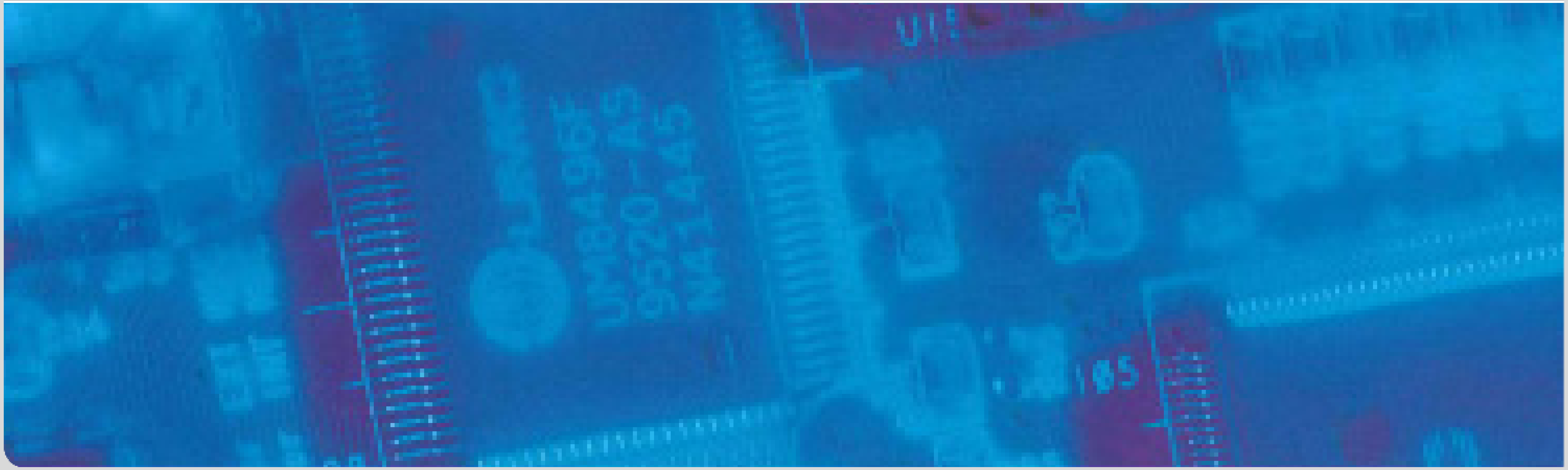
Karlsruhe Institute of Technology

Low Power Design

Volker Wenzel on behalf of Prof. Dr. Jörg Henkel
Summer Term 2016

CES – Chair for Embedded Systems



ces.itec.kit.edu

Lecture Slides Download

Overview Low Power Design Lecture

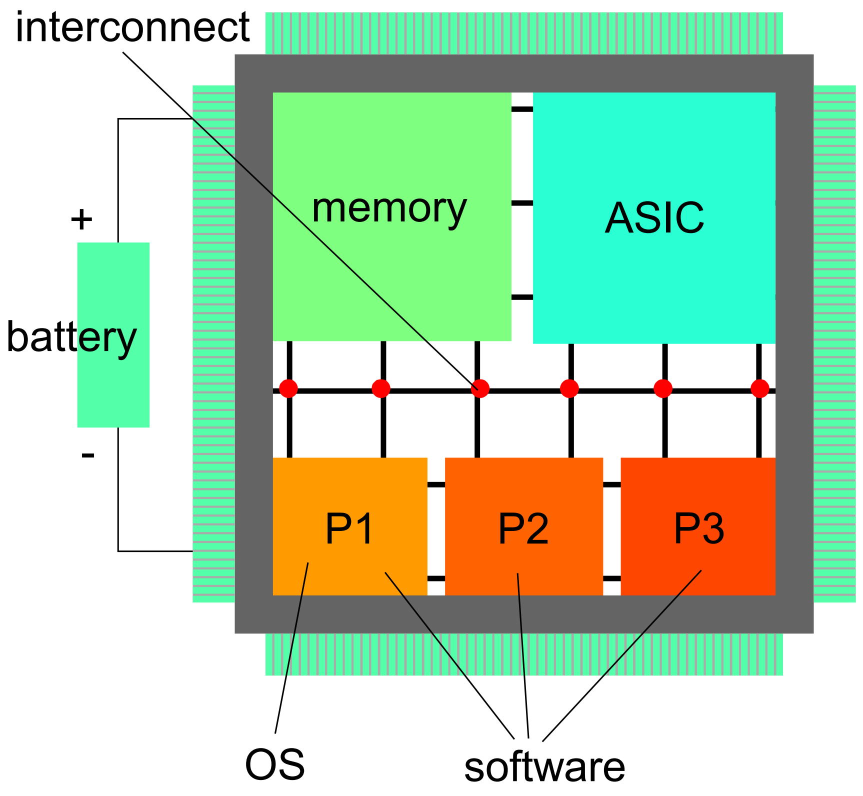
- Introduction and Energy/Power Sources (1)
- Energy/Power Sources(2): Solar Energy Harvesting
- Battery Modeling – Part 1
- Battery Modeling – Part 2
- Hardware power optimization and estimation – Part 1
- Hardware power optimization and estimation – Part 2
- Hardware power optimization and estimation – Part 3
- Low Power Software and Compiler
- Thermal Management – Part 1
- Thermal Management – Part 2
- Aging Mechanisms in integrated circuits
- Lab Meeting

Overview for today

- power consumption in HW
- operator scheduling

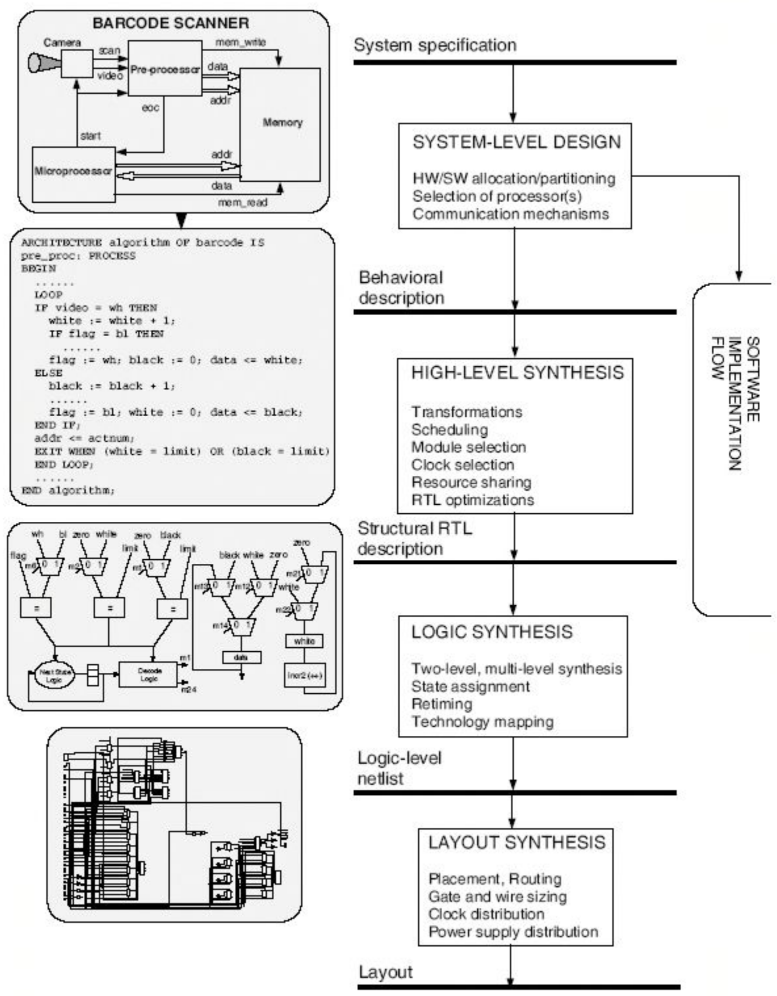
Overview

- Levels of abstraction
  - system
  - RTL
  - gate
  - transistor



- Challenges
  - optimize (ie. minimize for low power)
  - design /co-design (synthesize, compile, …)
  - estimate and simulate

