




Analisi MMU ARM tramite GEM5



Codice usato per la simulazione
Il codice va a creare un
checkpoint della simulazione FS,
così che poi rilanceremo con i
debug flag attivi.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <sys/mman.h>
4 #include <gem5/m5ops.h>
5
6
7 int main()
8 {
9     m5_checkpoint(0, 0);
10    m5_reset_stats(0, 0);
11
12    int *val1 = malloc(sizeof(int));
13    int *val2 = mmap(NULL, sizeof(int), PROT_READ | PROT_WRITE,
14                     MAP_PRIVATE | MAP_ANONYMOUS, 0, 0);
15
16    *val1 = 0;
17    *val2 = 0;
18
19    m5_dump_stats(0, 0);
20    m5_exit(0);
21    return 0;
22 }
```

command line: ./gem5/build/ARM/gem5.opt --debug-flags=TLB,TLBVerbose,PageTableWalker
gem5/configs/example/arm/starter_fs.py --restore m5out/cpt.67018537524750/



Output della simulazione FS con cpu atomic:


Viene richiesto al tlb di primo livello dedicato alle istruzioni (itb) di tradurre l'indirizzo 0x40080c.

A seguito di una miss si cerca una entry (anche parziale) nel tlb di secondo livello (l2_shared), un tlb più capiente e più lento, tuttavia si va incontro a miss anche in quel caso.

Quindi non essendoci altri livelli di tlb si inizia la page walk.

```
5000: cpus.mmu: translateFs addr 0x40080c, mode 2, st2 0, scr 0x30531 sctlr
0x3475d91d flags 0x100 tranType 0x0
5000: cpus.mmu: Translating VA=0x40080c context=11
5000: cpus.mmu.itb: Lookup 0x40080c, asn 0xb -> miss vmn 0x0 hyp 0 secure 0 ppn 0
size: 0 pa: 0 ap:0 ns:0 nstid:0 g:0 asid: 0 el: 0
5000: cpus.mmu.itb:
Passo al livello dopo di questo stage

5000: cpus.mmu.l2_shared: Lookup 0x40080c, asn 0xb -> miss vmn 0x0 hyp 0 secure 0
ppn 0 size: 0 pa: 0 ap:0 ns:0 nstid:0 g:0 asid: 0 el: 0
```



Per prima cosa la page walk delle istruzioni (`itb_walker`) cerca nel primo livello della translation table **L1** e trovo la traduzione parziale. Successivamente recupero l'indirizzo della entry di **L2** tramite l'indirizzo parziale trovato in L1.

```
5000: cpus.mmu: TLB Miss: Starting hardware table walker for 0x40080c(11:0)
5000: cpus.mmu.itb_walker: creating new instance of WalkerState
5000: cpus.mmu.itb_walker: Beginning table walk for address 0x40080c, TCR:
0x32b5593519
5000: cpus.mmu.itb_walker: - Selecting TTBR0 (AArch64)
5000: cpus.mmu.itb_walker: Fetching descriptor at address: 0xfba09000 stage2Req: 0
5000: cpus.mmu.itb_walker: L1 descriptor for 0x40080c is 0xfba09003 (AArch64)
5000: cpus.mmu.itb_walker: Analyzing L1 descriptor: 0xfba09003, type: 1
5000: cpus.mmu.itb_walker: L1 descriptor points to L2 descriptor at: 0xfba09010 (ns)
5000: cpus.mmu.itb_walker: Inserting Table descriptor into TLB
5000: cpus.mmu.itb_walker: - N:30 pfn:0xfba09000 size:0x3fffffff global:0 valid:1
5000: cpus.mmu.itb_walker: - vpn:0 xn:0 pxn:0 ap:3 domain:1 asid:11 vmid:0 hyp:0
nc:0 ns:1
5000: cpus.mmu.itb_walker: - domain from L1 desc:1 data:0xfba09003
5000: cpus.mmu.itb_walker:
La table walk prosegue, inserisco la entry PARZIALE trovata

5000: cpus.mmu.itb_walker: Fetching descriptor at address: 0xfba09010 stage2Req: 0
5000: cpus.mmu.itb_walker: L2 descriptor for 0x40080c is 0xfbb8b003 (AArch64)
5000: cpus.mmu.itb_walker: Analyzing L2 descriptor: 0xfbb8b003, type: 1
5000: cpus.mmu.itb_walker: L2 descriptor points to L3 descriptor at: 0xfbb8b000 (ns)
5000: cpus.mmu.itb_walker: Inserting Table descriptor into TLB
5000: cpus.mmu.itb_walker: - N:21 pfn:0xfbb8b000 size:0x1ffffff global:0 valid:1
5000: cpus.mmu.itb_walker: - vpn:0x2 xn:0 pxn:0 ap:3 domain:1 asid:11 vmid:0 hyp:0
nc:0 ns:1
5000: cpus.mmu.itb_walker: - domain from L2 desc:1 data:0xfbb8b003
5000: cpus.mmu.itb_walker:
La table walk prosegue, inserisco la entry PARZIALE trovata
```



Trovata la traduzione parziale di L2, si procede ad **inserirlo nel tlb di secondo livello** (l2_shared).

Si prosegue nell'ultimo livello della translation table, il livello L3.

Una volta **trovata l'ultima traduzione in L3**, si inserisce la entry **sia nel tlb di primo che di secondo livello**.

Finita la page walk, **si ripete la lookup del tlb** che avrà sicuramente una hit.

