How to create a project for STM32H7 with Ethernet and LwIP stack working

Adam BERLINGER
ST Employee

on 2020-03-11 07:50 AM - edited on 2024-06-17 06:55 AM by ADMIN **Laurids_PETERSEN**

This readme is intended for STM32CubeIDE version 1.9.0 and STM32CubeH7 version 1.10.0. For older tool versions please see older version of this readme in the repository.

If you have any questions regarding Ethernet topics in general, please visit our MCU forum and create a thread. Doing so will provide you with a shorter response time.

Example project code and older version of this article is provided on Github: https://github.com/stm32-hotspot/STM32H7-LwIP-Examples. Detailed how to step-by-step is provided below.

Features

- Fixed IP address 192.168.1.10
- Code should work even when re-generating the code in STM32CubeMX
- Changes in code can be find by searching for

Feedback

```
ETH_CODE
```

keyword

Release notes

Version	STM32CubeIDE version	STM32CubeH7 version	Description / Update	Date
1.2	1.9.0	1.10.0	Ported to new IDE/library version. Ethernet driver reworked in new library release. Added iperf measurement and TCP/IP settings tuned. Published on Github	August 9th 2022
1.1	1.6.1	1.9.0	Added Cortex-M4 base examples	July 19th 2021
1.0	1.6.1	1.9.0	Initial release on ST community (with minor changes on Github)	June 21st 2021

Using GIT tags it should be easy to find examples for particular version of STM32CubeIDE and HAL library

TCP/IP configuration in LwIP

Below configuration is necessary to achieve good TCP/IP performance

Parameter	Value	Formula	Needs to be changed in MX
TCP_MSS	1460	`1500-40`	**yes**
TCP_SND_BUF	5840	`4 * TCP_MSS`	**yes**
TCP_WND	5840	`4 * TCP_MSS`	no
TCP_SND_QUEUELEN	16	`UPPER((4 * TCP_SND_BUF) / TCP_MSS)`	**yes**

Memory layout

On STM32H74x/H75x devices, all data related to Ethernet and LwIP are placed in D2 SRAM memory (288kB). First 128kB of this memory are reserved for Cortex-M4 on dual-core devices. On single core devices this part can be used for other purposes.

Variable	STM32H74x/H75x address	Cortex-M4 alias	Size	Source file
DMARxDscrTab	0x30040000	0x10040000	96 (256 max.)	ethernetif.c
DMATxDscrTab	0x30040100	0x10040100	96 (256 max.)	ethernetif.c
memp_memory_RX_POOL_base	0x30040200	0x10040200	12*(1536 + 24)	ethernetif.c
LwIP heap	0x30020000	0x10020000	131048 (128kB - 24)	lwipopts.h

For STM32H72x/H73x devices, the D2 SRAM is more limited (only 32kB). The RX buffers need to be placed in AXI SRAM, since they won't fit to D2 RAM, together with LwIP heap. The LwIP heap is reduced to fit the rest of D2 RAM together with DMA descriptors.

Variable	STM32H72x/H73x address	Size	Source file
DMARxDscrTab	0x30000000	96 (256 max.)	ethernetif.c
DMATxDscrTab	0x30000100	96 (256 max.)	ethernetif.c
memp_memory_RX_POOL_base	AXI SRAM (32-byte aligned)	12*(1536 + 24)	ethernetif.c
LwIP heap	0x30000200	32232 (32kB - 512 - 24)	lwipopts.h

Value provided here are an example implementation, other configurations are also possible in order to optimize memory usage. When changing memory layout, the MPU configuration needs to be updated accordingly.

License

Libraries and middleware is taken from STM32CubeH7 package. So the same licenses apply to the these examples. There is minimum code added on top of STM32CubeMX and HAL libraries, this code is provided AS-IS.

How to create project from scratch

Goal

Goal of this example is to:

- Configure project in STM32CubeMX for STM32H750-Discovery
- Configure FreeRTOS and LwIP middlewares correctly
- Send UDP message periodically (optional)

Although the example is using STM32H750-Discovery, it might be easy to use the same steps for other STM32H7 based boards. The main differences are usually pinout and clock configuration. You might also need to check board solder bridges to make sure the Ethernet is connected to MCU.

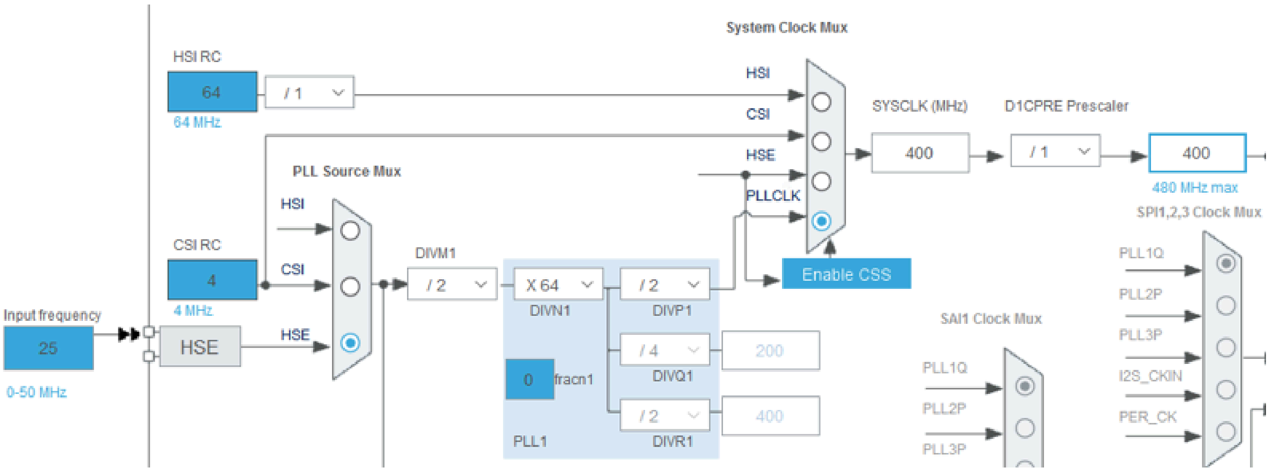
STM32CubeMX project configuration

- Create new project in STM32CubeMX, select STM32H750-Discovery board and select "No" to "Initialize all peripherals in default mode?" pop-up.
  - This will help with pin assignment.

Basic configuration

Configure clock tree:

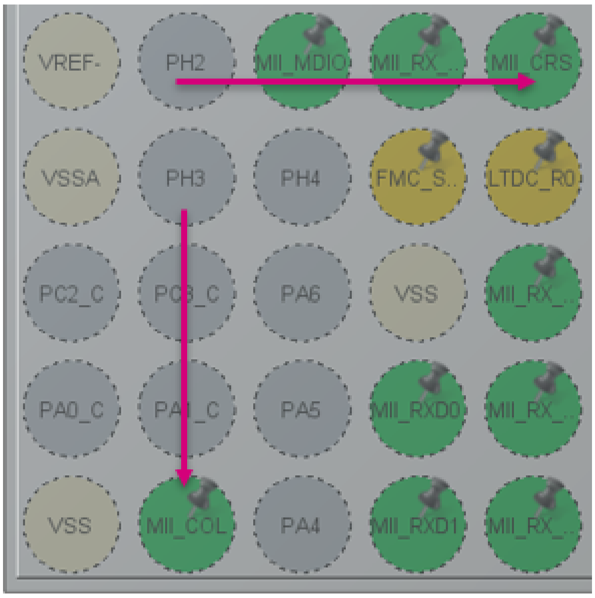
- In pinout/RCC configure HSE in bypass mode
- In clock tree configure 400MHz for core



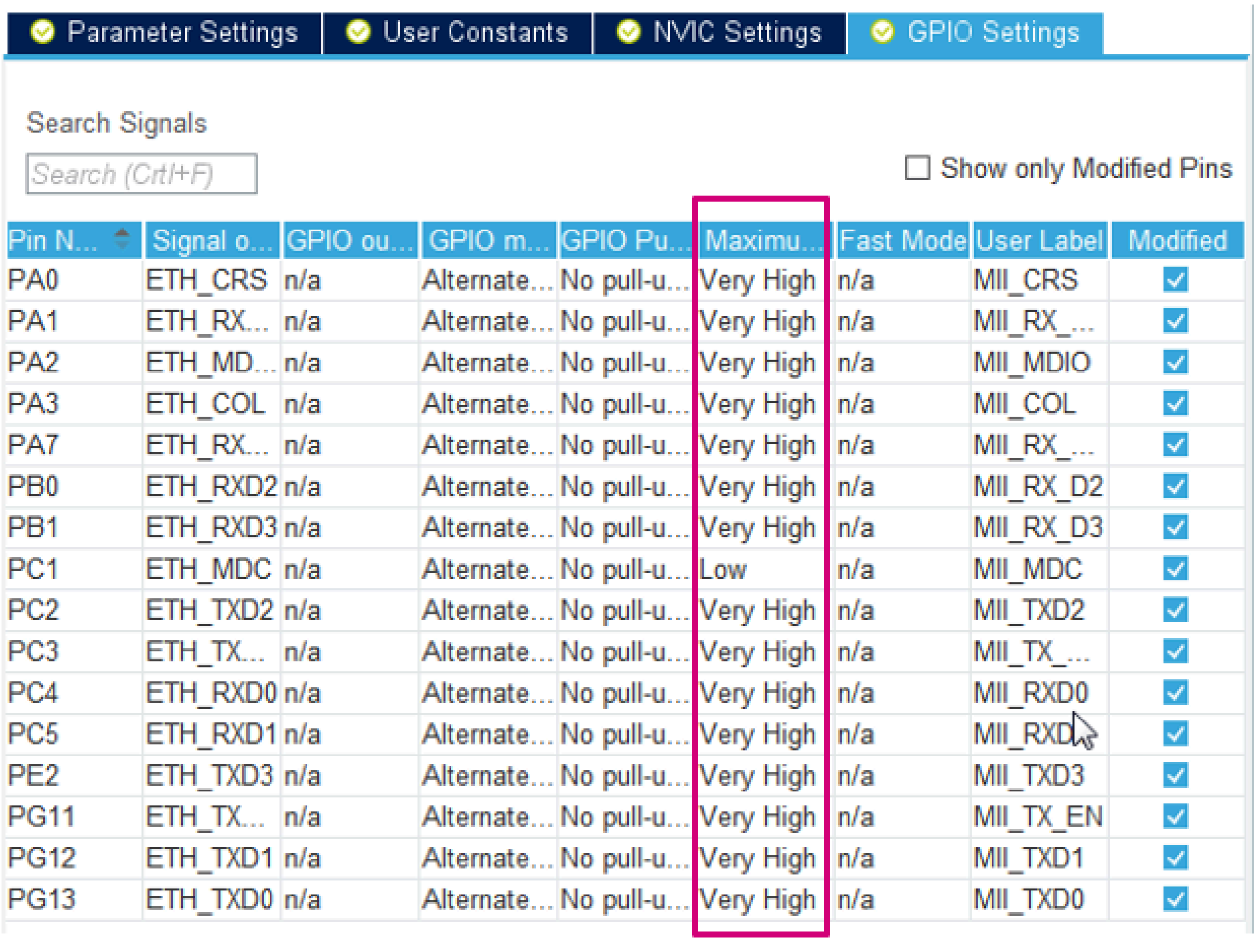
- In pinout/SYS configure different timebase than SysTick (recommended when using FreeRTOS)
  - TIM6 is usually a good option, since it is a simple timer

Ethernet configuration

- Enable Ethernet peripheral in pinout view in MII mode (MII used on the board).
- Enable Ethernet interrupt and set preemption priority to 5. This is required by FreeRTOS in order to call its functions from interrupt handler.
- Relocate Ethernet CRS and COL signals from PH2/PH3 to PA0/PA3
  - These signals are optional in full-duplex mode and not connected in default configuration
  - This will also allow use PH2/PH3 for QSPI
- Other pins should be correctly placed, since we create the project from board selector.



- Set GPIO pin speed to Very High.