Generic HW synthesis flow



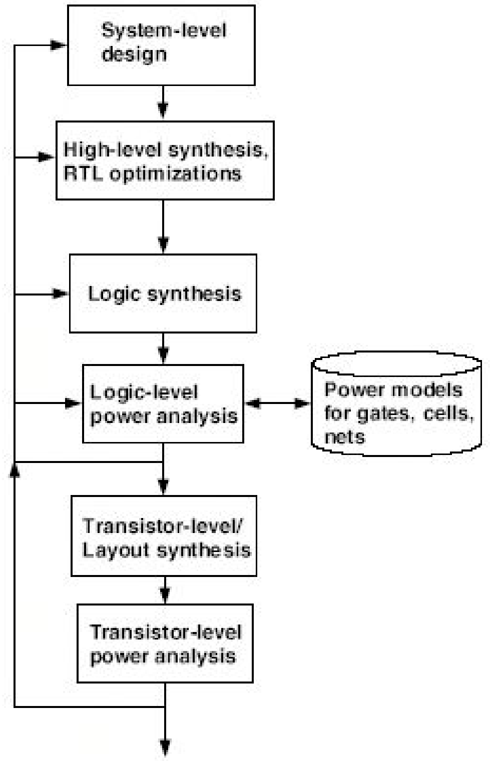
```
ARCHITECTURE algorithm OF barcode IS
pre_proc: PROCESS
BEGIN
  .......
  LOOP
  IF video = wh THEN
    white := white + 1;
    IF flag = bl THEN
      .......
    flag := wh; black := 0; data <= white;
  ELSE
    black := black + 1;
    .......
    flag := bl; white := 0; data <= black;
  END IF;
  addr <= actnum;
  EXIT WHEN (white = limit) OR (black = limit)
  END LOOP;
  .......
END algorithm;
```

- System-Level Design
- High-Level Synthesis
- Logic Synthesis
- Layout Synthesis

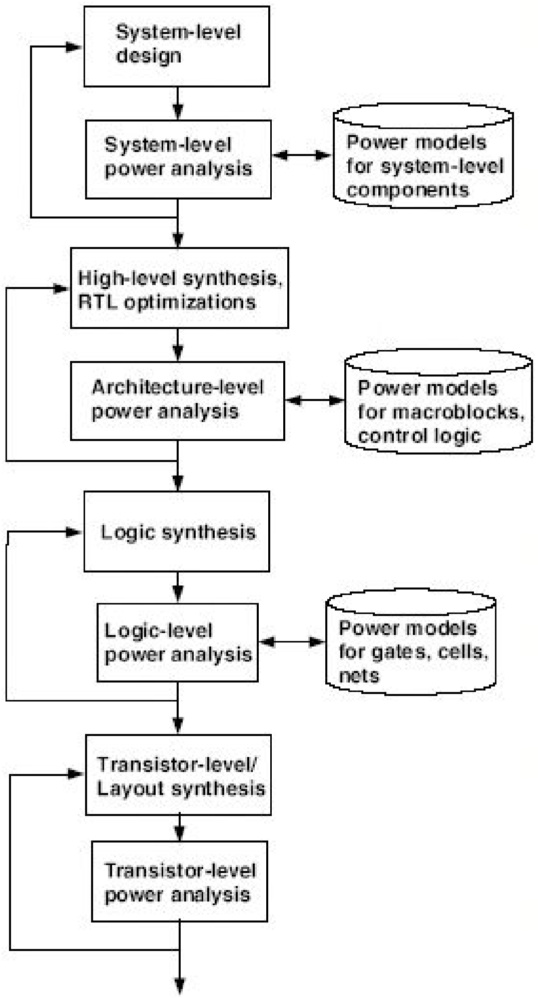
(src.: [Anand98])

Low power HW design flow

- Energy/power needs to be analyzed/optimized at each level of abstraction
- Necessitates appropriate models for each level


System-level design
High-level synthesis, RTL optimizations
Logic synthesis
Logic-level power analysis
Power models for gates, cells, nets
Transistor-level/ Layout synthesis
Transistor-level power analysis

design flow
w/o energy/power


System-level design
System-level power analysis
Power models for system-level components
High-level synthesis, RTL optimizations
Architecture-level power analysis
Power models for macroblocks, control logic
Logic synthesis
Logic-level power analysis
Power models for gates, cells, nets
Transistor-level/ Layout synthesis
Transistor-level power analysis

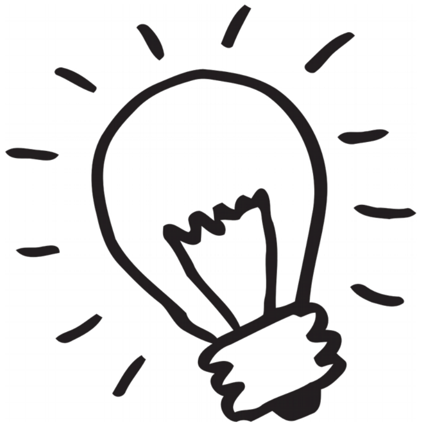
design flow
w energy/power

(src.: [Anand98])

Power consumption in HW

$$P_{avg} = P_{sw.cap.} + P_{short-circuit} + P_{leakage} + P_{static}$$

- In general, four components:
  - $P_{sw.cap}$ switching capacity power
  - $P_{short\text{-}circuit}$ short-circuit power
  - $P_{leakage}$ leakage power
  - P_{static} static power



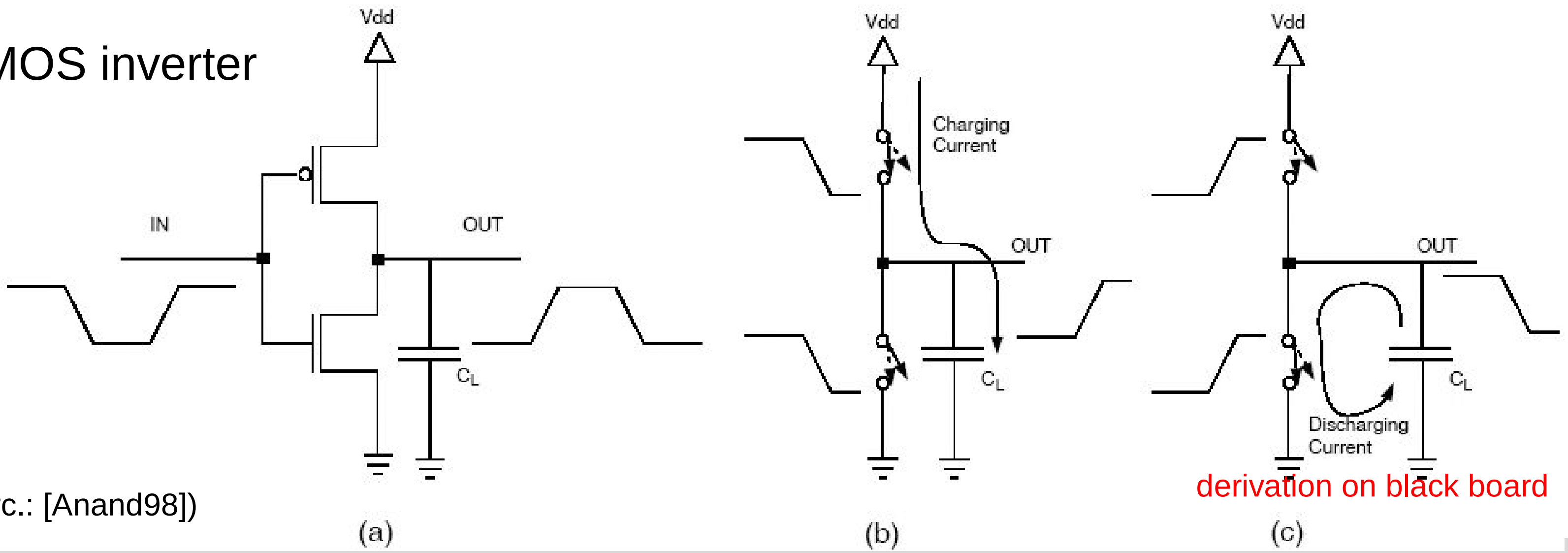
(src.: [Anand98])

Switching capacity power

- Caused by charging/discharging of parasitic capacitances
  - C_L: cumulative parasitic capacitance
  - N: expected # of transitions per clock cycle
  - f: clock frequency

$$P_{sw.cap} = \frac{1}{2} C_L V_{dd}^2 N f$$

CMOS inverter



derivation on black board

(src.: [Anand98])

