CoreMark移植过程记录

以Cortex M0为例

1. 下载CoreMark

官网地址：http://www.eembc.org/coremark/index.php

代码在github上：https://github.com/eembc/coremark

2. 拷贝对应的文件到自己的工程文件夹，注意main函数已经在core_main.c中。

```
core_list_join.c
core_main.c
core_matrix.c
core_state.c
core_util.c
coremark.h
simple/core_portme.c
simple/core_portme.h
```

其中core_portme.c和core_portme.h是和MCU相关的，主要移植这个文件

3. 将coremark.h所在文件夹位置设为include的路径

4. 修改core_portme.c中的portable_init，加入平台的初始化代码

5. 在core_portme.h里面添加宏定义，数字根据具体的MCU填写。

#define ITERATIONS　2000

这个宏定义表示迭代次数，CoreMark要求程序运行的最短时间至少是10s, 根据使用的系统时钟等情况设置这个参数。

6. 根据编译环境和配置修改宏定义COMPILER_FLAGS

```
#ifndef COMPILER_FLAGS
 #define COMPILER_FLAGS \
   "-O"/* "Please put compiler flags here (e.g. -o3)" */
 #endif
```

7. 如不使用标准的printf函数，需要定义ee_printf，在里面去掉HAS_PRINTF的定义，然后增加ee_printf的定义

```
/*
 #ifndef HAS_PRINTF
 #define HAS_PRINTF 1
 #endif
 */
 #define ee_printf Printf
```

8. 修改Timer部分的代码，使用SysTick计数

```
void start_time(void){
    //getmytime(&start_time_val);
    gTick = 0;
    SysTick_Config(()SystemCoreClock/1000)*1);
```

```
}

void stop_time(void){
    /getmytime(&stop_time_val);
    SysTick->CTRL &= 0xfffffffe;
}

CORE_TICKS get_time(void){
    //CORE_TICKS elapsed
    //  = (CORE_TICKS)(MYTIMEDIFF(stop_time_val,start_time_val));
    return (CORE_TICKS)gTick;
}
```

9. 添加Systick中断函数

```
uint32_t gTick=0;
void SysTick_Handler(void)
{
    gTick++;
}
```

10. 修改宏定义EE_TICKS_PER_SEC，改为Systick的中断时间修改，表示每秒多少个Tick。

```
#define EE_TICKS_PER_SEC          1000
```

一些MCU的CoreMark得分：

- MM32L073： 44.44
- FT900： 142.85
- STM32F103：86.95