PIC32MX6/7 family with embedded Ethernet controller – PHY support

This document briefly explains how to add a new PHY to the MCHP TCPIP-BSD stack.

Note: The same approach applies to the MCHP TCPIP stack.

Table of Contents

Supported PHY's

- The PIC32MX6-7 families with embedded Ethernet controller have support for external MII/RMII connected PHY's. Currently we support:
    - National DP83848 PHY.
    - SMSC LAN 8700 PHY.

Support for other PHY's will be added in the future.

How to add a new PHY

Microchip provides a PHY interface library that is being used as part of the MAC driver for the TCPIP stack.
The basic concepts behind this PHY library are:

1. Communication to the PHY is done using MIIM interface. For access to the external PHY registers over the MIIM interface the communication functions provided by the Ethernet Controller Peripheral Library are used ("..path\Microchip\MPLAB C32\pic32mx\include\peripheral\eth.h").
2. All PHYs support a standard set of registers with well known functionality. (As part of the IEEE 802.3 Clause 22).
3. Few functions that are vendor specific have to be added for particular PHY's.

The functions that the current TCPIP stack MAC driver uses for communicating and configuring the external PHY are prototyped in
"..install path\Microchip\Include\TCPIP-BSD\eth_phy.h"
and implemented in the
"..install path\Microchip\TCPIP-BSD\eth_phy\eth_phy.c".

To add support for a new PHY you have to add the implementation for the following **4** functions (the prototypes of the functions are in "..install path\Microchip\Include\TCPIP-BSD\eth_phy.h"):

1. **extern unsigned int EthPhyMIIMAddress(void):**
   a. does not take parameters;
   b. Should return the address the PHY responds to. Depends on the hardware design.
      Note: all PHY's respond to address 0.
   c. Can be inlined/#defined
2. **extern unsigned int EthPhyMIIMClock(void):**
   a. does not take parameters;
   b. Should return the maximum clock frequency that the PHY can use for the MIIM transactions.
      i. Most PHY's should support 2 MHz.
      ii. National DP83848C supports 25 MHz, for example.
   c. Can be inlined/#defined
3. **extern eEthRes EthPhyConfigureMII(eEthPhyCfgFlags cFlags):**
   a. This function should configure the PHY in one of MII/RMII operation modes.
   b. The relevant input flags are:
      i. ETH_PHY_CFG_RMII if RMII configuration is requested
      ii. ETH_PHY_CFG_MII if MII configuration is requested
   c. It should return:
      i. ETH_RES_OK for success
      ii. An error code if the requested configuration is not supported.
4. **extern eEthRes EthPhyConfigureMdix(eEthOpenFlags oFlags):**
   a. The function should configure the MDIX mode for the PHY.
   b. The relevant input flags are:
      i. ETH_OPEN_MDIX_AUTO if auto MDIX is requested
      ii. else ETH_OPEN_MDIX_NORM/ETH_OPEN_MDIX_SWAP for normal/swap mode.
   c. It should return:
      i. ETH_RES_OK for success
      ii. An error code if the requested mode is not supported.

Additional Notes:

1. All functions in the PHY library are defined as weak. You can replace any of the functions with your own implementation if need arises. The function header in the "..install path\Microchip\Include\TCPIP-BSD\eth_phy.h" explains what input/output and behavior the respective function should have.
2. Take a look in the "..install path\Microchip Solutions\Microchip\TCPIP-BSD\eth_phy\nat_dp83848c.c" or "..install path\Microchip Solutions\Microchip\TCPIP-BSD\eth_phy\smsc_8700.c" for an implementation example. Follow the same approach.
3. All functions needed to access the PHY MIIM registers are part of the provided Ethernet library.