Parasitic capacitance

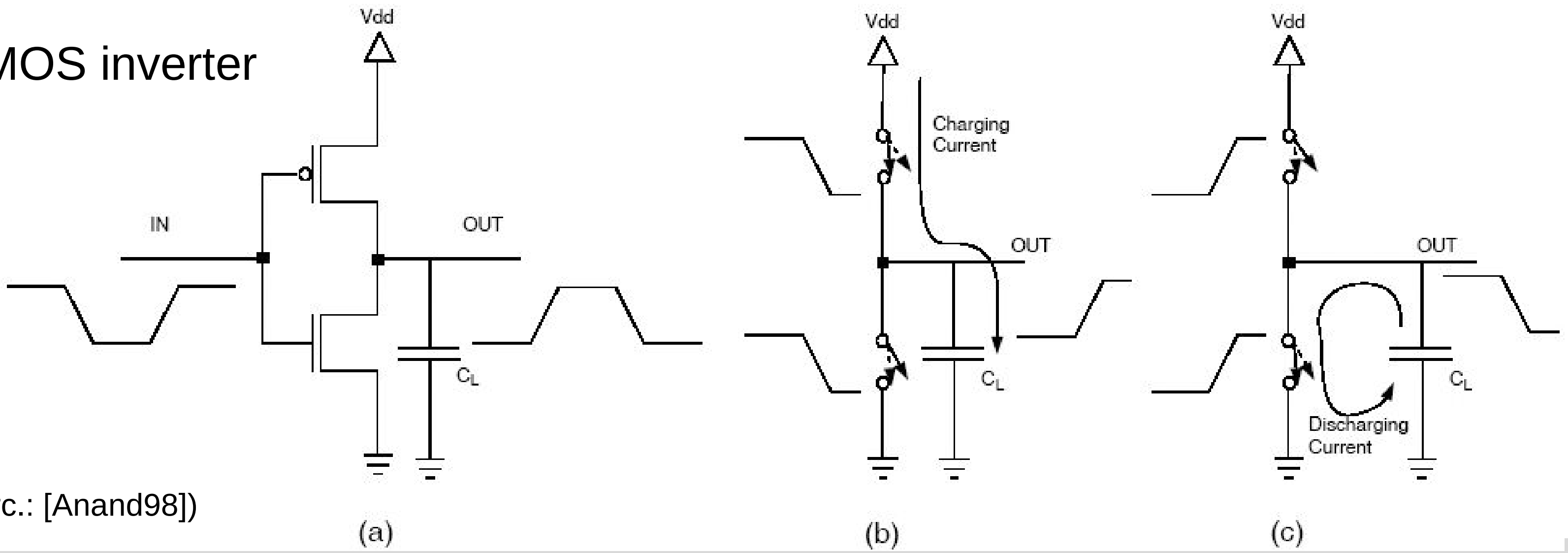
- aka **stray capacitance**
- exists between all electronic components due to proximity
- exist between conductors
- problematic in high frequency circuits

Short-circuit power

- Caused by direct supply-to-ground path
  - K: constant (transistor size, – technology)
  - V_T: threshold voltage
  - τ: input rise/fall time

$$P_{short-circuit} = K\left(V_{dd} - 2V_T\right)^3 \tau N f$$

CMOS inverter



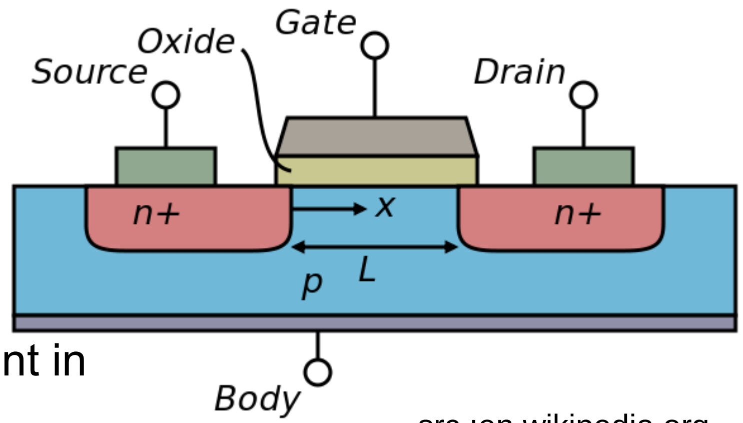
(src.: [Anand98])

Leakage power

$$P_{leakage} = (I_{subthreshold} + I_{oxide} + I_{diode}) \cdot V_{dd}$$

- I_{diode}: diodes formed between diffusion region and substrate
- I_{oxide}: electrons tunneling through the gate oxide
  - drops exponenitally with gate length

$$I_{subthreshold} = K W_{eff} e^{\frac{V_{in} - V_{T_n}}{S}}$$



- ‘off’ transistors still conduct some current
- K, S, technology parameters
- W_{eff} effective transistor channel width
- NOTE: leakage power is predicted to be dominant in future silicon technologies

(src.: [Anand98])

src.:en.wikipedia.org

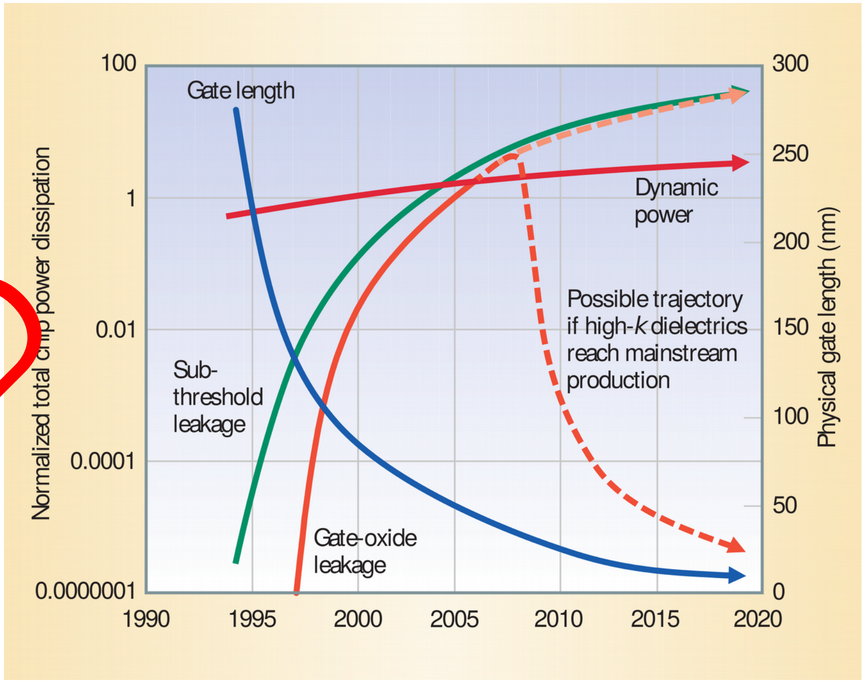
Static power

- not relevant in CMOS circuits
- Note: in some literature leakage power is denoted as “static power”
- relevant in
  - other logic families
  - some nMOS circuits where there is a constant path supply-to-ground (e.g. ECL)

Breakdown of power consumption in HW

- “Leakage power will dominate in future (<100nm) silicon technologies“
- one means to reduce leakage power is to deploy dielectrics with a high k-value

OUTDATED



(src.: [Kim])

Mark Bohr IDF14 stolen slides

- http://www.intel.com/content/dam/www/public/us/en/documents/pdf/foundry/mark-bohr-2014-idf-presentation.pdf

Hardware synthesis for low power

- Considered here: high-level synthesis (HLS) e.g.:
  - Operator scheduling
  - Module selection
  - Glitch power reduction
  - State transition reduction
  - …

Operator scheduling for low power

Scheduling (in the context of high-level synthesis)

- assigns operations in the behavioral description to control steps or controller states.
- determines cycle-by-cycle behavior (i.e. sequence in which operations are performed)
- determines the sequence in which the various operations of the behavioral description are performed
- dictates which operations and variables can share the same functional units and registers.
- can be used to enable resource sharing for low power by ensuring that correlated variables and operations with correlated operands are appropriately sequenced so that they can share the same resources

Some repetition from ESI:

- multicycling (operation can take more than one cycle)
- chaining (multiple operations executed in one cycle)

(src.: [Anand98])

Operator scheduling for low power (cont'd)

Scheduling can be performed so as to enable maximum resource sharing between operations that belong to instances of the same computational pattern, resulting in maximal exploitation of regularity during resource sharing

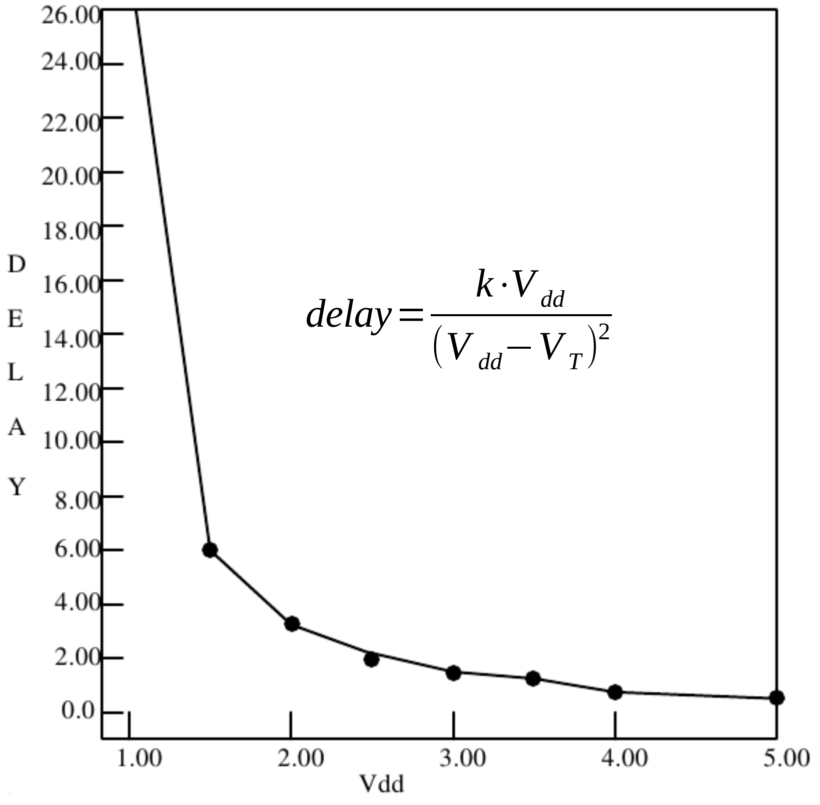
Scheduling can be used to distribute the slacks or mobilities of various operations in the DFG appropriately so that some operations may be performed using slower, more energy-efficient functional units. Thus, scheduling has an impact on the power trade-offs through module selection

Scheduling determines the distribution of operations over time, and hence affects the profile of the power consumption in the implementation over time (control steps or clock cycles). **Reducing** peak power is important due to packaging, cooling, and reliability considerations. The effect of scheduling on peak power will be illustrated later.