Search Signals

Search (Crtl+F)

□ Show only Modified Pins

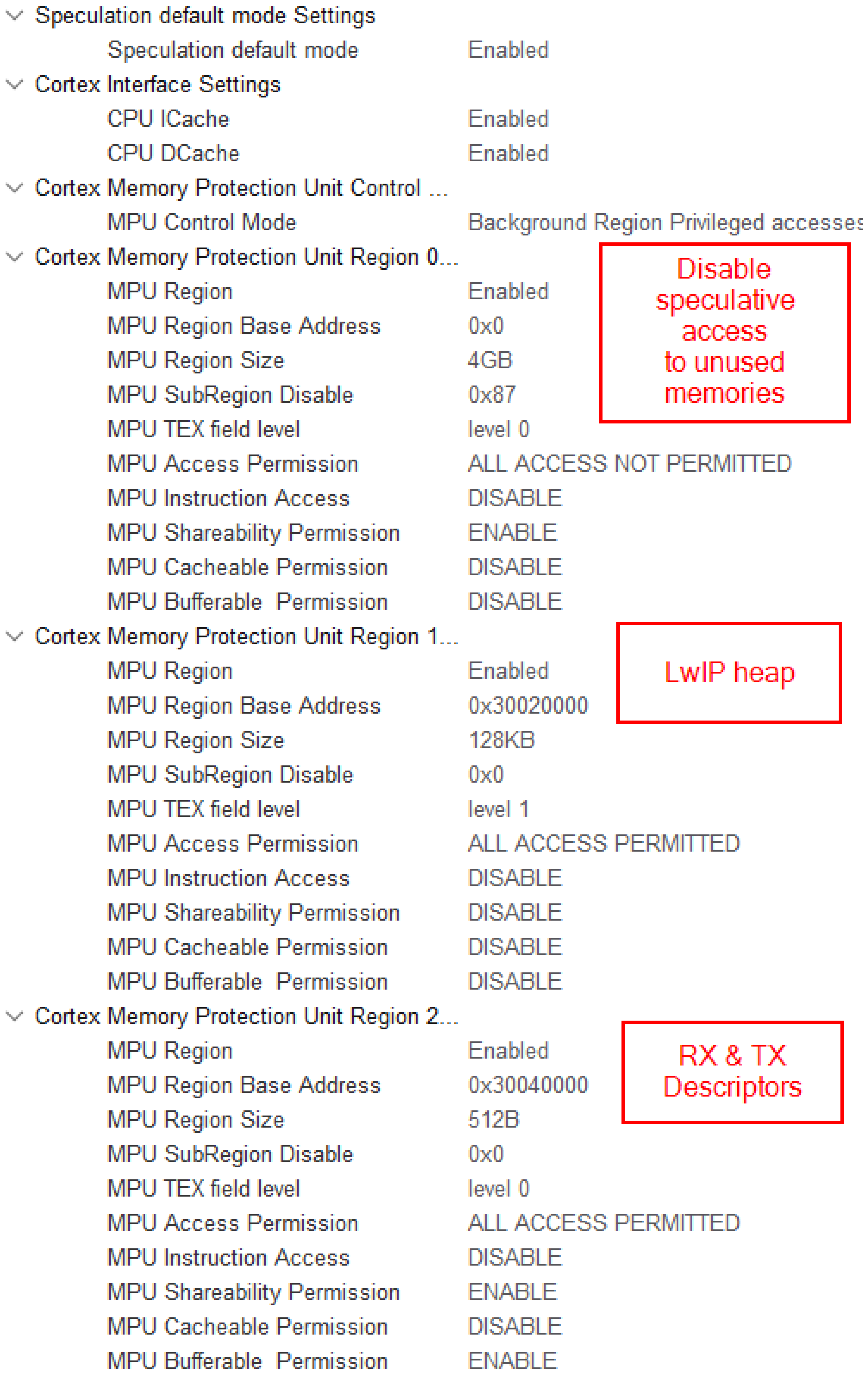
Pin N...	Signal o...	GPIO ou...	GPIO m...	GPIO Pu...	Maximu...	Fast Mode	User Label	Modified
PA0	ETH_CRS	n/a	Alternate...	No pull-u...	Very High	n/a	MII_CRS	☑
PA1	ETH_RX...	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RX_...	☑
PA2	ETH_MD...	n/a	Alternate...	No pull-u...	Very High	n/a	MII_MDIO	☑
PA3	ETH_COL	n/a	Alternate...	No pull-u...	Very High	n/a	MII_COL	☑
PA7	ETH_RX...	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RX_...	☑
PB0	ETH_RXD2	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RX_D2	☑
PB1	ETH_RXD3	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RX_D3	☑
PC1	ETH_MDC	n/a	Alternate...	No pull-u...	Low	n/a	MII_MDC	☑
PC2	ETH_TXD2	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TXD2	☑
PC3	ETH_TX...	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TX_...	☑
PC4	ETH_RXD0	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RXD0	☑
PC5	ETH_RXD1	n/a	Alternate...	No pull-u...	Very High	n/a	MII_RXD	☑
PE2	ETH_TXD3	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TXD3	☑
PG11	ETH_TX...	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TX_EN	☑
PG12	ETH_TXD1	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TXD1	☑
PG13	ETH_TXD0	n/a	Alternate...	No pull-u...	Very High	n/a	MII_TXD0	☑

The ETH_MDC speed couldn't be changed for some reason, but it doesn't affect the application and it was already fixed in new versions.

Cortex-M7 configuration

This step can be skipped when using Cortex-M4 core.

- Enable ICache and DCache.
- Enable memory protection unit (MPU) in “Background Region Privileged access only + MPU Disabled ...” mode. Configure regions according to the picture below:



```
∨ Speculation default mode Settings
        Speculation default mode          Enabled
∨ Cortex Interface Settings
        CPU ICache                        Enabled
        CPU DCache                        Enabled
∨ Cortex Memory Protection Unit Control ...
        MPU Control Mode                  Background Region Privileged accesses
∨ Cortex Memory Protection Unit Region 0...
        MPU Region                        Enabled
        MPU Region Base Address           0x0
        MPU Region Size                   4GB
        MPU SubRegion Disable             0x87
        MPU TEX field level               level 0
        MPU Access Permission             ALL ACCESS NOT PERMITTED
        MPU Instruction Access            DISABLE
        MPU Shareability Permission       ENABLE
        MPU Cacheable Permission          DISABLE
        MPU Bufferable  Permission        DISABLE
∨ Cortex Memory Protection Unit Region 1...
        MPU Region                        Enabled
        MPU Region Base Address           0x30020000
        MPU Region Size                   128KB
        MPU SubRegion Disable             0x0
        MPU TEX field level               level 1
        MPU Access Permission             ALL ACCESS PERMITTED
        MPU Instruction Access            DISABLE
        MPU Shareability Permission       DISABLE
        MPU Cacheable Permission          DISABLE
        MPU Bufferable  Permission        DISABLE
∨ Cortex Memory Protection Unit Region 2...
        MPU Region                        Enabled
        MPU Region Base Address           0x30040000
        MPU Region Size                   512B
        MPU SubRegion Disable             0x0
        MPU TEX field level               level 0
        MPU Access Permission             ALL ACCESS PERMITTED
        MPU Instruction Access            DISABLE
        MPU Shareability Permission       ENABLE
        MPU Cacheable Permission          DISABLE
        MPU Bufferable  Permission        ENABLE
```

Disable speculative access to unused memories

LwIP heap

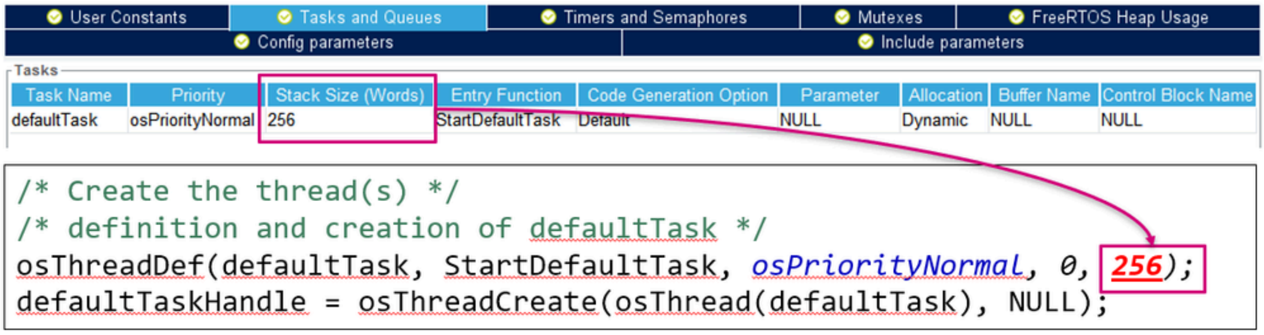
RX & TX Descriptors

Above example is for STM32H743 device. For other devices or Cortex-M4 core on dual-core device, different addresses and size might be necessary. Please refer to section Memory layout

When using dual-core device and running Ethernet on Cortex-M7 core, it must be ensured that memory used by Ethernet is not used by Cortex-M4. Also note the Cortex-M4 can use different address alias for D2 RAM

FreeRTOS configuration

- Enable the FreeRTOS with CMSIS_V1 API.
- Increase the size of defaultTask stack to 512 words[1]
- Lower stack values cause memory corruptions
- Please check also that the generated code is correct, since there is bug when increasing the MINIMAL_STACK_SIZE and there might be old value in code (this should be fixed in new versions)



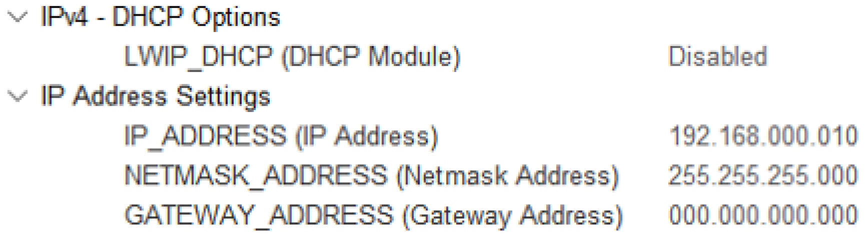
Tasks

Task Name	Priority	Stack Size (Words)	Entry Function	Code Generation Option	Parameter	Allocation	Buffer Name	Control Block Name
defaultTask	osPriorityNormal	256	StartDefaultTask	Default	NULL	Dynamic	NULL	NULL

```
/* Create the thread(s) */
/* definition and creation of defaultTask */
osThreadDef(defaultTask, StartDefaultTask, osPriorityNormal, 0, 256);
defaultTaskHandle = osThreadCreate(osThread(defaultTask), NULL);
```

LwIP configuration

- Enable LwIP in middleware.
- In "General settings" tab, disable DHCP server and configure fixed IP address (unless you know how to configure and use DHCP).



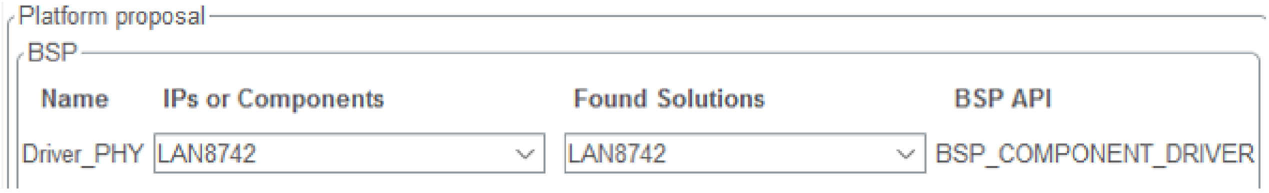
IPv4 - DHCP Options

LWIP_DHCP (DHCP Module)	Disabled

IP Address Settings

IP_ADDRESS (IP Address)	192.168.000.010
NETMASK_ADDRESS (Netmask Address)	255.255.255.000
GATEWAY_ADDRESS (Gateway Address)	000.000.000.000

- In the attached examples, the 192.168.1.10 IP address is used (instead of 192.168.0.10 shown on the screenshot).
- In "Platform settings" tab select "LAN8742" in both select boxes. The LAN8742 driver is also compatible with LAN8740 device, which is actually present on the STM32H750-Discovery board. The main difference between these devices is support of MII interface. On other boards LAN8742 PHY chip is used.



