Summary
Multiscalar Processors

Mrinal Nath (ID: 3307043)
January 31, 2005

Summary Part 1

What are the problems solved by this paper?
Superscalar processors have problems in exploiting large amounts of ILP from programs. This is because their instruction windows are not very large and the branch prediction accuracy limits the ILP extracted. Also, the complexity of the issue logic increases very fast ($n^2$) and there is the complexity of examining each instruction in the window to check if it can be issued. A lot of checks have to be performed before memory accesses can be reordered.
This paper attempts to alleviate most of these problems by proposing a new microarchitecture called Multiscalar.

What are the approaches attempted by this paper?
This paper uses a combination of software (compiler) and hardware (microarchitectural) techniques to solve the above problems.
The compiler uses the CFG of the program to partition the program into tasks. Information about the possible successor tasks is stored in a task descriptor in the multiscalar binary. Additional information like forward bits and stop bits are embedded in the multiscalar binary. (The multiscalar binary can be generated from an existing binary and this extra information can be stored in a table in the new binary)
The multiscalar hardware consists on multiple processing units. A sequencer takes the information about the tasks from the task descriptor and assigns tasks to the individual PUs. A communication ring is provided for intertask communication. ARB (address resolution buffer) is used to validate the memory accesses, before actually updating the data caches.

What are the main conclusions of this paper?
According to the results presented in the paper, enormous speed-ups are possible using the Multiscalar approach. Due to the fact that several tasks are running in parallel, the amount of ILP that can be extracted is very high. Due to this same reason, the instruction window can be very large (several hundred instructions) and since the innards of a task are not examined, the complexity found in the issue stage of superscalar processors is avoided. The memory accesses are issued independently in Multiscalar, so the various checks that have to be done in superscalar are avoided. Tasks are issued speculatively to the individual PUs, and each task may contain several conditional branches. So Multiscalar can effectively speculate beyond a large number of branches.