(src.: [Anand98])

Recap: Delay vs. V_{dd}

normalized



$$delay = \frac{k \cdot V_{dd}}{(V_{dd} - V_T)^2}$$

[V]

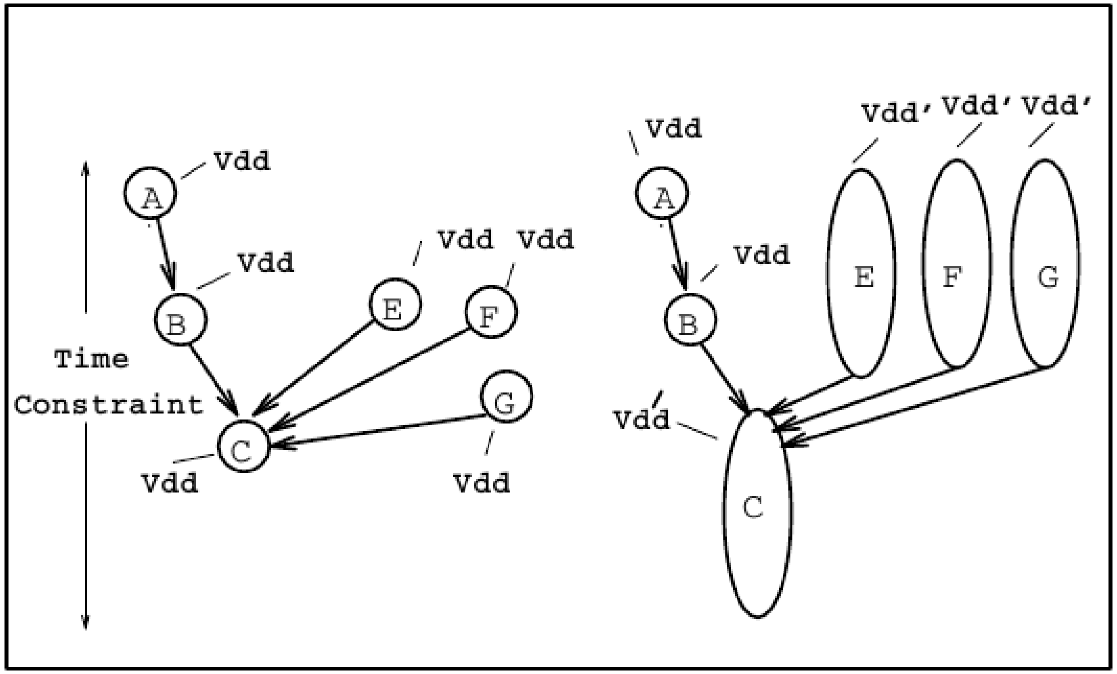
$$P_{dyn} \sim V_{dd}^2 \, C_{load} \, f_{switch}$$

Problem:
How to assign voltages to functional units?

(src.: [Sarraf95])

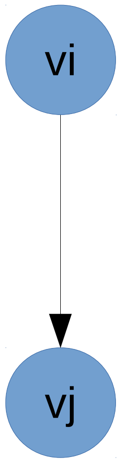
Normalized Delay vs Vdd

Recap: Data Flow Graphs (DFGs)



- DFG: directed acyclic graph G = (V,E)
- v_i must preced v_j : directed edge

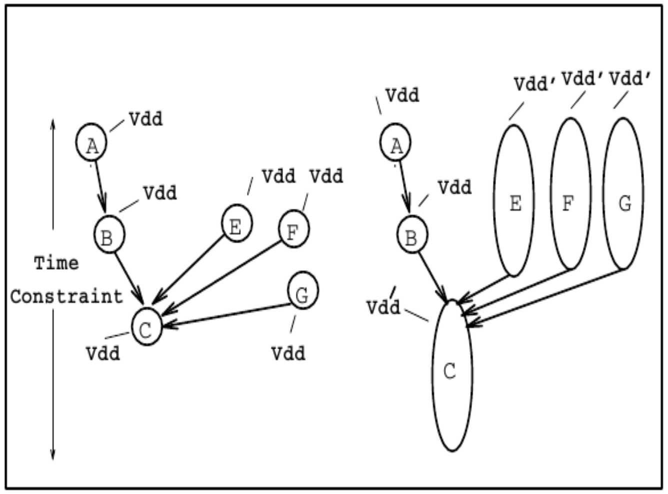
  from v_i to v_j



(src.: [Sarraf95])

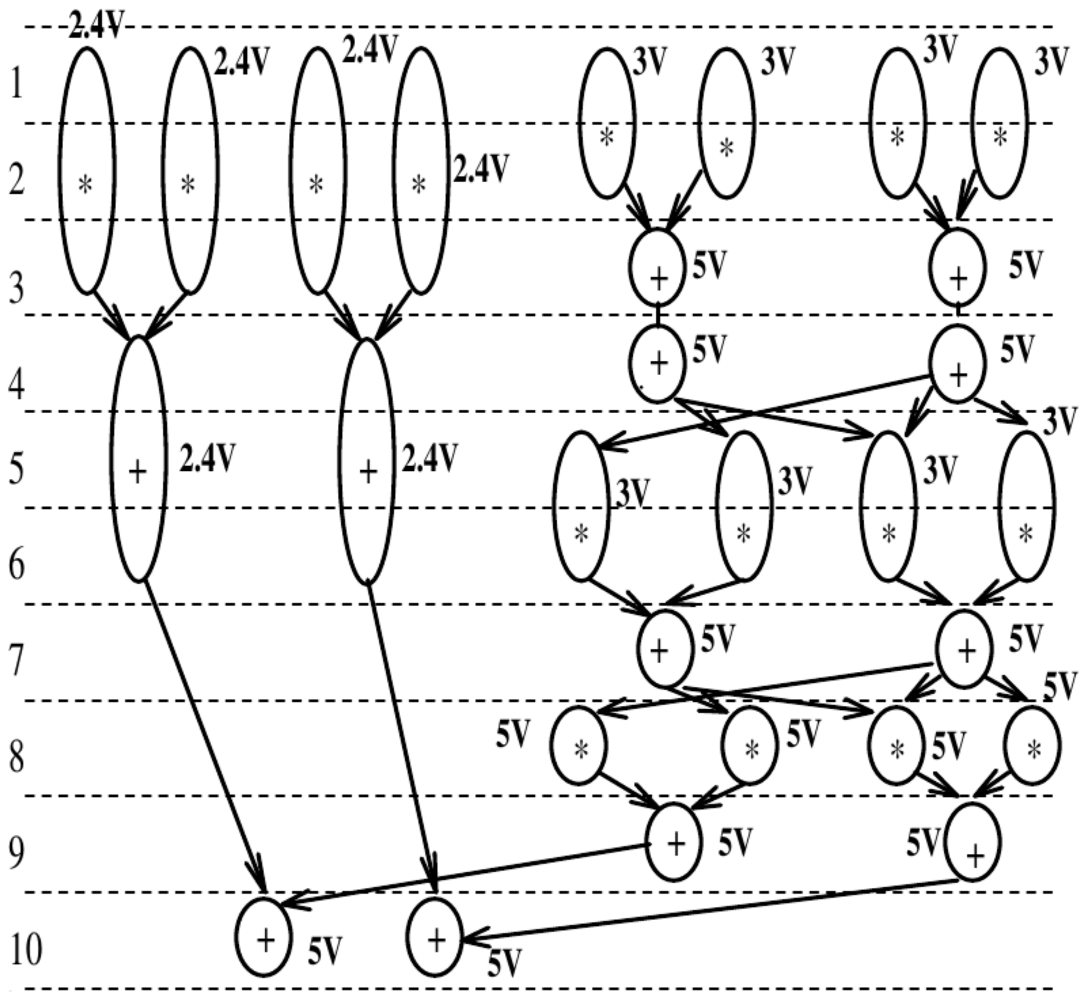
Definition: Critical Path

„A critical path of a system is defined as the path in the DFG, $\{v_1,v_2,\ldots,v_k\}$, such that the summation of the latencies of the nodes in the path is maximal among all the paths of the DFG. The sum C_p is termed as the critical path length“



Motivation

How do we know the optimal voltages?