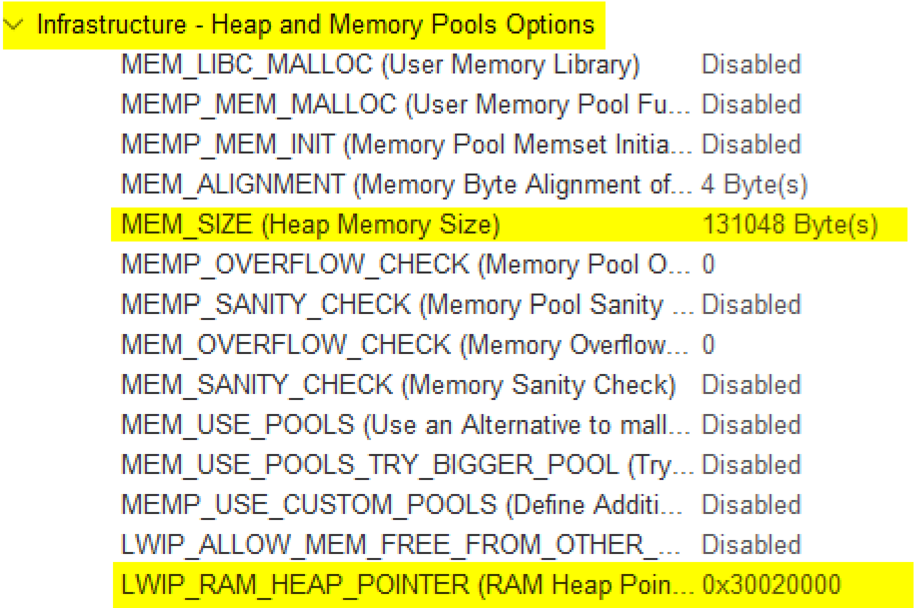
Platform proposal

BSP

Name	IPs or Components	Found Solutions	BSP API
Driver_PHY	LAN8742	LAN8742	BSP_COMPONENT_DRIVER

In "Key options" tab:

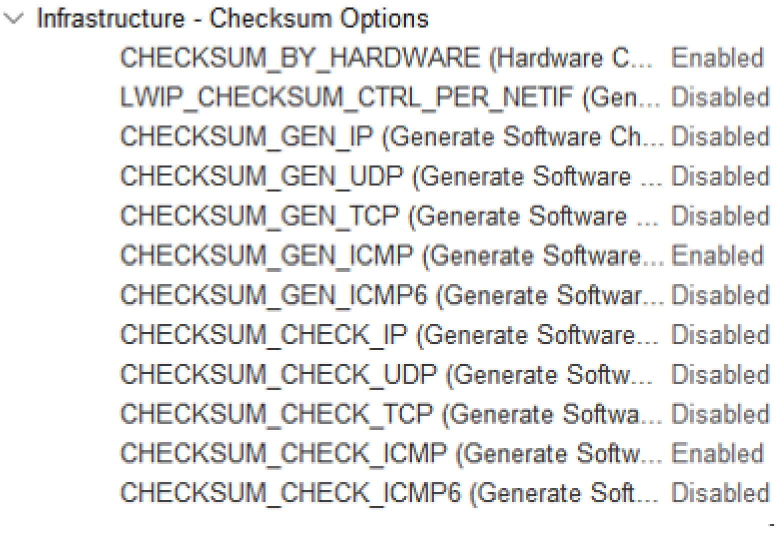
- Configure MEM_SIZE to 16360. This specifies the heap size, which we will relocated to D2 SRAM (16kb minus 24 bytes for allocator metadata).
- Also enable LWIP_NETIF_LINK_CALLBACK (needed for cable plugging/unplugging detection).
- Set the LWIP_RAM_HEAP_POINTER to proper address[2]
- Set the MEM_SIZE parameter to proper size[2]



Infrastructure - Heap and Memory Pools Options

MEM_LIBC_MALLOC (User Memory Library)	Disabled
MEMP_MEM_MALLOC (User Memory Pool Fu...	Disabled
MEMP_MEM_INIT (Memory Pool Memset Initia...	Disabled
MEM_ALIGNMENT (Memory Byte Alignment of...	4 Byte(s)
MEM_SIZE (Heap Memory Size)	131048 Byte(s)
MEMP_OVERFLOW_CHECK (Memory Pool O...	0
MEMP_SANITY_CHECK (Memory Pool Sanity ...	Disabled
MEM_OVERFLOW_CHECK (Memory Overflow...	0
MEM_SANITY_CHECK (Memory Sanity Check)	Disabled
MEM_USE_POOLS (Use an Alternative to mall...	Disabled
MEM_USE_POOLS_TRY_BIGGER_POOL (Try...	Disabled
MEMP_USE_CUSTOM_POOLS (Define Additi...	Disabled
LWIP_ALLOW_MEM_FREE_FROM_OTHER_...	Disabled
LWIP_RAM_HEAP_POINTER (RAM Heap Poin...	0x30020000

- (Should be done automatically by new CubeMX) In "Checksum" tab enable CHECKSUM_BY_HARDWARE. Other options should automatically reconfigure and you can leave them in this state.



Infrastructure - Checksum Options

CHECKSUM_BY_HARDWARE (Hardware C...	Enabled
LWIP_CHECKSUM_CTRL_PER_NETIF (Gen...	Disabled
CHECKSUM_GEN_IP (Generate Software Ch...	Disabled
CHECKSUM_GEN_UDP (Generate Software ...	Disabled
CHECKSUM_GEN_TCP (Generate Software ...	Disabled
CHECKSUM_GEN_ICMP (Generate Software...	Enabled
CHECKSUM_GEN_ICMP6 (Generate Softwar...	Disabled
CHECKSUM_CHECK_IP (Generate Software...	Disabled
CHECKSUM_CHECK_UDP (Generate Softw...	Disabled
CHECKSUM_CHECK_TCP (Generate Softwa...	Disabled
CHECKSUM_CHECK_ICMP (Generate Softw...	Enabled
CHECKSUM_CHECK_ICMP6 (Generate Soft...	Disabled

Generate project

Save project to some folder of your selection. Now you can generate the project for IDE. We will use STM32CubeIDE in this example, but it should work with other IDEs.

Modyfying the code (STM32CubeIDE)

There are several places where some additional code should be placed, also depending on selected device. In the examples all these places are marked with comment containing

```
ETH_CODE
```

and some basic explanation. Searching for

```
ETH_CODE
```

can show all these places.

Only main interesting points are mentioned below:

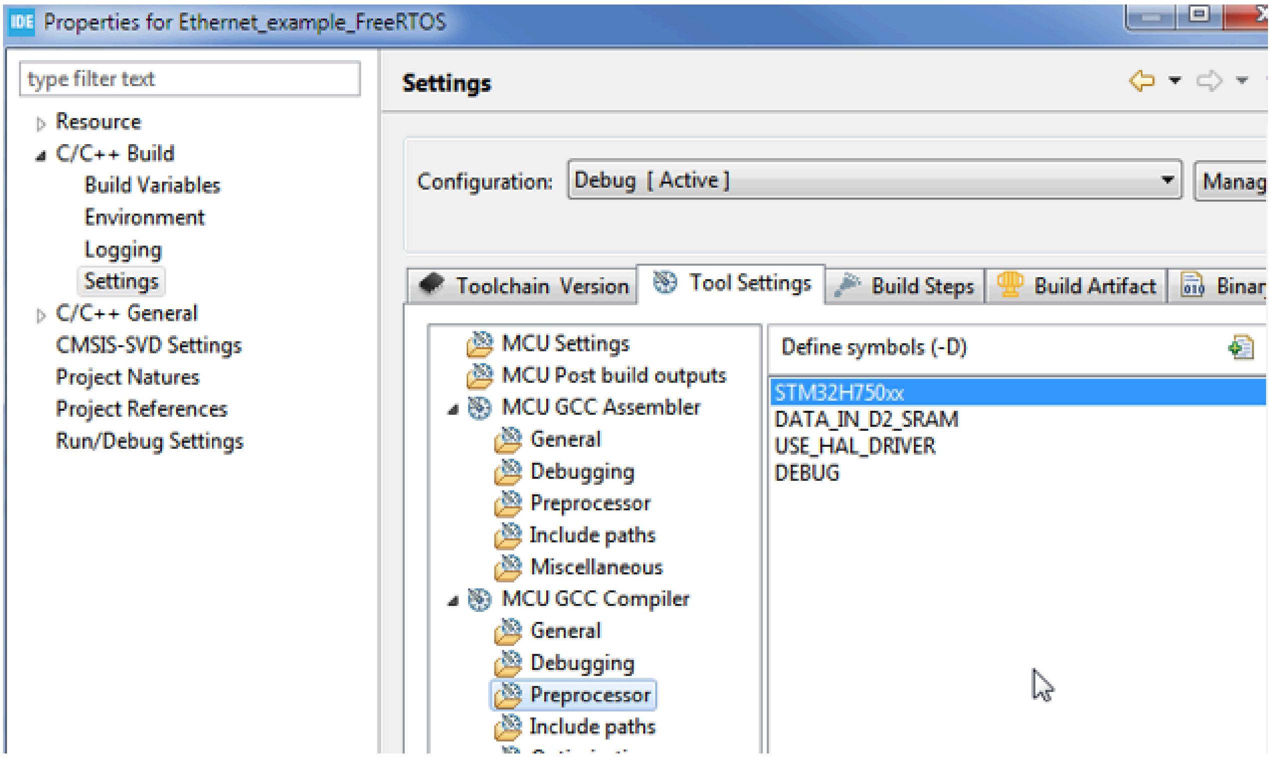
- Placement of the RX_POOL buffers (although we configured the address in CubeMX) in ethernetif.c[2]:

```
/* USER CODE BEGIN 2 */
#if defined ( __ICCARM__ ) /*!< IAR Compiler */
#pragma location = 0x30040200
extern u8_t memp_memory_RX_POOL_base[];

#elif defined ( __CC_ARM )  /* MDK ARM Compiler */
__attribute__((at(0x30040200)) 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 */
```

- In new STM32CubeIDE v1.9.0 the GCC was updated to version 10. This changes default handling of COMMON sections (https://gcc.gnu.org/gcc-10/changes.html). This can lead to compilation errors with errno variable being defined multiple times. This can happen when FreeRTOS + LwIP is enabled. To workaround this issue please put following code in lwipopts.h (https://github.com/STMicroelectronics/stm32_mw_lwip/issues/1

```
/* USER CODE BEGIN 1 /
#undef LWIP_PROVIDE_ERRNO
#define LWIP_ERRNO_STDINCLUDE
/ USER CODE END 1 */
```

- Add DATA_IN_D2_SRAM to macro definitions in project:



Configuration: Debug [Active]

Define symbols (-D)

```
STM32H750xx
DATA_IN_D2_SRAM
USE_HAL_DRIVER
DEBUG
```

Modify linkerscript (not valid for Keil/IAR)[2]

This step should be skipped for Keil and IAR, since they support placing variables at specific address in C code. Modify the linkerscript (*.ld) that the ETH descriptors and buffers are located in D2 SRAM. Also it is recommended to place all RAM to RAM_D1. In STM32CubeMX generated project, the "_FLASH" suffix linkerscript should be modified, which is used by default (e.g.: STM32H750XBHx_FLASH.ld). The "_RAM" suffix linkerscript is template for executing code from internal RAM memory.

```
  } >RAM_D1

  /* Modification start */
  .lwip_sec (NOLOAD) :
  {
    . = ABSOLUTE(0x30040000);
    *(.RxDecripSection)

    . = ABSOLUTE(0x30040060);
    *(.TxDecripSection)

    . = ABSOLUTE(0x30040200);
    *(.Rx_PoolSection)
  } >RAM_D2
  /* Modification end */

  /* Remove information from the compiler libraries */
  /DISCARD/ :
  {
    libc.a ( * )
    libm.a ( * )
    libgcc.a ( * )
  }
```

The memory definitions at the beginning of the linkerscript should look like:

```
MEMORY
{
  FLASH (rx)     : ORIGIN = 0x08000000, LENGTH = 128K
  DTCMRAM (xrw)  : ORIGIN = 0x20000000, LENGTH = 128K
  RAM_D1 (xrw)   : ORIGIN = 0x24000000, LENGTH = 512K
  RAM_D2 (xrw)   : ORIGIN = 0x30000000, LENGTH = 288K
  RAM_D3 (xrw)   : ORIGIN = 0x38000000, LENGTH = 64K
  ITCMRAM (xrw)  : ORIGIN = 0x00000000, LENGTH = 64K
}
```

For dual core devices it is better to restrict RAM_D2 section to avoid collision with Cortex-M4. Please see the linkerscripts in examples.

(Optional) Adding simple Hello UDP message

- Add following include files at the beginning of main.c:

```
#include “lwip/udp.h”
#include <string.h>
```

- Modify the StartDefaultTask in main.c with the following code:

```
/* USER CODE BEGIN 5 */
const char* message = "Hello UDP message!\n\r";

osDelay(1000);

ip_addr_t PC_IPADDR;
IP_ADDR4(&PC_IPADDR, 192, 168, 1, 1);

struct udp_pcb* my_udp = udp_new();
udp_connect(my_udp, &PC_IPADDR, 55151);
struct pbuf* udp_buffer = NULL;

/* Infinite loop */
for (;;) {
  osDelay(1000);
  /* !! PBUF_RAM is critical for correct operation !! */
  udp_buffer = pbuf_alloc(PBUF_TRANSPORT, strlen(message), PBUF_RAM);

  if (udp_buffer != NULL) {
    memcpy(udp_buffer->payload, message, strlen(message));
    udp_send(my_udp, udp_buffer);
    pbuf_free(udp_buffer);
  }
}
/* USER CODE END 5 */
```

Now you should be able to ping the device and receive UDP messages, assuming that you configure IP address 192.168.1.1 for the receiving device (the 192.168.1.0/24 network is used by attached examples). On Linux machine you can observe the messages with the following command:

```
netcat -ul 55151
```

Tips & common mistakes

1. For STM32H72x/H73x devices, the Ethernet buffers can't be placed in address range 0x30040000 - 0x30048000, since that range is not valid. D2 SRAM on those devices is much smaller, so the buffers need to be placed starting at 0x30000000. This affects RX & TX descriptors and RX buffer addresses (ETH configuration in CubeMX) and LWIP_RAM_HEAP_POINTER used for TX buffers (LWIP > Key options in CubeMX).

2. When running the stack on Cortex-M4, the buffers can be placed at the same address (0x30040000), but it is better to place them at 0x10040000 which is alias for the same address. This alias is accessible by Cortex-M4 D-bus and helps to utilize the Harvard architecture.
3. When not using FreeRTOS, the Ethernet interrupt should be disabled and MX_LWIP_Process should be called periodically (in main loop).
4. On STM32H747-Discovery board, modification needs to be done to default solder bridge configuration. SB8 should be closed and SB21 should be open for Ethernet to work, otherwise the MDC signal is not properly connected.

