.



A VNC server is running in the background. For further information please refer to *GUI_IP_ALPHA_TransparentDialogDemo.c* on page 63.

13.4.6 GUI_IP_WIDGET_GraphXYDemo.c

This sample shows a graph created with emWin widgets. Your display should show something similar to the screenshot below.



A VNC server is running in the background. For further information please refer to *GUI_IP_ALPHA_TransparentDialogDemo.c* on page 63.

13.4.7 GUI_IP_WIDGET_GraphYtDemo.c

This sample shows an oscilloscope-like graph sample calculated from random values displayed on a timeline. Your display should show something similar to the screenshot below.



A VNC server is running in the background. For further information please refer to *GUI_IP_ALPHA_TransparentDialogDemo.c* on page 63.

13.4.8 GUI_OS_Status.c

This sample shows current embOS values.

The following details are displayed:

- Number of tasks
- OS time
- System stack
- Interrupt stack

Furthermore the following details are displayed for each task:

- Task ID
- Task priority
- Task name
- Number of activations (context switches)
- Used task stack
- Size of task stack

13.4.9 GUI_USB_MSD_FS_Start.c

MSD sample using the file system driver as MSC storage driver. The GUI part is showing the enumeration status on the display.

13.5 embOS/IP samples

These samples use SEGGER embOS/IP to demonstrate TCP/IP applications.

13.5.1 OS_IP_DNSClient.c

Demonstrates the usage of the Domain Name System to resolve a hostname to an IP address.

13.5.2 OS_IP_FTPServer.c

Demonstrates the usage of the embOS/IP FTP server add-on.

You may test the FTP server functionality by connecting to port 21.

By default, the sample is configured for two user accounts:

User: Anonymous	Pass: *
User: Admin	Pass: Secret

The storage space for the FTP server is provided by the emFile file system.

13.5.3 OS_IP_Shell.c

Demonstrates the usage of the IP-shell to diagnose the IP stack.

It opens TCP-port 23 (telnet) and waits for a connection. The actual Shell server is part of the stack, which keeps the application program nice and small.

To connect to the target, use the command line:

```
> telnet <target-ip>
```

where <target-ip> represents the IP address of the target.

13.5.4 OS_IP_SimpleServer.c

Demonstrates the usage of a simple Telnet server.

It opens TCP-port 23 (telnet) and waits for a connection. Once the connection is established, the server outputs the current system time for each character received.

To connect to the target, use the command line:

```
> telnet <target-ip>
```

where <target-ip> represents the IP address of the target.

13.5.5 OS_IP_SpeedClient_TCP.c

Provides a client sample for measurement of the network speed. The corresponding server sample must be executed on a Windows host.

The sample requires the configuration of the current IP address of your Windows host. To do so, simply modify the following line of the application's source code:

```
#define SERVER_IP_ADDR IP_BYTES2ADDR(192, 168, 88, 1)
```

The corresponding server sample, the "SpeedTestServer", is located at **Start\Windows\IP\SpeedTestServer**.

13.5.6 OS_IP_Start.c

Demonstrates the usage of the IP stack without any server or client program.

To ping the target, use the command line:

```
> ping <target-ip>
```

where <target-ip> represents the IP address of the target.

13.5.7 OS_IP_Start_NetBIOS.c

The sample will setup a responder to simple NetBIOS requests and will respond upon receiving a request with a configured NetBIOS name. The default name to respond to is "EVALBOARD".

To ping the target, use the command line on your PC:

```
> ping EVALBOARD
```

The PC now sends a discover request to find the target named "EVALBOARD". The target responds and the PC can now ping the target with the discovered IP.

13.5.8 OS_IP_UDPDiscover.c

Demonstrates the usage of the IP stack to discover a target device with unknown IP address through usage of UDP broadcasts.

The corresponding client sample, "UDPDiscoverGUI", must be executed on a Windows host and is located at **Start\Windows\IP\UDPDiscover**.

13.5.9 OS_IP_UDPDiscoverZeroCopy.c

Demonstrates the usage of the IP stack to discover a target device with unknown IP address through usage of UDP broadcasts. This sample also uses zero copy API to directly process the packet instead of retrieving it through the IP stack. This increases speed on time critical transfers.

The corresponding client sample, "UDPDiscoverGUI", must be executed on a Windows host and is located at **Start\Windows\IP\UDPDiscover**.

