Getting started with SEGGER Eval Software for RDKRX62N and HEW

Document AN00009

Date: December 10, 2010

A product of SEGGER Microcontroller GmbH & Co. KG

www.segger.com

Disclaimer

Copyright notice

Trademarks

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
Email: support@segger.com
Internet: http://www.segger.com

Manual versions

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

For further information on topics or routines not yet specified, please contact us.

Revision	Date	By	Explanation
1	101210	TS	Link to Renesas RDKRX62N website added.
0	101203	TS	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.
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 & GmbH 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 developmenttime 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 efficent 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 developes and produces programming tools for flash microcontrollers, 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. Starterkits, eval- and trial-versions are available.

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. The profiling PC tool embOSView is included.

emFile
File system
emFile is an embedded file system with FAT12, FAT16 and FAT32 support. emFile has been optimized for minimum memory consumption in RAM and ROM while maintaining high speed. Various Device drivers, e.g. for NAND and NOR flashes, SD/MMC and CompactFlash cards, are available.

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

SEGGER TOOLS

Flasher
Flash programmer
Flash Programming tool primarily for microcontrollers.

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.

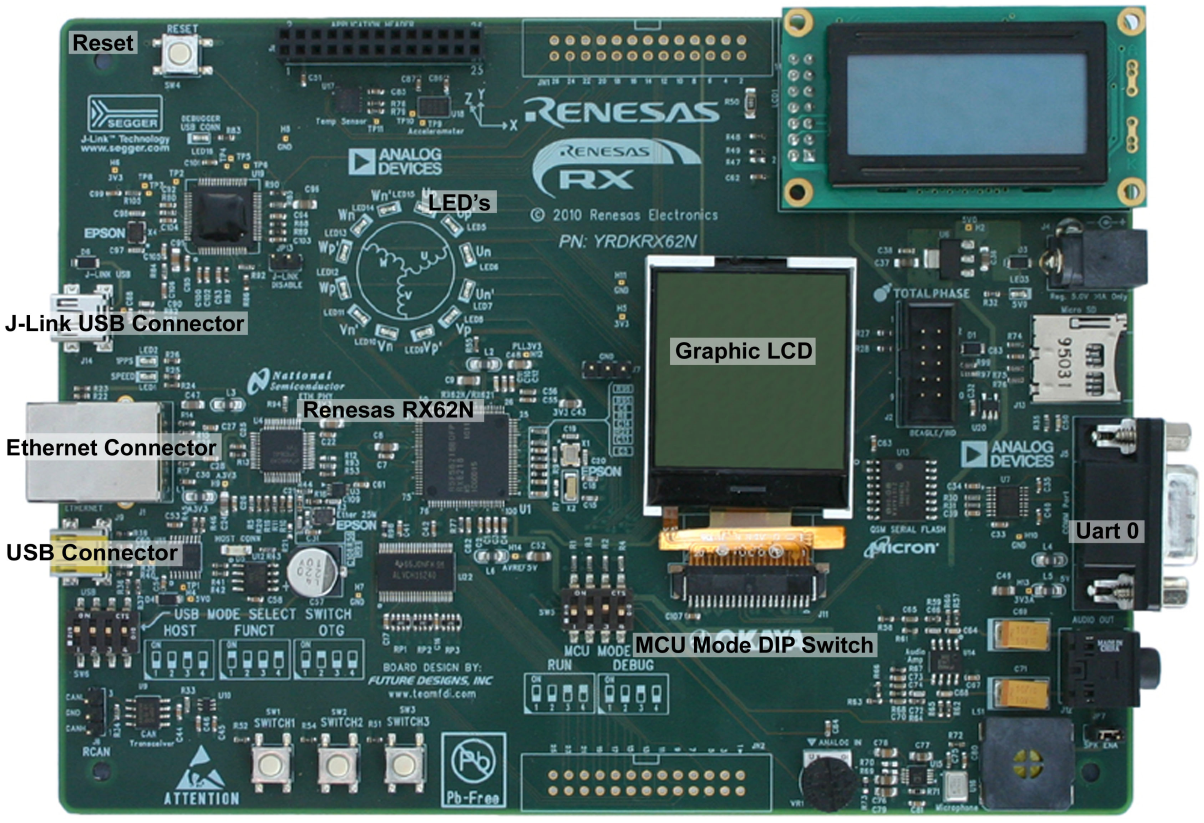
Table of Contents

Chapter 1

Renesas RDKRX62N evaluation board

This chapter contains all required information to start working with the Rensas RDKRX62N evaluation board and Segger Software.

1.1 Introduction



The evaluation board provides onboard J-Link emulator, ethernet interface, usb connector, three user push buttons, 12 user LEDs and a graphical monochrom LCD.

The processor on the board has an internal 512 KByte flash and 96 KByte of SRAM. Peripherals for several communications busses are available, including UART, SPI, I2C, USB, Ethernet, among others.