When facing issues with your own project:

1. First try example and see if there is proper configuration on PC side. With corporate firewalls and restrictions, it might be difficult to perform simple ping to specific IP address. In some cases it is easier to test from personal PC, or some PC with firewall disabled.
2. Check if Ethernet interrupt is called and if RX callback is called

    - If not, GPIOs are set to proper speed (very high)
    - and ETH global interrupt is enabled (only for FreeRTOS)

        - Interrupt priority should be 5 - preemption and 0 - subpriority. This is required by default FreeRTOS configuration.
3. If Hardfault is called, the problem might be in MPU configuration

    - Please check carefully (e.g. slight mistake like having "256KB" instead "256B" can make big difference) 1 RX callback is called but ping still not work
    - check that the buffers are properly placed in linkerscript (the one ending with "_FLASH.ld")
    - in STM32CubeIDE you can use "Build analyzer" window
4. When allocating buffers via pbuf_alloc (or similar), PBUF_RAM must be used as 3rd parameter. This is necessary to ensure that the allocated buffer is placed in D2 SRAM and synchronized with DMA
5. Not sufficient stack size for different thread can cause issues.

    - Enabling stack overflow detection can help identify where is the problem

Questions & Feedback

If you see any issue with these examples please fill an issue inside this repository. If you face some difficulties, but not sure if this is an issue you can start discussion inside this repository, or you can start thread on ST community

---

1. Exact size required for different stacks might depend on used compiler and optimization flags. Same goes for FreeRTOS heap size, since thread stacks are allocated from this heap.↩
2. Some addresses and sizes depend on the device or core used. Please refer to section Memory layout.↩↩↩↩

If you have any questions regarding Ethernet topics in general, please visit our MCU forum and create a thread. Doing so will provide you with a shorter response time.

STM32 MCU Products

1_STM32H7_ETH_Example_IDE1.6.1

STM32H7_ETH_Example_IDE1.6.1

7 Kudos

COMMENTS

JChar.3
Associate

2022-02-15 05:30 AM

It is nice to have the description, thanks. But normally I would expect CubeMX generate everything as needed - up to ping working, especially for distributed boards - currently talking about Nucleo H743ZI2. So embedded developers can concentrate on application development and not fight with any standard peripherals. I already spent many days in past years fighting with Eth on F7 and H7 boards - one problem being that mentioned LwIP_HTTP_Server_Netconn_RTOS have very different structure from code generated by CubeMX so it is almost impossible to merge these two (when LwIP_HTTP_Server_Netconn_RTOS works with some limitations, while default code from CubeMX does not).

I hope ST will move in that direction in next years, so CubeMX will generate everything as needed for Nucleo/Eval boards regarding Ethernet.

👍 3 Kudos

BBart.4
Associate

2022-03-23 02:28 AM

I followed the instructions and got the demo working on a Nucleo-H723ZG eval board. My setup was based on CubeMX V6.4.0 and STM32Cube FW_H7 V1.9.1.

Now, I updated to CubeMX V6.5.0 and it offered an update to STM32Cube FW_H7 V1.10.0. I migrated my project to the new firmware lib and now the demo does not work anymore, i.e. I had DHCP client and ping working before, but now I'm not getting an IP address via DHCP anymore.

There seem to be a lot of changes to the ethernet driver, e.g. the memory ranges for the RX / TX buffers, .RxArraySection is now complete gone. I guess that the MPU settings and the linker script need to be adjusted accordingly.

Can you please update the step-by-step guide and provide an updated example set?

At least for now, I'm going to stick with the old version and will not migrate my project.

👍 0 Kudos

heveskar
Senior

2022-04-06 12:07 AM

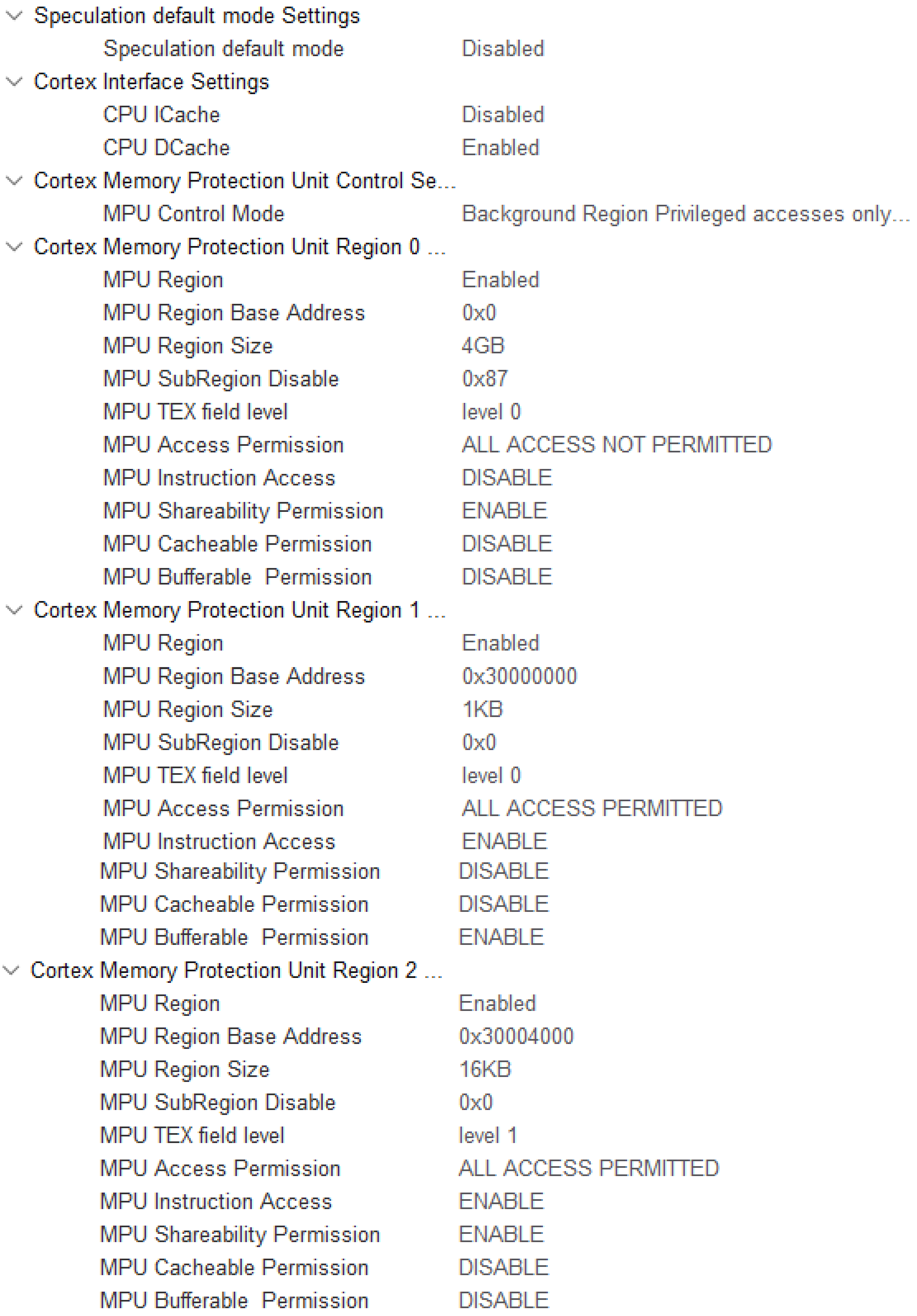
Hi @bernd Bartmann, I had also this problem, it seems there are not any working instructions for the new HAL version. What helped me was the example in *\STM32Cube\Repository\STM32Cube_FW_H7_V1.10.0\Projects\NUCLEO-*

H723ZG\Applications\LwIP\LwIP_HTTP_Server_Netconn_RTOS, there you can see how to set the addresses so that it is working. Apart from moving the addresses, one of the sections was also renamed (RxArraySection -> Rx_PoolSection).

In the linker file:

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

     . = ABSOLUTE(0x30000200);
     *(.TxDecripSection)

     . = ABSOLUTE(0x30000400);
     *(.Rx_PoolSection)
   } >RAM_D2 AT> ROM
```

In the MPU settings:



```
∨ Speculation default mode Settings
      Speculation default mode              Disabled
∨ Cortex Interface Settings
      CPU ICache                            Disabled
      CPU DCache                            Enabled
∨ Cortex Memory Protection Unit Control Se...
      MPU Control Mode                      Background Region Privileged accesses only...
∨ Cortex Memory Protection Unit Region 0 ...
      MPU Region                            Enabled
      MPU Region Base Address               0x0
      MPU Region Size                       4GB
      MPU SubRegion Disable                 0x87
      MPU TEX field level                   level 0
      MPU Access Permission                 ALL ACCESS NOT PERMITTED
      MPU Instruction Access                DISABLE
      MPU Shareability Permission           ENABLE
      MPU Cacheable Permission              DISABLE
      MPU Bufferable  Permission            DISABLE
∨ Cortex Memory Protection Unit Region 1 ...
      MPU Region                            Enabled
      MPU Region Base Address               0x30000000
      MPU Region Size                       1KB
      MPU SubRegion Disable                 0x0
      MPU TEX field level                   level 0
      MPU Access Permission                 ALL ACCESS PERMITTED
      MPU Instruction Access                ENABLE
      MPU Shareability Permission           DISABLE
      MPU Cacheable Permission              DISABLE
      MPU Bufferable  Permission            ENABLE
∨ Cortex Memory Protection Unit Region 2 ...
      MPU Region                            Enabled
      MPU Region Base Address               0x30004000
      MPU Region Size                       16KB
      MPU SubRegion Disable                 0x0
      MPU TEX field level                   level 1
      MPU Access Permission                 ALL ACCESS PERMITTED
      MPU Instruction Access                ENABLE
      MPU Shareability Permission           ENABLE
      MPU Cacheable Permission              DISABLE
      MPU Bufferable  Permission            DISABLE
```

And finally in ethernetif.c, there is missing setting the rx pool base to the correct address:

```
#if defined ( __ICCARM__ ) /*!< IAR Compiler */
#pragma location = 0x30000400
extern u8_t memp_memory_RX_POOL_base[];

