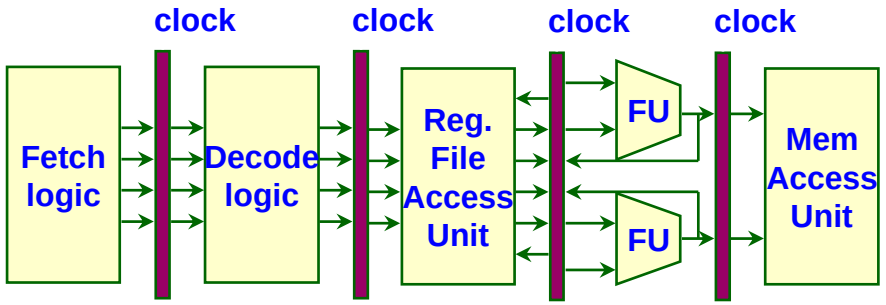
A glance on the analytical model of power-performance trade-off in VLSI microprocessor design

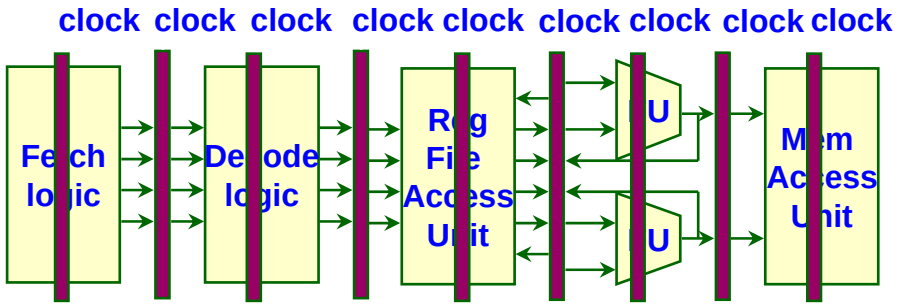
Sapienza University of Rome

Introductive ideas



V_{dd} = 3.3 V
T_{CK} = 5 ns
Power = 20 W
IPS = 200 mips

0.1 W / mips
0.00050 W / mips^2

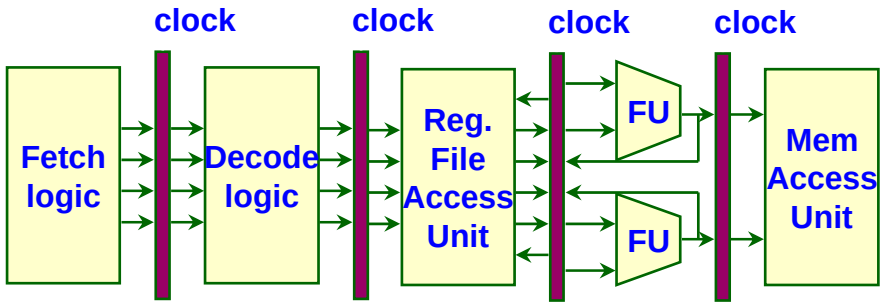


V_{dd} = 3.3 V
T_{CK} = 2.5 ns
Power = 40 W
IPS = 300 mips

0.13 W / mips
0.00044 W / mips^2

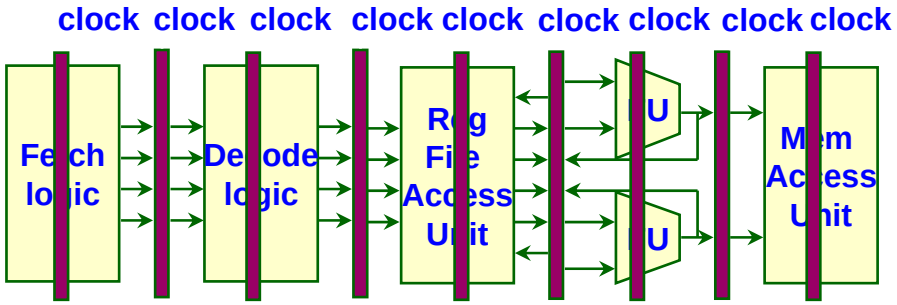
The second implementation is less power-efficient due to heavier pipe stall penalties