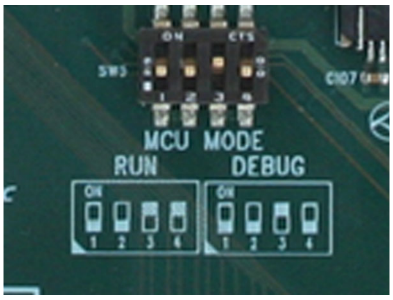
1.2 Evaluation board setup

1. Power

For basic operation the RDKRX62N only needs USB power. The Aux Power Jack J4 (+5v Regulated input) is only necessary in high current operation.

2. MCU Mode

Please ensure that MCU mode DIP switch SW5 is set to "DEBUG" (OFF, OFF, ON, OFF).



3. J-Link software download

Please download and install the latest J-Link RX software and device driver. It is downloadable at *http://www.segger.com/cms/j-link-rx.html.*

4. J-Link connection

Connect one end of the included USB cable to your compute and the other end to J14 of the RDK (USB port to the left of the LED Ring). This is the debug interface and where we power the board from. The Found New Hardware Wizard may appear. Please install the J-Link driver.

5. Ethernet connection

Connect the eval board to your network with the included network cable. Please ensure that a DHCP server is running in your network. If no DHCP server is running you can also setup a static ip address in the Segger evaluation software. Please refer to the following chapters how to setup a static ip address.

For more information please have a look in the Renesas RDKRX62N evaluation board user manual "reu10b0009_YRDKRX62N_user_manual.pdf". It is downloadable at www.renesas.com/rdkrx62n.

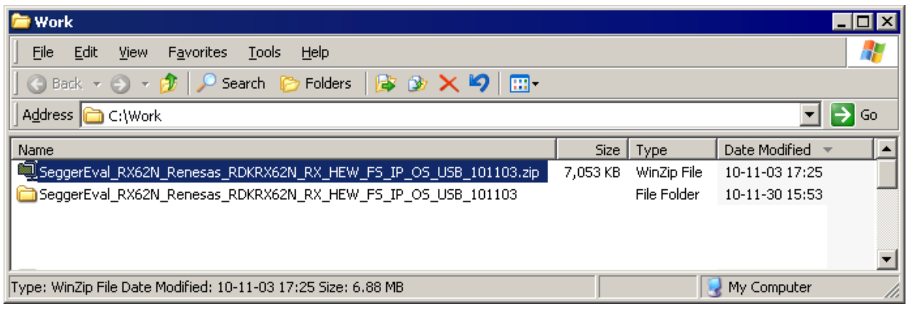
Chapter 2

Webserver sample application

The Segger evaluation software includes many sample applications for Segger middleware. The next steps describes how to get the default sample application running. The default sample application is a webserver running on embOS and embOS/IP.

1. Segger project for HEW

The Segger evaluation software for Renesas RDKRX62N is downloadable at segger.com/cms/renesas-rx62n-rdk.html. The file contains the Segger evaluation software, sample applications as also product documentation. Unzip the file to the folder of your choice, e.g. c:\work.



Name	Size	Type	Date Modified
SeggerEval_RX62N_Renesas_RDKRX62N_RX_HEW_FS_IP_OS_USB_101103.zip	7,053 KB	WinZip File	10-11-03 17:25
SeggerEval_RX62N_Renesas_RDKRX62N_RX_HEW_FS_IP_OS_USB_101103		File Folder	10-11-30 15:53

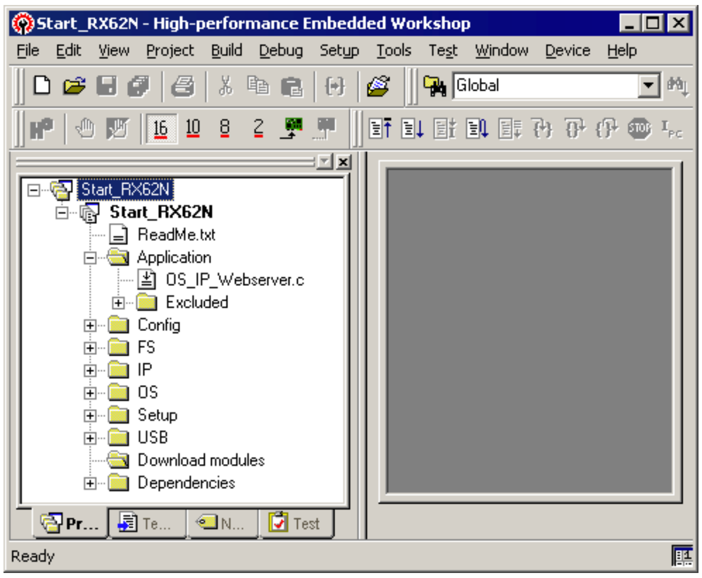
Type: WinZip File Date Modified: 10-11-03 17:25 Size: 6.88 MB   My Computer

2. HEW project

This documentation assumes that you have already installed HEW and Renesas RX compiler. If not please download and install HEW and Renesas RX compiler. You can download a complete Renesas installation CD from *http://www.renesas.com/rdkrx62ninstall*. The Renesas RX compiler is 128 KByte code size limited after 60 days of use. Most Segger sample applications in the Release build have less than 128 KByte code size.