Behavioral Synthesis = High Level Synthesis

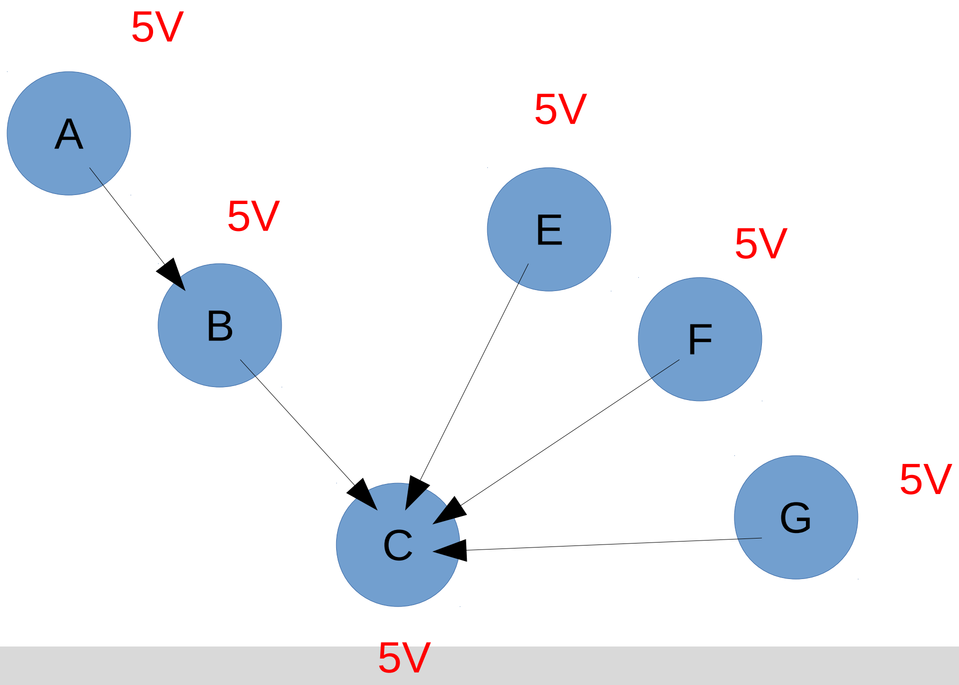
Algorithm Overview

- Step 1: Initialization
- Step 2: Computation of Slack
- Step 3: Maximal slack value
- Step 4: Dual graph Hp
- Step 5: Weight Assignment
- Step 6: Longest weighted path
- Step 7: Reassigning voltages to nodes in the longest path
- Step 8: Goto Step 2

Step 1 : Initialization

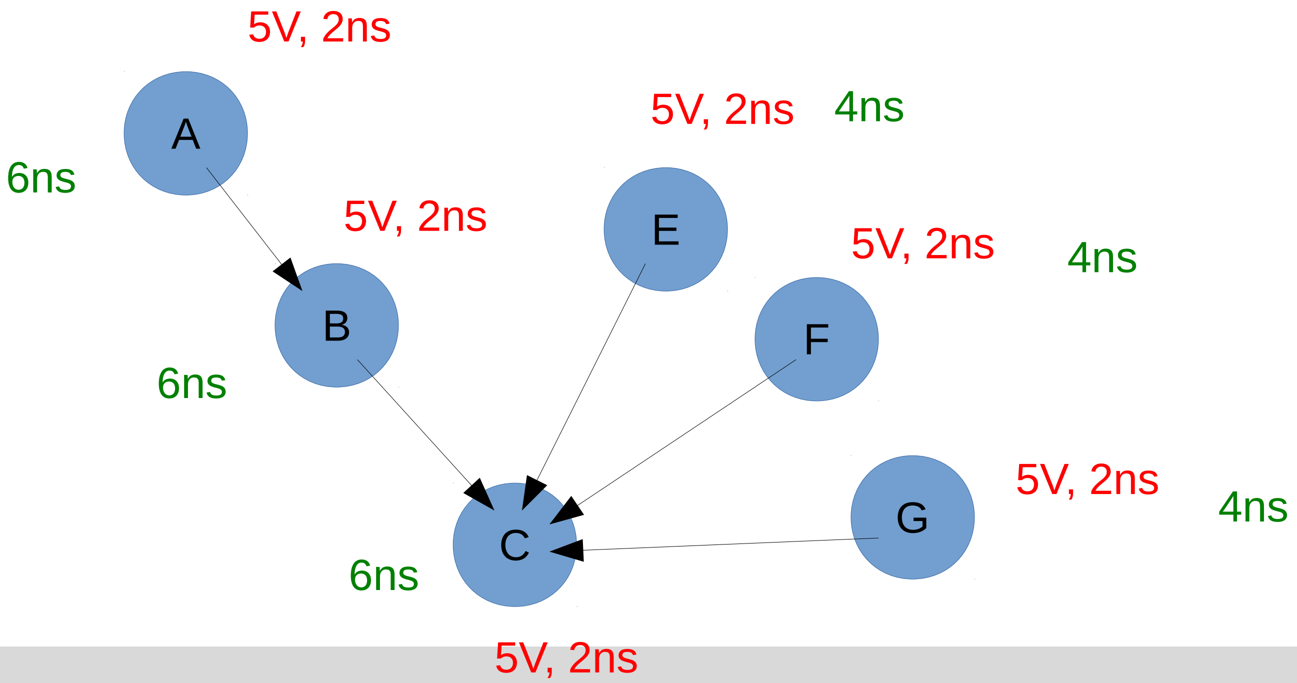
Step 1: *Initialization*
Each of the nodes, $v \in V$, in $G(V,E)$ is originally assigned a voltage V_c (also denoted by V_{c_0}), $\tau(v) = V_c$. $d(v)$ value is therefore initialized.

$$S = \{5\,\mathrm{V}, 3\,\mathrm{V}, 2.4\,\mathrm{V}\}$$

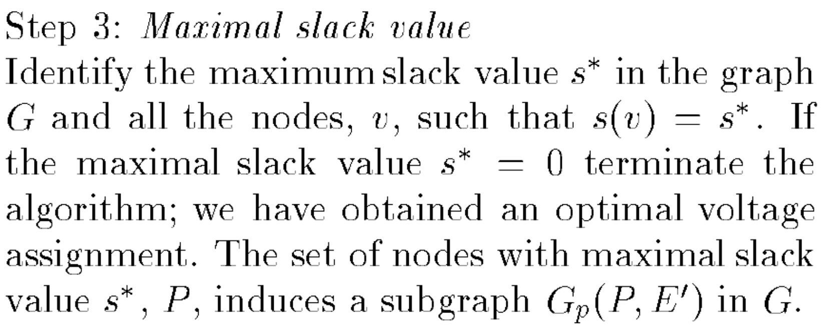


Step 2: Computation of slack

Step 2: *Computation of slack*
Using *depth first search* calculate $l(v)$ values for each of the nodes $v \in V$ and hence obtain $s(v) = kt_c - l(v)$.

