Network Driver for Microchip PIC32 User Guide

Version 1.10

For use with Network Driver for Microchip PIC32 module versions 1.01 and above

Date: 16-Jun-2017 10:46

Table of Contents

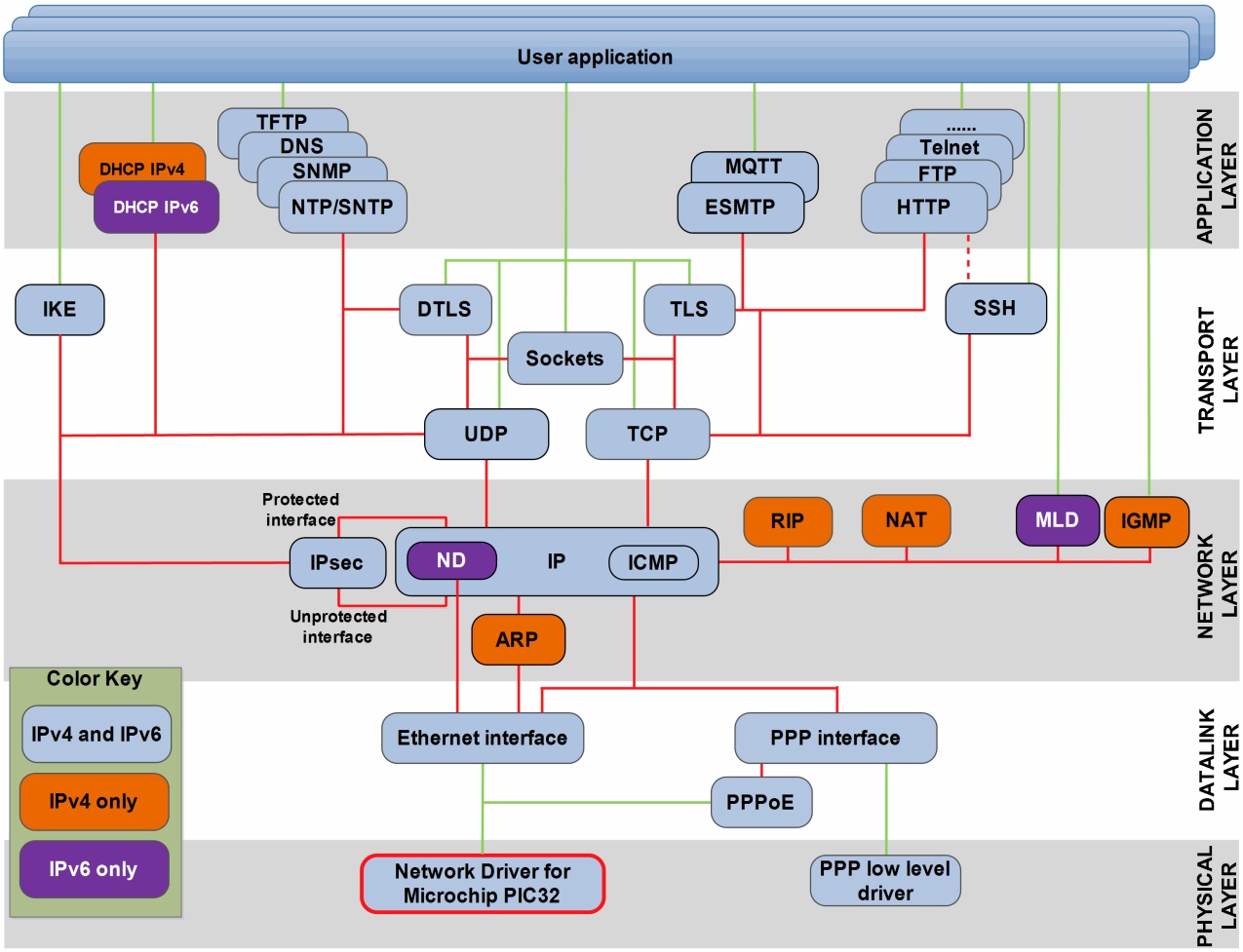
1 System Overview

1.1 Introduction

This guide is for those who want to implement a network driver for Microchip PIC32 32 bit microcontrollers. These devices are produced by Microchip Technology Inc.

The purpose of this driver is to provide an Ethernet physical port interface at the device end of a USB connection, so that the host system sees that remote physical port as a local Ethernet port.

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.
- Fully compatible with the HCC Network Driver Interface specification.
- Designed for integration with both RTOS and non-RTOS based systems.
- Conforms to the HCC Coding Standard.
- Supports Microchip PIC32 devices.
- HCC provides fully tested reference drivers for this module.

1.3 Device Description

This table summarizes the properties of the supported Microchip PIC32 devices. Both series have a large number of devices offering many options.

Series	Features
PIC32MX	Up to 120 MHz/150 DMIPS, MIPS® M4K core.
PIC32MZ	PIC32MZ with Floating Point Unit (FPU) Series. Up to 252 MHz/415 DMIPS,M-Class core with DSP instructions.