The Segger evaluation software includes a workspace and project for HEW.
Start HEW and open the workspace *Start_RX62N.hws* at
\SeggerEval_RX62N_Renesas_RDKRX62N_RX_HEW_FS_IP_OS_USB_xxx\Start\

Once you have opened the workspace it should look like the following one:

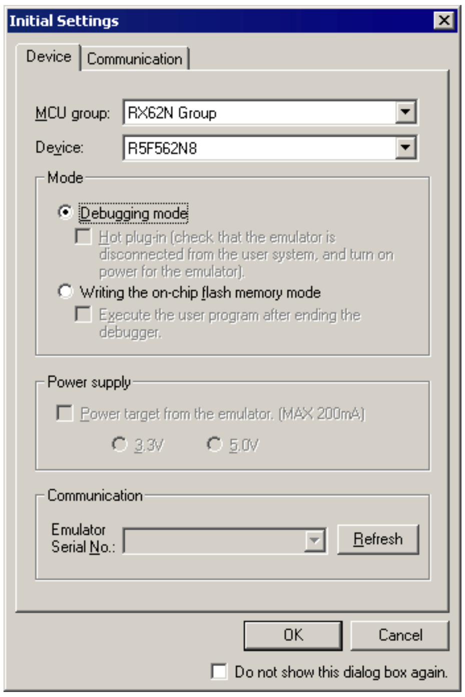


3. Project build

Build the project with [F7] or select from the top menu bar, "Build", then "Build All". (HEW Menu Bar > Build > Build All)

4. J-Link Settings

Connect HEW to the RDK through the JLink Debugger, select from the top menubar, "Debug", then "Connect". (HEW Menu Bar > Debug > Connect). Please check the following settings in the "Initial Settings" dialog box and click <Ok>:
- MCU Group is set to "RX62N Group"
- Device is set to "R5F562N8"
- Verify Mode is set to "Debugging mode"



Initial Settings

Device | Communication

MCU group: RX62N Group

Device: R5F562N8

Mode
- [✓ Debugging mode]
- Hot plug-in (check that the emulator is disconnected from the user system, and turn on power for the emulator).
- Writing the on-chip flash memory mode
- Execute the user program after ending the debugger.

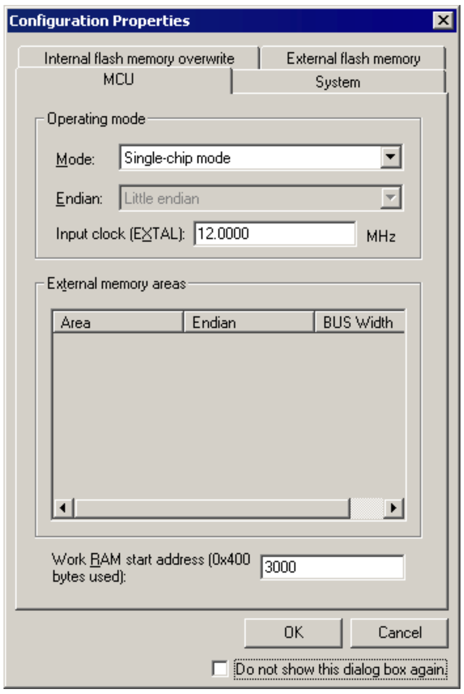
Power supply
- Power target from the emulator. (MAX 200mA)
- 3.3V 5.0V

Communication

Emulator Serial No.: Refresh

OK Cancel

Do not show this dialog box again.



Configuration Properties

Internal flash memory overwrite | External flash memory | MCU | System

Operating mode

Mode: Single-chip mode

Endian: Little endian

Input clock (EXTAL): 12.0000 MHz

External memory areas

Area	Endian	BUS Width

Work RAM start address (0x400 bytes used): 3000

OK Cancel

Do not show this dialog box again.

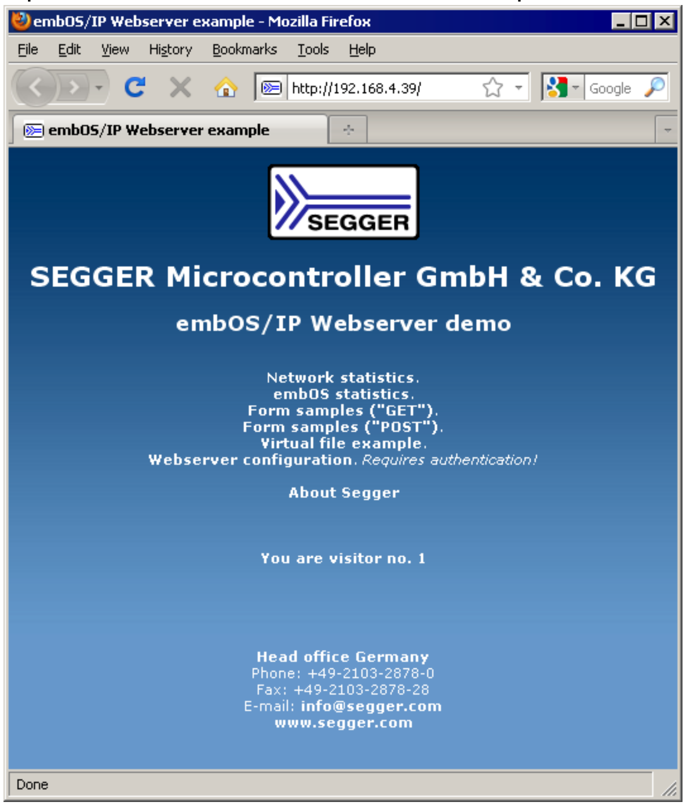
The first time you connect to the RDK, the J-Link debugger may need to updateits firmware. If the "Confirm Firmware" window on the right appears, click <Yes>. You will see some messages in a Connecting dialog box, then the "Configuration Properties" dialog box will be displayed. Click <Ok> to agree with the settings.

