Network Driver for TI EMAC Devices User Guide

Version 1.30

For use with Network Driver for TI EMAC Devices module versions 3.05 and above

Exported on 08/22/2018

Table of Contents

1 System Overview 3
1.1 Introduction 4
1.2 Feature Check 5
1.3 Device Description 6
1.4 Packages and Documents 7
Packages 7
Documents 7
1.5 Change History 8
2 Source File List 9
2.1 API Header File 9
2.2 Configuration File 9
2.3 System Files 9
2.4 Version Files 9
2.5 Platform Support Package (PSP) Files 10
3 Configuration Options 11
4 Application Programming Interface 13
4.1 emac_ethdriver_init 13
4.2 Error Codes 14
5 Integration 15
5.1 OS Abstraction Layer 15
5.2 Utilities 15
5.3 PSP Porting 16
psp_eth_ti_emac_init 17
psp_eth_ti_emac_start 18
psp_eth_ti_emac_stop 19
psp_eth_ti_emac_delete 20

1 System Overview

This chapter contains the fundamental information for this module.

The component sections are as follows:

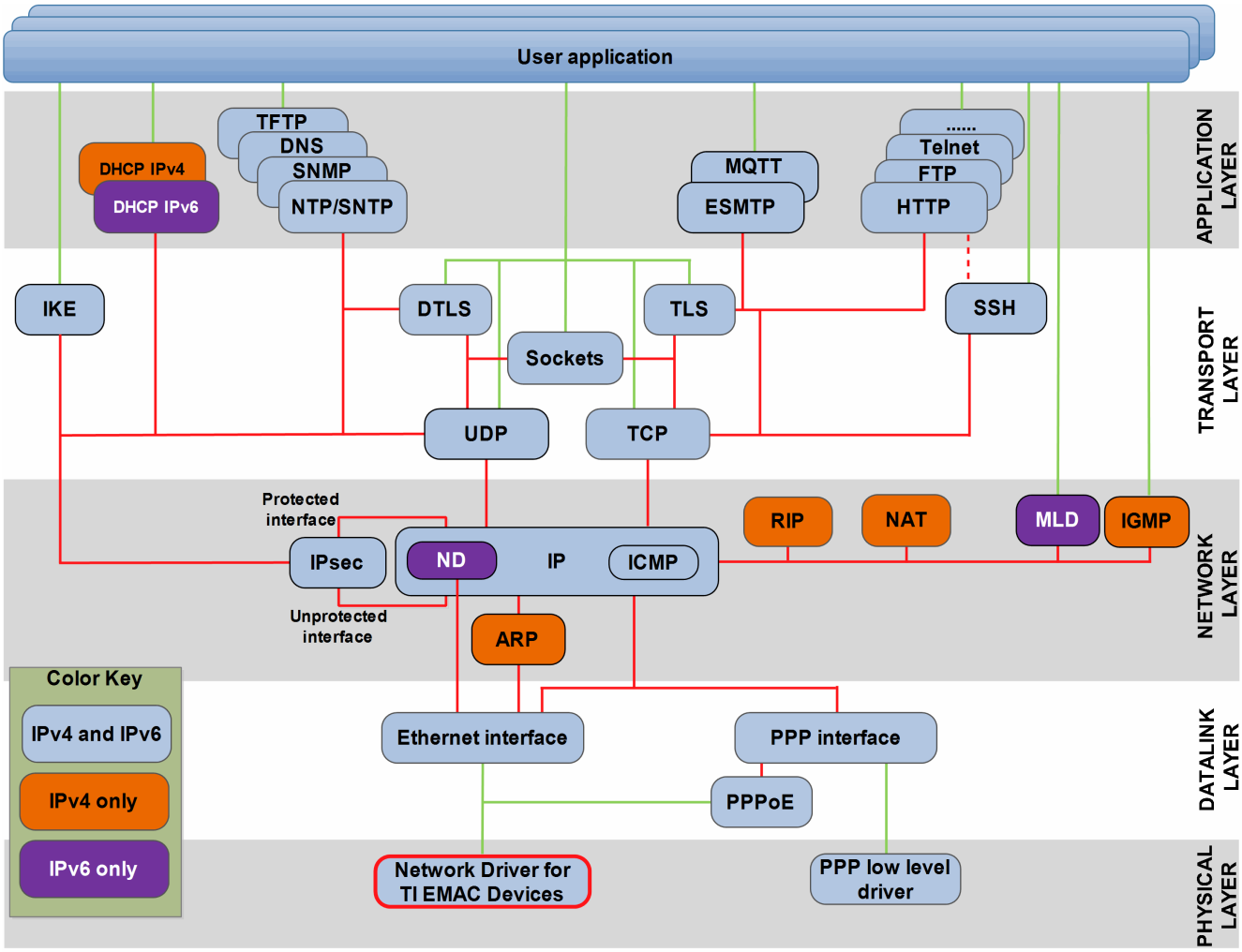
- Introduction – describes the main elements of the module. This section includes a diagram showing the position of the driver interface within HCC's TCP/IP stack.
- Feature Check – summarizes the main features of the module as bullet points.
- Device Description – summarizes the properties of the supported devices.
- Packages and Documents – the *Packages* section lists the packages that you need in order to use this module. The *Documents* section lists the relevant user guides.
- Change History – lists the earlier versions of this manual, giving the software version that each manual describes.

