Chapter 1 introduces the basic idea about pipelined processor including the motivation and the problems which should be solved and how to solve them.

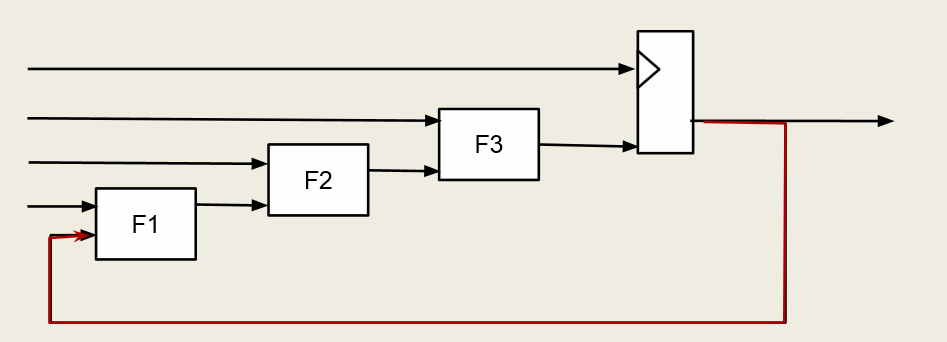
First, what is pipeline? Pipeline model was first used in UNIX operating system. It is a ‘chain model’ for connecting a chain of processing elements (processes, threads, coroutines, etc.), arranged so that the output of each element is the input of the next，forming a linear workflow. Thus, for a given complete input, after co-processing of the individual components, we can get the only final output. Connecting elements into a pipeline is analogous to functional composition.

What I have mentioned above is application of pipeline model in the area of software. With the development of science and technology, application development model to the pipeline among the various fields. Pipeline model is almost universally seen most of the mainstream model to process the requests, these designs are not surprised, because pipeline model specific idea seems to make us feel it is to handle requests generated. In fact, the application of pipeline model is much more than web server application architecture.

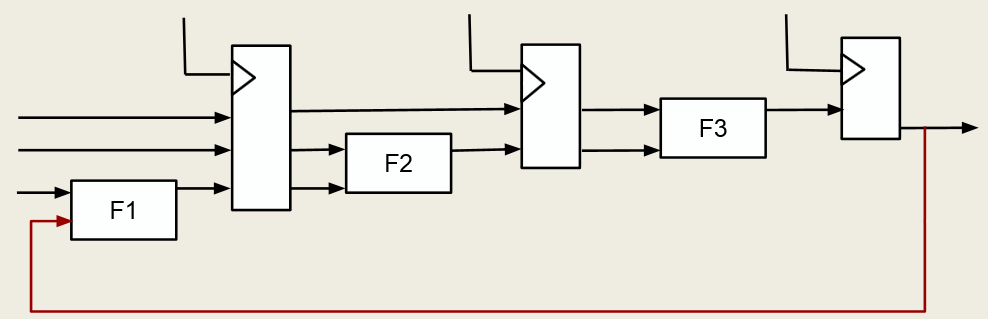
In computers, the same basic logic applies but rather than producing something physical on an assembly line, it is the workload itself that gets broken down into smaller stages, called pipeline. It is a set of data processing elements connected in series, where the output of one element is the input of the next one. In this paper, we are focused on the instruction pipelines, such as the classic RISC pipeline, which are used in central processing units (CPUs) to allow overlapping execution of multiple instructions with the same circuitry. The circuits is usually divided into several stages, including instruction decoding, arithmetic, and register fetching stages, wherein each stage processes one instruction at a time.

Second, why we used pipeline model in the design of central processing units? Let’s come to the huge advantages of pipelining. The cycle time of the processor is reduced and the instruction throughput is increased while applying pipeline model. Pipelining does not reduce the time it takes to complete an instruction, instead it increases the number of instructions that can be processed simultaneously and reduces the delay between completed instructions. The CPU arithmetic logic unit can be designed faster if pipelining is used. Besides, pipelining increases performance over an un-pipelined core by a factor of the number of stages and the code is ideal for pipeline execution. Briefly state, pipeline can make complex operations faster and more economically.

Assume the circuit shown below, consists of 3 combinational functions F1, F2 and F3, one flip-fop after F3. Let the 3 combinational functions have their own propagation delays T1, T2 and T3 respectively, then the total delay of the combinational is T=T1+T2+T3. Hence, the maximum frequency of this circuit f(max)=1/(T1+T2+T3).



Instead of the original circuit, we apply pipeline model, added flip-flops between the stages as shown below, while pipelining is a technique to increase the delay of a combinational circuit slightly but allows different components of the logic at the same time. Then, the delay for the slowest path is equal to the maximum among T1, T2 and T3. Hence, this new circuit with pipelining can be clocked with frequency: f(max)=1/max(T1,T2,T3).



Compared the different maximum frequency of processing in the two logic circuit above, we can tell the frequency of the pipelining circuit is higher than which in the original circuit clearly. If the functions F1, F2 and F3 had equal propagation delays, then the maximum frequency of the new circuit would have tripled compared to the old circuit. The reason for the improved throughput is that the different stages of a pipeline work in parallel while without pipelining the entire logic would be occupied by a single operation.

However, the speed advantage is diminished when execution encounters hazards. Then come to the 3 well-known hazards in pipelined CPU: structural hazard, data hazard and control hazard.