STM32CubeIDE: ethernet hardware and lwip setup for NUCLEO-H723ZG

The following version of STM32CubeIDE was used for the setup:
(Collected through STM32CubeIDE: Help -> About STM32CubeIDE : Right click the new window and "Copy Build Id Information to Clipboard")

- STM32CubeIDE Version: 1.11.2
- Build: 14494_20230119_0724 (UTC)
- OS: Linux, v.5.15.0-60-generic, x86_64 / gtk 3.24.20, WebKit 2.38.4
- Java vendor: Eclipse Adoptium
- Java runtime version: 11.0.16+8
- Java version: 11.0.16

Useful guides

The following guides provide a more in depth guide on how to configure the ethernet peripheral and lwip.

ST Community: How to STM32H7 with ETH and LwIP

Youtube Playlist: Tutorial about Cortex M7 and STM32CubeIDE Ethernet + LwIP (Videos 3-9)

Necessary ST Documents

STM32-H723ZG Refernce Manual (Downloads pdf)

Nucleo-H723ZG Schematics (online viewable)

Nucleo-H723ZG

This document is for the Nucleo-H723ZG Development-Board. Other architectures or Development-Boards may have other pinouts and basic configurations and my require different memory addresses.

STM32CubeIDE

Create a new STM32 project

- File -> New -> STM32 Project
- A new "STM32 Project" window appears
- Select "Board Selector"
- In the "Commercial Part Number" text field enter the name of your board. We are using "NUCLEO-H723ZG" here.
- Select "NUCLEO-H723ZG" in the Board List.
- The "Next >" Button should appear green.
- Click on "Next >".
- Give your project a name. We are using "nucleo-h723zg-ethernet-lwip" here.
- Select "Yes" for "Initialize all peripherials with their default Mode?".
- Click on the *.ioc file.