Introductive ideas



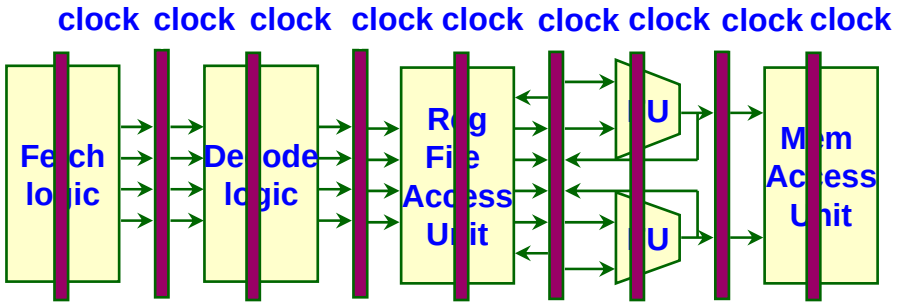
V_{dd} = 3.3 V
T_{CK} = 5 ns
Power = 20 W
IPS = 200 mips

0.1 W / mips
0.00050 W / mips^2



V_{dd} = 3.3 V
T_{CK} = 2.5 ns
Power = 40 W
IPS = 300 mips

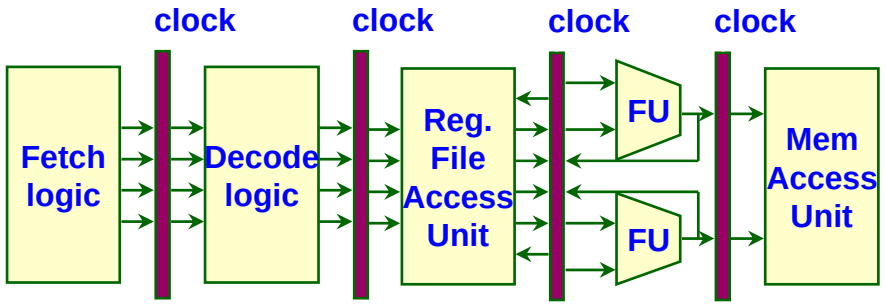
0.13 W / mips
0.00044 W / mips^2



V_{dd} = 2.1 V
T_{CK} = 3.75 ns
Power = 11.8 W
IPS = 200 mips

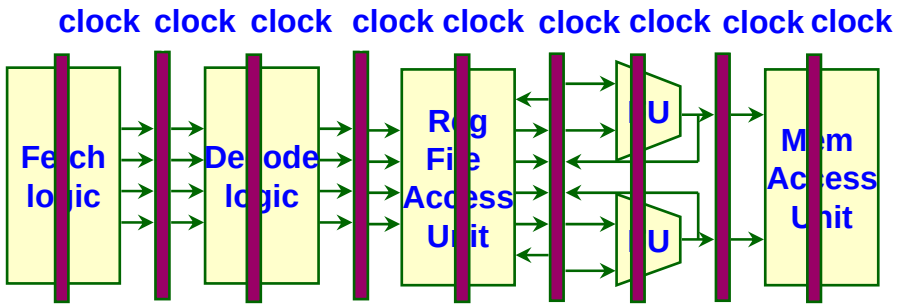
0.059 W / mips
0.00029 W / mips^2

Introductive ideas



V_{dd} = 2.1 V
T_{CK} = 7.5 ns
Power = 5.9 W
IPS = 133 mips

0.044 W / mips
0.00033 W / mips^2



