STM32 以太网 MAC 地址 Hash 过滤

关键词：MAC 地址，Hash 过滤

前言

网络中传递着各种各样的数据包，当设备连接到网络后，为了减少对接收到的数据进行处理的负荷，就需要对设备接收到的数据包进行过滤。STM32MCU 的以太网外设提供多种数据包过滤的模式。可以根据以太网帧的目标 MAC 地址，源 MAC 地址进行过滤，STM32H7 系列还提供对 VLAN tag 和 IP 地址，UDP/TCP 端口的过滤。

拿 MAC 地址过滤来说，SM32MCU 支持：单播目标地址过滤，多播目标地址过滤，单播源地址过滤和广播地址过滤。单播目标地址过滤和多播目标地址过滤又分为：Perfect 地址过滤和 Hash 地址过滤。
perfect 地址过滤就是把接收到的以太网帧中的目标地址与 MAC 地址寄存器中保存的地址进行比较，如果匹配，数据包就被接受，否则就被丢掉。还可以通过设置“反向过滤”，来翻转过滤的结果，接收到的以太网帧中的目标地址与 MAC 地址寄存器中保存的地址如果不匹配，数据包就被接收，否则就被丢掉。

Hash 地址过滤不是直接比较 MAC 地址，而是计算目标 MAC 地址的 CRC32 值，取其高 6 位作为索引去查询 Hash 表寄存器中对应的值，来判断是否接收该数据帧。Hash 地址过滤的方法稍微复杂，本文接下来将基于 STM32H743Nucleo 板，通过具体的例程介绍如何实现 Hash 地址过滤。

MAC 地址 Hash 过滤

过滤原理

在 Hash 地址过滤模式下，以太网 MAC 通过一张 64 位的 Hash 表来进行过滤。这张表存储在两个 32 位的寄存器中。STM32H743 的寄存器 ETH_MACHT0R 保存着 Hash 表的前 32 位，ETH_MACHT1R 中保存着 Hash 表的后 32 位值。
MAC 接收到以太网帧后，会自动计算目标 MAC 地址的 CRC 值，然后用该 CRC 值的高 6 位，作为索引号去前面提到的 Hash 表寄存器中查找对应位，如果该位的值是 1，则收到的以太网帧通过。否则就丢掉。例如，计算出的 CRC 高 6 位是 0，则对应 ETH_MACHT0R 的 bit0，如果该位是 1，则通过。
在初始化的时候，应该根据想要接收的目标 MAC 地址，先设置好 ETH_MACHT0R 和 ETH_MACHT1R 寄存器的值。Hash 地址过滤将 48 位的 MAC 地址，对应到 6 位的 Hash 值，肯定会出现多个 MAC 地址对应到一个 6 位 Hash 值的情况，所以这种过滤方式也被称作 imperfect 过滤模式。

Hash 值的计算方法

Hash 地址过滤模式，最关键的就是如何计算 6 位的 Hash 值。
在 RM0433 中介绍了 Hash 的产生方法，具体如下：

1.计算目标 MAC 地址的 CRC32 值。计算 CRC32 的方法参见 IEEE802.3 的第 3.2.8 章中 FCS 的说明 。根据 IEEE802.3 中 CRC 值的计算要求，和以太网帧中 MAC 地址传输的顺序，MAC 地址的 CRC 值计算方法如下：

- 第一个 32 位数据进行补码运算
- 输入的数据都进行按位反转顺序
- 进行 CRC32 计算，多项式为 0x4C11DB7
- 对最终输出数据进行补码运算

2. 对第一步的计算值进行按位反转顺序

3.取第二步计算值的高 6 位

然后就可以根据计算出来的 Hash 值，去设置 ETH_MACHT0R 和 ETH_MACHT1R 寄存器了。

MAC 地址过滤的寄存器配置

目标 MAC 地址过滤的寄存器配置见下表：

Table 521. Destination address filtering

Packet type	PR	HPF	HUC	DAIF	HMC	PM	DBF	DA filter operation
Broadcast	1	X	X	X	X	X	X	Pass
	0	X	X	X	X	X	0	Pass
	0	X	X	X	X	X	1	Fail
Unicast	1	X	X	X	X	X	X	Pass all packets
	0	X	0	0	X	X	X	Pass on Perfect/Group filter match
	0	X	0	1	X	X	X	Fail on Perfect/Group filter match
	0	0	1	0	X	X	X	Pass on Hash filter match
	0	0	1	1	X	X	X	Fail on Hash filter match
	0	1	1	0	X	X	X	Pass on Hash or Perfect/Group filter match
	0	1	1	1	X	X	X	Fail on Hash or Perfect/Group filter match
Multicast	1	X	X	X	X	X	X	Pass all packets
	X	X	X	X	X	1	X	Pass all packets
	0	X	X	0	0	0	X	Pass on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	0	1	0	X	Pass on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	0	1	0	X	Pass on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	X	X	1	0	0	X	Fail on Perfect/Group filter match and drop Pause packets if PCF = 0x
	0	0	X	1	1	0	X	Fail on Hash filter match and drop Pause packets if PCF = 0x
	0	1	X	1	1	0	X	Fail on Hash or Perfect/Group filter match and drop Pause packets if PCF = 0x

