Low Power Systems Design

School of EECS
Seoul National University

Introduction

- **Low power design**
  - **Increasing demand on performance and integrity of VLSI circuits**
  - **Popularity of portable devices**
- **Low power design at higher levels of abstraction**
  - **Faster design space exploration**
  - **Wider view**
  - **Higher power reduction**
  - **Less cost increase**

– **Opportunities for power reduction at every level of abstraction**

System	**50-90%**	**algorithms, HW-SW tradeoffs, supply voltage scaling**
Architecture	**40-70%**	**scheduling, resource binding, operand swapping**
Register-Transfer	**30-50%**	**clock gating, operand isolation, pre-computation, dynamic operand interchange, FSM encoding, bus encoding**
Gate / Logic	**20-30%**	**technology mapping, don't care optimization, de-glitching**
Transistor	**10-20%**	**transistor sizing**
Physical	**5-10%**	**interconnect capacitance reduction, clock-tree synthesis**

- **Power dissipation in CMOS circuits**
  - **Dynamic power dissipation (dominant)**
  - **Short-circuit power dissipation**
  - **Leakage power dissipation**
- **Dynamic power dissipation**

$$P_{dynamic} = C_{eff} V_{dd}^2 f_{clk}$$
$$= \alpha C_{phy} V_{dd}^2 f_{clk}$$

C_{eff} : **effective (switched) capacitance**

f_{clk} : **clock frequency**

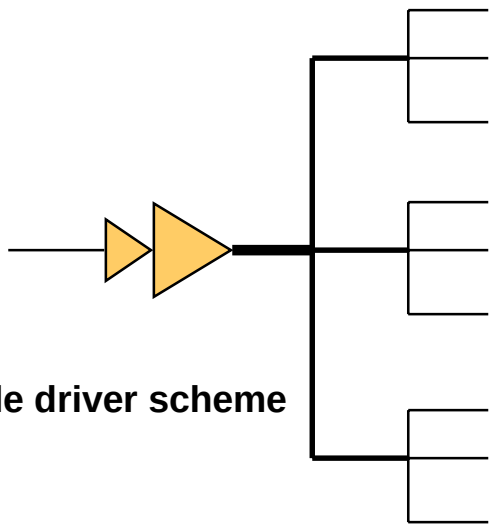
α : **switching activity**

V_{dd} : **supply voltage**

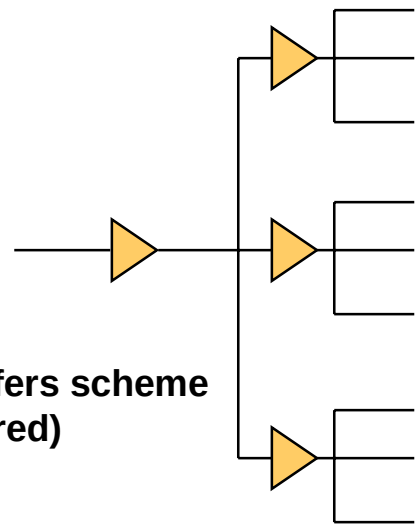
C_{phy} : **physical capacitance**

Physical/Transistor/Gate-Level Design

- **Interconnect capacitance reduction**
  - **Signals having high switching activity are assigned short wires**
- **Clock-tree synthesis**
  - **Clock is a major source of dynamic power dissipation**
  - **Clock of 200MHz DEC Alpha chip drives 3250pF load, 3.3V supply voltage => 7W (30% of the total power)**
  - **Clock skews must be controlled within tolerable values**