13.5.10 OS_IP_Webserver.c

Demonstrates the usage of embOS/IP to run a webserver. The webserver listens for incoming connections on port 80.

The sample uses a read-only file system using generated websites written in C-code. The storage space for the webserver is provided by the emFile filesystem.

13.6 emModbus samples

These samples use SEGGER emModbus to demonstrate the Modbus usage.

13.6.1 OS_IP_MB_MasterTCP.c

Demonstrates how to implement a Modbus master using the Modbus/TCP protocol. The sample connects to a Modbus/TCP slave via IP and toggles some LEDs on the slave.

The sample "Modbus/TCP Slave", located at **Start\Windows\MB\Modbus_Slave**, must be executed.

```
SEGGER Modbus/TCP slave V1.00b
Compiled on Aug  6 2015 14:29:18

Started server with slave addr. 1 and base addr. 1000 .

Slave status:

A master is connected to this slave.
```

Addr./		
Func.	1000	1001
Coils	[X]	[]
DI	[]	[]
Reg	0	0
IR	0	0

```
Press any key to close.
```

13.6.2 OS_IP_MB_SlaveTCP.c

Demonstrates how to implement a Modbus slave using the Modbus/TCP protocol. The sample will setup a Modbus/TCP slave via IP and provide access to some LEDs.

The sample "Modbus/TCP Master" must be executed, it is located at **Start\Windows\MB\Modbus_Master**. Use the IP address of the target for configuration.

```
SEGGER Modbus/TCP master V1.00b
Compiled on Aug  6 2015 14:28:46

Enter network address of Modbus/TCP slave [127.0.0.1]: 192.168.11.107
Enter slave address (dec.) [1]:
Enter base address of registers (dec.) [1000]:

Executing blinky on slave addr. 1 by toggling coils on addr. 1000 & 1001 .

Press any key to close.
```

13.7 embOS samples

These samples use SEGGER embOS to demonstrate RTOS features.

13.7.1 OS_EventObject.c

Application sample demonstrating the usage of EVENT objects.

13.7.2 OS_MeasureCPU_Performance.c

Application sample demonstrating performance measurements. It tests how many primes can be calculated within one second.

13.7.3 OS_MeasureCST_HRTimer_embOSView.c

Application sample measuring the context switch time, using embOSView for outputs on a Windows PC.

It is completely generic and runs on every target that is configured for embOSView.

13.7.4 OS_MeasureCST_HRTimer_Printf.c

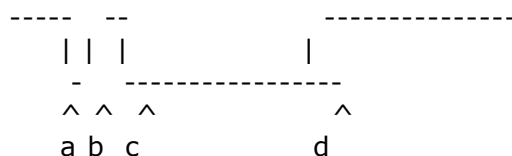
Application sample measuring the context switch time, using printf() for outputs to the debug terminal/console.

13.7.5 OS_MeasureCST_Scope.c

Application sample measuring the context switch time, designed to visualize results by usage of an oscilloscope. It sets and clears port pins, thereby allowing measurement of the context switch time with an oscilloscope.

The context switch time is

$$\text{Time} = (d - c) - (b - a)$$



The time between c and d is the context switch time, but note that the real context switch time is shorter, because the signal also contains the overhead of switching the LED on and off. The time of this overhead is also displayed on the oscilloscope as a small peak between a and b.

13.7.6 OS_Queue.c

Application sample demonstrating the usage of Queues.

13.7.7 OS_Start2Tasks.c

Application sample demonstrating the creation of tasks.

13.7.8 OS_Start2TasksEx.c

Application sample demonstrating the creation of extended tasks.

13.7.9 OS_StartLEDBlink.c

Application sample demonstrating the usage of two tasks that toggle LEDs.

13.8 emUSB samples

These samples use SEGGER emUSB to demonstrate USB communication.

13.8.1 USB_BULK_Echo1.c

This sample program for emUSB demonstrates a simple 1-byte USB BULK echo server.

The windows client that is required to evaluate this sample is located under **Start\Windows\USB\Bulk\SampleApp\Exe\Echo1.exe**.

13.8.2 USB_BULK_EchoFast.c

This sample program for emUSB demonstrates a fast echo server to test the stack.

The windows client that is required to evaluate this sample is located under **Start\Windows\USB\Bulk\SampleApp\Exe\EchoFast.exe**.

13.8.3 USB_BULK_ShowDeviceState.c

This sample program for emUSB sample shows the status of the current USB state in the debugger terminal I/O.