```
5000: cpus.mmu.l2_shared: Inserting entry into TLB with pfn:0xfbb8b000 size:0x1fffff
vpn: 0x2 asid:11 vmid:0 N:21 global:0 valid:1 nc:0 xn:0 ap:0x3 domain:0x1 ns:1
nstdid:1 isHyp:0
5000: cpus.mmu.itb_walker: Fetching descriptor at address: 0xfbb8b000 stage2Req: 0
5000: cpus.mmu.itb_walker: L3 descriptor for 0x40080c is 0x20000080b8cfd3 (AArch64)
5000: cpus.mmu.itb_walker: Analyzing L3 descriptor: 0x20000080b8cfd3, pxn: 1, xn: 0,
ap: 3, af: 1, type: 3
5000: cpus.mmu.itb_walker: memAttrsAArch64 AttrIdx:0x4 sh:0x3
5000: cpus.mmu.itb_walker: Inserting Page descriptor into TLB
5000: cpus.mmu.itb_walker: - N:12 pfn:0x80b8c size:0xfff global:0 valid:1
5000: cpus.mmu.itb_walker: - vpn:0x400 xn:0 pxn:1 ap:3 domain:1 asid:11 vmid:0 hyp:0
nc:0 ns:1
5000: cpus.mmu.itb_walker: - domain from L3 desc:1 data:0x20000080b8cfd3
5000: cpus.mmu.itb_walker:
La table walk ha finito, inserisco la entry trovata

5000: cpus.mmu.itb: Inserting entry into TLB with pfn:0x80b8c size:0xfff vpn: 0x400
asid:11 vmid:0 N:12 global:0 valid:1 nc:0 xn:0 ap:0x3 domain:0x1 ns:1 nstdid:1 isHyp:0
5000: cpus.mmu.l2_shared: Inserting entry into TLB with pfn:0x80b8c size:0xfff vpn:
0x400 asid:11 vmid:0 N:12 global:0 valid:1 nc:0 xn:0 ap:0x3 domain:0x1 ns:1 nstdid:1
isHyp:0
5000: cpus.mmu.itb: Lookup 0x40080c, asn 0xb -> hit vmn 0x0 hyp 0 secure 0 ppn
0x80b8c size: 0xfff pa: 0x80b8c80c ap:3 ns:1 nstdid:1 g:0 asid: 11 el: 0
```

Dopo aver caricato la nuova
entry **seguono delle hit** su
quest'ultima

Inizio a tradurre nello stage 1

```
5000: cpus.mmu: Checking S1 permissions: ap:3, xn:0, pxn:1, r:0, w:0, x:1, is_priv:
0, wxn: 0
5000: cpus.mmu: Setting memory attributes: shareable: 0, innerAttrs: 0, outerAttrs:
0, mtype: 2, stage2: 0
5250: cpus.mmu: CPSR is priv:0 UserMode:0 secure:0 S1S2NsTran:0
5250: cpus.mmu: translateFs addr 0x400728, mode 2, st2 0, scr 0x30531 sctlr
0x3475d91d flags 0x100 tranType 0x0
5250: cpus.mmu: Translating VA=0x400728 context=11
5250: cpus.mmu.itb: Lookup 0x400728, asn 0xb -> hit vmn 0x0 hyp 0 secure 0 ppn
0x80b8c size: 0xfff pa: 0x80b8c728 ap:3 ns:1 nstid:1 g:0 asid: 11 el: 0
5250: cpus.mmu:
Inizio a tradurre nello stage 1
```

