

Network Driver for Synopsys Devices User Guide

Version 1.40

For use with Network Driver for Synopsys[®] Devices
module versions 3.07 and above

Date: 30-Aug-2017 11:04

All rights reserved. This document and the associated software are the sole property of HCC Embedded. Reproduction or duplication by any means of any portion of this document without the prior written consent of HCC Embedded is expressly forbidden.

HCC Embedded reserves the right to make changes to this document and to the related software at any time and without notice. The information in this document has been carefully checked for its accuracy; however, HCC Embedded makes no warranty relating to the correctness of this document.

Table of Contents

System Overview	3
Introduction	3
Feature Check	4
Packages and Documents	5
Packages	5
Documents	5
Change History	6
Source File List	7
API Header File	7
Configuration File	7
System Files	7
Platform Support Package (PSP) Files	8
Version Files	8
Configuration Options	9
Application Programming Interface	10
synopsys_eth_drv_init	10
Error Codes	11
Integration	12
OS Abstraction Layer	12
Utilities	12
PSP Porting	13
psp_synopsys_eth_init	14
psp_synopsys_eth_start	15
psp_synopsys_eth_stop	16
psp_synopsys_eth_delete	17
psp_synopsys_get_buf	18

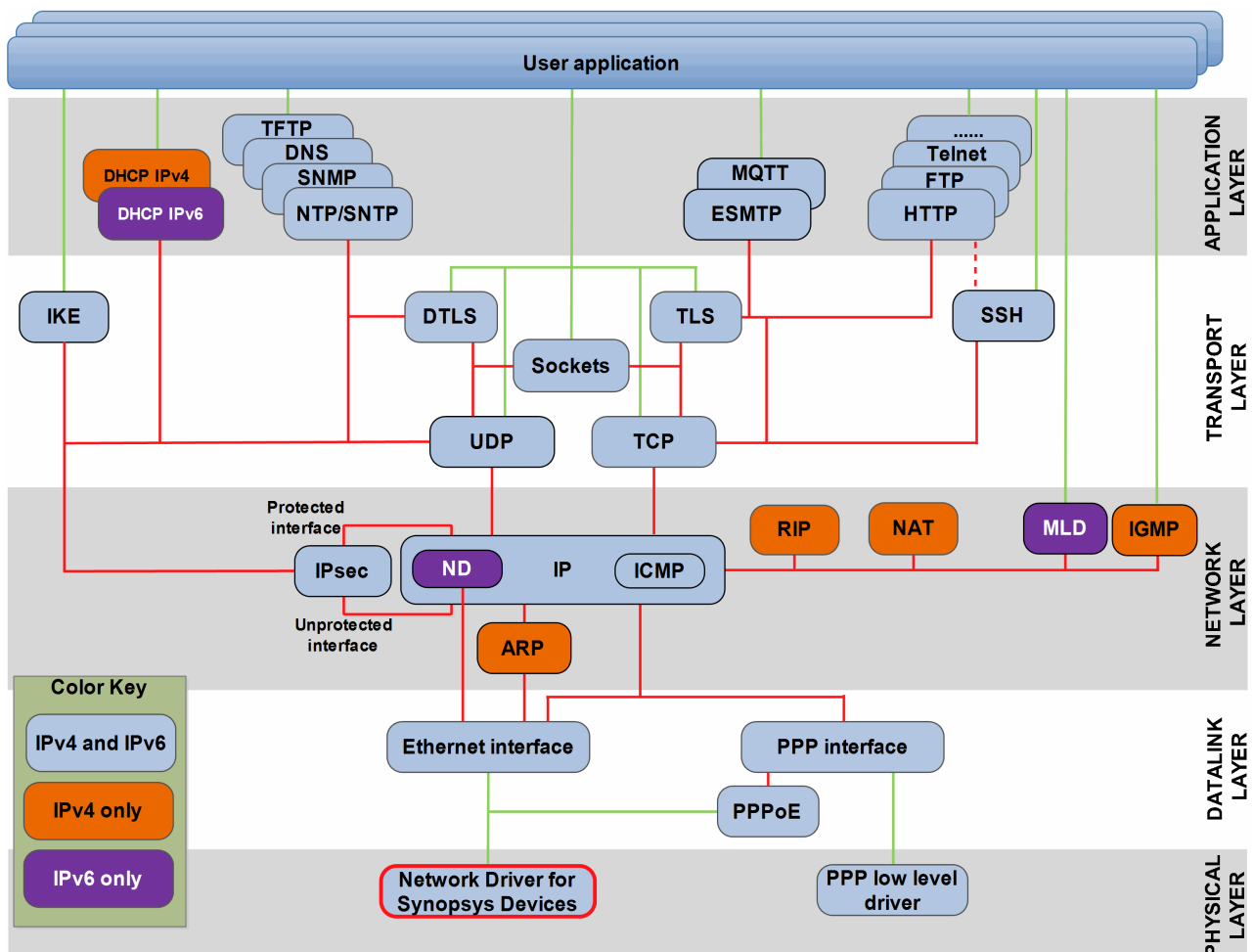
1 System Overview

1.1 Introduction

This guide is for those who want to implement a network driver for the Synopsys® Ethernet IP core (that is, for devices including STM32, LPC435x, and XMC4xxx).