Note: To download this manual as a PDF, see Network Driver PDFs.

1.1 Introduction

This guide is for those who want to implement a network driver for Hercules RM46, RM48, and TMS570 microcontrollers from Texas Instruments Incorporated and other TI devices that use the similar Ethernet controller IP.

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 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 Coding Standard, including full MISRA compliance.
- Fully compatible with the HCC Network Driver Interface specification.
- Supports Hercules RM46, RM48 and TMS570 devices from Texas Instruments Incorporated.
- Supports the DP83xxx Ethernet PHY family from Texas Instruments Incorporated.
- HCC provides fully tested reference drivers for this module.

1.3 Device Description

The driver was tested with the RM48xl50 and TMS570LS31x.

This table summarizes the properties of the supported devices:

	Hercules RM46	Hercules RM48	TMS570
Flash	1 to 1.25 MB	2 to 3 MB	256 KB to 4 MB
RAM	128 or 192 KB	192 or 256 KB	32 to 512 KB
ECC	Yes	Yes	Yes

1.4 Packages and Documents

Packages

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

Package	Description
hcc_base_doc	This contains the two guides that will help you get started.
nw_drv_base	The network driver base package. This is the base system on which the TI EMAC driver is built.
nw_drv_eth_ti_emac	The Network Driver for TI EMAC Devices package.

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 TI EMAC Devices User Guide

This is this document.

1.5 Change History

This section describes past changes to this manual.

- To download this manual or a PDF describing an earlier software version, see Network Driver PDFs.
- For the history of changes made to the package code itself, see History: nw_drv_eth_ti_emac.

The current version of this manual is 1.30. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.30	2018-08-22	3.05	Changed *System Overview* and *Change History*.
1.20	2018-08-13	3.05	Corrected number of ISRs in *OS Abstraction Layer*.
1.10	2017-06-16	3.03	New *Change History* format.
1.00	2017-04-25	3.03	First online version.

2 Source File List

This section describes all the source code files included in the system. These files follow the HCC Embedded 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_ti_emac.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 emac_ethdriver_init function.

2.2 Configuration File

The file **src/config/config_ethdriver_ti_emac.h** contains all the configurable parameters of the system. Configure these as required. For details of the options, see Configuration Options.

2.3 System Files

The following files are in the folder **src/driver/network/ethernet/ti_emac**. **These files should only be modified by HCC**.

File	Description
eth_phy_dp83xx.c	Ethernet driver code for DP83xxx family.
eth_phy_dp83xx_reg.h	Registers header file for DP83xxx family.
eth_ti_emac.c	Ethernet driver source code.

2.4 Version Files

The following files contain the version number of the components of this module. The version number is checked by all modules that use a module to ensure system consistency over upgrades.

File	Description
eth_ethdriver_ti_emac.h	Network driver version.
eth_psp_eth_phy_dp83xx.h	DP83xxx driver component version.

