You can get performance by :

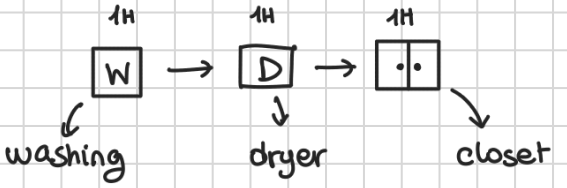
- parallelism

↓

PIPELINING

EXAMPLE: Suppose you have to do the laundry. You can split this in some phases

$n = 3 = w, d, c$



Pipelining is useful when you have several instances of a work to do.

$n + 2$ → time to perform

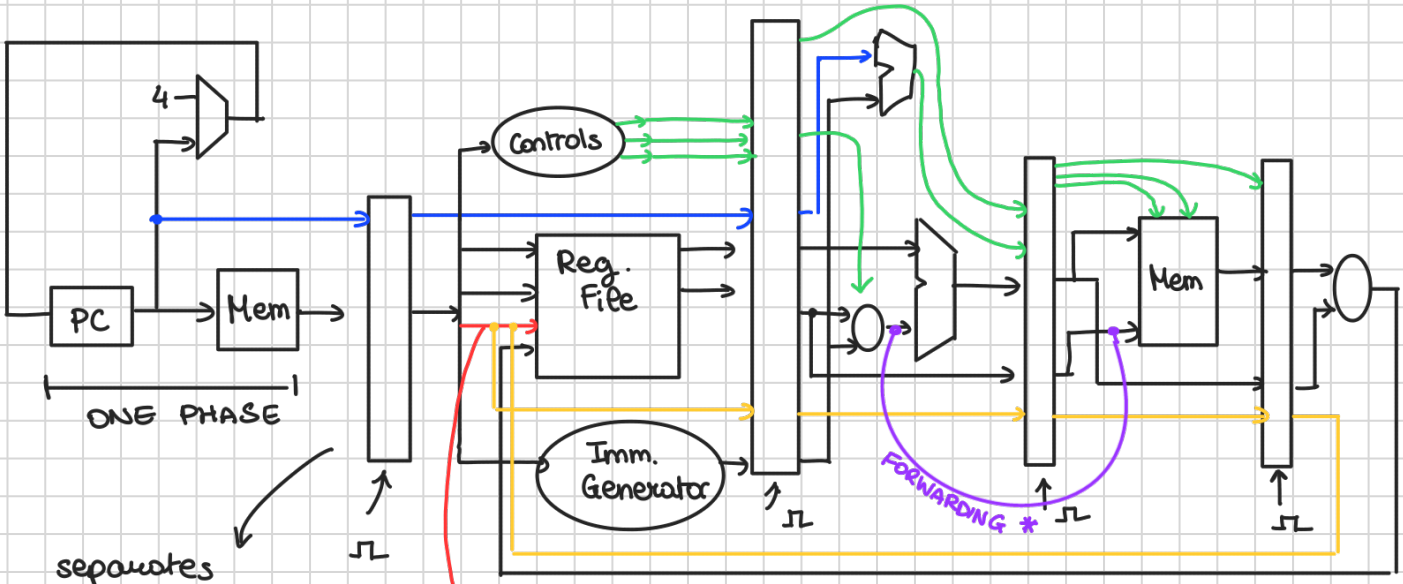
worst time ↖ $\frac{3n}{n+2}$ → SPEED - UP

↖ best time

$\frac{3n}{n+2} \xrightarrow[n \to \infty]{} 3$

THREE REQUIREMENTS FOR PIPELINING

1. You must have a process that can be divided into phases.
2. Each phase has to use different hardware
3. Each phase needs to have approx. same time length.



separates one phase from another (PIPELINE REG.)

this is wrong because it will have other information about other instructions.

(A) ADD x2, x3, x5
(B) SUB x6, x7, x8
(C) OR x9, x10, x11
(D) AND x12, x13, x14
(E) XOR x15, x16, x17

- we take this content along so that it can't be lost
- same thing with PC, we carry it along
- same thing with the controls

This is 5x faster than our previous architecture.

One problem is that we might execute two instructions that share same registers which may affect the results / values.

To overcome this, we use 2 bubbles between 2 instructions.

↳ a clock cycle to slow down

Building unit - FORWARDING TECHNIQUE *

↓

basically, we replace wrong value with correct one if we are in a situation where the next instruction has the register value of its previous one

We call these problems "HAZARD" Ex. (1) add x5, x6, x7 (2) sub x6, x10, x11

HOW CAN WE NOT LOSE CLOCK CYCLES WHEN WE HAVE TO JUMP ?

LOOP UNROLLING → reduce number of branches. Done by compiler

INTEL PENTIUM (2GH) → has a lot of pipelining components

IBM (Motorola) architecture — PowerPC — G4

→ Apple had an agreement with IBM to use PowerPC G4's.