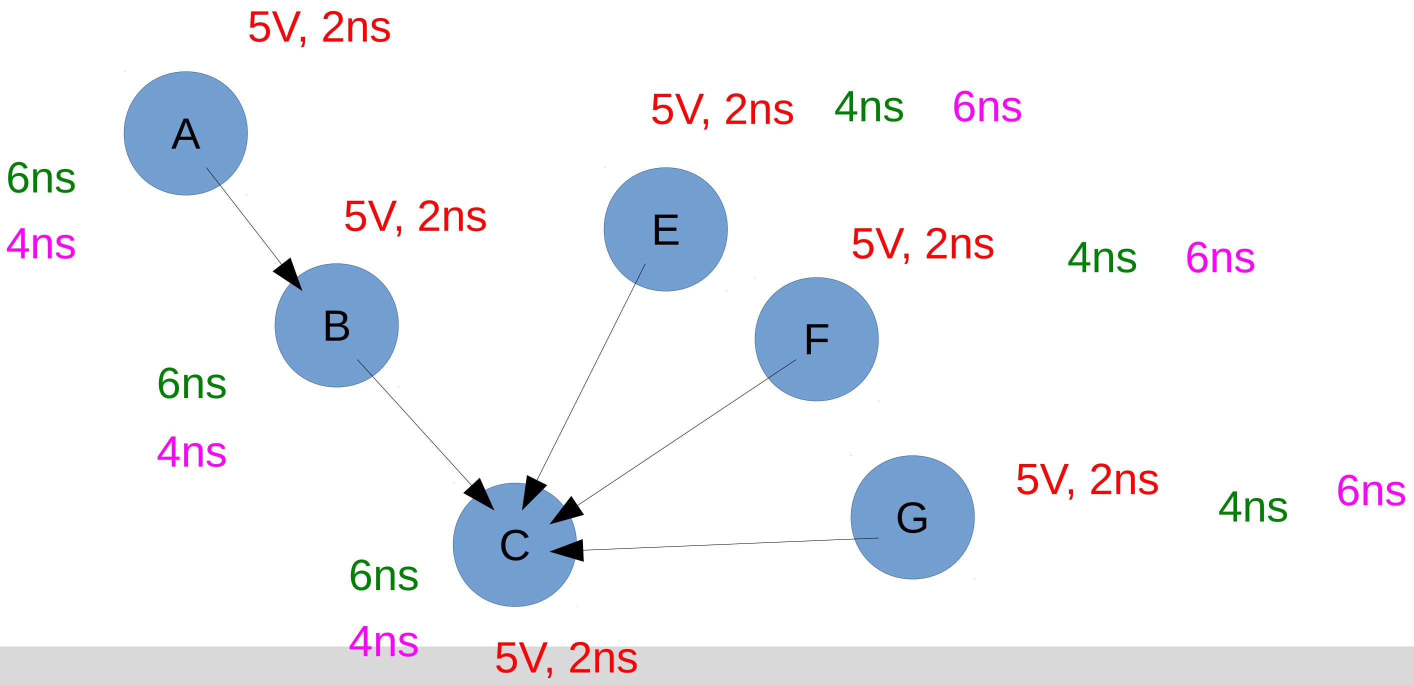


Step 3: Maximal slack value



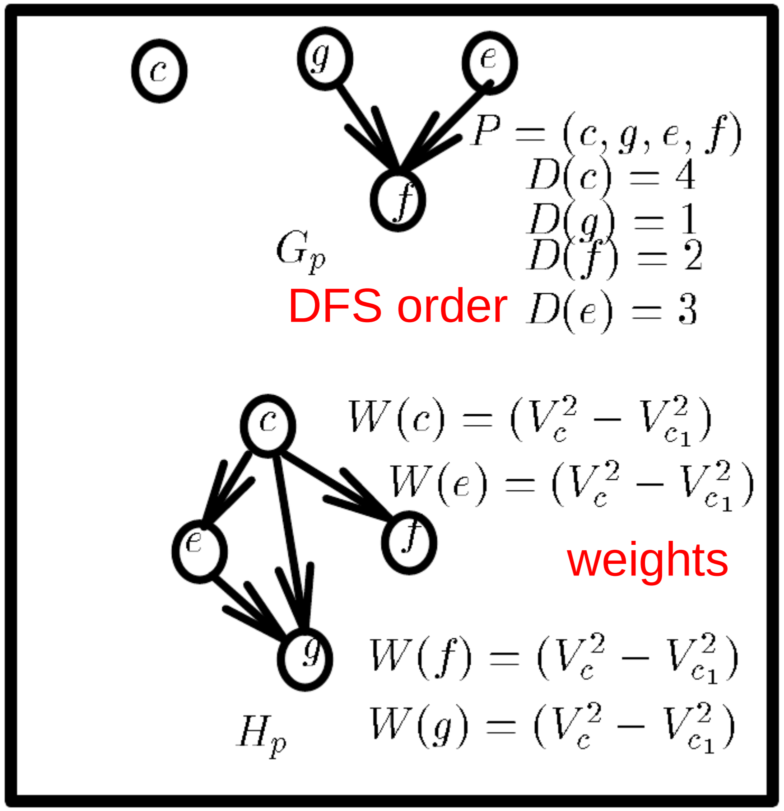
Step 3: *Maximal slack value*
Identify the maximum slack value s^* in the graph G and all the nodes, v, such that $s(v) = s^*$. If the maximal slack value $s^* = 0$ terminate the algorithm; we have obtained an optimal voltage assignment. The set of nodes with maximal slack value s^*, P, induces a subgraph $G_p(P, E')$ in G.

Assume: ktc = 10ns



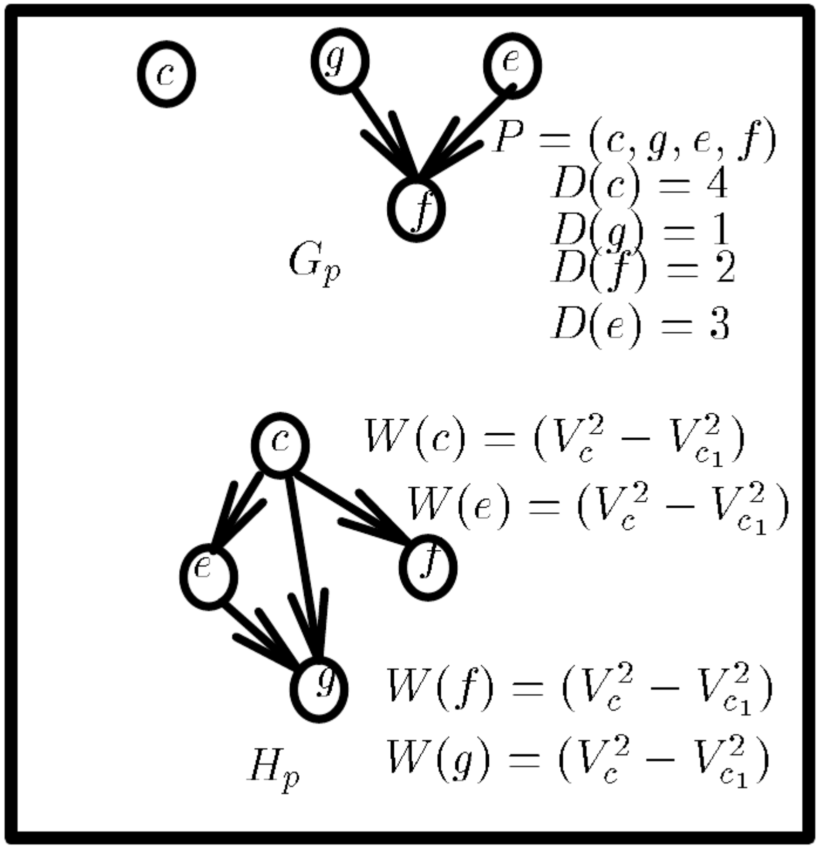
Step 4: Dual graph of H_p

Step 4: *Dual graph* H_p
A dual graph $H_p(P, E_p)$ is obtained for the graph $G_p(P, E')$: Obtain a Depth-First Search (DFS) ordering of the graph G_p. Let $D(v)$ be the order in which the node v is visited during the DFS. The dual graph, H_p, is constructed on the node set P. If $u, v \in P$ and there does not exist a path from v to u and from u to v in G_p, then if $D(u) > D(v)$ then $(u, v) \in E_p$.



c
g
e
f
$P = (c, g, e, f)$
$D(c) = 4$
$D(g) = 1$
$D(f) = 2$
G_p
DFS order
$D(e) = 3$
$W(c) = (V_c^2 - V_{c_1}^2)$
$W(e) = (V_c^2 - V_{c_1}^2)$
weights
$W(f) = (V_c^2 - V_{c_1}^2)$
H_p
$W(g) = (V_c^2 - V_{c_1}^2)$

Step 5: Weight Assignment



c
g
e
f
$P = (c, g, e, f)$
$D(c) = 4$
$D(g) = 1$
$D(f) = 2$
$D(e) = 3$
G_p
$W(c) = (V_c^2 - V_{c_1}^2)$
$W(e) = (V_c^2 - V_{c_1}^2)$
$W(f) = (V_c^2 - V_{c_1}^2)$
$W(g) = (V_c^2 - V_{c_1}^2)$
H_p

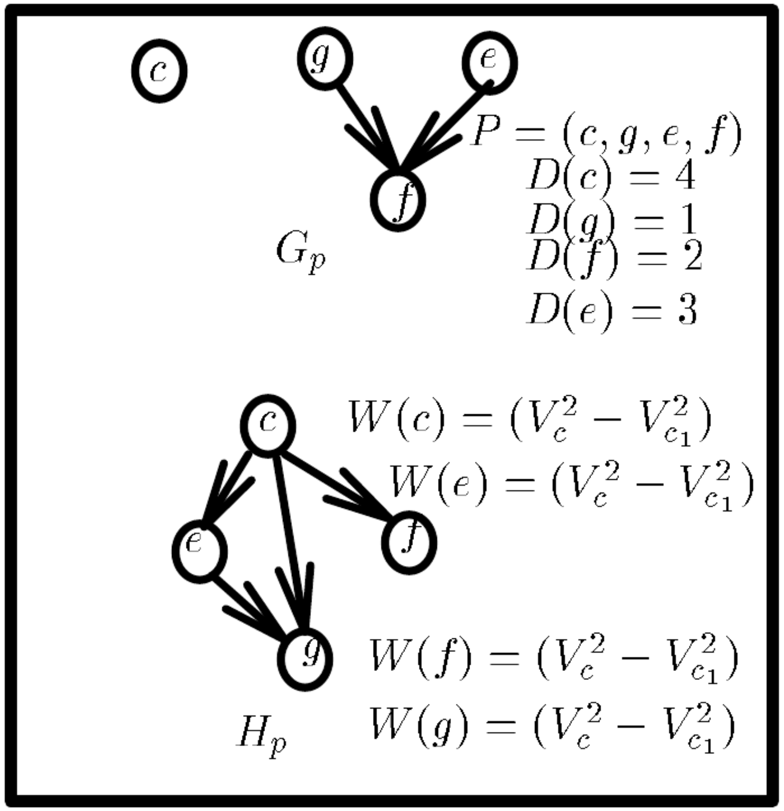
Step 5: *Weight assignment*
If a node $v \in P$ is assigned a voltage $\tau(v) = V_{c_k}$ then assign a weight $W(v) = (V_{c_k}^2 - V_{c_{k+1}}^2)$ in H_p.

