2. Discuss the technique of **PIELING** a processor design. What are the tricky implementation issues? What is the effect of pipelining on processor performance? (support your answer quantitatively)

**Extra credit:** what is the effect of pipelining on power dissipation? (support your answer quantitatively)

Pipelining a processor is the process of taking a the steps of a processor: Fetch, decode, Execute, Mem, and Write Back, etc. and putting state registers in between the stages to store the current state of the processor. What this provides is a way to increase the clock speed of the system to match the longest of the stages of the CPU. This means that in a single cycle of the CPU that multiple instructions can be operated upon concurrently, i.e.

```
add r0, r1, r2
addi r3, r2, r3
lui r4, 154
add r3, r4, r5
add r4, r5, r6
```

This can bring about implementation issues such as dependent instruction and load-use penalties. Most of these issues can be solved with data forwarding. For the last case, where data is needed immediately after a load the fetch, EX, and ID registers must be held by holding their values and inserting a NOP into the EX register. Additionally, branch instructions can be processed by inserting a Pipeline Stage.

The upside to all of this complexity is that there is a performance gain due to the fact that there are 5 instructions being worked upon at once. even though overall one per instruction basis. Each instruction takes longer to run than the non-pipelined CPU the added throughput of the system increases performance dramatically.

Finally, the effect of pipelining on power dissipation is that there is increased power being used for the extra control logic and pipe registers so the power dissipation can be used to drive the data lines. Since the distances to the endpoints of the system are less long, so overall it is a balancing game that requires analysis on how much each part of the system.****