例程说明

下面我们将用一个例子来说明如何配置 Hash 地址过滤。

在该例程中，我们希望 STM32H743Nucleo 板只接收广播，发往自己的单播 MAC 地址的消息，以及两个特定多播 MAC 地址的消息。

单播 MAC 地址为：00:80:E1:00:00:00，

多播 MAC 地址为：01:0c:0d:01:01:03 和 01: 00: 5e: a8: 00: 0a。

例程中，我们需要做以下设置：

1. 设置数据包过滤寄存器 ETH_MACPFR 中相关位设置，使能单播 perfect 过滤，多播 Hash 过滤，不屏蔽广播消息。

PR	HPF	HUC	DAIF	HMC	PM	DBF
0	0	0	0	1	0	0

2.将单播地址设置到 ETH_MACA0HR 和 ETH_MACA0LR 中，并使能该地址。那么所有发往 00:80:E1:00:00:00 的单播数据包都能被收到，其他的单播数据包将被丢掉。

3.设置 Hash 过滤表寄存器。在初始化以太网外设时，利用 STM32H743 的 CRC 外设自动计算 MAC 地址的 CRC32 值，再得到对应的 Hash 值，根据该值去初始化 ETH_MACHT0R 和 ETH_MACHT1R 寄存器。H743Nucleo 将可以接收发往 01:0c:0d:01:01:03 和 01: 00: 5e: a8: 00: 0a MAC 地址的多播消息，其他的多播消息都被丢掉。

CRC 外设初始化代码：

```
CRC_HandleTypeDef hcrc;
void MX_CRC_Init(void)
{

  hcrc.Instance = CRC;
  hcrc.Init.DefaultPolynomialUse = DEFAULT_POLYNOMIAL_ENABLE;
  hcrc.Init.DefaultInitValueUse = DEFAULT_INIT_VALUE_ENABLE;
  hcrc.Init.InputDataInversionMode = CRC_INPUTDATA_INVERSION_BYTE;
  hcrc.Init.OutputDataInversionMode = CRC_OUTPUTDATA_INVERSION_ENABLE;
  hcrc.InputDataFormat = CRC_INPUTDATA_FORMAT_BYTES;
  if (HAL_CRC_Init(&hcrc) != HAL_OK)
  {
    Error_Handler();
  }

}
```

计算并使能 Hash MAC 地址过滤的代码：

```
#define MULTI_MAC_ADDR_MAX_NUM          2
uint8_t multi_addr[MULTI_MAC_ADDR_MAX_NUM][6]= {
  {0x01, 0x0c, 0x0d, 0x01, 0x01, 0x03},
  {0x01, 0x00, 0x5e, 0xa8, 0x00, 0x0a}
};

void EnableHashMulticastFilter(uint8_t* multi_mac_addr)
{
  uint8_t i=0;
  uint32_t CRCvalue=0;
  uint32_t Hashvalue=0;

  for(i=0;i<MULTI_MAC_ADDR_MAX_NUM;i++)
  {
```

```
    //##-1 - calculate the hash value of MAC address
    CRCvalue = HAL_CRC_Calculate(&hcrc, (uint32_t *)(multi_mac_addr+i*6), 6);

    /*##-2 - complement the CRC result and reverse the bit order ###############*/
    CRCvalue = bitReverse(~CRCvalue);

    /*##-3 - Get the upper 6 bits of CRC32MAC
################################################*/
    Hashvalue = CRCvalue >> 26;

    /*##-4 - set HASH table register
################################################*/
    if(Hashvalue >=32)
      heth.Instance->MACHT1R |= 1<<(Hashvalue-32);
    else
      heth.Instance->MACHT0R |= 1<<Hashvalue;
  }
  //enable hash filter for multicast
  heth.Instance->MACPFR |= ETH_MACPFR_HMC;
}

uint32_t bitReverse(uint32_t data)
{
  uint32_t result = 0;
  uint32_t i =0;

  for(i=0;i<32;i++)
  {
    if(data & (1<< i))
    {
      result += (1<<(31-i));
    }
  }
  return result;
}
```