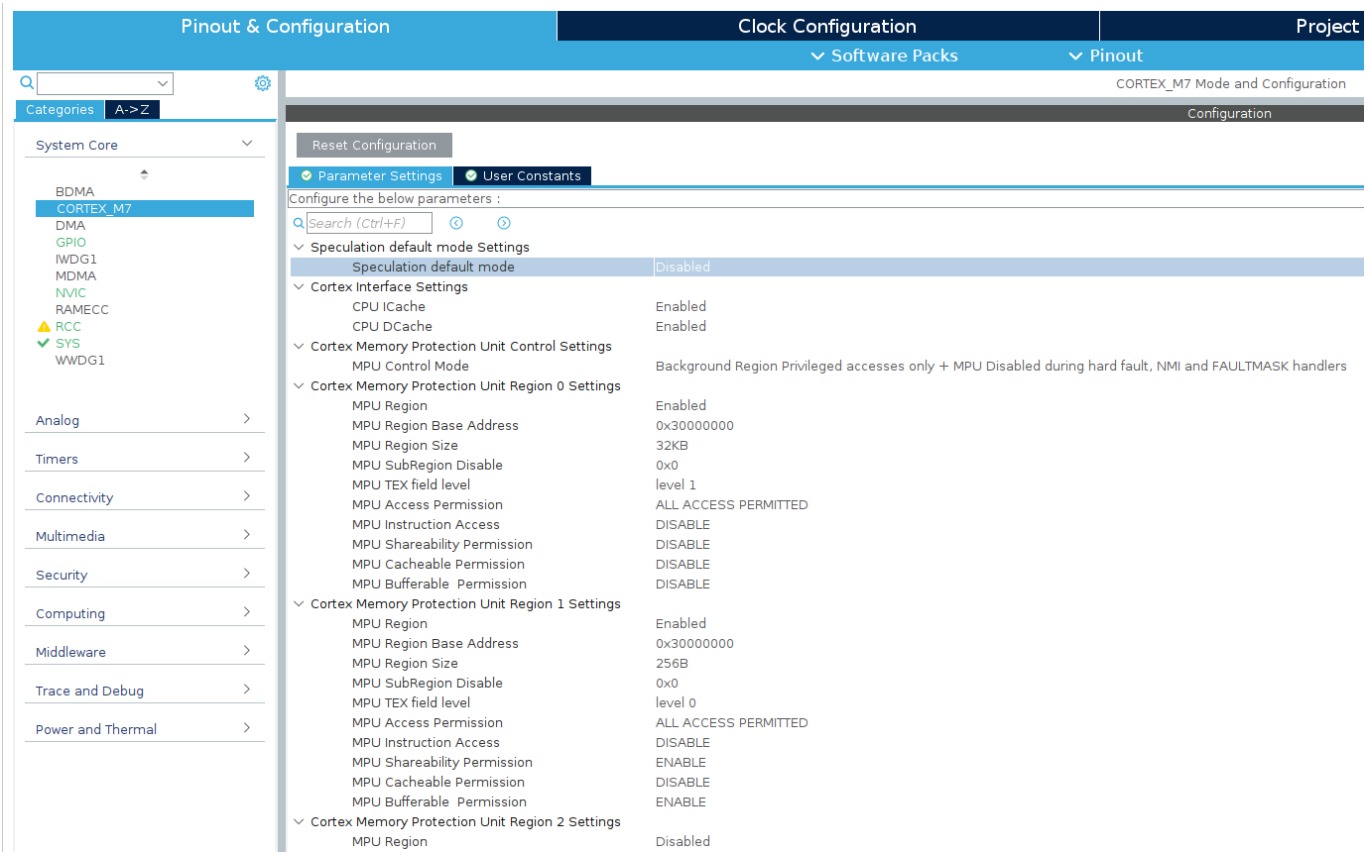
Configuration through *.ioc file

Cortex_M7

Enable the Instruction and Data Cache for the Cortex_M7 processor:

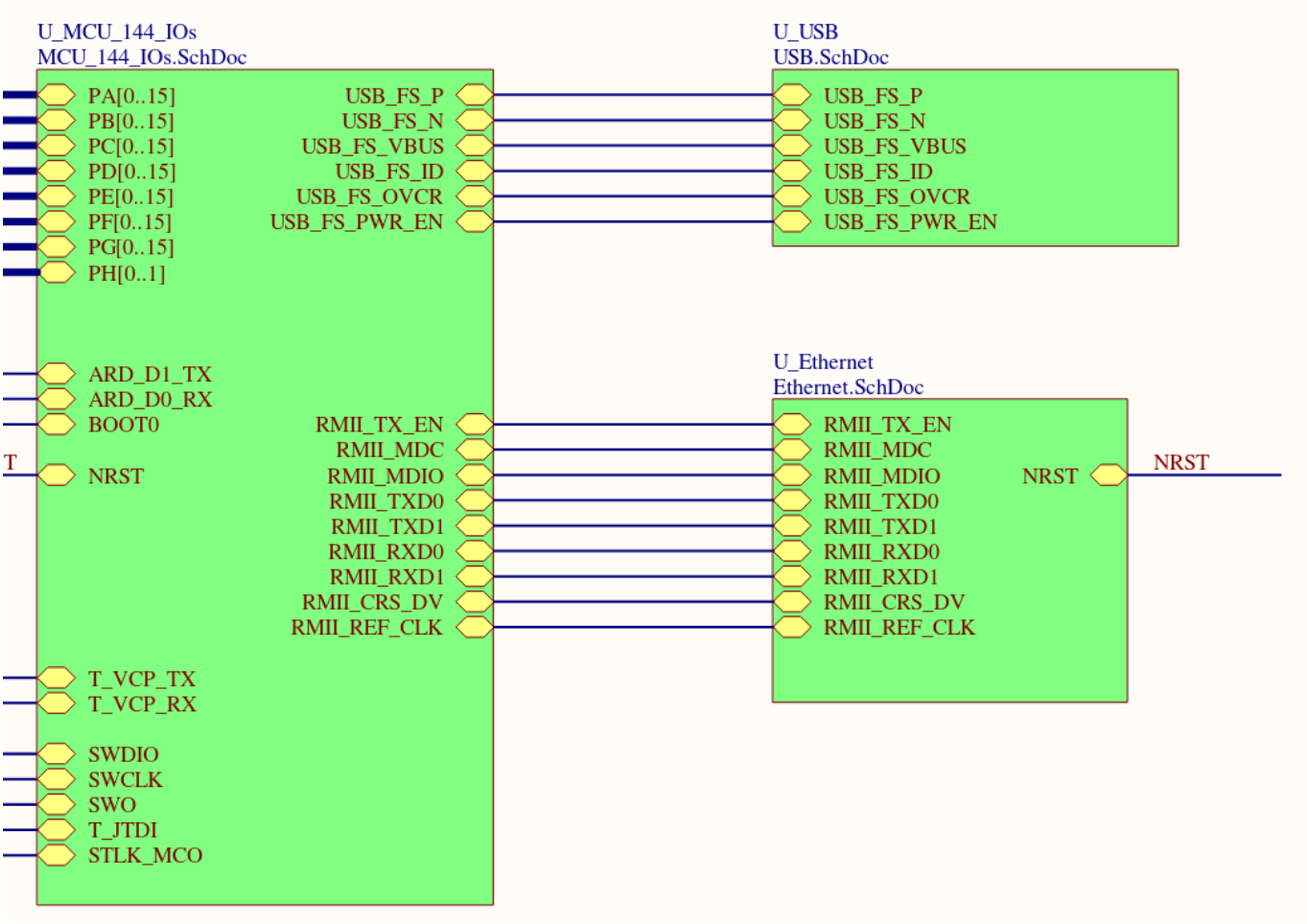
- CPU ICache -> Enabled
- CPU DCache -> Enabled