5. Start sample application

From the top menu bar, select "Debug", then "Download Modules", finally "All Download Modules". The "Downloading" window will show the progress of the download, once completed it will disappear. Run the project with [Shift F5] or choose from the top menu bar "Debug", then "Reset Go", (HEW Menu Bar > Debug > Reset Go).

As soon as the webserver sample application receives an ip address from your DHCP server it is displayed on the Lcd.

Open a web browser and enter the ip address.



SEGGER Microcontroller GmbH & Co. KG

embOS/IP Webserver demo

Network statistics.
embOS statistics.
Form samples ("GET").
Form samples ("POST").
Virtual file example.
Webserver configuration. *Requires authentication!*

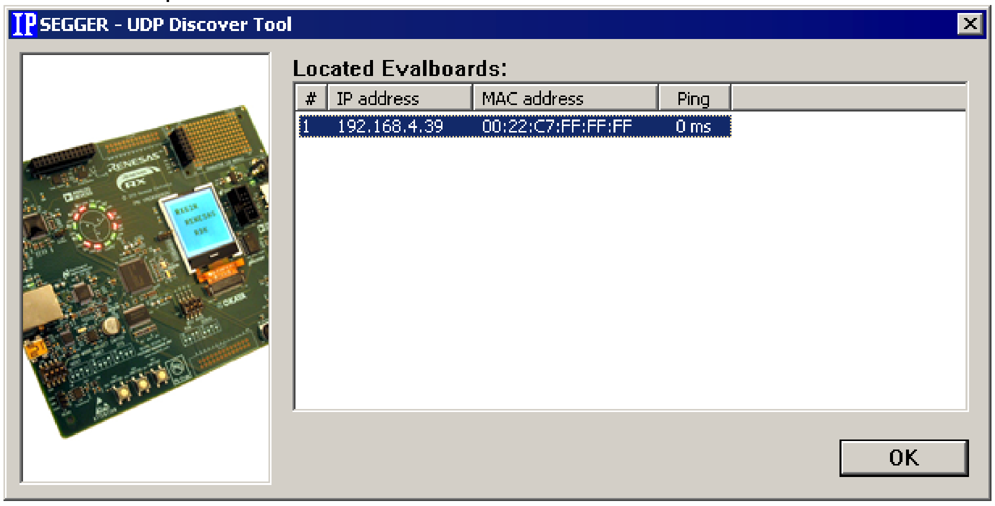
About Segger

You are visitor no. 1

Head office Germany
Phone: +49-2103-2878-0
Fax: +49-2103-2878-28
E-mail: info@segger.com
www.segger.com

You can now browse through the web sites and evaluate the different webserver features like dynamic web page generating.

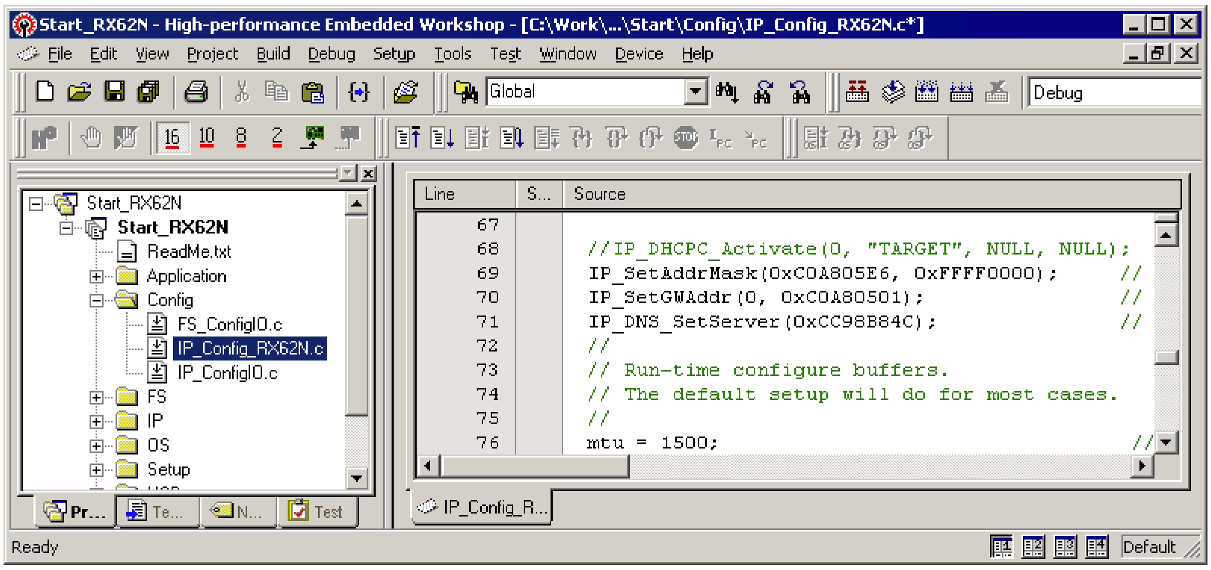
You can also start the PC tool "UDPDiscover_RDKRX62N.exe" from the directory "Start\Win-dows\TCPIP\UDPDiscover\". It searches the RDKRX62N eval board and shows the ip address:



Located Evalboards:

#	IP address	MAC address	Ping
1	192.168.4.39	00:22:C7:FF:FF:FF	0 ms

If you are not using a DHCP server in your network you will need to enter an IP address and subnetmask into the file "Start\Config\IP_Config_RX62N.c" manually. The screenshot below shows an example for the IP address *192.168.5.230* with sub-netmask *255.255.0.0*.



```
  //IP_DHCPC_Activate(0, "TARGET", NULL, NULL);
  IP_SetAddrMask(0xC0A805E6, 0xFFFF0000);     //
  IP_SetGWAddr(0, 0xC0A80501);                //
  IP_DNS_SetServer(0xCC98B84C);               //
  //
  // Run-time configure buffers.
  // The default setup will do for most cases.
  //
  mtu = 1500;                                 //
```

Chapter 3

Software components

Not every Segger eval package contains all Segger products, e.g. emWin will only be included if your target eval board has a display.

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 software that has been optimized for speed, versatility and memory footprint.

emFile documentation can be found under "`Doc\UM02001_emFile.pdf`".

emWin

emWin is SEGGER's embedded Graphical User Interface (GUI) using a feature rich API and providing an efficient, processer- and LCD controller-independent GUI for any application that operates with a display.

emWin documentation can be found under "`Doc\UM03001_emWinUser.pdf`".

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.

embOS documentation can be found under "`Doc\UM01003_embOS_Generic.pdf`".

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.

embOS/IP documentation can be found under "`Doc\UM07001_embOSIP.pdf`".

emUSB

emUSB is SEGGER's embedded USB device stack. It is written in ANSI C and features bulk communication as well as device classes such as MSD, CDC or HID.

emUSB documentation can be found under "`Doc\UM09001_emUSB.pdf`".

emUSB/Host

emUSB is SEGGER’s embedded USB Host stack. It is written in ANSI C. Segger’s USB host software stack implements full USB host functionality, including external hub support, and optionally provides device class drivers. It enables developers to easily add USB host functionality to embedded systems.

emUSB documentation can be found under "`Doc\UM10001_emUSBH.pdf`".

Setup

Requirements

In order to recompile the projects you will need the HEW Workbench as indicated in the ReadMe.txt file.

In order to recompile the emWin simulation you will need a C-compiler. The shipping contains an ready to go project for Microsoft Visual C++ 6.0. Microsoft Visual C++ 6.0 or Microsoft Visual Studio .Net are required to use this project.

Installation

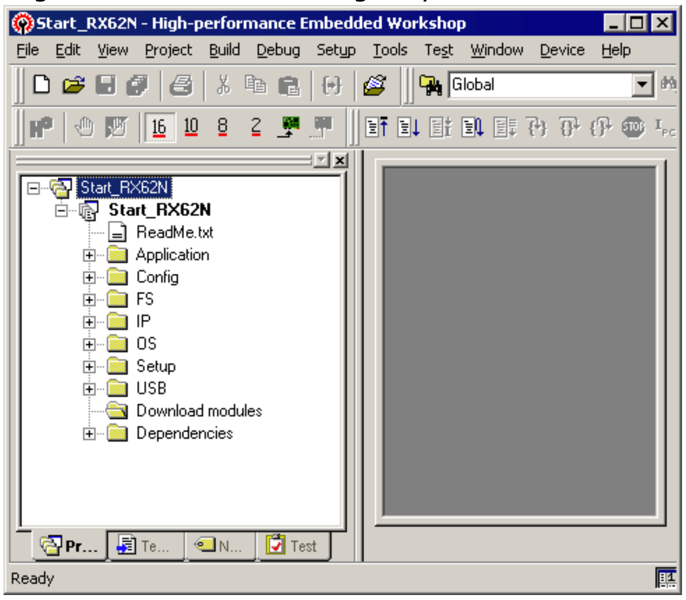
All eval packages are supplied as a zip-file. The latest version of the software can be archieved from the following location:
`http://www.segger.com/evalboards.html`

Extract it to any folder of your choice, preserving the directory structure of the zip-file. Assuming that you are using the HEW Workbench project manager to develop your application, no further installation steps are required. You will find prepared sample start applications, which you should use and modify to write your application.

All products can be combined and purchased separately. For ordering information, please contact SEGGER, *www.segger.com*.

3.1 Project structure

All components of the eval package are provided as libraries and the required header files. The sample applications are provide in source form. The sample projects are organized in the following way:



The root directory of the eval package includes three folders, "`Doc`", "`Start`" and this documentation.

