L1 cache设置

一.设置步骤

- 开mmu之前失效L1 i-cache和 L1 d-cache
  - 直接写寄存器ICIALLU来失效L1 i-cache
  - 按照set/way方式失效 L1 d-cache
- 开mmu时使能L1 cache

二.设置代码

```
    @@ 失效L1 i-cache和 L1 d-cache,使能之前需要,在
    mov     r0, #0                  @ r0 = 0x00000000
    mcr     p15, 0, r0, c7, c5, 0   @ 使失效I-cache,ICIALLU,Invalidate all instruction
caches to PoU
    mcr     p15, 2, r0, c0, c0, 0   @ 将0x00000000写入CSSELR(Cache Size Selection Register,
选择当前CCSIDR),选择L1 D-cache的CCSIDR

    mrc     p15, 1, r0, c0, c0, 0   @ r0 = CCSIDR(Cache Size ID Registers,Provides
information about the architecture of the caches selected by CSSELR)
    ldr     r1, =0x7fff             @ r1 = 0x7fff
    and     r2, r1, r0, lsr #13     @ r2 = r1 & (r0 >> 13), NumSets - 1

    ldr     r1, =0x3ff              @ r1 = 0x3ff
    and     r3, r1, r0, lsr #3      @ r3 = r1 & (r0 >> 3) ,NumWays - 1

    add     r2, r2, #1              @ r2 = NumSets

    @@ now r0 = CCSIDR, r1 = NumWays - 1, r2 = NumSets
    and     r0, r0, #0x7    @ r0 = r0 & 7,(Log2(Number of words in cache line)) - 2 =
log2(N/4)
    @ add     r0, r0, #4    @ SetShift , r0 = log2(N/4) + 4 = log2(N) + 2 , 应该加错了
    add     r0, r0, #2      @ SetShift , r0 = log2(N/4) + 2 = log2(N)

    clz     r1, r3          @ WayShift , r1 = (NumWays - 1)的前导零数
    add     r4, r3, #1      @ NumWays , r4 = NumWays

    @@ now r0 = log2(N), N = Number of words in cache line
    @@      r1 = WayShift,ARMV7 B4.2.1
    @@      r2 = NumSets
    @@      r4 = NumWays
1:
    sub     r2, r2, #1      @ NumSets--
    mov     r3, r4          @ Temp = NumWays
2:
    subs    r3, r3, #1              @ r3 = NumWays - 1
    mov     r5, r3, lsl r1          @ r5 = (NumWays -1)<<WayShift
    mov     r6, r2, lsl r0          @ r6 = (NumSets-1)<<SetShift
```

```
    orr     r5, r5, r6              @ Reg = ((NumWays -1)<<WayShift)|((NumSets-1)
<<SetShift),ARMV7 B4.2.1
    mcr     p15, 0, r5, c7, c6, 2   @ DCISW, Invalidate data cache line by set/way
    bgt     2b                      @ 内循环way
    cmp     r2, #0
    bgt     1b                      @ 外循环set
    dsb
    isb
```

三.使能代码

```
        /* b0000 0000 0000 0000 0001 1000 0000 0101 ,开mmu和l1 cache,分支预测 */
/*      asm __volatile__("dsb\n"
        "isb\n"
        "mrc    p15, 0, r1, c1, c0, 0\n"
        "ldr    r2, =0x1805\n"
        "orr    r1, r1, r2\n"
        "mcr    p15, 0, r1, c1, c0, 0\n"
        "dsb\n"
        "isb\n"
        : : :"memory", "cc");
*/
```