V_{dd} = 2.1 V
T_{CK} = 3.75 ns
Power = 11.8 W
IPS = 200 mips

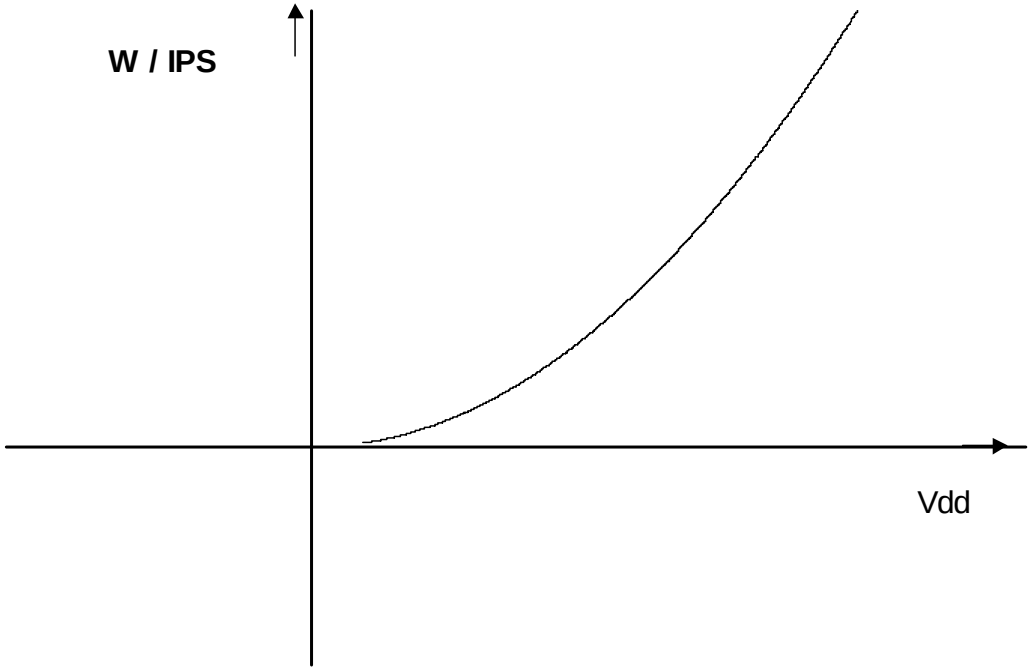
0.059 W / mips
0.00029 W / mips^2

The latter implementation is more power-efficient thanks to better pipeline/V_{dd} sizing

Introductive idea: analytical view

- $W = C_{eff}\, V_{dd}^{\,2} / T_{CK}$
- $IPS = 1 / (CPI \;\; T_{CK})$

$W / IPS = CPI\, C_{eff}\, V_{dd}^{\,2}$



However, we cannot arbitrarily improve the power efficiency
by simple V_{dd} and T_{CK} scaling if we have a performance constraint IPS > IPS*
(…see next slide)