#elif defined ( __CC_ARM )  /* MDK ARM Compiler */
__attribute__((section(".Rx_PoolSection"))) 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
```

I have taken those things directly from the example.

Hope that this helps

Karel

0 Kudos

BBart.4

Associate

2022-04-07 01:15 AM

Thanks a lot Karel! This made it working again for me as well.

The only thing I need to figure out is an assert that I'm getting during startup:

Assertion "pbuf_free: p->ref > 0" failed at line 753 in ../Middlewares/Third_Party/LwIP/src/core/pbuf.c

I'll debug this further.

Best regards,

Bernd

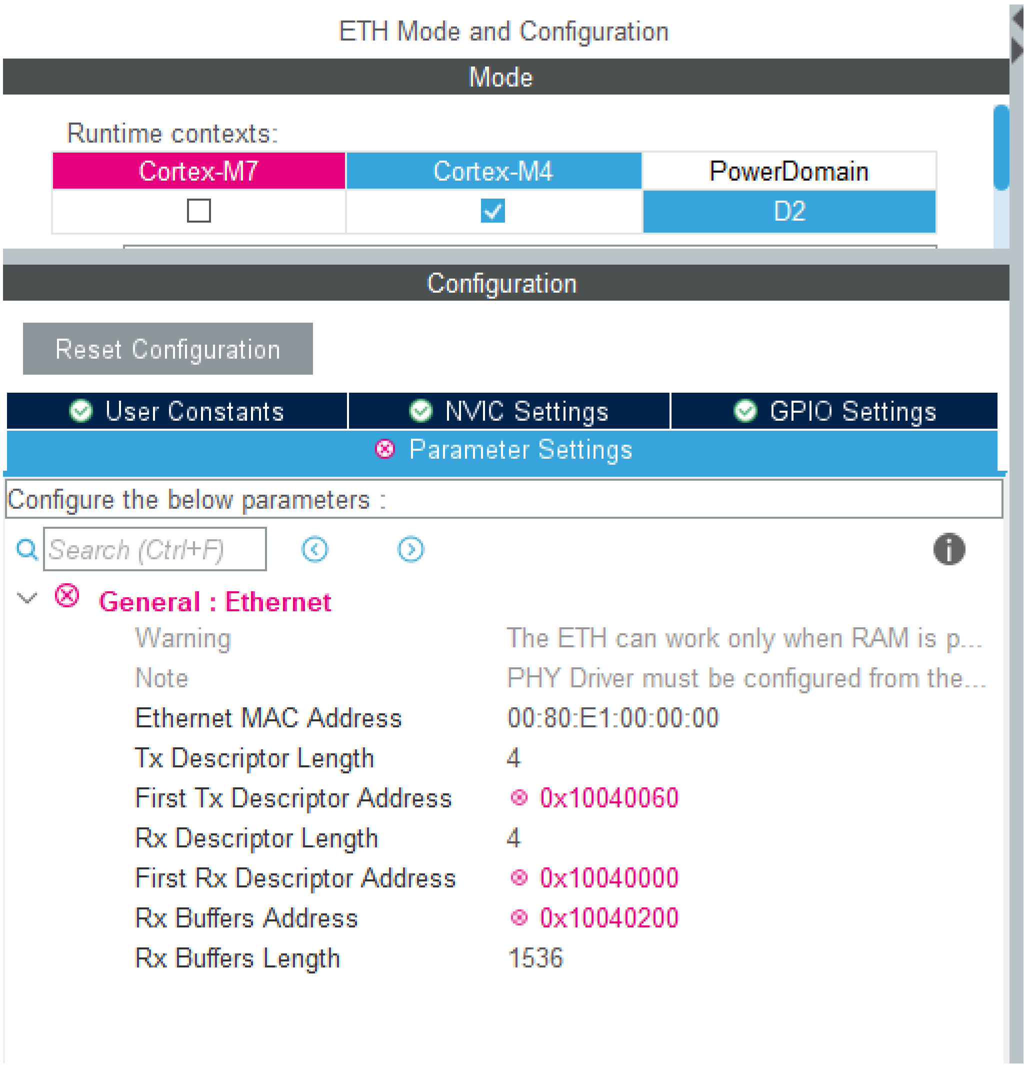
0 Kudos

lOvch
Associate II

2022-06-17 04:30 AM

Starting with the **STM32 Cube FW_H7 1.10.0** version, it becomes impossible to specify the correct location of the ethernet descriptors and buffers officially recommended here in paragraph 10.2 for CortexM4.

Migrating example (p.11. Attached examples) *STM32H745_Nucleo_M4_ETH.ioc* from 1.9 to 1.10 gives the result shown in the picture.



Runtime contexts:

Cortex-M7	Cortex-M4	PowerDomain
□	☑	D2

Configure the below parameters :

General : Ethernet

Warning	The ETH can work only when RAM is p...
Note	PHY Driver must be configured from the...
Ethernet MAC Address	00:80:E1:00:00:00
Tx Descriptor Length	4
First Tx Descriptor Address	☒ 0x10040060
Rx Descriptor Length	4
First Rx Descriptor Address	☒ 0x10040000
Rx Buffers Address	☒ 0x10040200
Rx Buffers Length	1536

If you are trying to run Ethernet on Cortex M4 while having all it in the M7 region (0x3000000), it still works. But it works very bad and unstable, even a regular ping test gives from 3 to 20% of failed packets, and if you increase the size of the ping packet, it falls immediately.

So... can you please you please update your examples to current STM32 Cube FW_H7 version ?

0 Kudos

Adam BERLINGER

ST Employee

2022-06-17 07:42 AM

Hello @IOvch ,

thank you for your feedback.

I have in plan to update this material to be compatible with new release of STM32Cube_FW_H7. The Ethernet driver in v1.10.0 was reworked, so there could be some other catches in the configuration.

However the issue you describe is quite strange. Are you sure that the memory used for Ethernet (0x30000000) is not used by the program for other purposes through the alias (0x10000000)?

There could be some slow down, but I would expect it to fail completely.

Best regards,

Adam Berlinger

👍 **0 Kudos**

IOvch

Associate II

2022-06-20 09:50 PM

Hello @Adam BERLINGER !

Thank you for your reply!

I didn't implement anything else - only ethernet on M4 core. You can try to create a new empty project to run **ethernet on M4 with STM32 Cube FW_H7 1.10.0**. And this does not work at all when used in RTOS mode.

👍 **0 Kudos**

IOvch

Associate II

2022-06-30 12:02 AM

@Adam BERLINGER It seems to be a more serious problem than just a descriptors location:

https://community.st.com/s/question/0D53W00001VYkb5SAD/stm32f417-lwip-udp-packets-are-being-truncated

👍 **0 Kudos**

Adam BERLINGER

ST Employee

2022-06-30 07:13 AM

Hello @IOvch ,

the community thread you linked is regarding STM32F4 devices, which use different Ethernet driver, so I don't this is relevant for STM32H7. I also roughly checked in H7 driver code and I don't see the same issue(s).

I added new section "**10. STM32Cube_FW_H7_V1.10.0 (and newer) specific issues**" maybe it will help with your issue.

Best regards,

Adam Berlinger

👍 **0 Kudos**

Scott Feister

Scott Feister

Associate

2022-07-01 08:35 AM

Hi Laura, I have really appreciated the work you've done here! I am wondering if you could shed some light on the "32KB" region size for the MPU region? I would have expected that it would be a 288KB region size, to encapsulate the entire D2 SRAM. I am working to develop a better understanding of this as I am trying to blend this knowledge with the article for getting DMA to work with the Data Cache Enabled: Knowledge Article "DMA is not working on STM32H7 devices" -- and so I am adding a ".dma_buffer" region to the modifications described here with the ".lwip_sec" region. Thanks!

👍 0 Kudos

LauraCx
ST Employee

2022-07-01 10:08 AM

Hi @Scott F,

@Adam BERLINGER is the author of the article, he will be able to help you!

👍 0 Kudos

ktrofimo
Senior III

2022-07-02 04:15 AM

Exactly the same problem :(

👍 0 Kudos

ktrofimo
Senior III

2022-07-02 04:16 AM

Please, check default value for:

#define ETH_RX_BUFFER_SIZE   1000

looks like it should be 1536 (please, see comment from Alexander Kvashin at https://community.st.com/s/question/0D73W000001OsEOSA0/detail?s1oid=00Db0000000YtG6&s1nid=0DB0X000000DYbd&emkind=chatterCommentNotification&s1uid=0050X000007vIPS&emtm=1656688370859&fromEmail=1&s1ext=0) as well as all discussion regarding DMA errors

0 Kudos

IOvch
Associate II

2022-07-04 03:57 AM

Hello @Adam BERLINGER ,

thanks for your reply and updating the article!

But I'm talking about ethernet running on the **M4 CORE** as part of STM32H7 with new version of FW_H7_V1.10 in RTOS mode. On the M7 core it works as is... and on the M4 its not working even with your last reccomendations.

Your attached examples are perfect, but can you please update just one of them? Just a simple ping test on **M4 CORE** of STM32H7, please! It would remove SO MUCH questions at once! Such as:

1. Do I need to configure MPU in M7 when using Ethernet on **M4 core**?
2. You have write "LWIP_RAM_HEAP_POINTER set to **0x30044B00**" not shure, but maybe you mean **0x30044800** instead? Cause its equals to 18kb.
3. You have write "This should ensure that all Ethernet specific buffers/descriptors are placed in **SRAM3**". I thought that we use only **RAM_D2** for it.
4. What means "MEM_SIZE12000"? Should I change something?
5. What means "size of the memory pool is roughly 1.5kB × 12 = 18kB"? It is a **MEM_SIZE (Heap Memory Size)**? So, should I still subtract 24 bytes for allocator metadata

6. Should I change PBUF_POOL_SIZE and PBUF_POOL_BUFSIZE? Or just keep in mind that the number of RX buffers is fixed to 12?

In the link I sent, in comments, It talks exactly about the poor transfer of the Ethernet driver of the M7 core to the M4 as part of the STM32H7 in FW_H7_V1.10. There is even an internal ticket number 126954. But if this has nothing to do with this - Ok.

I just want to make ethernet work on M4 of STM32H7 with FW_H7_V1.10 and FreeRTOS.

Best regards!

IOvch

0 Kudos

Noti

Associate

2022-07-07 07:04 AM

Very nice article. It has helped me a lot in order to be able to build a working Ethernet project.

However, I'd like to point a possible bug I've found just in case it avoids some headache to anyone else. If you run the app and unplug and plug the Ethernet cable back, it stops working. Modifying a line towards the end of ethernet_link_thread() function (in ethernetif.c) makes it work correctly:

```
if(linkchanged)
    {
      /* Get MAC Config MAC */
      HAL_ETH_GetMACConfig(&heth, &MACConf);
      MACConf.DuplexMode = duplex;
      MACConf.Speed = speed;
      HAL_ETH_SetMACConfig(&heth, &MACConf);
      HAL_ETH_Start(&heth);     // <-------------------   This should be HAL_ETH_Start_IT(&heth);
      netif_set_up(netif);
      netif_set_link_up(netif);
    }
```

I'm using STM32CubeIDE 1.10.0 and STM32CubeMX 6.6.0 with STM32H745 MCU.

0 Kudos

heveskar
Senior

2022-07-08 12:44 AM

Hi, @Adam BERLINGER, I second the request for comment about points 2, 3 and 4. As @IOvch mentions, in the text you first want us to set **LWIP_RAM_HEAP_POINTER** to *0x30044000* and **MEM_SIZE** to *16360*, then in the last generic section you say

```
And LWIP_RAM_HEAP_POINTER set to 0x30044B00, with MEM_SIZE12000
```

So, which one is correct? For me the first set of values worked.

Regards,

Karel

0 Kudos

Noti
Associate

2022-07-08 08:50 AM

Hi all,

```
You have write "LWIP_RAM_HEAP_POINTER set to 0x30044B00" not shure, but maybe you me
an 0x30044800 instead? Cause its equals to 18kb.
```

In the new section 10, Adam explains it. The size **of the memory pool** is 1536 bytes x 12 = 18432 bytes = 18 KB = 0x4800. Because the Rx_PoolSection section (memory pool) is located at 0x30040200:

```
.lwip_sec (NOLOAD) :
  {
    . = ABSOLUTE(0x30040000);
    *(.RxDecripSection)

    . = ABSOLUTE(0x30040060);
    *(.TxDecripSection)

    . = ABSOLUTE(0x30040200);
    *(.Rx_PoolSection)
  } >RAM_D2
```

if you add 0x4800 (18KB), you get 0x30044A00. This is the position where the memory pool ends. I think Adam has chosen the next "round" value of 0x30044B00, making 0x100 = 256 bytes of gap between this pool memory and the lwIP RAM heap pointer. So this lwIP RAM heap is placed at 0x30044B00.

Then, because SRAM3 ends at 0x30048000, you have a free memory space of 0x30048000 - 0x30044B00 = 0x3500 = 13568 bytes = 13,2KB. I think Adam prefers to be "safe" and use only 12 KB of that memory. That's why he says that you should set a MEM_SIZE of 12000 (Section 7. lwIP configuration, "...in key options tab configure MEM_SIZE to ~~16360~~...)

```
You have write "This should ensure that all Ethernet specific buffers/descriptors are placed in SRAM3". I thought that we use only RAM_D2 for it.
```

I think you get domains and RAMs mixed up. RAM_D2 (Domain 2) is composed of SRAM1, SRAM2 and SRAM3.

Best regards,

Noti.

👍 0 Kudos

Ec1

Associate II

2022-07-25 12:50 AM

It says 192.168.1.10 IP address is used in the LwIP configuration. Is this IP address the IP address it gives to the kit it uses? Or is it the PC!'s IP address? and according to which notation it gives this IP address.

👍 0 Kudos

IOvch
Associate II

2022-07-26 12:51 AM

Hello! @Adam BERLINGER @David Notivoli @heveskar

During all this time, I have not been able to run ethernet on the M4 core under FreeRTOS. I'm not doing anything superfluous - just an new empty project with HAL **FW_H7 1.10.0**. Surprisingly, ethernet works on the **M4 CORE** out of the box even without any changes (except for increasing the GPIO speed).

But... only **without FreeRTOS**. If you add it, then none of the manipulations suggested here or in neighboring branches help.

I also noticed that with FreeRTOS for M4, it generates in **ethernetif.c** file a call to **SCB_InvalidateDCache_by_Addr** that cannot be executed, so I'm just removing that call...

Maybe in this case it is necessary to configure the MPU on both cores?

I have attached two projects - there is only basic functionality, nothing but ping. They are almost compatible with Nucleo-745 except for a couple of pins. It works without FreeRTOS (**M4 core**), but if you added it.. nothing helps.

I have lost hope for a working ethernet example on a new version of the libraries. But maybe there will be some ideas?

👍 0 Kudos

heveskar
Senior

2022-07-28 06:52 AM

@IOvch Unfortunately I do not have a board with both cores, but for FreeRTOS, wouldn't help you replacing