$P_{dyn} \sim V_{dd}^2 C_{load} f_{switch}$

Step 6: Longest Weighted Graph

Step 6: *Longest weighted path*
Obtain a longest weighted path in H_p. The longest weighted path in a directed acyclic graph is defined as the path in the graph which has the

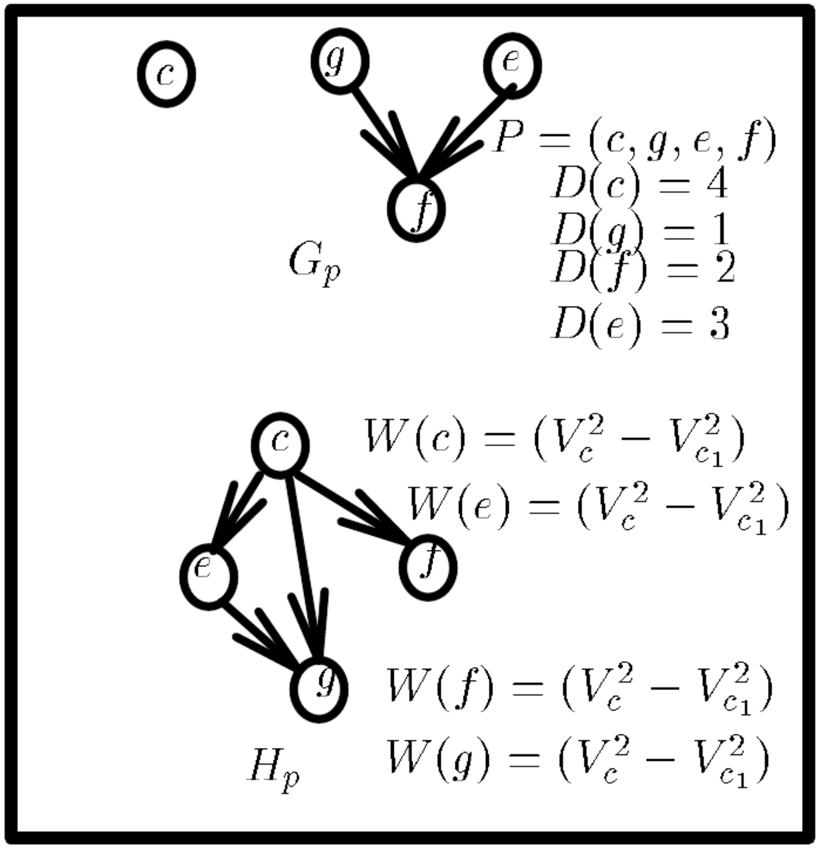
maximum total node weight and it can be obtained using a single *breadth first search*.



G_p
$P = (c, g, e, f)$
$D(c) = 4$
$D(g) = 1$
$D(f) = 2$
$D(e) = 3$
H_p
$W(c) = (V_c^2 - V_{c_1}^2)$
$W(e) = (V_c^2 - V_{c_1}^2)$
$W(f) = (V_c^2 - V_{c_1}^2)$
$W(g) = (V_c^2 - V_{c_1}^2)$

Step 7: Reassigning voltages to nodes in the longest path

Step 7: *Reassigning voltages to nodes in the longest path*
Reassign voltages to nodes in the longest weighted path obtained in the previous step. If a node in the longest path, v, has a prior voltage assignment $\tau(v) = V_{c_k}$ then change this assignment to $\tau(v) = V_{c_{k+1}}$. The new assignment of voltages to nodes in P changes the delay values ($d(v)$ values) for nodes.


c
g
e
f
G_p
$P = (c, g, e, f)$
$D(c) = 4$
$D(g) = 1$
$D(f) = 2$
$D(e) = 3$
c
e
f
g
H_p
$W(c) = (V_c^2 - V_{c_1}^2)$
$W(e) = (V_c^2 - V_{c_1}^2)$
$W(f) = (V_c^2 - V_{c_1}^2)$
$W(g) = (V_c^2 - V_{c_1}^2)$

Step 8

- Goto Step 2

Theorem 1: Correctness

Theorem 1 *Given a set of allowable voltages* S *and* data flow graph $G(V, E)$, *the above algorithm produces a mapping* $\tau : V \rightarrow S$ *that minimizes* $\sum_{v_i \in V} \tau(v_i)^2$.

$P_{dyn} \sim V_{dd}^2\, C_{load}\, f_{switch}$

no proof in this lecture!

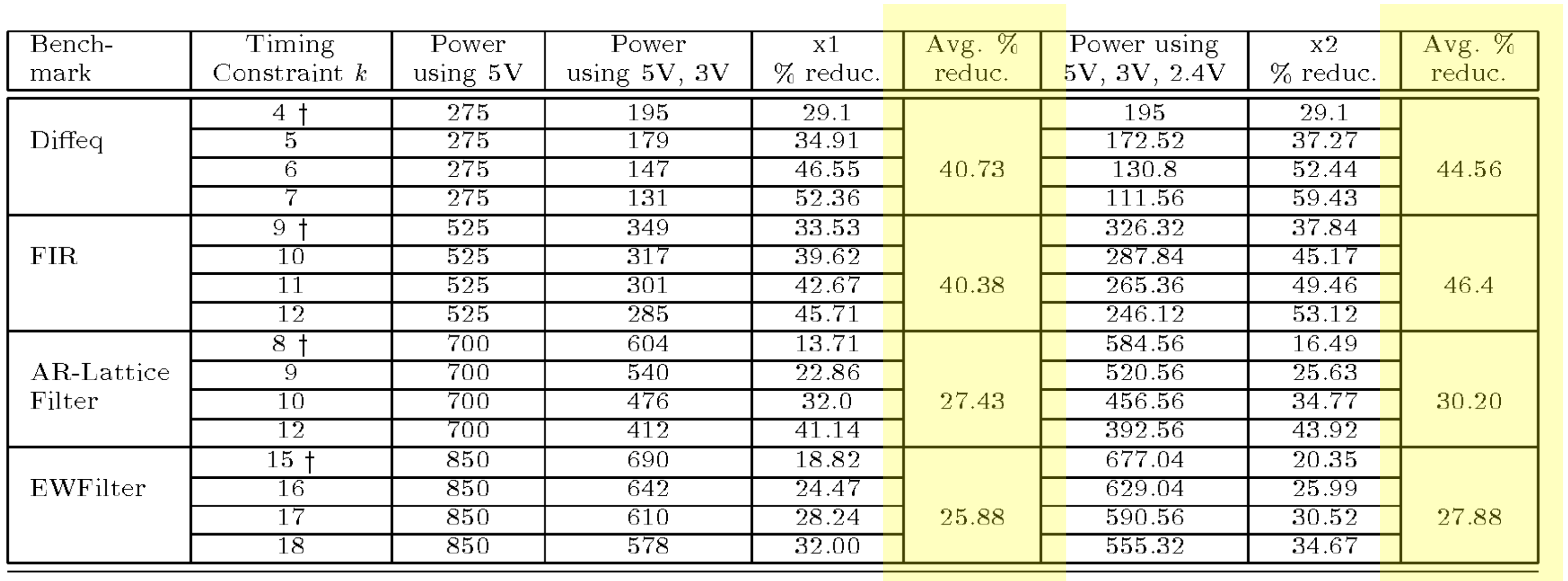
Theorem 2: Complexity

Theorem 2 *The time complexity of the algorithm is* $O(kn^2)$*, where* kt_c *is the timing constraint and* n *is the number of nodes in* G*.*

no proof in this lecture!