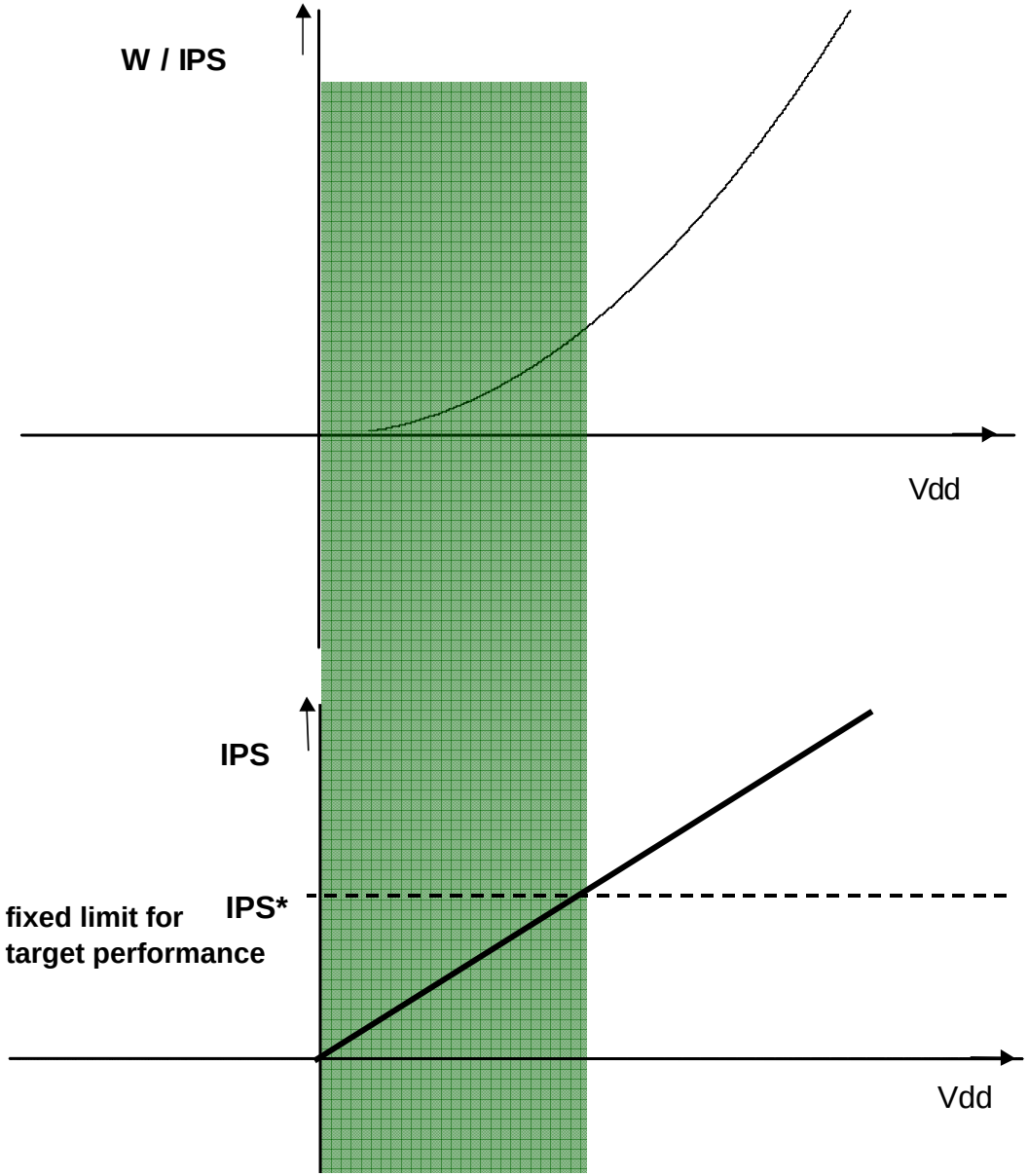
Introductive idea: analytical view

- $W = C_{eff}\, V_{dd}^{\,2} / T_{CK}$
- $IPS = 1 / (CPI \;\; T_{CK})$

$W / IPS = CPI\, C_{eff}\, V_{dd}^{\,2}$

- $T_{CK} \propto 1 / V_{dd}$ *(rough approx.)*
- $IPS = 1 / (CPI \;\; T_{CK})$

- $IPS \propto V_{dd} / CPI$



Problem Formulation

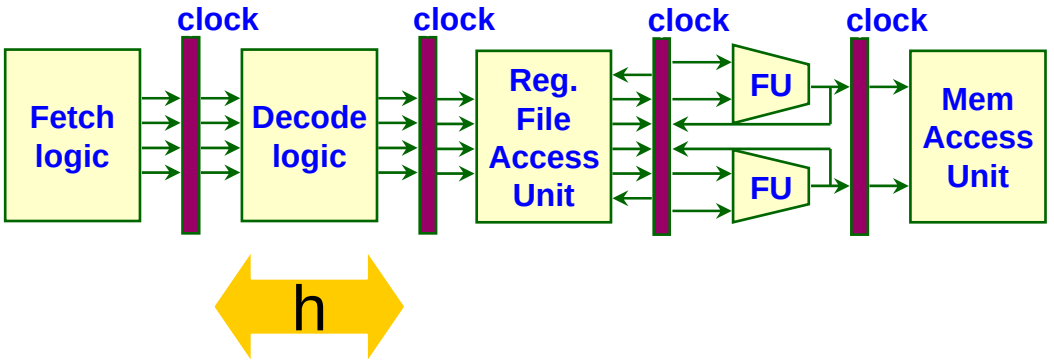
The relevant aspects of the trade-off evaluation are manifold:

- The $\mathbf{V_{dd}}$ vs. $\mathbf{T_{CK}}$ relation (technology dependent)
- The $\mathbf{V_{dd}}$, $\mathbf{T_{CK}}$, **IPS** vs. **Power** relation (involving switching capacitance estimation)
- The **pipeline depth** vs. **IPS** relation (involving pipe stall estimation)

Target: to define a unified model of all these aspects, obviously assuming simplifying hypotheses, which may represent a starting point for more detailed technology-specific and architecture-specific analyses