Configure MPU:



ETH

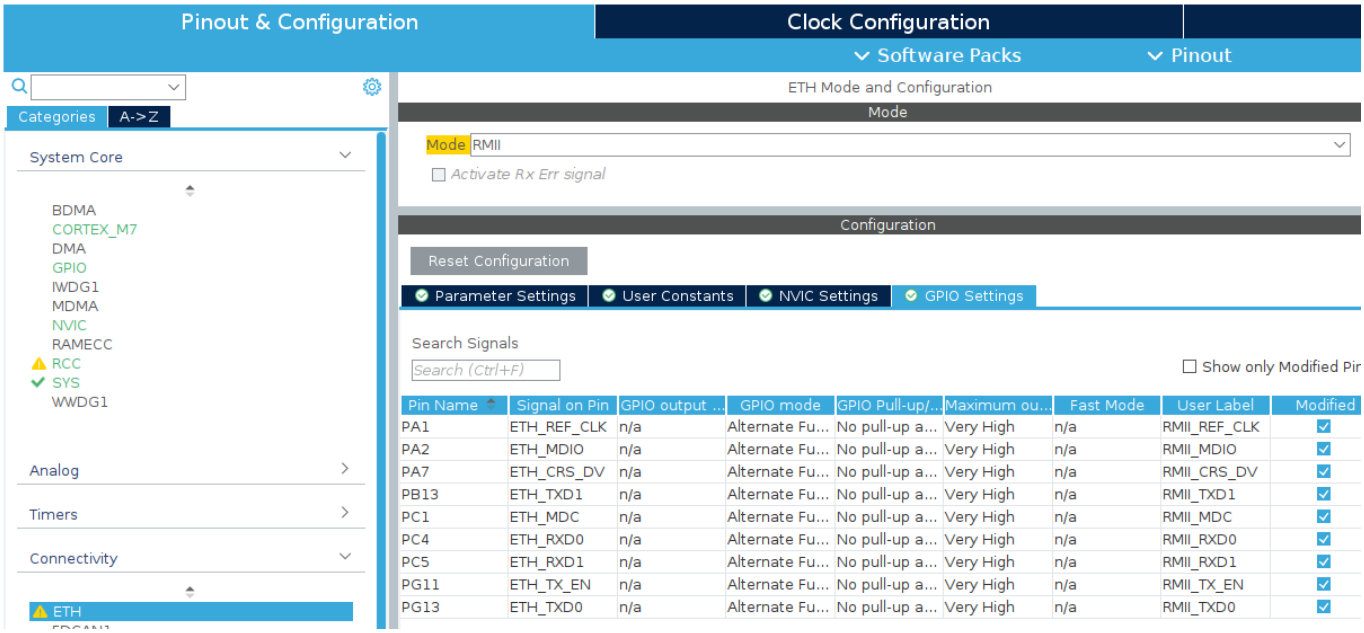
Make sure the correct Mode for the ETH peripheral is used. To find out which mode is supported by the hardware refer to the schematic. The Nucleo-H723ZG Schematics show that the RMII mode is supported.