The "`Doc`"-folder contains the user guides for the included SEGGER Eval Software and the "`License.txt`". The "`Start`"-folder contains all files necessary for the SEGGER Eval Software itself.

The "`Start`"-folder may contain an additional directory (`Windows`), which is not included in the project tree of the HEW Workbench project.

Directory	Description
Application	Includes the sample applications.
Setup	Includes the Board Support Package (BSP). The BSP consists all files which are required to initialize the hardware and build a project.
Config	Includes the configuration files of the included software packages.
FS	Includes the emFile header files and the emFile libraries.
Inc	Includes utility header files which are not directly related to one of the included software packages.
IP	Includes the embOS/IP header files and the embOS/IP libraries.
OS	Includes the embOS header files and the embOS libraries and `main.c`.
USB	Includes the header files and the emUSB libraries.

Table 3.1: Eval package structure

3.2 Evaluation limitations

The included evaluation 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 run without a time limit with a maximum of three tasks. If your application creates more than three tasks stops embOS after a time limit of 15 minutes.
emOS/IP	The eval version of the embOS/IP libraries have a time limit of 15 minutes on the connection.
emUSB	The eval version of the emUSB libraries have a time limit of 15 minutes on the connection.
emWin	The eval version of the emWin library shows an evaluation notification before the actual application starts.

Table 3.2: Limitations of eval package components

Your use of the eval package or of any part included in the project indicates your acknowledgment and agreement to the SEGGER eval software license `License.txt` is located in the `Doc` directory of the eval package.

Chapter 4

Sample applications

embOS comes with many sample applications. The sample applications are included as source code and can be used to evaluate the Segger middleware products.

4.1 List of sample applications

The list below contains the most important sample applications that are shipped. All sample applications can be found in the "`Start\Application`"-folder.

File(s)	Description
embOS/IP samples	
OS_IP_DNSClient.c	DNS client sample application
OS_IP_FTPServer.c	FTP server example using the filesystem as storage space.
OS_IP_NonBlocking_Connect.c	Sample program using none blocking connection.
OS_IP_Ping.c	ICMP client application.
OS_IP_SendMail.c	SMTP client application.
OS_IP_Shell.c	Shell server example listening on port 23.
OS_IP_SimpleServer.c	Server example listening on port 23. Returns the actual OS time on keypress.
OS_IP_SpeedClient_TCP.c	Speed client for TCP/IP stack using socket interface.
OS_IP_Start.c	Sample using the TCP/IP stack.
OS_IP_UDPDiscover.c	Demonstrates setup of a simple UDP application.
OS_IP_UDPDiscoverZeroCopy.c	Demonstrates setup of a simple UDP application using zero copy.
OS_IP_Webserver.c	Webserver example using a readonly filesystem.
OS_IP_Webserver_Lcd.c	Webserver example using the RDKRX62N LCD
embOS samples	
OS_Main_EVENT.c	Sample program for embOS using EVENT object.
OS_Main_OS_Q.c	Sample program for embOS using queues.
OS_Main_TaskEx.c	Sample program for embOS using extended task contexts.
OS_MeasureCST_HRTimer_embOSView.c	Performance test program for embOS using embOS-View for outputs.
OS_MeasureCST_HRTimer_Printf.c	Performance test program for embOS
OS_MeasureCST_Scope.c	Performance test program for embOS designed for oscilloscope measurement.
OS_PerformanceTest.c	embOS sample used for measuring the RTOS.
OS_Start_2Tasks.c	Sample program for embOS using 2 tasks.
OS_Start_LEDBlink.c	Sample program for embOS using 2 tasks to toggle LEDs.
emUSB samples	
USB_BULK_Echo1.c	USB BULK sample showing a simply 1-byte echo server.
USB_BULK_Echo1_Timed.c	USB BULK sample showing a simply 1-byte echo server
USB_BULK_EchoFast.c	USB BULK sample fast echo server to test the stack.
USB_BULK_ShowDeviceState.c	USB BULK sample showing the status of the current USB state.
USB_BULK_Test.c	USB BULK sample to test the stack.
USB_CDC_Echo.c	Sample showing a simple USB2COM echo server.
USB_CDC_Start.c	emUSB sample showing usage of virtual COM port.
USB_HID_Mouse.c	emUSB sample showing a USB mouse moving.
USB_HID_MouseByJoystick.c	Demonstrates usage of the HID component
USB_HID_CDROM_Start.c	Sample startup, using the file system driver as MSC storage driver.

Table 4.1: Sample applications

File(s)	Description
USB_MSD_FS_DisconnectOnWrite.c	Sample showing how to use the FS and USB mass storage device simulateously.
USB_MSD_FS_Start.c	MSD sample using the filesystem driver as MSC storage driver.
USB_MSD_FS_WriteOnDisconnect.c	Sample showing how to use the FS and USB mass storage device simulateously.
USB_MSD_Start_StorageByName.c	Sample startup, using the file system driver as MSC storage driver.
USB_MSD_Start_StorageRAM.c	Sample startup, using a simple RAM disk driver.
USB_Printer.c	Sample implementation of USB printer device class
emFile samples	
FS_CheckDir.c	Sample program demonstrating FS_CheckDisk functionality
FS_DirOperations.c	Sample program for creation files and directorys
FS_Performance.c	Sample program which is designed to take performance measurements.
FS_Start.c	Start application for file system.
FS_Start_ReadWriteHook.c	Sample program with hook function.