A simple model

- We refer to a pipelined multiple-issue architecture



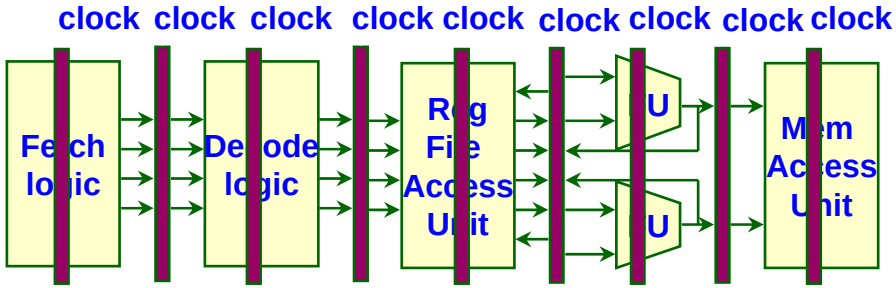
- Let us identify some parameters
  - δ, the **minimal gate delay** in the target CMOS technology (FO1)
  - **h,** the clock cycle time measured in δ units
  - $\mathbf{C_{eff}}$**,** effective switching capacitance per cycle

- *h* is a measure of the logic complexity in a pipe stage, and indirectly of the pipeline depth

"First level" (and very limited) model

- One simplified but representative sub-micron CMOS delay model is
  - $\delta = V_{dd}\, q_0 / (V_{dd} - 2v_t)$ (q_0 is a constant) (*Chenming Hu, ISSCC 94*)
- The whole system is governed by the following set of equations:
  - $\delta = V_{dd}\, q_0 / (V_{dd} - 2v_t)$
  - $TCK = h\, \delta$
  - $W = C_{eff}\, V_{dd}\, 2 / TCK$
  - $IPS = 1 / (CPI\ \ TCK)$
  - $CPI = f$ (architecture_parameters) ⬅ *from insn level simulation!*
- Assume a target IPS = *IPS**. By substituting δ, we can derive the optimal sizing for *h*, in order to maintain $IPS \equiv IPS^*$ despite *Vdd* scaling:
  - $IPS = (V_{dd} - 2v_t) / (CPI\ h\ V_{dd}\ q_0)$ ⇨ $h = (V_{dd} - 2v_t) / (CPI\ V_{dd}\ q_0\ IPS^*)$.
- BUT we are assuming only circuit/logic level optimization of *h:* no effect on CPI, no architecture re-design. Not very realistic unless for little V_{dd} adjustments.

"Second level" (realistic) model



- Typically, reducing *h* will involve less logic operations within each clock cycle, i.e. pipe stage re-design.
- This will change the **pipeline depth** and thus the CPI rate, due to pipe stalls.
- Let us identify some new parameters:
  - CPI_{ideal} , average *CPI* assuming no pipe stalls
  - H_{stall} , average duration of a pipe stall measured δ in units
  - C_{eff0} , effective switching capacitance per cycle *not due to any instruction processing* (i.e. clock distribution, etc.)
  - C_{effPI} , effective switching capacitance per cycle due to *one* instruction processing (fetch, decode, functional unit operation, ecc.)

"Second level" (realistic) model

- The actual *CPI* of the pipelined processor is
- $CPI = CPI_{ideal} + H_{stall} / h$

- The switching activity per cycle due to the actual *instruction per cycle* rate is $C_{eff} = Ceff_0 + C_{effPI} / CPI$

- The new set of equations governing our system is
  - $\delta = Vdd\ q_0 / (V_{dd} - 2v_t)$
  - $TCK = h\ \delta$
  - $W = C_{eff}\ V_{dd}^{\ 2} / TCK$
  - $IPS = 1 / (CPI\ \ TCK\ )$
  - $CPI_{ideal} = f$ (architecture_parameters) ⬅ *insn level simulation*
  - $CPI = CPI_{ideal} + H_{stall} / h$
  - $C_{eff} = C_{eff0} + C_{effPI} / CPI$

"Second level" model

- By substitution, we can derive the desired value for *h* :

$$h = \frac{V_{dd} - 2v_t - V_{dd}\, q_0\, H_{stall}\, IPS^*}{V_{dd}\, q_0 IPS^*\, CPI_{ideal}}$$