```
#define INTERFACE_THREAD_STACK_SIZE ( 350 )
```

in ethernetif.c with

```
#define INTERFACE_THREAD_STACK_SIZE ( 512 )
```

I do not know if it will help you, but I have discovered that 350 is not enough.

0 Kudos

IOvch
Associate II

2022-07-28 11:20 PM

@heveskar Thanks for your advice! It didn't help, but I'll use it by default in the future

I think some deeper MPU setup is needed here. But... it works without RTOS, so it's not a fact that this is the case.

0 Kudos

FBelt.1
Associate II

2022-08-01 07:22 AM

Hi,

if I try to modify ETH_RX_BUFFER_SIZE to 1536 or 1524 project doesn't work, even if the memory is always rigth placed and aligned.

Anyone knows why this value is 1000?

0 Kudos

MDega.2
Associate II

2022-08-04 01:50 AM

Hi,

Is there a reason to select the clock at 400MHz, instead of the maximum allowed?

I am using H723 and the clock can be 550MHz, do I have to reduce it to 400?

0 Kudos

Adam BERLINGER
ST Employee

2022-08-10 04:04 AM

Hello @ktrofimo and @FBelt.1 ,

The ETH_RX_BUFFER_SIZE should be set to 1536. Sometimes (for unknown reason to me) the STM32CubeMX selects 1524. So far I haven't faced 1000 value on STM32H7.

I know there are were some issues on F4 family where this was hardcoded to the driver.

If this issues is not related to CubeMX configuration, could you be please more specific? Maybe I misunderstood.

Adam

0 Kudos

Adam BERLINGER
ST Employee

2022-08-10 04:16 AM

Hello @Ec ,

the 192.168.1.10 IP is for the board. I always configure 192.168.1.1 on the PC side, however different IP address from the subnet could be also viable.

0 Kudos

victagayun

Senior III

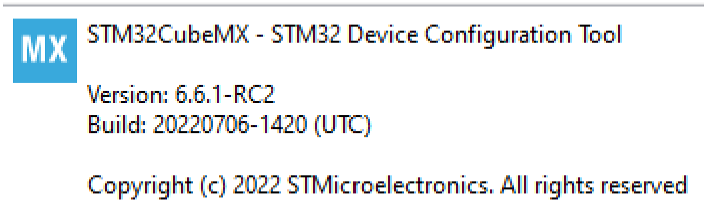
2022-08-13 06:57 AM

Hi,

Newbie here for ETH.

Seems that my Nucleo-H743ZI is not showing in my router.

My setup is Win10 with the ff info:



STM32CubeMX - STM32 Device Configuration Tool

Version: 6.6.1-RC2
Build: 20220706-1420 (UTC)

Copyright (c) 2022 STMicroelectronics. All rights reserved

I have modified the code like so:

in lwipopts.h

```
/* USER CODE BEGIN 1 /
#undef LWIP_PROVIDE_ERRNO
#define LWIP_ERRNO_STDINCLUDE
#define LWIP_RAM_HEAP_POINTER (0x30044000)
/ USER CODE END 1 */
```

in ethernetif.c

```
/* USER CODE BEGIN 2 */
#if defined ( __ICCARM__ ) /*!< IAR Compiler */
#pragma location = 0x30040200
extern u8_t memp_memory_RX_POOL_base[];

#elif defined ( __CC_ARM )  /* MDK ARM Compiler */
__attribute__((at(0x30040200)) 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 */
```

in LWIP.c

```
IP_ADDRESS[0] = 192;
  IP_ADDRESS[1] = 168;
  IP_ADDRESS[2] = 50;
  IP_ADDRESS[3] = 10;
  NETMASK_ADDRESS[0] = 255;
  NETMASK_ADDRESS[1] = 255;
  NETMASK_ADDRESS[2] = 255;
  NETMASK_ADDRESS[3] = 0;
  GATEWAY_ADDRESS[0] = 192;
  GATEWAY_ADDRESS[1] = 168;
  GATEWAY_ADDRESS[2] = 50;
  GATEWAY_ADDRESS[3] = 1;
```

My STM32H743ZITX_FLASH.ld

```
/* Entry Point */
ENTRY(Reset_Handler)

/* Highest address of the user mode stack */
_estack = 0x24080000;    /* end of RAM */
/* Generate a link error if heap and stack don't fit into RAM */
_Min_Heap_Size = 0x200 ;      /* required amount of heap  */
_Min_Stack_Size = 0x400 ; /* required amount of stack */

/* Specify the memory areas */
MEMORY
{
  FLASH (rx)     : ORIGIN = 0x08000000, LENGTH = 2048K
  RAM (xrw)      : ORIGIN = 0x20000000, LENGTH = 128K
  RAM_D1 (xrw)   : ORIGIN = 0x24000000, LENGTH = 512K
  RAM_D2 (xrw)   : ORIGIN = 0x30000000, LENGTH = 288K
  RAM_D3 (xrw)   : ORIGIN = 0x38000000, LENGTH = 64K
  ITCMRAM (xrw)  : ORIGIN = 0x00000000, LENGTH = 64K
}

/* Define output sections */
SECTIONS
{
  /* The startup code goes first into FLASH */
  .isr_vector :
  {
    . = ALIGN(4);
    KEEP(*(.isr_vector)) /* Startup code */
    . = ALIGN(4);
  } >FLASH

  /* The program code and other data goes into FLASH */
  .text :
  {
    . = ALIGN(4);
    *(.text)           /* .text sections (code) */
    *(.text*)          /* .text* sections (code) */
    *(.glue_7)         /* glue arm to thumb code */
    *(.glue_7t)        /* glue thumb to arm code */
    *(.eh_frame)

    KEEP (*(.init))
    KEEP (*(.fini))

    . = ALIGN(4);
    _etext = .;        /* define a global symbols at end of code */
  } >FLASH

  /* Constant data goes into FLASH */
  .rodata :
  {
    . = ALIGN(4);
    *(.rodata)         /* .rodata sections (constants, strings, etc.) */
    *(.rodata*)        /* .rodata* sections (constants, strings, etc.) */
    . = ALIGN(4);
  } >FLASH

  .ARM.extab   : { *(.ARM.extab* .gnu.linkonce.armextab.*) } >FLASH
  .ARM : {
    __exidx_start = .;
    *(.ARM.exidx*)
    __exidx_end = .;
  } >FLASH

  .preinit_array     :
  {
    PROVIDE_HIDDEN (__preinit_array_start = .);
    KEEP (*(.preinit_array*))
    PROVIDE_HIDDEN (__preinit_array_end = .);
  } >FLASH
  .init_array :
  {
    PROVIDE_HIDDEN (__init_array_start = .);
    KEEP (*(SORT(.init_array.*)))
    KEEP (*(.init_array*))
    PROVIDE_HIDDEN (__init_array_end = .);
  } >FLASH
  .fini_array :
  {
    PROVIDE_HIDDEN (__fini_array_start = .);
    KEEP (*(SORT(.fini_array.*)))
    KEEP (*(.fini_array*))
    PROVIDE_HIDDEN (__fini_array_end = .);
  } >FLASH

  /* used by the startup to initialize data */
  _sidata = LOADADDR(.data);

  /* Initialized data sections goes into RAM, load LMA copy after code */
  .data :
  {
    . = ALIGN(4);
    _sdata = .;        /* create a global symbol at data start */
  *(.data)           /* .data sections */
  *(.data*)          /* .data* sections */
  *(.RamFunc)        /* .RamFunc sections */
  *(.RamFunc*)       /* .RamFunc* sections */

  . = ALIGN(4);
  _edata = .;        /* define a global symbol at data end */
} >RAM_D1 AT> FLASH

/* Uninitialized data section */
. = ALIGN(4);
.bss :
{
  /* This is used by the startup in order to initialize the .bss secion */
  _sbss = .;         /* define a global symbol at bss start */
  __bss_start__ = _sbss;
  *(.bss)
  *(.bss*)
  *(COMMON)

  . = ALIGN(4);
  _ebss = .;         /* define a global symbol at bss end */
  __bss_end__ = _ebss;
} >RAM_D1

/* User_heap_stack section, used to check that there is enough RAM left */
._user_heap_stack :
{
  . = ALIGN(8);
  PROVIDE ( end = . );
  PROVIDE ( _end = . );
  . = . + _Min_Heap_Size;
  . = . + _Min_Stack_Size;
  . = ALIGN(8);
} >RAM_D1

.lwip_sec (NOLOAD) : {
  . = ABSOLUTE(0x30040000);
  *(.RxDecripSection)

  . = ABSOLUTE(0x30040060);
  *(.TxDecripSection)

  . = ABSOLUTE(0x30040200);
  *(.RxArraySection)
} >RAM_D2
```

```
  /* Remove information from the standard libraries */
  /DISCARD/ :
  {
    libc.a ( * )
    libm.a ( * )
    libgcc.a ( * )
  }

  .ARM.attributes 0 : { *(.ARM.attributes) }
}
```

Did I miss something?

Only the orange LED in the LAN port is blinking.

Can you possibly add UART for debug messages and/or the LED for indication?

0 Kudos

gpmaier

Associate

2022-08-17 04:54 AM

I also tried to migrate based on the How to here. Project set up with StmCube, then migrated to 1.10.x.

As with you both I also get one or some

```
Assertion "pbuf_free: p->ref > 0" failed at line 753 in ../Middlewares/Third_Party/L
wIP/src/core/pbuf.c
```

at startup on both a

- STM32H742 (descriptors and pool address are "vanilla" -> 0x30000000 and so on; heap at 0x30002000 because of the 742's limited memory) and a
- STM32H743 (here heap at 0x30008000).

I tried to debugging. No luck for 2, 3 days.

Aside from this (or because of this) the Ethernet communication is very instable.

I made the observation that - also at startup - some allocated RX pBufs in ethernetif.c never will be deallocated and therefore sometimes the buffer pool is full (12, see ETH_RX_BUFFER_CNT) and I get communication issues at startup.

Have you managed the issue for now @Bernd Bartmann , @ktrofimo ?

👍 **0 Kudos**

Andrei Chichak

Lead

2022-08-17 11:10 AM

I believe that there are some issues with the MPU setup and the linker file locations for the TX descriptor section. Perhaps I don't understand some of the details though.

A ETH_DMADescTypeDef is 40 bytes, 10 integers of 4 bytes each, and DMARxDscrTab has 4 of them for 160 bytes or 0xA0. DMATxDscrTab is the same 160 bytes or 0xA0. These are placed via the info in the linker file at specific start locations.

So if DMARxDscrTab is placed by the linker file at 0x3004 0000 it will extend to 0x3004 009F,

DMATxDscrTab is placed at 0x3004 0060 for 0xA0 bytes, which is overlapping with DMARxDscrTab.

Ooops?

The MPU setup declares a 32k region starting t 0x3004 0000, and another with the same start address for 256B to contain the DMA descriptors. This second region would extend to 0x3004 00FF. The transmit and receive descriptors are a total of 160+160 = 320 bytes, so the 256B region won't apply the appropriate shareability or bufferabiltiy to the complete transmit descriptor table. A better length for the second region is 512B.

The receiver pool placement seems to be okay, but I don't understand what the LWIP_RAM_HEAP_POINTER is all about and what protections the RX Pool requires. There is a comment "**And LWIP_RAM_HEAP_POINTER set to 0x30044B00, with MEM_SIZE12000. This should ensure that all Ethernet specific buffers/descriptors are placed in SRAM3.**" But those values would have the heap completely in SRAM2. SRAM3 doesn't begin until 0x3008 0000, far above the 0x3004 79E0 end address of the pool. Can you clarify this?

👍 0 Kudos

PGrif.2
Associate III

2022-08-20 11:23 PM

since "Your administrator has disabled feed post and comment deletions."

please see below ..

👍 0 Kudos

PGrif.2
Associate III

2022-08-20 11:25 PM

Hi there!

I am using the current provided CubeIDE (STM32CubeIDE Version: 1.10.1 Build: 12716_20220707_0928 (UTC)) CubeMX 6.6.1 and H7 Firmware package F1.10.0

I've used the above step by step instruction to get the

- FreeRTOS
- LWIP
- as TCP Server

running on a STM32H723ZG Nucleo-Board including the described fixes

- 10. STM32Cube_FW_H7_V1.10.0 (and newer) specific issues
- ethernet_link_thread() by @Noti
- btw I agree with @JChar.3

Flash.ld file was changed to

```
MEMORY
{
 ITCMRAM (xrw)  : ORIGIN = 0x00000000, LENGTH = 64K
 DTCMRAM (xrw)  : ORIGIN = 0x20000000, LENGTH = 128K
 FLASH  (rx)  : ORIGIN = 0x08000000, LENGTH = 1024K
 RAM_D1 (xrw)  : ORIGIN = 0x24000000, LENGTH = 320K
 RAM_D2 (xrw)  : ORIGIN = 0x30000000, LENGTH = 288K
 RAM_D3 (xrw)  : ORIGIN = 0x38000000, LENGTH = 16K
}
```

and

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

  . = ABSOLUTE(0x30000060);
  *(.TxDecripSection)



  . = ABSOLUTE(0x30000200);
  *(.Rx_PoolSection)

 } >RAM_D2
 /* Modification end */
```

LWIP_RAM_HEAP_POINTER set to 0x30004B00, with MEM_SIZE12000

Testing is done with termial program Hercules with serveral ping and connect/diconnect cycles. Start up and connect diconnect run smooth at first, but after a while (or a unplug/replug of Ethernet cable) the device will still ping but refused connection.

Test:

Sending ICMP ECHO REQUEST to module <- frist ping on start up

Received ICMP ECHO REPLY

Connecting to 172.16.17.100 ...

Connected to 172.16.17.100 <- frist connect OK

Connection closed

[.. will continue to be OK for the next 20+ connections]

Connecting to 172.16.17.100 ...