13.8.4 USB_BULK_Test.c

This sample program for emUSB demonstrates a modified echo server to test the stack.

The windows client that is required to evaluate this sample is located under **Start\Windows\USB\Bulk\SampleApp\Exe\Test.exe**.

13.8.5 USB_CDC_Start.c

This sample program for emUSB demonstrates a simple USB2COM echo server.

You can connect to the echo server by connecting the target to your PC via USB and using a simple terminal program like Hyper Terminal to connect to the target via the virtual COM port.

13.8.6 USB_HID_Echo1.c

This sample program for emUSB demonstrates a simple 1-byte USB BULK echo server using HID drivers to eliminate the need for extra drivers.

The windows client that is required to evaluate this sample is located under **Start\Windows\USB\HID\SampleApp\Exe\HIDEcho1.exe**.

13.8.7 USB_HID_Keyboard.c

This sample for emUSB demonstrates the usage of the HID component of the USB stack. A predefined string like from a regular keyboard will be typed.

Note: It is recommend to open a notepad application before connecting the USB cable.

13.8.8 USB_HID_Keyboard_Mouse.c

This sample for emUSB demonstrates the usage of the HID component of the USB stack as a mouse and keyboard. It makes the mouse jump left & right as well as typing a predefined string like from a regular keyboard.

Note: It is recommend to open a notepad application before connecting the USB cable.

13.8.9 USB_HID_Mouse.c

This sample program for emUSB demonstrates the usage of the HID component of the USB stack. It makes the mouse jump left & right.

13.8.10 USB_MSD_FS_Start.c

This sample program for emUSB demonstrates a sample startup for MSD, using the file system driver as MSC storage driver.

13.9 emUSB Host samples

13.9.1 OS_USBH_CDC.c

This sample program demonstrates the usage of USBH, a USB communications Device Class compatible device is required.

For testing purposes a second device can be programmed with emUSB-Device's "USB_CDC_Echo.c" sample.

13.9.2 OS_USBH_HID.c

This sample is designed to present emUSBH's capability to enumerate Human Interface Devices and handle the input data accordingly.

This sample will try to enumerate a connected mouse or keyboard and output the keystrokes or mouse movements to the terminal.

13.9.3 OS_USBH_MSD.c

Demonstrates the capability of emUSB-Host to enumerate Mass Storage Devices and perform file system operations on them. This sample will try to mount a connected device, check if it is formatted and create a "TestFile.txt" in the root directory.

Note: A special configuration is needed for this sample: **Setup/FS_USBH_MSDConfig.c** must be included. This config file may only be used for this sample.

Chapter 14

Literature and references

This chapter lists documents which may be useful to gain a deeper understanding of technical details.

Reference	Title	Comments
[UM01001]	User & reference guide for embOS	This document gives information about using the generic parts of embOS. It is publicly available from SEGGER (https://www.segger.com/downloads/embos).
[UM010xx]	User & reference guide for [CPU] and [Compiler]	This document gives information about using the CPU and compiler specific parts of embOS. It is included in respective embOS trial packages available from SEGGER (https://www.segger.com/downloads/embos).
[UM02001]	User & reference guide for emFile	This document gives information about using SEGGER emFile. It is publicly available from SEGGER (https://www.segger.com/downloads/emfile).
[UM03001]	User & reference guide for emWin	This document gives information about using SEGGER emWin. It is publicly available from SEGGER (https://www.segger.com/downloads/emwin).
[UM07001]	User & reference guide for embOS/IP	This document gives information about using SEGGER embOS/IP. It is publicly available from SEGGER (https://www.segger.com/downloads/embosip).
[UM08001]	SEGGER J-Link / J-Trace User's Guide.	This document gives information about using SEGGER J-Link / J-Trace. It is publicly available from SEGGER (https://www.segger.com/jlink-software.html).
[UM09001]	User & reference guide for emUSB Device	This document gives information about using SEGGER USB Device. It is publicly available from SEGGER (https://www.segger.com/downloads/emusb_device).
[UM10001]	User & reference guide for emUSB Host	This document gives information about using SEGGER USB Host. It is publicly available from SEGGER (https://www.segger.com/downloads/emusb_host).
[UM14001]	User & reference guide for emModbus	This document gives information about using SEGGER Modbus. It is publicly available from SEGGER (https://www.segger.com/downloads/emmodbus).