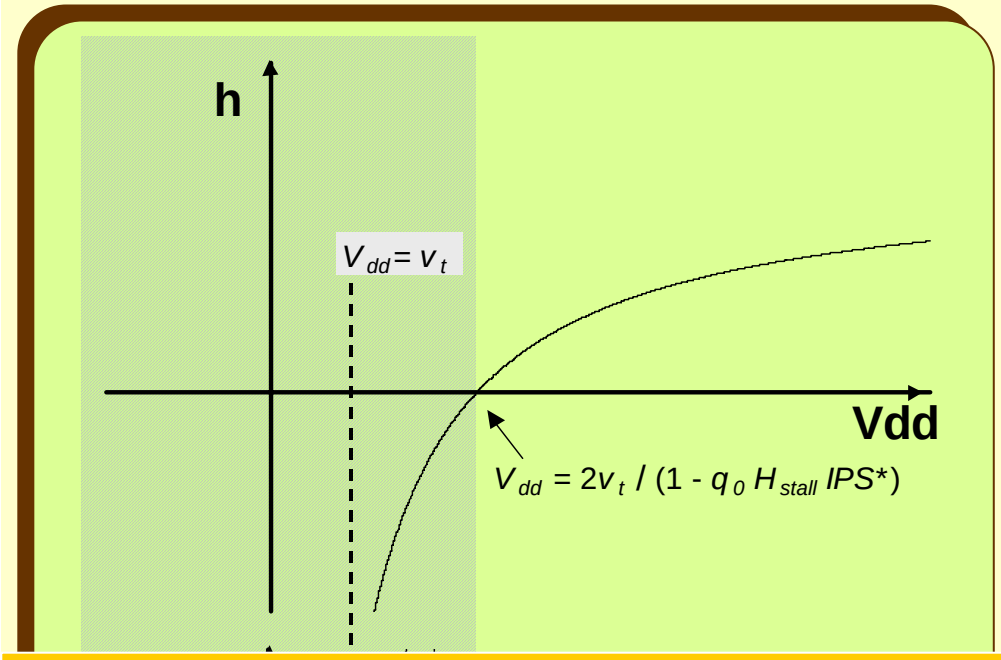
- The consequent power consumption of the system at fixed throughput IPS* is

$$W = \frac{C_{eff}\, V_{dd}^2 (V_{dd} - 2v_t) q_0\, CPI_{ideal}\, IPS^*}{V_{dd} - 2v_t - V_{dd}\, q_0\, H_{stall}\, IPS^*}$$

- *W* exhibits a minimum value for an optimal value of *Vdd*! (see next)