2.5 Platform Support Package (PSP) Files

These files provide functions the core code needs to call, depending on the hardware. The following general PSP implementation files are in the directory **src/psp**. 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.
- In the package these files are offset to avoid overwriting an existing implementation. Copy them to the root **hcc** directory for use.

File	Description
include/psp_eth_mii.h	Media Independent Interface (MII) header file.
include/psp_eth_phy.h	Ethernet PSP header file.
target/eth/psp_eth_ti_emac.c	PSP functions source code.
target/eth/psp_eth_ti_emac.h	PSP functions header file.
target/include/hcc_ti_rm48_tms570_reg.h	Registers header file.

3 Configuration Options

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

HCC_EMAC_BUF_SIZE

The Ethernet buffer size. The default is (16 * 1024).

HCC_EMAC_BUF_ALIGN_PWR_2

The buffer alignment as a power of 2. The default is 2.

HCC_EMAC_MEM_HDL

The memory handle. The default is HCC_MEM_UNCACHED.

HCC_EMAC_MAX_TX_BUFD

The maximum number of EMAC TX descriptors to use per interface. The default is 8.

HCC_EMAC_MAX_RX_BUFD

The maximum number of EMAC RX descriptors to use per interface. The default is 8.

HCC_EMAC_MAC_ADDRESS

The MAC address of the driver. The default is { 0x00, 0xA0, 0x91, 0x00, 0x93, 0xCD }.

HCC_EMAC_LINK_STA_POLL_INTERVAL

The link timer interval in ms. The default is 500.

HCC_EMAC_EXT_PORT_NUM

The number of connected external ports. The default is 2.

HCC_EMAC_PHY_MODE

The mode, Media-Independent Interface (MII) or Reduced Media-Independent Interface (RMII). The default is HCC_EMAC_PHY_MII_MODE.

HCC_EMAC_TX_ISR_ID

The EMAC TX interrupt ID. The default is PSP_EMAC_TX_ISR_ID.

HCC_EMAC_RX_ISR_ID

The EMAC RX interrupt ID. The default is PSP_EMAC_RX_ISR_ID.

HCC_EMAC_MISC_ISR_ID

The EMAC LINK interrupt ID. The default is PSP_EMAC_LINK_ISR_ID.

HCC_EMAC_TX_IT_PRIO

The EMAC TX interrupt priority. The default is 0.

HCC_EMAC_RX_IT_PRIO

The EMAC RX interrupt priority. The default is 0.

HCC_EMAC_MISC_IT_PRIO

The EMAC Link interrupt priority. The default is 0.

4 Application Programming Interface

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

4.1 emac_ethdriver_init

Use this function to initialize the network driver.

Format

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

Arguments

Parameter	Description	Type
param	The driver parameter.	uint32_t
p_ethdriver	Where to write the pointer to the 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	0
Events	0
ISRs	3

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 module makes use of the following standard PSP functions:

Function	Package	Element	Description
psp_memcpy()	psp_base	psp_string	Copies a block of memory. The result is a binary copy of the data.
psp_memset()	psp_base	psp_string	Sets the specified area of memory to the defined value.

The module 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 module makes use of the following functions that must be provided by the PSP. These are designed for you to port them easily to work with your hardware solution. The package includes samples in the **target/eth/psp_eth_ti_emac.c** file.

Function	Description
psp_eth_ti_emac_init()	Initializes the hardware for the Ethernet driver.
psp_eth_ti_emac_start()	Starts the driver.
psp_eth_ti_emac_stop()	Stops the driver.
psp_eth_ti_emac_delete()	Deletes the driver, releasing associated resources.

These functions are described in the following sections.

psp_eth_ti_emac_init

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

Format

```
t_nwdriver_ret psp_eth_ti_emac_init ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_eth_ti_emac_start

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

Format

```
t_nwdriver_ret psp_eth_ti_emac_start ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_eth_ti_emac_stop

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

Format

```
t_nwdriver_ret psp_eth_ti_emac_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_eth_ti_emac_delete

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

Format

```
t_nwdriver_ret psp_eth_ti_emac_delete( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.