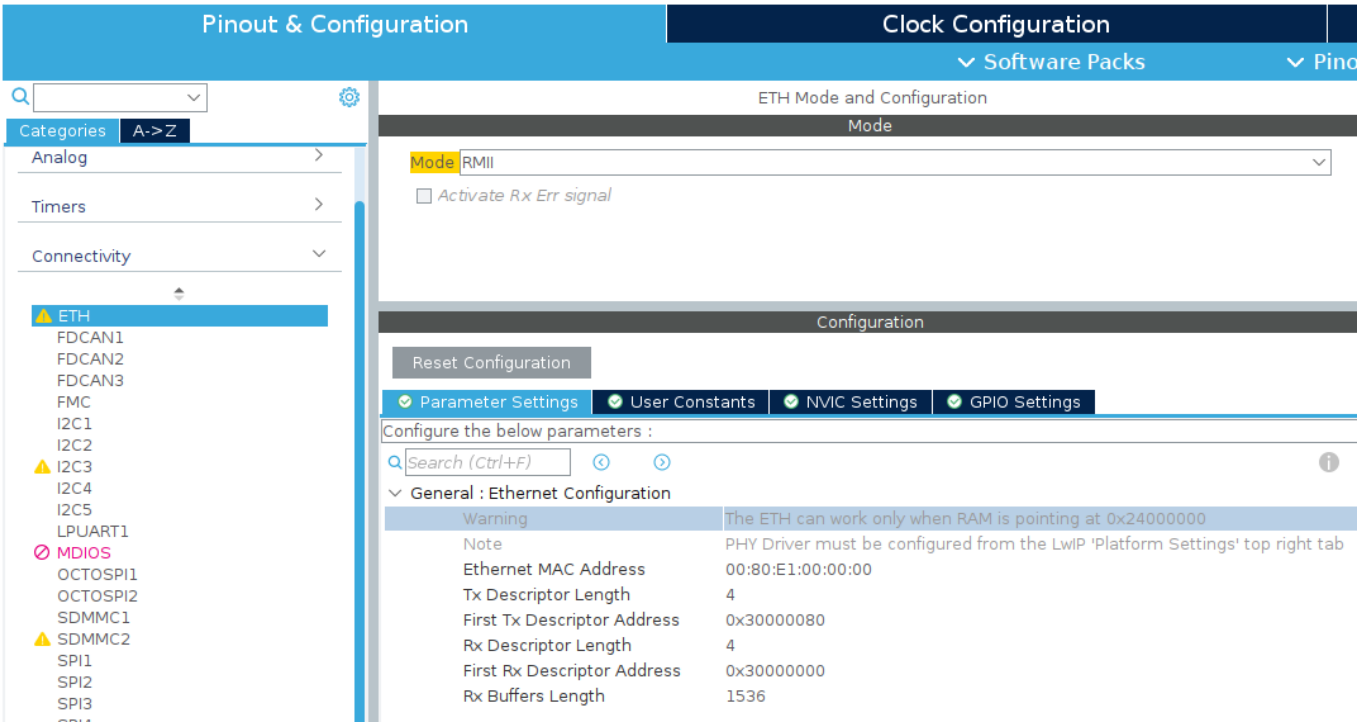
Confirm that the correct GPIOs are used. Using the schematic we can see the following configuration:

Pin	Function
PA1	RMII_REF_CLK
PA2	RMII_MDIO
PA7	RMII_CRS_DV
PB13	RMII_TXD1
PC1	RMII_MDC
PC4	RMII_RXD0
PC5	RMII_RXD1
PG11	RMII_TX_EN
PG13	RMII_TXD0

The correct GPIOs were selected by the default initialization. Set the output speed to "Very High".