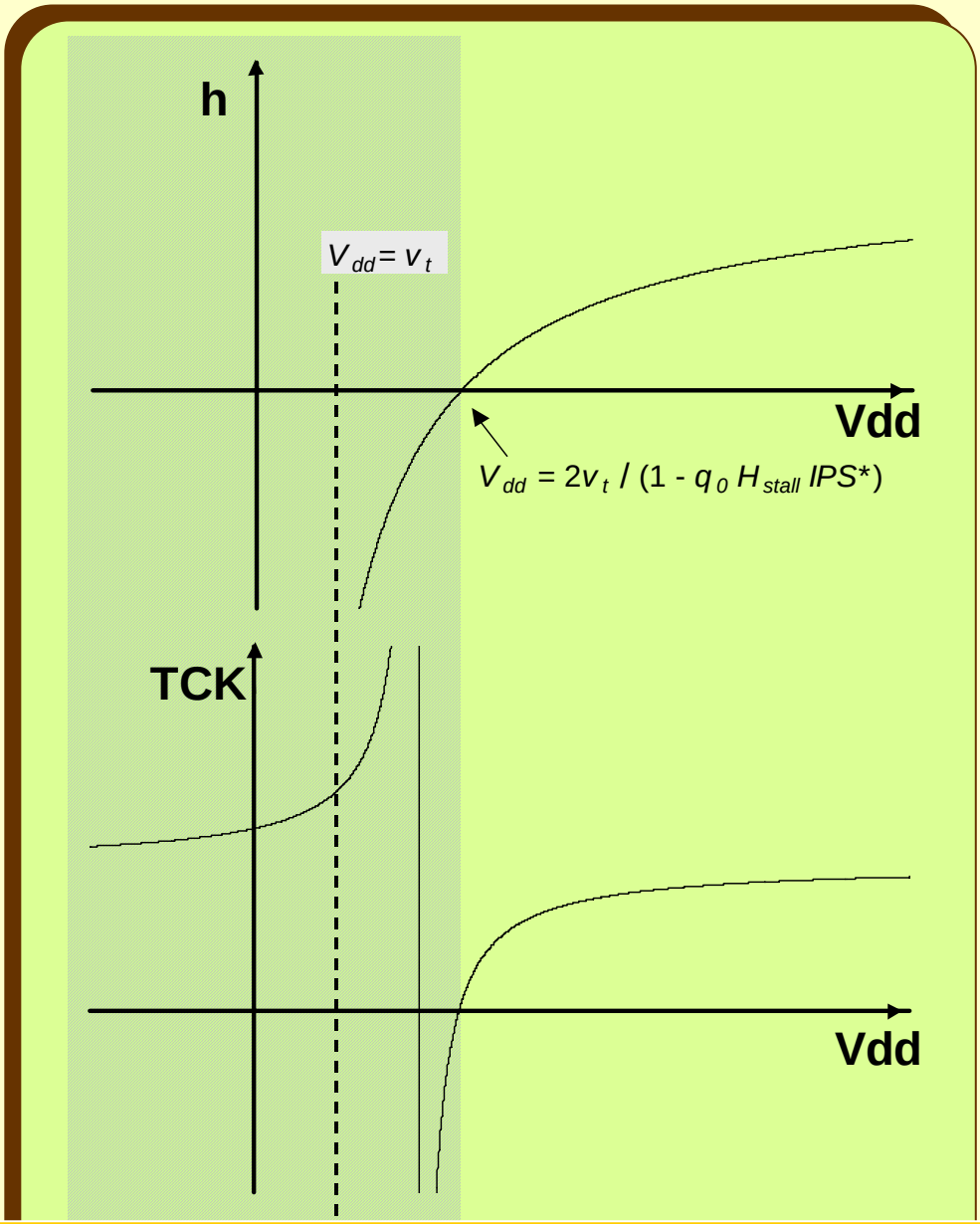
- Note: we neglect the effect of retiming (moving/adding pipe registers) on the effective capacitance. More accurate models can be developed.