Experimental Results

Experimental Setup: Simulation using High-Level Synthesis Benchmark Tools



Bench-mark	Timing Constraint k	Power using 5V	Power using 5V, 3V	x1 % reduc.	Avg. % reduc.	Power using 5V, 3V, 2.4V	x2 % reduc.	Avg. % reduc.
Diffeq	4 †	275	195	29.1	40.73	195	29.1	44.56
	5	275	179	34.91		172.52	37.27	
	6	275	147	46.55		130.8	52.44	
	7	275	131	52.36		111.56	59.43	
FIR	9 †	525	349	33.53	40.38	326.32	37.84	46.4
	10	525	317	39.62		287.84	45.17	
	11	525	301	42.67		265.36	49.46	
	12	525	285	45.71		246.12	53.12	
AR-Lattice Filter	8 †	700	604	13.71	27.43	584.56	16.49	30.20
	9	700	540	22.86		520.56	25.63	
	10	700	476	32.0		456.56	34.77	
	12	700	412	41.14		392.56	43.92	
EWFilter	15 †	850	690	18.82	25.88	677.04	20.35	27.88
	16	850	642	24.47		629.04	25.99	
	17	850	610	28.24		590.56	30.52	
	18	850	578	32.00		555.32	34.67	

Table 1: Power Consumption Results for smaller Timing Constraints
†: Corresponds to the longest path length for the *DFG*.

Experimental Results (Cont'd)

Bench-mark	Timing Constraint k	Power using 5V	Power using 5V, 3V	x1 % reduc.	Avg. % reduc.	Power using 5V, 3V, 2.4V	x2 % reduc.	Avg. % reduc.
	8	275	99	64		82.8	69.89	
Diffeq	12	275	99	64	64	63.36	76.96	73.43
	18	525	189	64		150.12	71.41	
FIR	27	525	189	64	64	120.96	76.96	74.19
AR-Lattice	16	700	252	64		232.56	66.77	
Filter	24	700	252	64	64	161.28	76.96	71.87
	30	850	306	64		280.08	67.05	
EWFilter	45	850	306	64	64	195.84	76.96	72.00

Table 2: Power Consumption Results for larger Timing Constraints

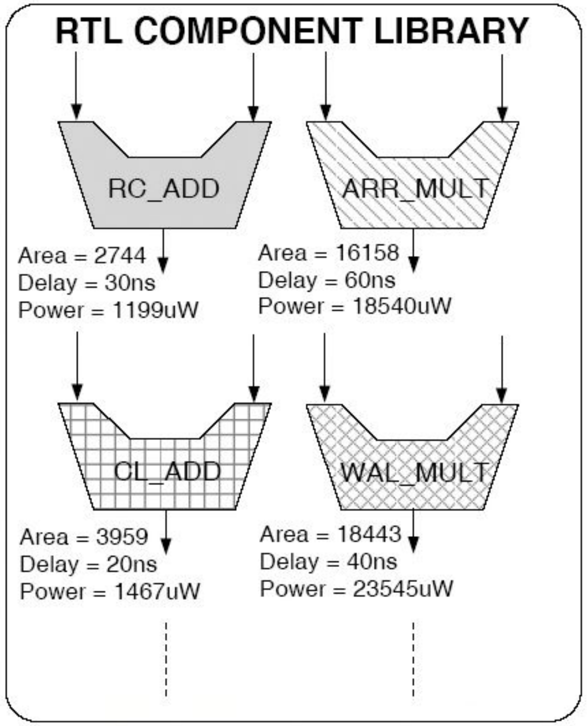
higher timing constraints here

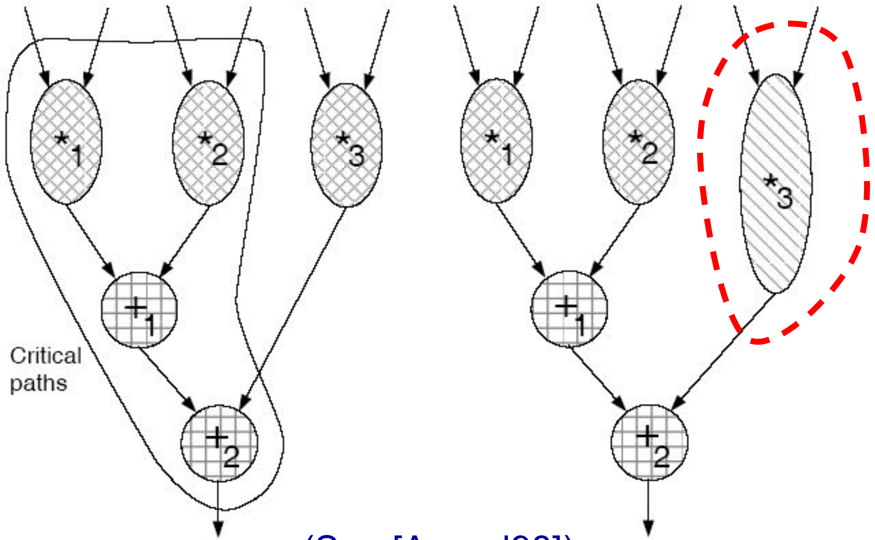
Module Selection

Module Selection

- ... is the process of mapping operations from the CDFG to component templates from the RTL library.
- Initially, only a **functional unit template**, not a specific instance, associated with each operation
- Example: „+“-Operation may be implemented using
  - ripple-carry adder (slower, but more switched capacitance efficient)
  - carry-lookahead adder (faster, incures higher switched capacitance)
  - carry-select adder
  - ...
- similar tradeoffs for other operations
- Idea: Exploit tradeoffs to fulfill power constraints through module selection

Module selection for LP (cont'd)





(Src: [Anand98])

- Each operation in the DFG (middle) has been mapped to fast component in order to meet performance constraints (in that case constraint: 85ns)
- But is that really necessary? => no, not all ops need necessarily be mapped to fastest module. Focus (for timing constraint) should rather be on critical path
- Idea: slack in off-critical path ops may be used to select slower functional units that may have a better efficiency in switched capacitance (see right DFG). There, mult op uses less power (but not less energy)
- Important: to have a large module library with distinct switching capacity efficiencies and performance characteristics

Summary

- Power consumption in HW:

$$P_{avg} = P_{sw.cap.} + P_{short-circuit} + P_{leakage} + P_{static}$$

- Operator Scheduling:
  - reread [Sarraf95] at home;
  - understanding notation takes a bit of time

issues in [Sarraf95]:

- τ is used for different things
- Φ should be substituted by ∅
- 'integral' → 'integer'

Sources

[Heer04] Ch. Herr, U. Schlichtmann, “Ultra-Low-Power Design: Device and logic design approaches”, pp. 1-20, in “Ultra Low-Power Electronics and Design” by Kluwer, 2004.

[Anand98] A. Raghunathan, N.K. Jha, S. Dey, “High-level power analysis and optimization”, Kluwer Academic Publishers,1998.

[Sarraf95] S. Raje, M. Sarrafzadeh, “Variable voltage scheduling”, IEEE/ACM ISLPED 1995. pp. 9-14, 1995.

[Knight] R.S. Martin, J.P. Knight, “Power-Profiler: Optimizing ASIC’s Power Consumption at the behavioral level”, Proc. Of IEEE/ACM Design Automation Conf. (DAC’95), pp.42-47,1995.

[Macii04] E. Macii (Ed.), “Ultra Low-Power Electronics and Design”, Kluwer Academic Publishers, 2004.

[Devadas] Alidina, M.; Monteiro, J.; Devadas, S.; Ghosh, A.; Papefthymiou, M.; “Precomputation-based Sequential Logic Optimization For Low Power”, Computer-Aided Design (ICCAD), 1994., IEEE/ACM International Conference on November 6-10, 1994 Page(s):74 – 81.

[Ragh99] Raghunathan, A.; Dey, S.; Jha, N.K.; “Register transfer level power optimization with emphasis on glitch analysis and reduction”, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 18, Issue 8, Page(s):1114 – 1131, Aug. 1999.

[Tivari] Tiwari, V.; Malik, S.; Ashar, P.; “Guarded evaluation: pushing power management to logic synthesis/design”, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 17, Issue 10, Page(s):1051 – 1060, Oct. 1998.

[Mehra] R. Mehra, J. Rabaey, “Exploiting Regularity for Low Power Design”, IEEE/ACM Intl’ Conference on Computer Aided Design (ICCAD96), pp. 166-172, 1996.

[Keating] Keating, Michael, David Flynn, Rob Aitken, Alan Gibbons, and Kaijian Shi. Low power methodology manual: for system-on-chip design. Springer Publishing Company, Incorporated, 2007.

[Kim] Kim, N.S.; Austin, T.; Baauw, D.; Mudge, T.; Flautner, K.; Hu, J.S.; Irwin, M.J.; Kandemir, M.; Narayanan, V., "Leakage current: Moore's law meets static power," Computer , vol.36, no.12, pp.68,75, Dec. 2003