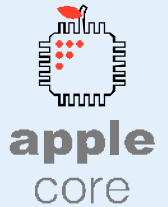


Micro-Threaded LEON3 Processor

<http://www.apple-core.info/>

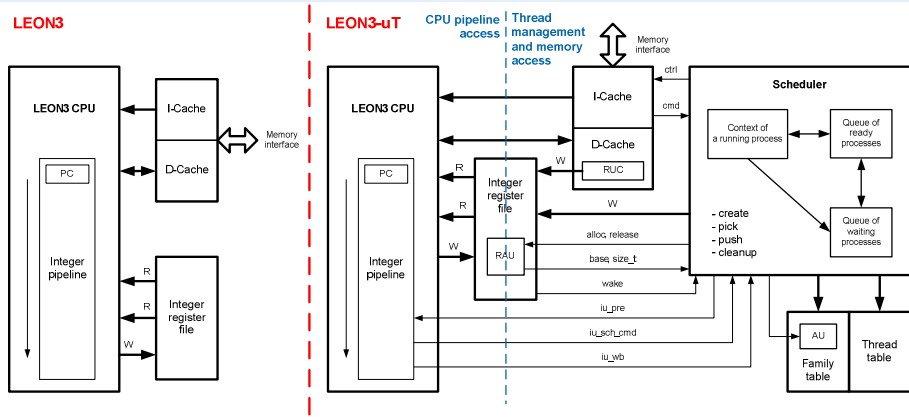


Motivation

Tolerate long-latency operations (LD/ST, FPop). Synchronize on register access.

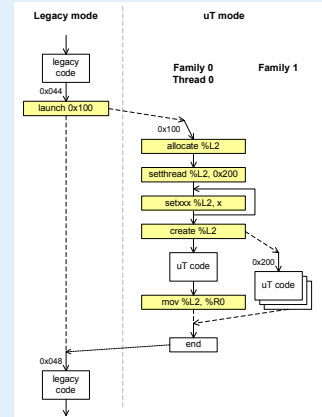
- ▶ Use self-synchronizing register file
 - ▶ Register status managed autonomously by the register file
- ▶ Prevent pipeline stall by thread switch
- ▶ Manage thread status & thread switch autonomously
 - ▶ Thread scheduler

LEON3 vs. LEON3-uT



10 New ASM Instructions

- ▶ launch
- ▶ allocate
- ▶ setstart
- ▶ setlimit
- ▶ setstep
- ▶ setthread
- ▶ create
- ▶ kill
- ▶ break



Micro-Threaded LEON3 Processor Characteristics

Self-synchronizing register file

- ▶ Register file ports
 - ▶ Integer pipeline: 2R, 1W
 - ▶ D-Cache and scheduler: 1W (shared)
 - ▶ Inner usage : 2R
 - TOTAL : 4R, 2W**
- ▶ Registers
 - ▶ 2b code {empty, pending, waiting, full}
 - ▶ 32b per register
 - Total 34b per register**
- ▶ Register allocation unit
- ▶ Thread wake up detection

Family and thread tables

- ▶ Fixed size: FT 1 line = 1 family, TT 1 line = 1 thread
- ▶ FT: 1 line = 159b, 8 lines
- ▶ TT: 1 line = 92b, 32 lines

Thread scheduler

- ▶ Executes in parallel with the integer pipeline
- ▶ Create – 5CC + 7CC per thread
- ▶ Pick – 0CC
- ▶ Push – 1CC (2CC)
- ▶ Cleanup – 2CC per thread

I-Cache/D-Cache

- ▶ Cache line: 16x 32b words
- ▶ Cache tag: write back mask / reference counter (16b), head (10b), tail (10b) ptr, request ID (4b), keep (2b), lock (2b), status (2b), address high (18b)
- Total: 64b**
- ▶ Memory read/write: asynchronous data read/write in one or two 16W burst transactions
- ▶ ICache: Each first word in a cache line contains 2b extensions – commands for the thread scheduler
 - ▶ *continue, switch, end*

Integer pipeline

- ▶ Next PC: the calculation is modified to skip the first word in a cache line

Execution Scenarios