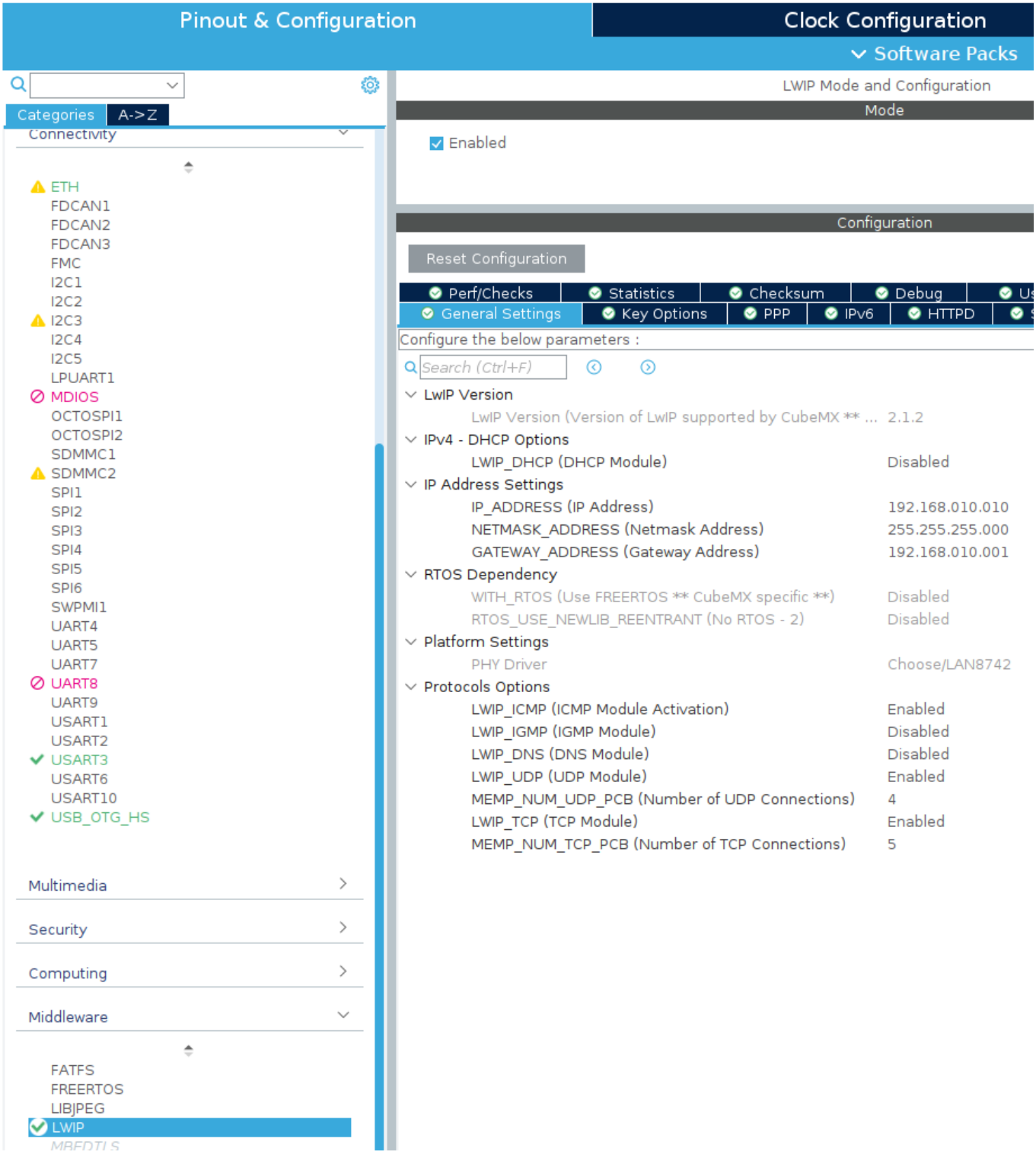


Select the memory addresses that you're using for the Rx- and Tx-Descriptors. The default configuration will use the memory in the D2 domain. The eastimated size for the descriptors are 128 B. (The actual size is currently 96 B, but the MPU Configuration is unable to configure a 192 B sized region.)



