Chip multi-processors using a micro-threaded model

Most microprocessor chips today use an out-of-order (OOO) instruction execution mechanism. This mechanism allows superscalar processors to extract reasonably high levels of instruction level parallelism (ILP). The most significant problem with this approach is a large instruction window and the logic to support instruction issue from it. This includes generating wake-up signals to waiting instructions and a selection mechanism for issuing them. Wide-issue width also requires a large multi-ported register file, so that each instruction can read and write its operands simultaneously. Neither structure scales well with issue width leading to poor performance relative to the gates used. Furthermore, to obtain this ILP, the execution of instructions must proceed speculatively. An alternative, which avoids this complexity in instruction issue and eliminates speculative execution, is the microthreaded model. This model fragments sequential code at compile time and executes the fragments OOO while maintaining in-order execution within the fragments. The fragments of code are called microthreads and they capture ILP and loop concurrency. Fragments can be interleaved on a single processor to give tolerance to latency in operands or distributed to many processors to achieve speedup. The major advantage of this model is that it provides sufficient information to implement a penalty free distributed register file organisation. However, the scalability of the microthreaded register file in terms of the number of required logical read and write ports is not clear yet. In this thesis, we looked at the distribution and frequency of access to the asynchronous (non-pipeline …