运行结果

将附件的例程烧录到 H743Nucleo 板，通过 XCAP 连续发送下面的 6 条消息。

178 29.556928	00:00:00_00:00:33	STMicroe_00:00:00	0x0601	28	Ethernet II
179 29.557240	00:00:00_00:00:33	02:00:00:00:00:00	0x0601	28	Ethernet II
180 29.558449	00:00:00_00:00:33	IPv4mcast_a8:00:0a	0x0601	28	Ethernet II
181 29.558819	00:00:00_00:00:33	Communic_01:01:03	0x0601	28	Ethernet II
182 29.559161	00:00:00_00:00:33	Communic_01:01:ff	0x0601	28	Ethernet II
183 29.559505	00:00:00_00:00:33	Broadcast	0x0601	28	Ethernet II

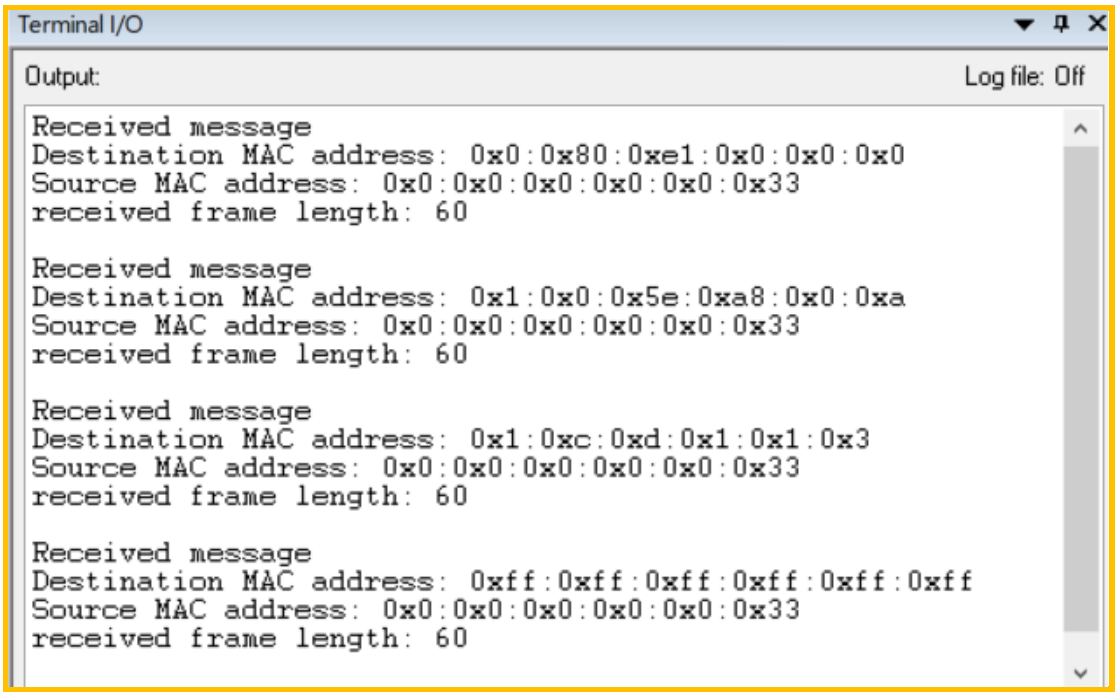
包括：

两条单播消息，目标 MAC 地址分别是：00:80:E1:00:00:00 和 02:00:00:00:00:00。

三条多播消息,目标 MAC 地址分别是：01:0c:0d:01:01:03，01: 00: 5e: a8: 00: 0a 和 01:0c:0d:01:01:ff。

一条广播消息。

从程序的打印信息里可以看到，H743Nucleo 板接收到了其中的 4 条消息，MAC 地址没有设置的一条单播消息（02:00:00:00:00:00）和一条多播消息（01:0c:0d:01:01:ff）都被过滤掉了。



.END.

重要通知 – 请仔细阅读

意法半导体公司及其子公司 （“ST”）保留随时对 ST 产品和 / 或本文档进行变更的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。 ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。若需 ST 商标的更多信息，请参考 *www.st.com/trademarks*。所有其他产品或服务名称均为其 各自所有者的财产。

本文档是 ST 中国本地团队的技术性文章，旨在交流与分享，并期望借此给予客户产品应用上足够的帮助或提醒。若文中内容存有局限或与 ST 官网资料不一致，请以实际应用验证结果和 ST 官网最新发布的内容为准。您拥有完全自主权是否采纳本文档（包括代码，电路图等）信息，我们也不承担因使用或采纳本文档内容而导致的任何风险。

本文档中的信息取代本文档所有早期版本中提供的信息。