This Synopsys® core is widely used in both Microcontrollers and System on Chip (SOC) solutions. This HCC driver can be used for all of these.

The driver's location within HCC's TCP/IP stack is shown below. (In this diagram green lines show interfaces available to users of the TCP/IP stack, red lines show internal TCP/IP interfaces.)



Note: Although every attempt has been made to simplify the system's use, you need a good understanding of the requirements of the systems you are designing in order to obtain the maximum practical benefits. HCC Embedded offers hardware and firmware development consultancy to help you implement your system.

1.2 Feature Check

The main features of the Synopsys® network driver are the following:

- Conforms to the HCC Advanced Embedded Framework.
- Designed for integration with both RTOS and non-RTOS based systems.
- Conforms to the [HCC Network Driver interface specification](#).

This driver can be used with the following LPC4xxx devices:

- LPC4076
- LPC4078
- LPC4088
- LPC4330
- LPC4333
- LPC4337
- LPC4350
- LPC4353
- LPC4357
- LPC4370

This driver can be used with the following STM32 devices:

- STM32F107
- STM32F207
- STM32F217
- STM32F4xx
- STM32F7xx

This driver can be used with the following XMC4xxx devices:

- XMC4400
- XMC4500
- XMC4700
- XMC4800

Other devices with the Synopsys® core can also be integrated.

1.3 Packages and Documents

Packages

The table below lists the packages which you need in order to use this module.

Package	Description
<code>hcc_base_doc</code>	This contains the two guides that will help you get started.
<code>nw_drv_base</code>	The network driver base package. This is the base system on which the Synopsys [®] driver is built.
<code>nw_drv_eth_synopsys</code>	The Network Driver for Synopsys [®] Devices package described in this document.
<code>psp_template_membar</code>	The Platform Support Package (PSP) Memory Barrier.

Documents

For an overview of HCC's TCP/IP stack software, see [Product Information](#) on the main HCC website.

Readers should note the points in the [HCC Documentation Guidelines](#) on the HCC documentation website.

HCC Firmware Quick Start Guide

This document describes how to install packages provided by HCC in the target development environment. Also follow the *Quick Start Guide* when HCC provides package updates.

HCC Source Tree Guide

This document describes the HCC source tree. It gives an overview of the system to make clear the logic behind its organization.

HCC Network Driver User Guide

This document describes the network driver base system.

HCC Network Driver for Synopsys Devices User Guide

This is this document.

1.4 Change History

This section describes past changes to this manual.

- To view or download earlier manuals, see [Archive: Network Driver for Synopsys User Guide](#).
- For the history of changes made to the package code itself, see [History: nw_drv_eth_synopsys](#).

The current version of this manual is 1.40. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.40	2017-08-30	3.07	Corrected <i>Packages</i> list.
1.30	2017-06-16	3.07	New <i>Change History</i> format.
1.20	2017-03-29	3.07	Changes to TCP Stack diagram.
1.10	2017-01-16	3.07	Changes to TCP Stack diagram.
1.00	2015-12-21	3.03	First online version.

2 Source File List

This section lists and describes all the source code files included in the system. These files follow HCC Embedded's standard source tree system, described in the [HCC Source Tree Guide](#). All references to file pathnames refer to locations within this standard source tree, not within the package you initially receive.

Note: Do not modify any files except the configuration file and PSP files.

2.1 API Header File

The file `src/api/api_ethdriver_synopsys.h` should be included by any application using the system. This is the only file that should be included by an application using this module. It defines the [synopsys_eth_drv_init](#) function.

2.2 Configuration File

The file `src/config/config_ethdriver_synopsys.h` contains all the configurable parameters of the system. Configure these as required. For details of these options, see [Configuration Options](#).

2.3 System Files

These files are in the folder `src/driver/network/ethernet/synopsys`. They should only be modified by HCC.

File	Description
<code>eth_phy_dp83848.c</code>	DP83848 source code.
<code>eth_phy_dp83848_reg.h</code>	DP83848 header file.
<code>eth_phy_lan8720.c</code>	LAN 8720 source code.
<code>eth_phy_lan8720_reg.h</code>	LAN 8720 header file.
<code>eth_synopsys.c</code>	Synopsys IP core source code.

2.4 Platform Support Package (PSP) Files

These files provide functions the core code needs to call, depending on the hardware. There is a folder named **psp_xxx** for each device type that HCC has implemented.

The following general PSP implementation files are in the directory **src/psp/target**. These provide templates for you to produce your own PSP.

Note: You must modify these PSP implementations for your specific microcontroller and development board; see [PSP Porting](#) for details.

File	Description
eth/psp_eth_synopsys.c	Functions source code.
eth/psp_eth_synopsys.h	Functions header file.
include/hcc_xxx_regs.h	Registers file. For example, for an LPC4350, this is hcc_lpc4350_regs.h .

2.5 Version Files

These files in the directory **src/version** contain the version number of each part of the module. This version number is checked by all modules that use a module to ensure system consistency over upgrades.