“Second level” model results



h vs. V_{dd} ; expresses required pipe stage complexity for target IPS*

“Second level” model results

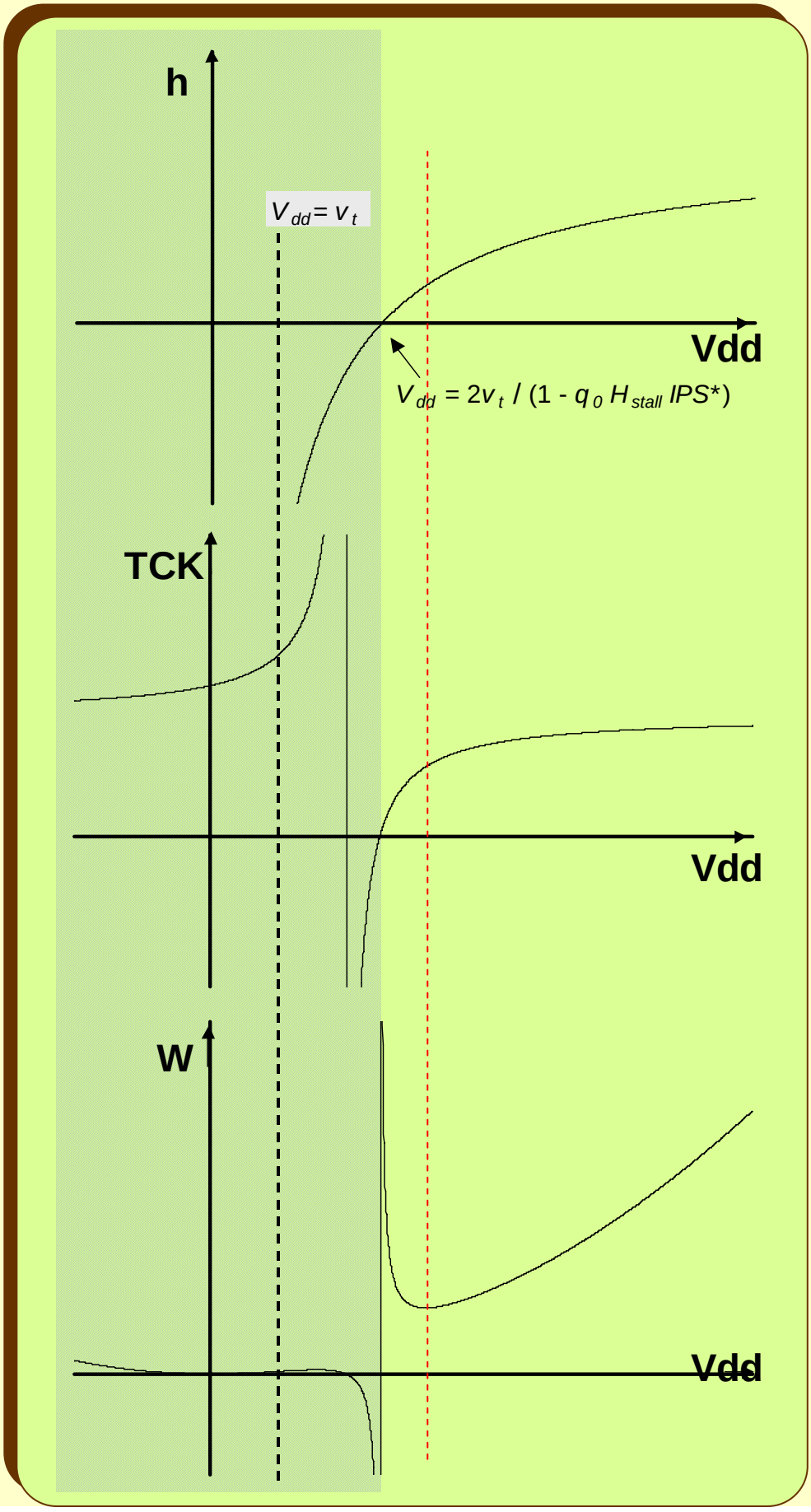


h vs. Vdd ; expresses required pipe stage complexity for target IPS*

TCK vs. Vdd ; expresses required TCK behavior, for target IPS*

W vs. Vdd ;

“Second level” model results



h vs. Vdd ; expresses required pipe stage complexity for target IPS*

TCK vs. Vdd ; expresses required TCK behavior, for target IPS*

W vs. Vdd ; expresses the resulting power dissipation behavior, for target IPS*

WHAT MORE? A LOT!

- Leakage power dissipation **must** be included!
  - A simple yet effective leakage model is needed
- Multi-core issues can be included!
  - Communication overhead
  - Local power down or frequency scaling of the cores
- Thermal effects can be included!
- How to integrate with multi-core simulators?

ASI2 short projects

- **Difficolta' bassa (da 0 a 1 pto)**
  - (Ricerca bibliografica) e realizzazione slide su circuiti digitali a CNTFET
- **Difficolta' media (da 0 a 2 pti, eccezionalmente 3 pti)**
  - Caratterizzazione leakage totale su librerie di celle e su macrocelle (ALU, memorie, ecc.) rispetto a Vdd in 45, 32, 22 nm (deterministico e MonteCarlo)
  - Caratterizzazione leakage totale su librerie di celle e su macrocelle (ALU, memorie, ecc.) rispetto a body biasing in 45, 32, 22 nm (deterministico e M.C.)
- **Difficolta' alta (da 0 a 3 pti, eccezionalmente 4 pti)**
  - ~~Sviluppo tool di post-processing di file VCD (generazione tracce)~~ *VENDUTO*
  - Sviluppo di tool di post processing di file VCD (consumo dinamico)
  - Sviluppo di tool di post processing di file VCD (consumo di leakage)
  - Modellazione VHDL di librerie di celle (consumo e ritardo)
  - Modellazione in VHDL di semplici core RISC (consumo dinamico e statico)
  - Modellazione SystemC o VHDL di semplici multi-core RISC (consumo din e stat)
  - Sintesi transistor-level di semplici core RISC (solo per gruppi motivati)