Table 4.1: Sample applications

4.2 embOS/IP samples

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

OS_IP_DNSClient.c

This sample demonstrates use of the integrated DNS client.

OS_IP_FTPServer.c

This sample application runs an FTP server. The FTP server listens on port 21 for an incoming connection.

You can easily login to the FTP by using the "`anonymous`" account.

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

OS_IP_NonBlockingConnect.c

Demonstrates how to connect to a server using non-blocking sockets.

OS_IP_Ping.c

This sample is a ICMP server and client. Please modify the ip addressin define HOST_TO_PING.

OS_IP_SendMail.c

SMTP client, please set the SMTP and email settings first.

OS_IP_Shell.c

This sample demonstrates using 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 keep 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.

OS_IP_SimpleServer.c

This sample demonstrates setup of a simple server which simply sends back the target system tick for every character received. It opens TCP-port 23 (telnet) and waits for a connection.

To connect to the target, use the command line:

> telnet <target-ip>

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

OS_IP_SpeedClient_TCP.c

This sample application is a client sample for measuring the network speed. For this sample you have to start the SpeedTestServer located under Start\Windows\TCPIP\SpeedTestServer.

You have to modify the IP address which can be found in the file Start\Config\IP_Config_RX62N.c.

Example

Change the lines

```
//IP_SetAddrMask(0xC0A80505, 0xFFFF0000);     // Assign IP addr. and subnet mask
  IP_DHCPC_Activate(0,"Target", NULL, NULL);
```

to

```
IP_SetAddrMask(0xC0A80505, 0xFFFF0000);     // Assign IP addr. and subnet mask
//IP_DHCPC_Activate(0,"Target", NULL, NULL);
```

for IP address 192.168.5.5 / 16.

OS_IP_Start.c

This sample demonstrates use 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.

OS_IP_UDPDiscover.c

This sample demonstrates use of the IP stack to discover a device without knowledge of the IP address by using UDP broadcast.

The windows client needed to evaluate this sample is located under "Start\Windows\TCPIP\UDPDiscover\UDPDiscover.exe".

OS_IP_UDPDiscoverZeroCopy.c

This sample demonstrates use of the IP stack to discover a device without knowledge of the IP address by using UDP broadcast. This sample 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 windows client needed to evaluate this sample is located under "Start\Windows\TCPIP\UDPDiscover\UDPDiscover.exe".

OS_IP_Webserver.c

This sample application runs a webserver. The webserver listens on port 80 for incoming connections.

The storage space for the webserver is provided by the emFile filesystem. The webserver sample uses a readonly filesystem using generated websites in C-code.

OS_IP_Webserver_Lcd.c

Same application as OS_IP_Webserver but the IP address is displayed on the LCD.

4.3 embOS samples

These samples use SEGGER embOS to demonstrate RTOS features.

OS_Main_EVENT.c

This sample program for embOS demonstrates usage of EVENT objects.

OS_Main_OS_Q.c

This sample program for embOS demonstrates usage of QUEUE objects.

OS_Main_TaskEx.c

This sample program for embOS demonstrates usage extended task contexts.

OS_MeasureCST_HRTimer_embOSView.c

This benchmark measures the OS context switch time and displays the result in the terminal window of embOSView.

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

OS_MeasureCST_HRTimer_Printf.c

This benchmark measures the context switch time and displays the result on CSpy's terminal I/O window. It runs with every workbench that supports terminal I/O with printf.

OS_MeasureCST_Scope.c

This benchmark uses the LED.c module to set and clear a port pin. This allows measuring the context switch time with an oscilloscope.

The context switch time is

Time = (d - c) - (b - a)

```
-----   --                 ---------------
    | |  |                |
     -   -----------------
    ^ ^  ^                ^
    a b  c              d
```

The time between c and d ist 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.

OS_PerformanceTest.c

This sample tests how many primes can be calculated in one second to determine the speed of the embOS RTOS.

OS_Start_2Tasks.c

These samples show some basic functionality of embOS by using two tasks.

OS_Start_LEDBlink.c

These samples show some basic functionality of embOS by toggling LEDs in two tasks.

4.4 emUSB samples

These samples use SEGGER emUSB to demonstrate USB communication. **Please include the file USB_Config_RX62N.c in the project.**

USB_BULK_Echo1.c

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

The windows client needed to evaluate this sample is located under "`Start\Windows\USB\Bulk\SampleApp\Echo1.exe`".

USB_BULK_EchoFast.c

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

The windows client needed to evaluate this sample is located under "`Start\Windows\USB\Bulk\SampleApp\EchoFast.exe`".

USB_BULK_ShowDeviceState.c

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

USB_BULK_Test.c

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

The windows client needed to evaluate this sample is located under "`Start\Windows\USB\Bulk\SampleApp\Test.exe`".