LWIP

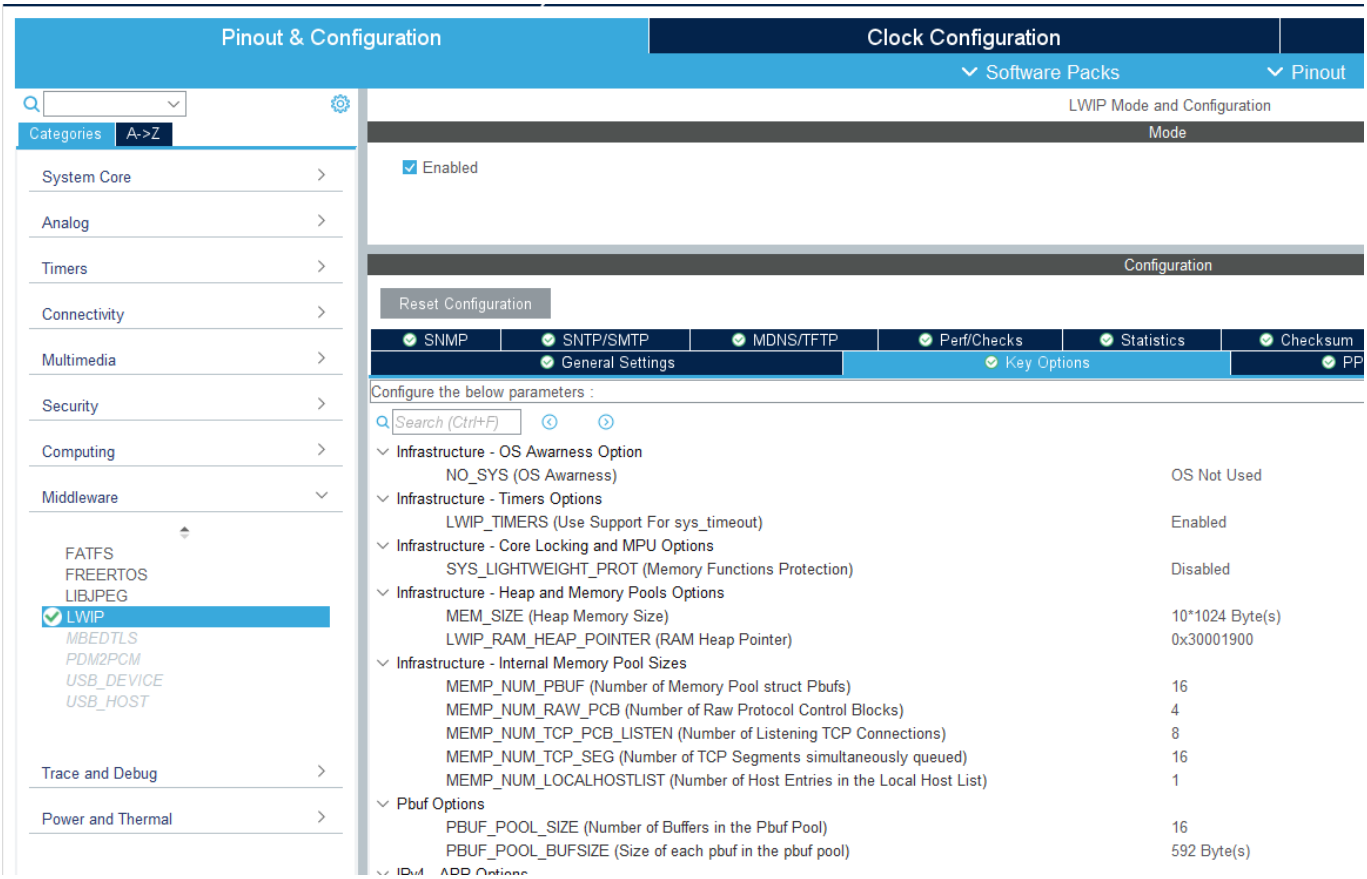
It may be easier to start with a static IP-Address:



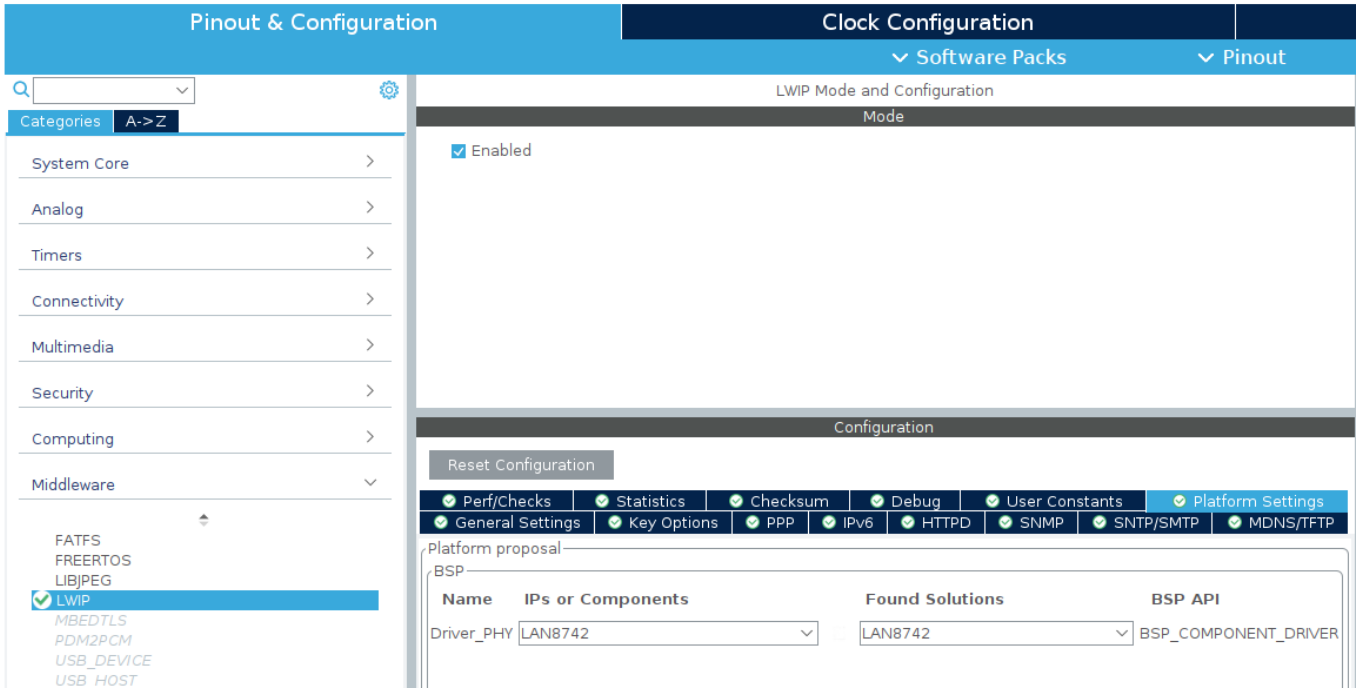
Select the desired size and address for the Heap. According to this guide we can calculate the position the following way:

usage	Address Space	size
Free	0x30004100 - 0x30007fff	16128 B
LwIP Heap	0x30001900 - 0x300040ff	10*1024 B
Rx Buffer	0x30000100 - 0x300018ff	4*Rx Buffers Length = 4*1536 B
Tx Descriptor	0x30000080 - 0x300000ff	128 B

usage	Address Space	size
Rx Descriptor	0x30000000 - 0x3000007f	128 B



Select a value for the "Found Solutions" in the "Platform Settings" tab.



When comparing the calulated memory regions with the actual memory regions provided by the "Build analyzer", we can see some difference in the actually needed space. The reason for this is unkown to me.

Memory Regions | Memory Details

Region	Start address	End address	Size	Free	Used	Usage (%)
ITCMRAM	0x00000000	0x0000ffff	64 KB	64 KB	0 B	0.00%
DTCMRAM	0x20000000	0x2001ffff	128 KB	128 KB	0 B	0.00%
FLASH	0x08000000	0x080fffff	1024 KB	938.8 KB	85.2 KB	8.32%
RAM_D1	0x24000000	0x2404ffff	320 KB	305.7 KB	14.3 KB	4.47%
RAM_D2	0x30000000	0x30007fff	32 KB	13.37 KB	18.63 KB	58.21%
RAM_D3	0x38000000	0x38003fff	16 KB	16 KB	0 B	0.00%

The actually free space is 13.37 KB which is equivalent to roughly 13690 Byte.

ethernetif.c

You need to add the following commands in the "User Code 2" Section of the "ethernetif.c" file. This shoves the "memp_memory_RX_POOL_base" pointer into the ".Rx_PoolSection" section. The ".Rx_PoolSection" section is then later used in the "STM32H723ZGTX_FLASH.ld" file to move the array into the memory in the D2 domain.

```
...
/* USER CODE BEGIN 2 */
#if defined ( __ICCARM__ ) /*!< IAR Compiler */
#pragma location = 0x30000100
extern u8_t memp_memory_RX_POOL_base[];

#elif defined ( __CC_ARM )  /* MDK ARM Compiler */
__attribute__((at(0x30000100)))) extern u8_t memp_memory_RX_POOL_base[];

#elif defined ( __GNUC__ ) /* GNU Compiler */
__attribute__((section(".Rx_PoolSection"))) extern u8_t
memp_memory_RX_POOL_base[];

#endif
/* USER CODE END 2 */
...
```

STM32H723ZGTX_FLASH.ld

Now we select the address for the different variables. We will move all variables into memory in the D2 domain.

```
...
  } >RAM_D1

  /* Modification start */
  .lwip_sec (NOLOAD) : {
    . = ABSOLUTE(0x30000000);
    *(.RxDecripSection)
```

```
    . = ABSOLUTE(0x30000080);
    *(.TxDecripSection)

    . = ABSOLUTE(0x30000100);
    *(.Rx_PoolSection)
  } >RAM_D2
  /* Modification end */

  /* Remove information from the standard libraries */
  /DISCARD/ :
  {
...
```

main.c

In order to process the received packets you need to add the following command to your super loop.

```
...
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    MX_LWIP_Process();
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
...
```

