XCore Fetch No-Operation Modeling

In addition to the instructions defined in the ISA, a further operation, Fetch No-Op (FNOP) can be issued by the processor. This is issued when the instruction buffer of a thread does not contain sufficient data to issue the next instruction into the rest of the pipeline. The fetch stage of the pipeline is then used to fetch the missing data, at the cost of the other stages being idle for the affected thread. Thus, the thread effectively performs a no-op during its scheduled slot. The conditions that produce FNOPs are documented [1]and deterministic. For example, repeated sequences of memory operations prevent fetches from occurring, requiring FNOP operations to refill the instruction buffer. The FNOP has no effect on the actual result of the program, but does affect time and energy consumption. The energy model contains the FNOP power as an additional entry into the list of possible instructions, which can be utilized by a simulator that traces FNOPs.

To account for FNOPs in static analysis of a program, the programs CFG at ISA level is analyzed. We model the behavior of the instruction buffer logic that deter- mines where in a basic block FNOPs will occur. A potential inaccuracy in this approach is when an FNOP insertion depends on the dynamic behavior of the program. When a branch instruction reaches a basic block, the instruction buffer is flushed prior to loading the target address. If the target is unaligned, then an FNOP may be required for long instructions, or one may be needed in the subsequent cycle if the target instruction includes a memory operation. For blocks that have more than one predecessor, and one of the predecessors passes execution to the block without branching, then there is a dynamic decision as to whether an FNOP is required upon entry to the block. This is analogous to assigning FNOPs to edges of the CFG, rather than within the basic blocks. The effect of an FNOP arising from such branches is significant if the basic block is a loop entry point. To produce safe upper bounds in our WCEC analysis we assume that there is always an FNOP at the entry of a basic block which has multiple predecessors and its first instruction is unaligned and either long or a memory operation. In our experimental evaluation, we demonstrate that this does not have a significant effect on the tightness of our bounds. This is because the vast majority of static FNOP predictions will be correct during the repeated iteration of a loop, typically with a single mis-prediction upon entry to the loop.

[1]The XMOS XS1 Architecture 2009a