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 with your PC via USB and using a simple terminal program like Hyper Terminal to connect to the target via the virtual COM port. You can select every baudrate you like.

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 needed to evaluate this sample is located under "Start\Windows\USB\HID\SampleApp\Exe\HIDEcho1.exe".

USB_HID_Mouse.c

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

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.

4.5 emFile samples

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

FS_APIValidation.c

Sample program showing a simple generic API test.

Console output

```
High level formatting volume...Ok

Simple file test
Creating file...Ok
Write some data...Ok
Check file pos.......Ok
Read written data back...Ok
Write Burst test...Ok
Read Burst test...Ok
Creating a lot of file in root directory.......................................Ok
Test FS_FOpen modes
Mode 'w'...OK
Mode 'a'...OK
Mode 'r+'...OK
Mode 'w+'...OK
Mode 'a+'...OK

Directory API test
Checking FS_MkDir()...Ok
Checking FS_CreateDir()...Ok
Checking FS_FindFirstFile()/FS_FindNextFile...Ok
Checking FS_RmDir()....Ok

Extended API test
Preparing test...Ok
Checking FS_CopyFile()...Ok
Checking FS_Move()...Ok
Checking FS_Remove()...Ok
Checking FS_Rename()...Ok
Finished
```

FS_CheckDisk.c

This sample shows a simple checkdisk program.
This sample has no function when using a RAMDisk as filesystem.

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
```

FS_Performance.c

Sample program for measure the performance.

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...
```

FS_Start.c

Start application for 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
```

FS_Start_ReadWriteHook.c

Console output

```
High level formatting
Running sample on
Open/create file
  Read  : StartSector: 0x00000000, NumSectors: 0x00000001, SectorType: DATA
  Read  : StartSector: 0x00000000, NumSectors: 0x00000001, SectorType: DATA
  Read  : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
   Write : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR  1st Write (4
bytes)to file
  Read  : StartSector: 0x00000001, NumSectors: 0x00000001, SectorType: MAN
  Write : StartSector: 0x00000029, NumSectors: 0x00000001, SectorType: DATA
  Read  : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
  Write : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
   Write : StartSector: 0x00000001, NumSectors: 0x00000001, SectorType: MAN  2nd write (511
bytes) to file.
  Read  : StartSector: 0x00000029, NumSectors: 0x00000001, SectorType: DATA
  Write : StartSector: 0x00000029, NumSectors: 0x00000001, SectorType: DATA
  Write : StartSector: 0x0000002a, NumSectors: 0x00000001, SectorType: DATA
  Read  : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
  Write : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR  Close file
  Read  : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
  Write : StartSector: 0x00000019, NumSectors: 0x00000001, SectorType: DIR
Finished
```

FS_WritePerformanceTest.c

Sample program for measure the performance.

Console output

```
High level formatting
Writing chunks of 2048 x 1024 Bytes
(Min/Avg/Max/Total): 0, 0, 1, 55
Writing chunks of 2048 x 1024 Bytes
(Min/Avg/Max/Total): 0, 0, 1, 27
Finished...
```

Chapter 5

FAQ

What is the purpose of this eval package ?

This eval package has been designed to provide customers and potential customers (you) with a complete, easy to use software package for the specified target hardware and HEW Workbench for Renesas (target compiler).

It allows you to easily check out the target hardware, the target compiler and our software components. This evaluation process typically does not take a lot of time since the software can be easily recompiled and downloaded to the target.

What are the components of the software package?

The SEGGER software components are provided in library form, the applications are provided in source code form. Most packages also come with a "`Prebuild`"-folder, containing prebuild executables, which simply need to be downloaded to the target.

Can I recompile the application supplied?

Yes. All you need is the target compiler/eval version of the target compiler. Refer to "`ReadMe.txt`" in the "`Start`"-folder for more information.

Can I write my own applications with this eval package?

Yes, you can write your own applications. However the purpose of this eval package is to test the soft- and hardware for fitnes. You may not use the result in a product.

Chapter 6

Licensing

License terms

The 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 superseed this description.

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

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.

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 7

Literature and references

Reference	Title	Comments
[UM08001]	SEGGER J-Link / J-Trace User's Guide.	This document gives information about using the SEGGER J-Link / J-Trace ARM. It is publicly available from SEGGER (*www.segger.com*).
[UM01003]	User's & reference manual for embOS	This document gives information about using the generic parts of embOS. It is publicly available from SEGGER (*www.segger.com*).
[UM07001]	embOS/IP User Guide	This document gives information about using the SEGGER IP stack. It is publicly available from SEGGER (*www.segger.com*).
[UM09001]	User's and reference manual for emUSB	This document gives information about using the SEGGER USB stack. It is publicly available from SEGGER (*www.segger.com*).
[UM02001]	emFile User's Guide	This document gives information about using the SEGGER embedded filesystem. It is publicly available from SEGGER (*www.segger.com*).
[UM03001]	User's and reference manual for emWin	This document gives information about using the SEGGER GUI software. It is publicly available from SEGGER (*www.segger.com*).

Table 7.1: Literature and References