Test LWIP

Open up a terminal. Configure one of your network interfaces to the same network (here 192.168.10.0/24) as your specified IP-address. Once that is done ping your microcontroller. The pings should get a response.

```
ping 192.168.10.10
```

Common errors

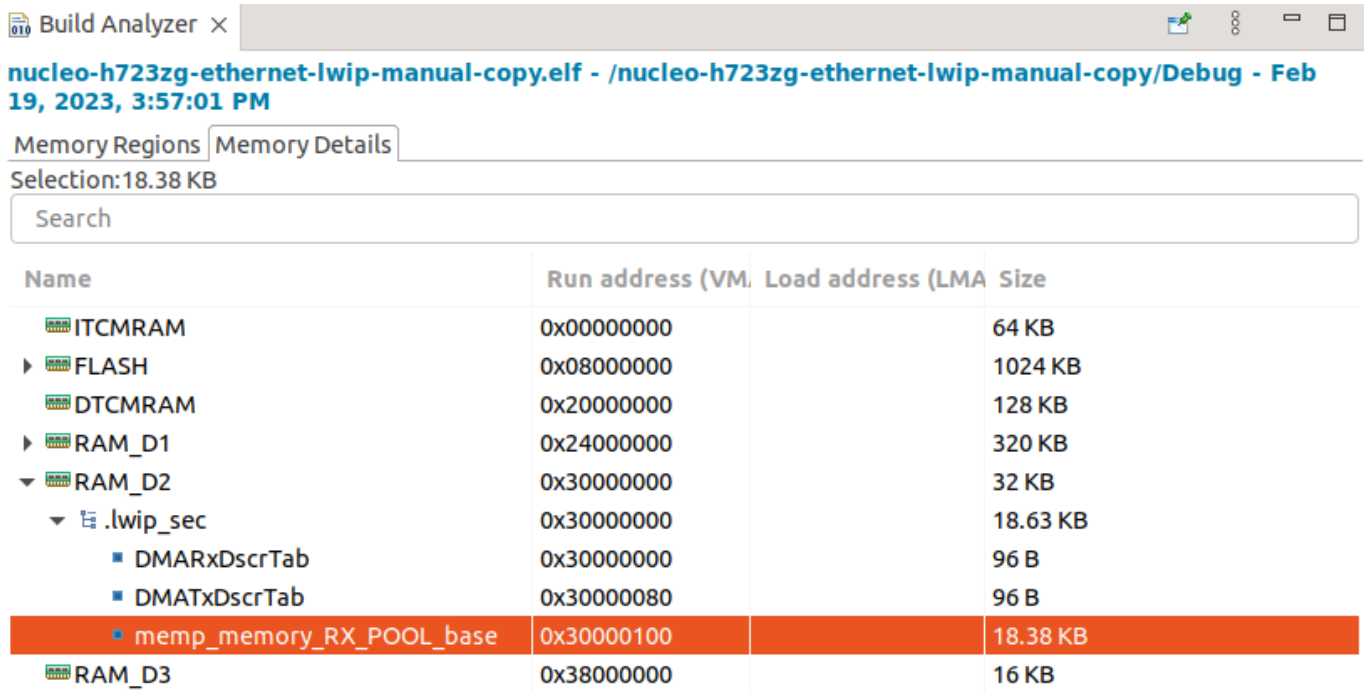
Hardfault

If you encounter the "Hard Fault" error when debugging the code, there is probably an error with the configuration for the MPU or the selected addresses.

Microcontroller responds to 3-5 pings, then no responses

I encountered this error when the ".RxArraySection" was not correctly configured. Make sure that "memp_memory_RX_POOL_base" shows up in the "Build Analyzer" in the "Memory Details" tab. For me the reason was, that the name for the section was different in both the "ethernetif.c" and the "STM32H723ZGTX_FLASH.ld" file. They have to be the same, otherwise the "memp_memory_RX_POOL_base" pointer will not be moved to the desired memory address.

Correct: (example for used names ".RxArraySection" and ".RxArraySection")



Wrong: (example for used names ".RxArraySection_not_the" and ".RxArraySection_same_name")