CM32M4xxR PMP基础功能应用样例

Create Time	@July 22, 2021 8:59 AM
End Time	
Start Date	@July 22, 2021
Tags	已完成

0.功能说明

该应用样例展示了如何利用PMP的内存隔离特性实现对存储单元的访问控制，展示了PMP配置样例，方便开发者快速集成；

1.运行环境

环境

Name	版本
软件开发环境	NucleiStudio

2.外设资源

资源

Name	配置	描述
串口	UART5（TX-PE8, RX-PE9）	115200-8-1-n

3.功能描述

本应用例程中定义了一个共享内存空间test_array，将通过对该数据空间的存储空间进行定制化配置，展示PMP的访问控制功能；

由于PMP对地址有对齐要求，应保障地址空间与PMP匹配长度保持对齐，因此将test_array重定向至user_ram段，定位至0x2001A000;

> 💡 例如目标匹配的PMP长度为1KB，则PMP的起始地址应为 0xZZZZZ400/0xZZZZZ800/0xZZZZZC00/0xZZZZZ000

本应用样例将展示在PMP配置生效前后，分别对test_array进行读取写入访问，验证PMP的功能生效；

在PMP配置生效后，将触发PMP访问异常，并在异常处理中打印异常访问地址；

运行日志如下：

```
PMP Example Start!!!!!!

Reading test_array Success, before PMP Configuration

Write test_array Success, before PMP Configuration

Setting PMP Configuration!

Reading  test_array with read-only permission Success, after PMP Configuration

PMP Exception Occures Run at 0x20000476, Fetching Memory at 0x2001a00f
```

4.示例设计说明

4.1 PMP配置

本示例中，将针对共享内存test_array设置为只读权限，并在PMP配置生效后，对test_array分别进行读取及写入操作；读取指令将被放行，而写入指令将被拦截；同时注册异常访问的PMP处理函数；

```
  PMP_Region_InitTypeDef pmp_init;

  pmp_init.Number = PMP_REGION_NUMBER0;
  pmp_init.Enable = PMP_REGION_ENABLE;   // Enable Configuration
  pmp_init.Lock = PMP_REGION_LOCK;      // Configuration Lock, may not modify unless reset, and also match in Machine Mode
  pmp_init.BaseAddress = TEST_REGION_BASE;  //Seting test_array Base
  pmp_init.Size = PMP_REGION_SIZE_128B;    //Setting array size
  pmp_init.AddressMatching = PMP_ADDRESS_MATCHING_NAPOT;  //Setting PMP Size to NAPOT mode -> 2^n
  pmp_init.AccessPermission = PMP_ACCESS_R;     //Setting array permission is Read Only

  PMP_ConfigRegion(&pmp_init);

  // Register PMP Exception Handler
  Exception_Register_EXC(EXC_INS_ACCESS_FAULT, PMP_Exception_Handler);
  Exception_Register_EXC(EXC_LOAD_ACCESS_FAULT, PMP_Exception_Handler);
  Exception_Register_EXC(EXC_STORE_AMO_ACCESS_FAULT, PMP_Exception_Handler);
```

由于当前运行在机器模式，为了保障PMP配置项适用于机器模式，通常有两种方法：
1.PMP配置项锁定；
2.将mstatus.MPRV置为1，尝试以用户模式进行权限匹配；
本示例中采用第一种方式；

CM32M4xxR共支持8组PMP配置入口，最小配置颗粒度为128字节，仅支持NAPOT模式（2^n）；

4.2 流程说明

示例在系统启动后，将首先对test_array共享内存进行读取及写入访问，随后将对该区域设置为只读模式，并同时对内存进行读取及访问，已验证PMP的权限配置是否生效；

```
char temp = test_array[15];
  printf("\n\r Reading test_array Success, before PMP Configuration\n\r");
  //try to write test_array without PMP Configuration, may be pass
  test_array[15] = 1;
  printf("\n\r Write test_array Success, before PMP Configuration\n\r");

  //Configure test_array PMP, set to Read Only
  PMP_Config();
  printf("\n\r Setting PMP Configuration!\n\r");

  //try to read test_array with read-only permission, may be pass
  temp = test_array[15];
  printf("\n\r Reading  test_array with read-only permission Success, after PMP Configuration\n\r");

  //try to write test_array with read-only permission, could be failure
  test_array[15] = 1;
  //could not run to here
  printf("\n\r !!!!!!!!!!!!!!!!!!!!!!!!!Fail!!!!!!!!!!!!!!!!!!!\n\r");
  printf("\n\r Writting test_array with read-only permission , after PMP Configuration\n\r");
```

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/73ee4e01-de85-4593-b583-28a70e75be26/PMP.vsdx