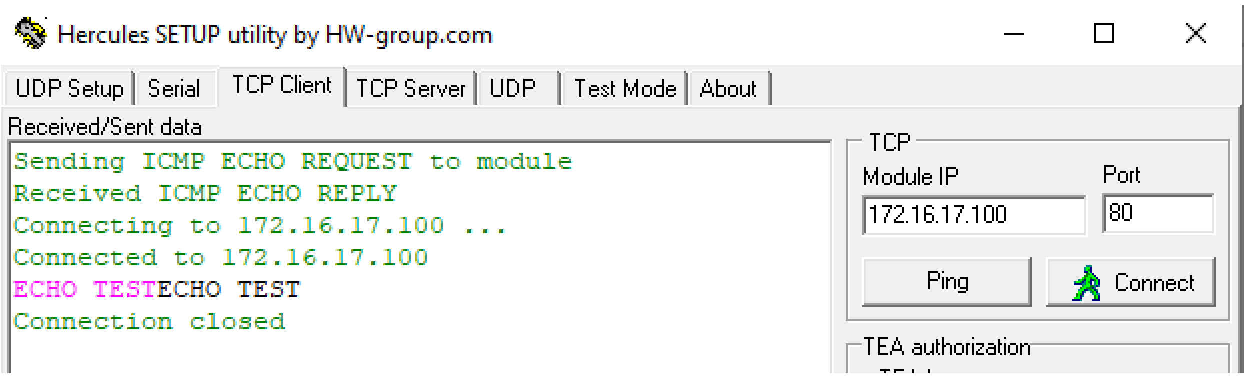
```
Connected to 172.16.17.100

// diconnect cable

Connection refused by remote host        <- terminal recognized disconnect OK

Connection closed

// reconnect cable

Sending ICMP ECHO REQUEST to module <- frist ping on reconnect OK

Received ICMP ECHO REPLY

Connecting to 172.16.17.100 ...

Connected to 172.16.17.100           <- frist connect OK

Connection closed

Connecting to 172.16.17.100 ...

Connected to 172.16.17.100            <- second connect OK

Connection closed

Connecting to 172.16.17.100 ...

Connected to 172.16.17.100            <- third connect FAIL

Connection refused by remote host

Connection closed

Sending ICMP ECHO REQUEST to module <- but ping still works

Received ICMP ECHO REPLY
```

I think it might be some MPU setup or MEMORY assignment issue.

For convienece I'va added a minimal example

address is 172.16.17.100 Port 80

ping, connect and send a string (e.g. ECHO TEST) the server will reply the string and diconnect



```
Sending ICMP ECHO REQUEST to module
Received ICMP ECHO REPLY
Connecting to 172.16.17.100 ...
Connected to 172.16.17.100
ECHO TESTECHO TEST
Connection closed
```

Module IP: 172.16.17.100
Port: 80

Any help would be highly appreciated!

Regards!

0 Kudos

EOzde.1

Associate III

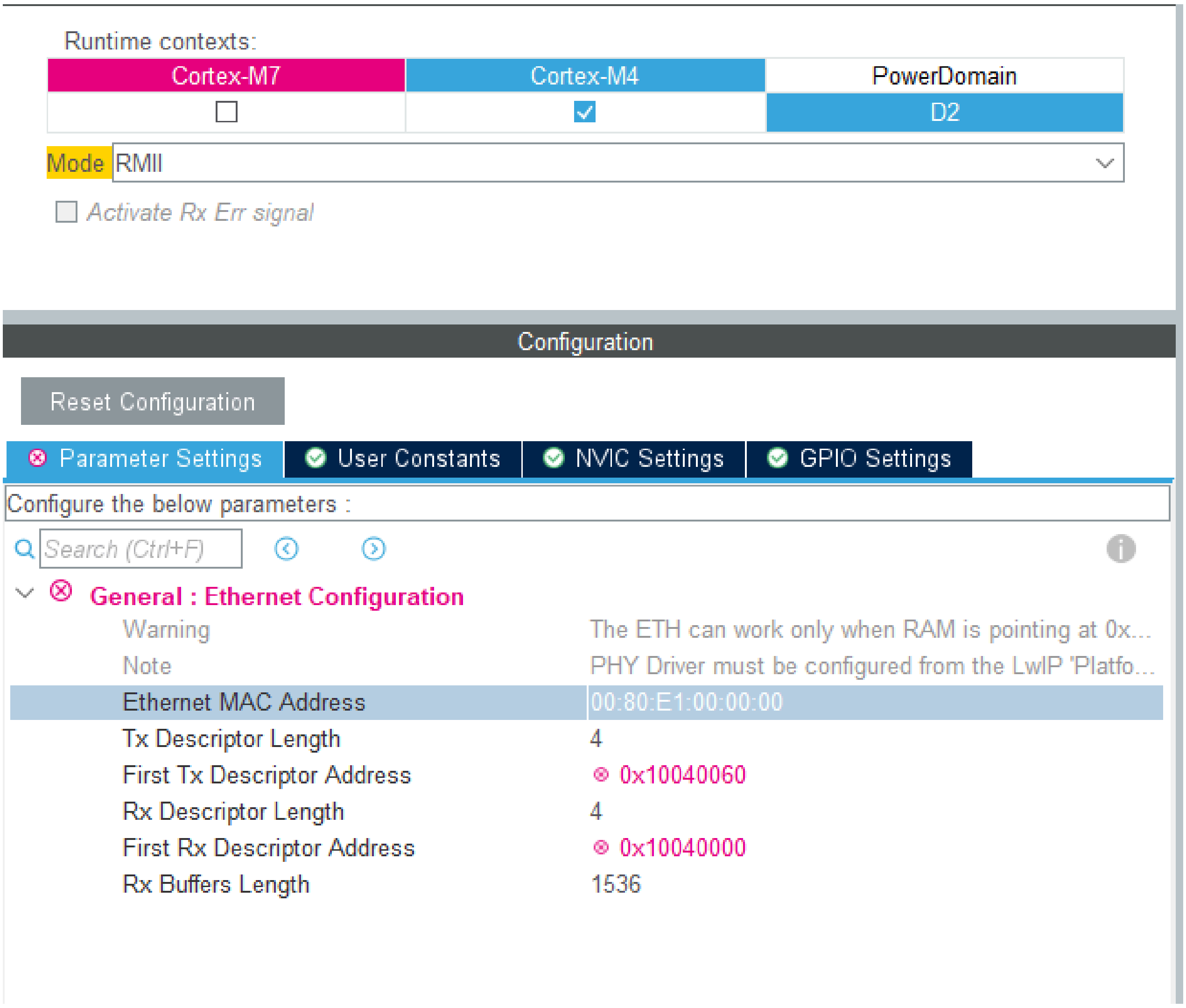
2022-08-23 02:09 AM

Hi,

After migrating my project to 1.10.1, I have made the necessary changes as this article describes.

But here is the picture of the cubemx perspective. I cannot figure out what to do to fix it.

Have a nice day.



Runtime contexts:

Cortex-M7	Cortex-M4	PowerDomain
□	☑	D2

Mode RMII

□ *Activate Rx Err signal*

Configuration

Configure the below parameters :

⊗ General : Ethernet Configuration

Warning	The ETH can work only when RAM is pointing at 0x...
Note	PHY Driver must be configured from the LwIP 'Platfo...
Ethernet MAC Address	00:80:E1:00:00:00
Tx Descriptor Length	4
First Tx Descriptor Address	⊗ 0x10040060
Rx Descriptor Length	4
First Rx Descriptor Address	⊗ 0x10040000
Rx Buffers Length	1536

0 Kudos

JYI

Associate II

2022-11-04 08:38 AM

This article is very helpful with MPU and _FLASH.LD settings to make project running on NUCLEO-H723ZG without freeROTS for Ethernet.

The link looks working since plugging and unplugging cable indicating "Link Up" and "Link Down".

However, the Ping is not working.

The setting LWIP_ICMP is 1.

Not sure any functions need to be called after MX_LWIP_Init() to make Ping working?

JYI

👍 0 Kudos

Riscy
Senior

2022-11-19 01:04 PM

I was attempted to apply a similar method to Nucleo-F746ZG without FreeRTOS, just UDP transmission and it did not work. Shall I send the attached project? I'm using Cube 1.10.1, STM32Cube FW_F7 V1.17.0 and most recently CubeMX. I do note, it is a single-core M7 and a single RAM bank. I have placed the RAM within the linker and set 192.168.41.5 as IP address which did not appear on Wireshark. I also have WIndow IPV4 setting to 192.168.41.42 with subnet mask 255.255.255.0. It did not respond to the ping test, and no UDP emitted from 500mSec loop.

👍 0 Kudos

Riscy
Senior

2022-11-19 01:05 PM

In addition: the hardware is working fine via earlier code (System Workspace) which means my setup is working and also tested with ethernet STM demo code.

👍 0 Kudos

XLei.1
Associate II

2022-12-04 08:53 PM

Hi,

I followed this article to create the test project on my evaluation board, but ping is not working. I did some debug, the ARP message can be received correctly, but looks like the data can't be sent out, even the call HAL_ETH_Transimit() returns OK. Do you have any idea what else should be checked here?

0 Kudos

RTsai.2
Associate

2023-04-18 08:17 AM

Hello there, I'm a new one at STM32H745-DISCO, I download the ethernet example from this website,

added the DATA_IN_D2_SRAM and Hello UDP message, then Build and Run the code on my board STM32H745XIH6 successfully.

but I can't receive "Hello UDP message!" when I ping to 192.168.1.10,

can anyone help me to figure it out? I will be thankful.

0 Kudos

JN. G.1
Associate

2023-05-04 01:36 PM

Great article on creating a project for STM32H7 with Ethernet and LwIP stack! The step-by-step instructions and clear explanations make it easy to follow. It's a helpful guide for anyone looking to work with these technologies. Well done!

1 Kudo

AHuss.9
Associate II

2023-05-25 09:54 AM

Hi all,

I am new to STM32 processors and know very little.

I have a Nucleo 144 with STM32F756 processor and my web pages are loading slowly - i.e., they take a few seconds to load.

HTML pages served are only a few kilobytes in size before flattening them with the makefsdata tool.

Otherwise they are working fine.

Is there any tweaks I can do to increase the server page responses?

Thanks in advance.

Abid

👍 **0 Kudos**

JojoS
Associate III

2023-06-08 04:28 AM

I agree this is very helpful.

One remaining problem:

My configuration on a H743 is not working when I follow the example MPU configuration. The MPU setting for the Rx_PoolSection (I'm using the same address 0x30040200 as in the example) is missing. The DMA is reading and writing the buffers in the memory pool and so it must be set to not shareable, not cacheable and not bufferable.

👍 **0 Kudos**

ManishNair
Associate III

2023-06-20 03:33 AM

Thanks a lot for sharing... Just following this post and making necessary changes, my STM32H753VI is now capable of sending 86Mbps continuous data thru UDP.

0 Kudos

JYI
Associate II

2023-06-29 02:53 PM

Hi, Adam,

I am referencing you and others posts for the STM32Hx Ethernet issues to try to make my own board to work.

Unfortunately, my board is still not working after about two weeks debugging.

Basically, I am using STM32H723VG with LAN8740 for MII with development tools STM32CubeIde Version 1.11.2 and STM32Cube_FW_H7_V1.11.0 + LWIP without RTOS.

Referencing the posts here, I've made some adjustments for MPU and _Flash.ld.

The Rx Buffer is located at D1(0x2400C000) and set LWIP_RAM_HEAP_POINTER to 0x30000400.

I have tested the hardware interface by verification LAN8740 registers and cable connection indicating that hardware is working since link status and auto-negotiation working properly and Rx and Tx signals are observed once the DHCP is starting. But STM has not received the message from the router since DHCP will be time out.

I've reviewed lots of posts about H7 ethernet issue, but still couldn't figure out what is missing.

Any suggestions will be appreciated.

I've posted some info below for the reference.

Thanks.

