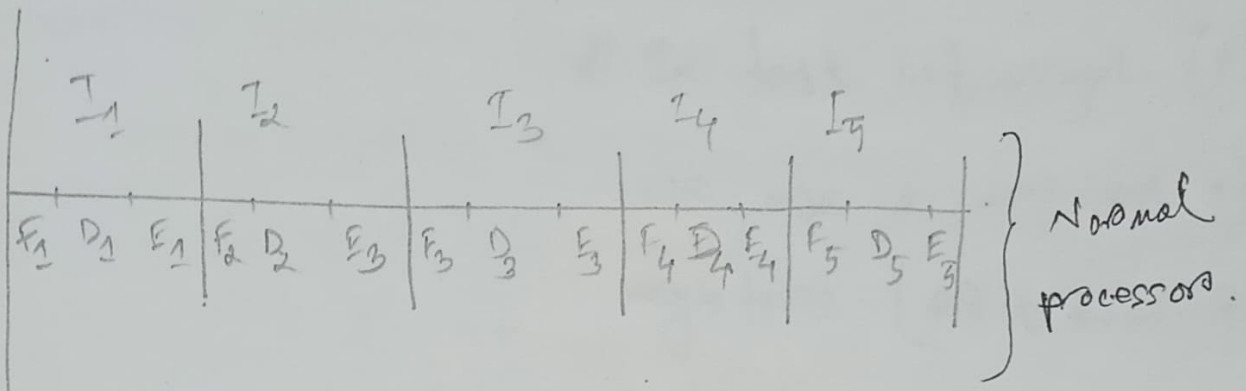
ARM7 - Pipelining

Lec 06
16 August 2022

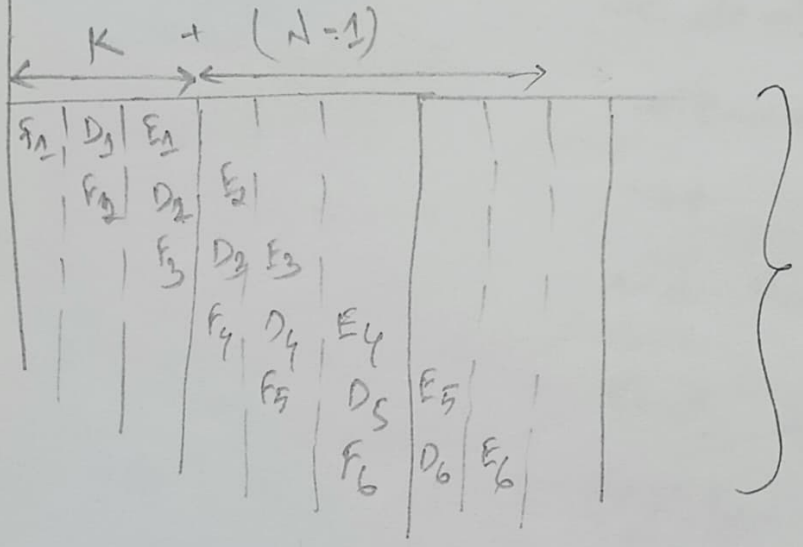
K = No. of stages
N = No. of instruction.

#pipelining: dividing the different parts of instruction and overlapping them.
ARM uses three stage pipeline.



Normal processor.

Total cycle = K×N



pipelined processor.

Total cycles = K + (N-1)

Normal processors fetch the first, then decode the first instr, then execute the third instr.
Then fetch the second instr, then decode the second instr, then execute the second instr.
So for each instruction requires three cycle.

For pipelined processors (ARM7, 3 stage processor)
In first cycle, fetch the first instr, then decode the first instr, while decoding first instr, fetch the first instr then execute the first instr at the same time decoding the second instr and fetching the third instr. In fourth cycle execute the second instr, decode the third instr and fetch the fourth instr.

Ex: K = 3, N = 5

Total cycle (Normal processor) $= 3 \times 5 = 15$

Total cycle (pipelined processor) $= 3 + (5 - 1) = 3 + 4$
$= 7$

In pipelined processors,

first instr requires = 3 cycle.

every instr after first requires = 1 cycle.

If there is 100 inst.

T.C = 100 × 3 = 300 cycle (Normal)

T.C = 3 + (100 - 1) = 102 cycle (pipelined).

So 3 stage pipelined processors

is 3 times faster than normal

processors

fetching, decoding and execution

for each instruction must be done

in one cycle to maintain pipeline

~~processor~~ process.

All the instruction in ARM is

exactly 32 bit so fetching, decoding

and execution take exactly 1 cycle of

of time.

pipeline fails when there is bubble or

branch in the pipeline.

(Drawbacks in ARM)

~~# #~~ Higher microprocessor like pentium there is "branch prediction algorithm" which protects from bubble ~~from~~ in pipelining.