1.4 Packages and Documents

Packages

The table below lists the packages that 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 Microchip PIC32 driver is built.
nw_drv_eth_microchip_pic32	The Network Driver for Microchip PIC32 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 Microchip PIC32 User Guide

This is this document.

1.5 Change History

This section describes past changes to this manual.

- To view or download earlier manuals, see Archive: Network Driver for Microchip PIC32 Devices User Guide.
- For the history of changes made to the package code itself, see History: nw_drv_eth_microchip_pic32 .

The current version of this manual is 1.10. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.10	2017-06-16	1.01	New *Change History* format.
1.00	2017-04-25	1.01	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_microchip_pic32.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 microchip_pic32_eth_drv_init() function.

2.2 Configuration File

The file **config_ethdriver_microchip_pic32.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

The following files are in the directory **src/driver/network/ethernet/ microchip_pic32**. **These files should only be modified by HCC**.

File	Description
eth_microchip_pic32.c	Ethernet driver code.
eth_phy_dp83848.c	Source file for DP83848.
eth_phy_dp83848_reg.c	DP83848 registers header file.
eth_phy_ksz8031.c	Source file for KSZ8031.
eth_phy_ksz8031_reg.c	KSZ8031 registers header file.
eth_phy_lan8720.c	Source file for LAN8720.
eth_phy_lan8720_reg.c	LAN8720 registers header file.

2.4 Version Files

These files in the directory **src/version** contain the version numbers of components of this module. The version number is checked by all modules that use a component to ensure system consistency over upgrades.

File	Description
ver_ethdriver_microchip_pic32.h	Module version file
ver_psp_eth_mii.h	Media Independent Interface (MII) version file.
ver_psp_eth_phy.h	Ethernet PSP version file
ver_psp_eth_phy_dp83848.h	DP83848 version file.
ver_psp_eth_phy_lan8720.h	LAN8720 version file.

2.5 Platform Support Package (PSP) Files

These files provide functions the core code needs to call, depending on the hardware. The following generalized PSP implementation files are in the directory **src/psp**. These provide templates for you to produce your own PSP.

> **Note:** You may need to modify these PSP implementations for your specific microcontroller and development board; see PSP Porting for details.

File	Description
include/psp_eth_mii.h	Media Independent Interface (MII) header file.
include/psp_eth_phy.h	Ethernet PSP header file.

3 Configuration Options

Set the system configuration options in the file **src/config/config_ethdriver_microchip_pic32.h**, as described below. This section lists the available configuration options and their default values.

MICROCHIP_PIC32_ETHERNET_BUF_SIZE

The Ethernet buffer size. This must be larger than MAX_RX_DESC * 1536 + MAX_TX_DESC * 1536.

The default is 16384.

ETH_ISR_ID

The ISR ID. The default is 153.

ETH_ISR_PRIO

The ISR priority. The default is 2.

ETH_PHY_DEV_ADDR

The external PHY address. The default is 0x01.

ETH_HANDLE_PHY

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

ETHDRV_LINK_STA_POLL_INTERVAL

The link status poll interval in milliseconds. The default 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.

4 Application Programming Interface

This section describes the single Application Programming Interface (API) function and the error codes it may return.

4.1 microchip_pic32_eth_drv_init

Use this function to initialize the network driver.

Format

```
t_nwdriver_ret microchip_pic32_eth_drv_init (
    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	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_WR_LE16	psp_base	psp_endianness	Writes a 16 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_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/eth_microchip_pic32.c**.

Function	Description
psp_microchip_pic32_eth_init()	Initializes the hardware for ETH usage.
psp_microchip_pic32_eth_start()	Starts the module.
psp_microchip_pic32_eth_stop()	Stops the module.
psp_microchip_pic32_eth_delete()	Deletes all resources created by the module.
psp_microchip_pic32_get_buf()	Gets the contents of a buffer.

These functions are described in the following sections.

psp_microchip_pic32_eth_init

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

Format

```
t_nwdriver_ret psp_microchip_pic32_eth_init ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_microchip_pic32_eth_start

This function is provided by the PSP to start the Microchip PIC32 Ethernet driver.

Format

```
t_nwdriver_ret psp_microchip_pic32_eth_start ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_microchip_pic32_eth_stop

This function is provided by the PSP to stop the Microchip PIC32 Ethernet driver.

Format

```
t_nwdriver_ret psp_microchip_pic32_eth_stop ( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_microchip_pic32_eth_delete

This function is provided by the PSP to delete the Microchip PIC32 Ethernet driver, releasing the associated resources.

Format

```
t_nwdriver_ret psp_microchip_pic32_eth_delete( void )
```

Arguments

None.

Return Values

Return value	Description
NWDRIVER_SUCCESS	Successful execution.
NWDRIVER_ERROR	Operation failed.

psp_microchip_pic32_get_buf

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

Format

```
t_nwdriver_ret psp_microchip_pic32_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.