Junbo

```
// MPU setting
static void MPU_Config(void)
{
MPU_Region_InitTypeDef MPU_InitStruct;

/* Disable the MPU */
HAL_MPU_Disable();

/* Configure the MPU as Strongly ordered for not defined regions
* Disable speculative access to unused memories
*/
MPU_InitStruct.Number = MPU_REGION_NUMBER0;
MPU_InitStruct.Enable = MPU_REGION_ENABLE;
MPU_InitStruct.BaseAddress = 0x00;
MPU_InitStruct.Size = MPU_REGION_SIZE_4GB;
MPU_InitStruct.SubRegionDisable = 0x87;
MPU_InitStruct.TypeExtField = MPU_TEX_LEVEL0;
MPU_InitStruct.AccessPermission = MPU_REGION_NO_ACCESS;
MPU_InitStruct.DisableExec = MPU_INSTRUCTION_ACCESS_DISABLE;
MPU_InitStruct.IsCacheable = MPU_ACCESS_NOT_CACHEABLE;
MPU_InitStruct.IsBufferable = MPU_ACCESS_NOT_BUFFERABLE;
MPU_InitStruct.IsShareable = MPU_ACCESS_SHAREABLE;

HAL_MPU_ConfigRegion(&MPU_InitStruct);

/* Configure the MPU attributes as Device not cacheable
   for ETH DMA descriptors */
MPU_InitStruct.Number = MPU_REGION_NUMBER1;
MPU_InitStruct.Enable = MPU_REGION_ENABLE;
MPU_InitStruct.BaseAddress = 0x30000000;
MPU_InitStruct.Size = MPU_REGION_SIZE_1KB;
MPU_InitStruct.SubRegionDisable = 0x00;
MPU_InitStruct.TypeExtField = MPU_TEX_LEVEL0;
MPU_InitStruct.AccessPermission = MPU_REGION_FULL_ACCESS;
MPU_InitStruct.DisableExec = MPU_INSTRUCTION_ACCESS_DISABLE;
MPU_InitStruct.IsCacheable = MPU_ACCESS_NOT_CACHEABLE;
MPU_InitStruct.IsBufferable = MPU_ACCESS_BUFFERABLE;
MPU_InitStruct.IsShareable = MPU_ACCESS_SHAREABLE;

HAL_MPU_ConfigRegion(&MPU_InitStruct);

/* Configure the MPU attributes as Normal Non Cacheable
    for LwIP RAM heap which contains the Tx buffers  */
MPU_InitStruct.Number = MPU_REGION_NUMBER2;
MPU_InitStruct.Enable = MPU_REGION_ENABLE;
MPU_InitStruct.BaseAddress = 0x30000400;
MPU_InitStruct.Size = MPU_REGION_SIZE_32KB;
MPU_InitStruct.SubRegionDisable = 0x00;
MPU_InitStruct.TypeExtField = MPU_TEX_LEVEL1;
MPU_InitStruct.AccessPermission = MPU_REGION_FULL_ACCESS;
MPU_InitStruct.DisableExec = MPU_INSTRUCTION_ACCESS_DISABLE;
MPU_InitStruct.IsCacheable = MPU_ACCESS_NOT_CACHEABLE;
MPU_InitStruct.IsBufferable = MPU_ACCESS_NOT_BUFFERABLE;
MPU_InitStruct.IsShareable = MPU_ACCESS_NOT_SHAREABLE;
   HAL_MPU_ConfigRegion(&MPU_InitStruct);
*/
   /*
    * Enable the MPU
    */
   HAL_MPU_Enable(MPU_PRIVILEGED_DEFAULT);
}

// _Flash.ld

// RX buffer
.lwip_rx_pool_sec (NOLOAD) :
{
. = ABSOLUTE(0x2400C000);
*(.Rx_PoolSection)
} >RAM_D1 AT> FLASH
```

```
/* lwip: add placement of DMA descriptors */
.lwip_dma_desc_sec (NOLOAD) :
{
. = ABSOLUTE(0x30000000);
*(.RxDecripSection)

. = ABSOLUTE(0x30000200);
*(.TxDecripSection)
} >RAM_D2 AT> FLASH

// .map
.lwip_rx_pool_sec
0x000000002400b740 0x5453 load address 0x0000000008030454
0x000000002400c000 . = ABSOLUTE (0x2400c000)
*fill* 0x000000002400b740 0x8c0
*(.Rx_PoolSection)
.Rx_PoolSection
0x000000002400c000 0x4b93 ./LWIP/Target/ethernetif.o
0x000000002400c000 memp_memory_RX_POOL_base
.lwip_dma_desc_sec
0x0000000030000000 0x2c0 load address 0x0000000008030454
0x0000000030000000 . = ABSOLUTE (0x30000000)
*(.RxDecripSection)
.RxDecripSection
0x0000000030000000 0xc0 ./LWIP/Target/ethernetif.o
0x0000000030000000 DMARxDscrTab
0x0000000030000200 . = ABSOLUTE (0x30000200)
*fill* 0x00000000300000c0 0x140
*(.TxDecripSection)
.TxDecripSection
0x0000000030000200 0xc0 ./LWIP/Target/ethernetif.o
0x0000000030000200 DMATxDscrTab
// DHCP log
Hard Reset PHY LAN8740....
LWIP Init....
netif: added interface st IP
```

```
addr
0.0.0.0
netmask
0.0.0.0
gw
0.0.0.0


netif: setting default interface st

Link UP(5)
FULL-DUPLEX(10M)
dhcp_start(netif=0x2400507c) st0

dhcp_start(): mallocing new DHCP client

dhcp_start(): allocated dhcp
dhcp_start(): starting DHCP configuration

udp_bind(ipaddr =
0.0.0.0
, port = 68)

udp_bind: bound to
0.0.0.0
, port 68)

udp_connect: connected to
0.0.0.0
, port 67)

dhcp_discover()

dhcp_create_msg()

transaction id: xid(4bb5f646)

dhcp_discover: making request

dhcp_discover: sendto(DISCOVER, IP_ADDR_BROADCAST,
LWIP_IANA_PORT_DHCP_SERVER)

udp_send: added header in given pbuf 0x24009958

udp_send: sending datagram of length 316

udp_send: UDP packet length 316
```

```
udp_send: UDP checksum 0xf701

udp_send: ip_output_if (,,,,0x11,)

ip4_output_if: st0

IP header:

+-------------------------------+

| 4 | 5 | 0x00 | 336 | (v, hl, tos, len)

+-------------------------------+

| 0 |000| 0 | (id, flags, offset)

+-------------------------------+

| 255 | 17 | 0xba9d | (ttl, proto, chksum)

+-------------------------------+

| 0 | 0 | 0 | 0 | (src)

+-------------------------------+

| 255 | 255 | 255 | 255 | (dest)

+-------------------------------+

ip4_output_if: call netif->output()

etharp_output()

ethernet_output()

ethernet_output: sending packet 0x24009958

dhcp_discover: deleting()ing

dhcp_discover: SELECTING

dhcp_discover(): set request timeout 2000 msecs

dhcp_fine_tmr(): request timeout

dhcp_timeout()

dhcp_timeout(): restarting discovery
```

👍 **0 Kudos**

HLe.2

Associate II

2023-07-13 10:45 PM

Hello Mr. Adam BERLINGER

Thank you so much for the very helpful article.

I'm making a simple board based on STM32H7 and LAN8720 (currently I don't have LAN8742). I just started using STM32, I follow your tutorial and samples code on github, ping is OK but it disconnects after some time (please see the log below).

Do I need to change any parameters? Please give me some suggestions.

```
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=2ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=2ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=2ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=3ms TTL=255
Reply from 192.168.1.123: bytes=32 time=7ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=3ms TTL=255
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.1.123: bytes=32 time=2ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=3ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=3ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time<1ms TTL=255
Reply from 192.168.1.123: bytes=32 time=1ms TTL=255
```

👍 **0 Kudos**

arena_chris1seto

Associate II

2023-08-16 10:41 PM

Bumping this.

I have the exact issue others are talking about in this thread with the pbuf assert

```
Assertion "pbuf_free: p->ref > 0" failed at line 753 in ../Middlewares/Third_Party/LwIP
```

There is so much conflicting information and so many different memory configurations being discussed on these forums and in the ST code. The information from ST has been extremely hard to follow. For example, why does the memory config from this repo differ from the CubeH7 V1.11?

I am using CubeH7 V1.11.0 with the exact memory layout it recommends:

"

```
.lwip_sec (NOLOAD) :
{
. = ABSOLUTE(0x30000000);
*(.RxDescripSection)

. = ABSOLUTE(0x30000200);
*(.TxDescripSection)


. = ABSOLUTE(0x30000400);
*(.Rx_PoolSection)

} >RAM_D2
```

"

@Adam BERLINGER @STOne-32 Can we please get an update with a working setup, which does not have pbuf assert issues, with v1.11? *Have the CubeH7 Stm32H723 examples ever actually been seen to work without memory issues?*

Thanks,

0 Kudos

Snm.1

Associate III

2023-08-16 10:43 PM

Hello I want to use stm32h747 s ethernet periferal. But ı have to use Rmii and without rtos mode. I gues can you help me??

0 Kudos

STOne-32

ST Employee

2023-08-17 12:05 PM

Dear @arena_chris1seto ,

Our technical teams are working on it right now in the Opened ticket and will be back to you shortly, we need absolutely to replicate the behavior using our environment of board and thank you again for your patience and collaboration to go to the end.

Ciao

STOne-32

0 Kudos

Adam BERLINGER

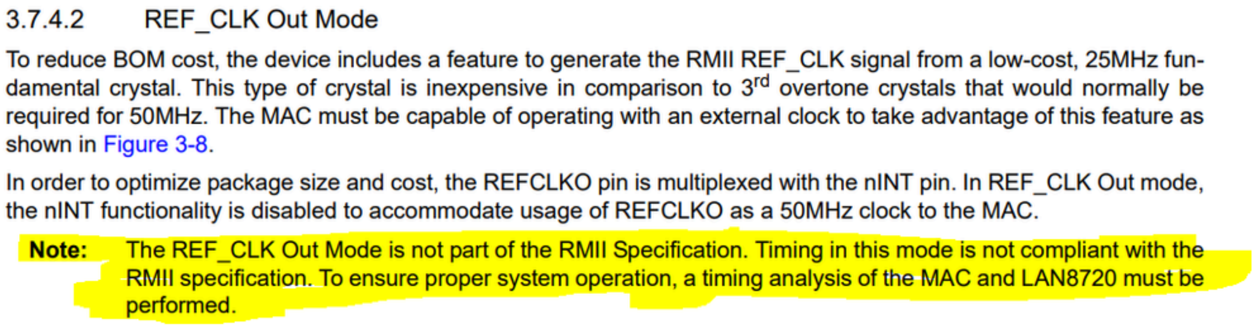
ST Employee

2023-08-18 05:20 AM

Dear all, @HLe.2, @arena_chris1seto

unfortunately **when LAN8720 is used in REF_CLK_OUT mode (RMII clocks generated by PHY), the timing is not compatible with STM32H7**. In that case the RXD output hold is 1.4ns minimum on PHY side, while STM32H7 requires at least 2ns.

Also the datasheets of LAN8720 warns about this compatibility issue.



3.7.4.2 REF_CLK Out Mode

To reduce BOM cost, the device includes a feature to generate the RMII REF_CLK signal from a low-cost, 25MHz fundamental crystal. This type of crystal is inexpensive in comparison to 3rd overtone crystals that would normally be required for 50MHz. The MAC must be capable of operating with an external clock to take advantage of this feature as shown in Figure 3-8.

In order to optimize package size and cost, the REFCLKO pin is multiplexed with the nINT pin. In REF_CLK Out mode, the nINT functionality is disabled to accommodate usage of REFCLKO as a 50MHz clock to the MAC.

Note: The REF_CLK Out Mode is not part of the RMII Specification. Timing in this mode is not compliant with the RMII specification. To ensure proper system operation, a timing analysis of the MAC and LAN8720 must be performed.

Such issue is not present when using LAN8742, or supplying 50MHz RMII clock externally. Also please note that STM32H7 PLL output through MCO (Micro-controller clock output) is not suitable for such signal, due to long-term jitter requirements.

Note: STM32F2/F4 and STM32F7 should be still compatible with LAN8720, since they have different timing constraints than STM32H7. But it is better to double-check the particular device.

Best regards,

Adam Berlinger

👍 **1 Kudo**

arena_chris1seto
Associate II

2023-08-21 11:43 PM

Thanks @Adam BERLINGER , yeah I will admit we were caught by this. I'll rework a new RMII reference clock onto the board. Any thoughts on the pbuf issues?

I bought some STM32 Nucleo dev boards to hopefully replicate the issue on official ST hardware with completely unmodified example code.

👍 **0 Kudos**

ADMIN **Laurids_PETERSEN**
Community manager

2023-08-22 12:57 AM

Hi everyone! I'm excited to see all the comments and questions regarding this article.

However, to prevent bloating the comment section and to give you the best possible guidance to your question, I strongly recommend creating a post in our STM32 MCU forum (click here).

This gives you the best possible opportunity to get the answer you need. We have active members that are happy to answer your questions.

Best regards,
Laurids

👍 0 Kudos

Zzaaack
Associate III

2023-09-29 08:51 AM

Hello everyone,

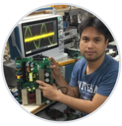
I know it's been some time since the beginning of the discussion, but the topic is as hot as the first time it was shared.

@IOvch already solved my issues a long time ago, but i couldn't find the uploaded files he talks about in one of his replies. I'm developing the lwip stack on M4 device (stm32H755 nucleo) and it seems to be very tricky, also listening all the relpies and comments, the memory allocation. I have one question especially: is MPU mandatory for M4 config? They are still out of my understanding the definitions of cacheable, bufferable and shareable, could you providwe some documentation about it please?

Thanks in advance

Zaaack

👍 0 Kudos



victagayun
Senior III

2023-10-15 06:43 PM

Hello,

It is best to provide IOC files on your next release of the CubeMX update.

0 Kudos

Previous 1 2 Next

Version history

Last update:

2024-06-17 06:55 AM

Updated by:

ADMIN **Laurids_PETERSEN**

Contributors

Laurids_PETERSEN

Adam BERLINGER