```
5250: cpus.mmu: Checking S1 permissions: ap:3, xn:0, pxn:1, r:0, w:0, x:1, is_priv:
0, wxn: 0
5250: cpus.mmu: Setting memory attributes: shareable: 0, innerAttrs: 0, outerAttrs:
0, mtype: 2, stage2: 0
5500: cpus.mmu: CPSR is priv:0 UserMode:0 secure:0 S1S2NsTran:0
5500: cpus.mmu: translateFs addr 0x40072c, mode 2, st2 0, scr 0x30531 sctlr
0x3475d91d flags 0x100 tranType 0x0
5500: cpus.mmu: Translating VA=0x40072c context=11
5500: cpus.mmu.itb: Lookup 0x40072c, asn 0xb -> hit vmn 0x0 hyp 0 secure 0 ppn
0x80b8c size: 0xfff pa: 0x80b8c72c ap:3 ns:1 nstid:1 g:0 asid: 11 el: 0
```



Ad un certo punto si **uscirà dal range di istruzioni** presenti nella pagina tradotta dalla entry inserita precedentemente nel tlb itb. Tuttavia è possibile **trovare una traduzione parziale nel tlb di secondo livello** (l2_shared), la quale era stata inserita durante la page walk precedente solo in questo tlb più capiente.

6750: cpus.mmu:

Inizio a tradurre nello stage 1

6750: cpus.mmu: Checking S1 permissions: ap:3, xn:0, pxn:1, r:0, w:0, x:1, is_priv:0, wxn:0

6750: cpus.mmu: Setting memory attributes: shareable:0, innerAttrs:0, outerAttrs:0, mtype:2, stage2:0

7000: cpus.mmu: CPSR is priv:0 UserMode:0 secure:0 S1S2NsTran:0

7000: cpus.mmu: **translateFs** addr **0x405c40**, mode 2, st2 0, scr 0x30531 sctlr 0x3475d91d flags 0x100 tranType 0x0

7000: cpus.mmu: Translating VA=0x405c40 context=11

7000: cpus.mmu.**itb**: **Lookup 0x405c40, asn 0xb -> miss** vmn 0x0 hyp 0 secure 0 ppn 0 size:0 pa:0 ap:0 ns:0 nstid:0 g:0 asid:0 el:0

7000: cpus.mmu.itb:

Passo al livello dopo di questo stage

7000: cpus.mmu.**l2_shared**: **Lookup 0x405c40, asn 0xb -> hit** vmn 0x0 hyp 0 secure 0 ppn 0xfbb8b000 size:0x1ffffff pa:0x1f771600005c40 ap:3 ns:1 nstid:1 g:0 asid:11 el:0

7000: cpus.mmu: TLB Miss: Starting hardware table walker for 0x405c40(11:0)



Quindi si **inizia la page walk**, ma invece che partire dal primo livello della translation table, si parte dall'indirizzo parziale trovato nella tlb l2_shared (ovvero la traduzione parziale del livello L2).

Inoltre **grazie alla cache del page walker** in questo caso non vi è bisogno neanche di accedere alla RAM, ma si trova la traduzione finale direttamente nella cache.

Quindi, come prima, si inserisce la traduzione finale (L3) sia in **tlb di primo livello** itb, sia in **tlb l2_shared**.

```
7000: cpus.mmu.itb_walker: Beginning table walk for address 0x405c40, TCR:
0x32b5593519
7000: cpus.mmu.itb_walker: - Selecting TTBR0 (AArch64)
7000: cpus.mmu.itb_walker: Walk Cache hit: va=0x405c40, level=2, table
address=0xfbb8b000
7000: cpus.mmu.itb_walker: Fetching descriptor at address: 0xfbb8b028 stage2Req: 0
7000: cpus.mmu.itb_walker: L3 descriptor for 0x405c40 is 0x200000fa687fd3 (AArch64)
7000: cpus.mmu.itb_walker: Analyzing L3 descriptor: 0x200000fa687fd3, pxn: 1, xn: 0,
ap: 3, af: 1, type: 3
7000: cpus.mmu.itb_walker: memAttrsAArch64 AttrIndx:0x4 sh:0x3
7000: cpus.mmu.itb_walker: Inserting Page descriptor into TLB
7000: cpus.mmu.itb_walker: - N:12 pfn:0xfa687 size:0xfff global:0 valid:1
7000: cpus.mmu.itb_walker: - vpn:0x405 xn:0 pxn:1 ap:3 domain:1 asid:11 vmid:0 hyp:0
nc:0 ns:1
7000: cpus.mmu.itb_walker: - domain from L3 desc:1 data:0x200000fa687fd3
7000: cpus.mmu.itb_walker:
La table walk ha finito, inserisco la entry trovata

7000: cpus.mmu.itb: Inserting entry into TLB with pfn:0xfa687 size:0xfff vpn: 0x405
asid:11 vmid:0 N:12 global:0 valid:1 nc:0 xn:0 ap:0x3 domain:0x1 ns:1 nstid:1 isHyp:0
7000: cpus.mmu.l2_shared: Inserting entry into TLB with pfn:0xfa687 size:0xfff vpn:
0x405 asid:11 vmid:0 N:12 global:0 valid:1 nc:0 xn:0 ap:0x3 domain:0x1 ns:1 nstid:1
isHyp:0
7000: cpus.mmu.itb: Lookup 0x405c40, asn 0xb -> hit vmn 0x0 hyp 0 secure 0 ppn
0xfa687 size: 0xfff pa: 0xfa687c40 ap:3 ns:1 nstid:1 g:0 asid: 11 el: 0
```




Lo stage 2 non viene mai invocato poichè la full system viene eseguita senza un hypervisor al di sotto del sistema operativo che usiamo per eseguire il programma, quindi viene usato solo lo stage 1.