File	Description
ver_ethdriver_synopsys.h	Synopsys files version number.
ver_psp_eth_mii.h	MII file version.
ver_psp_eth_phy.h	Ethernet PSP version.
ver_psp_eth_phy_dp83848.h	DP83848 version.
ver_psp_eth_phy_lan8720.h	LAN 8720 version.

3 Configuration Options

Set the system configuration options in the file `src/config/config_ethdriver_synopsys.h`. This section lists the available options and their default values.

SYNOPSYS_ETHERNET_BUF_SIZE

The Ethernet buffer size. The default value is 16384.

ETH_ISR_ID

The ISR ID. The default value is 0.

ETH_ISR_PRIO

The ISR priority. The default value is 0.

ETH_PHY_DEV_ADDR

The external physical address. The default value is 0x00.

ETH_HANDLE_PHY

Keep the default value of 1 to configure and use the MDIO interface. Set this to 0 to disable the interface

ETH_LINK_STA_POLL_INTERVAL

The polling interval in ms. The default value is 500.

MAX_RX_DESC

The maximum number of RX descriptors. The default value is 4.

MAX_TX_DESC

The maximum number of TX descriptors. The default value is 4.

MAC_ADDRESS

The MAC address of the driver. The default value is { 0x00, 0xA2, 0x92, 0x00, 0x92, 0xAA }.

4 Application Programming Interface

This section describes the single API function and the error codes it may return.

4.1 synopsys_eth_drv_init

Use this function to initialize the driver.

Format

```
t_nwdriver_ret synopsys_eth_drv_init (
    uint32_t      param,
    t_nwdriver * * const p_ethdriver )
```

Arguments

Parameter	Description	Type
param	The driver parameter (not used).	uint32_t
p_ethdriver	Where to write the pointer to the Ethernet driver.	t_nwdriver * *

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

4.2 Error Codes

This table lists all the error codes that may be generated by the API calls:

Error code	Value	Meaning
NWDRIVER_SUCCESS	0	Execution successful.
NWDRIVER_ERROR	1	Operation failed.

5 Integration

This section describes all aspects of the network driver that require integration with your target project. This includes porting and configuration of external resources.

5.1 OS Abstraction Layer

The network driver uses the OS Abstraction Layer (OAL) that allows it to run seamlessly with a wide variety of RTOSes, or without an RTOS.

The network driver uses the following OAL components:

OAL Resource	Number Required
Tasks	0
Mutexes	1
Events	0
ISRs	1

5.2 Utilities

The code creates and uses a single timer in the **hcc_timer** module.

The **hcc_timer** module is included in your system when you install the base network driver module.

5.3 PSP Porting

The Platform Support Package (PSP) is designed to hold all platform-specific functionality, either because it relies on specific features of a target system, or because this provides the most efficient or flexible solution for the developer. For full details of its functions and macros, see the *HCC Base Platform Support Package User Guide*.

The driver makes use of the following standard PSP macros:

Macro	Package	Element	Description
PSP_RD_LE16	psp_base	psp_endianness	Reads a 16 bit value stored as little-endian from a memory location.
PSP_RD_LE32	psp_base	psp_endianness	Reads a 32 bit value stored as little-endian from a memory location.

The driver makes use of the following standard PSP function:

Function	Package	Element	Description
psp_membar()	psp_base	psp_membar	Memory barrier.
psp_memcpy()	psp_base	psp_string	Copies a block of memory. The result is a binary copy of the data.

The driver makes use of the following PSP functions which must be provided by the Platform Support Package. These are designed for you to port them easily to work with your hardware solution. The package includes samples in the file **include/psp_eth_synopsys.c**.

Function	Description
psp_synopsys_eth_init()	Initializes the hardware for ETH usage.
psp_synopsys_eth_start()	Starts the module.
psp_synopsys_eth_stop()	Stops the module.
psp_synopsys_eth_delete()	Deletes all resources created by the module.
psp_synopsys_get_buf()	Gets the contents of a buffer.

These functions are described in the following sections.

psp_synopsys_eth_init

This function is provided by the PSP to initialize the Synopsys Ethernet driver.

Format

```
t_nwdriver_ret psp_synopsys_eth_init ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_synopsys_eth_start

This function is provided by the PSP to start the Synopsys Ethernet driver.

Format

```
t_nwdriver_ret psp_synopsys_eth_start ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_synopsys_eth_stop

This function is provided by the PSP to stop the Synopsys Ethernet driver.

Format

```
t_nwdriver_ret psp_synopsys_eth_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_synopsys_eth_delete

This function is provided by the PSP to delete the Synopsys Ethernet driver, releasing the associated resources.

Format

```
t_nwdriver_ret psp_synopsys_eth_delete( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_synopsys_get_buf

This function is provided by the PSP to get the address of the Ethernet buffer.

Format

```
t_nwdriver_ret psp_synopsys_get_buf ( uint8_t * * const pp_buf )
```

Arguments

Parameter	Description	Type
pp_buf	Where to store the buffer address.	uint8_t **

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.