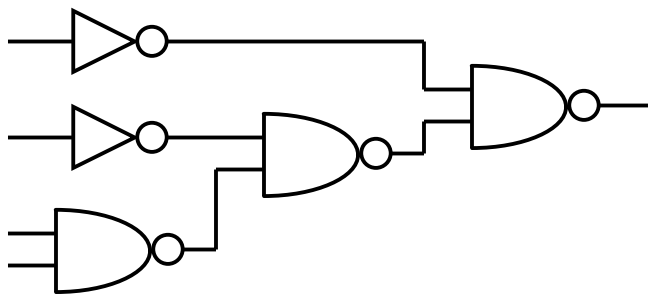


Single driver scheme

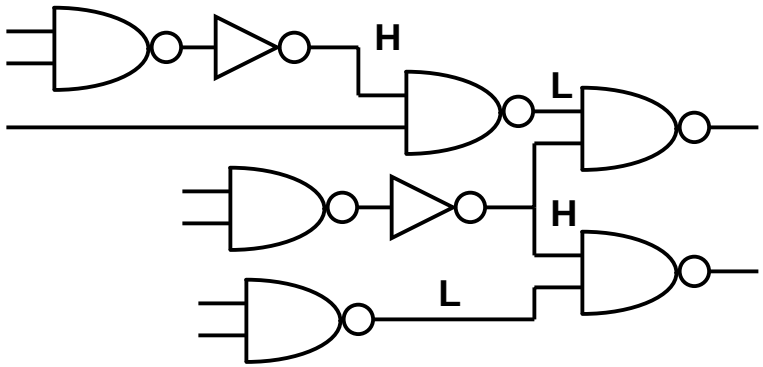


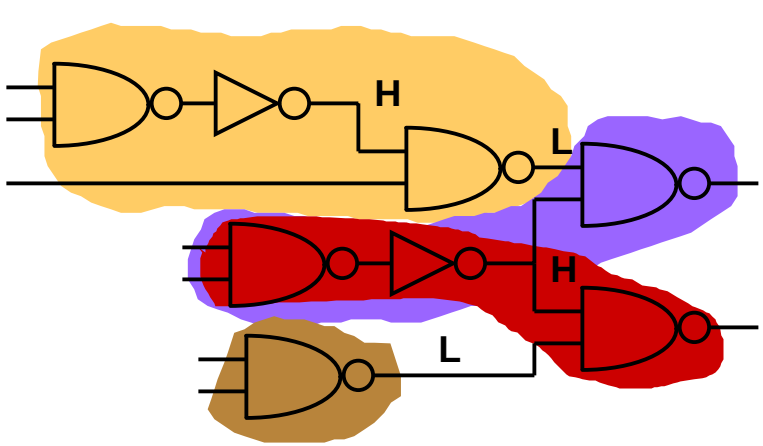
Distributed buffers scheme (preferred)

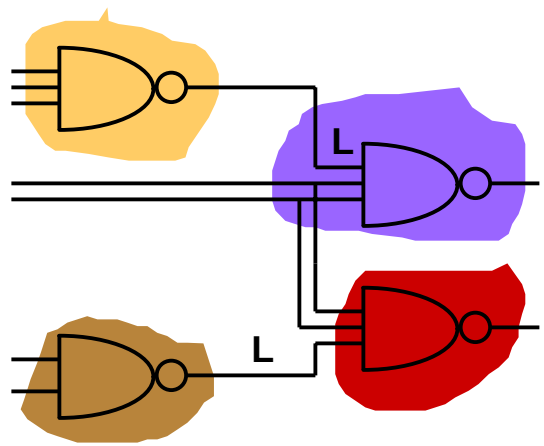
- **Transistor sizing**
  - **Compute the slack at each gate**
  - **Sizes of the transistors in the gate are reduced until the slack becomes zero**
  - **Reduced size => reduced capacitance => reduced power**
  - **Critical path is not affected**
  - **Path balancing => reduced glitch => reduced power**



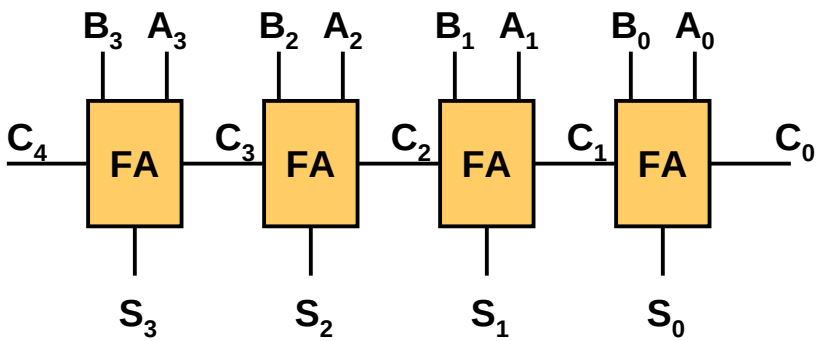
- **Technology mapping**
  - **V. Tiwari, P. Ashar, and S. Malik, "Technology mapping for low power," *Proc. of Design Automation Conference*, pp. 74-79, June 1993**
  - **Hide nodes with high switching activity inside the gates where they drive smaller load capacitances**







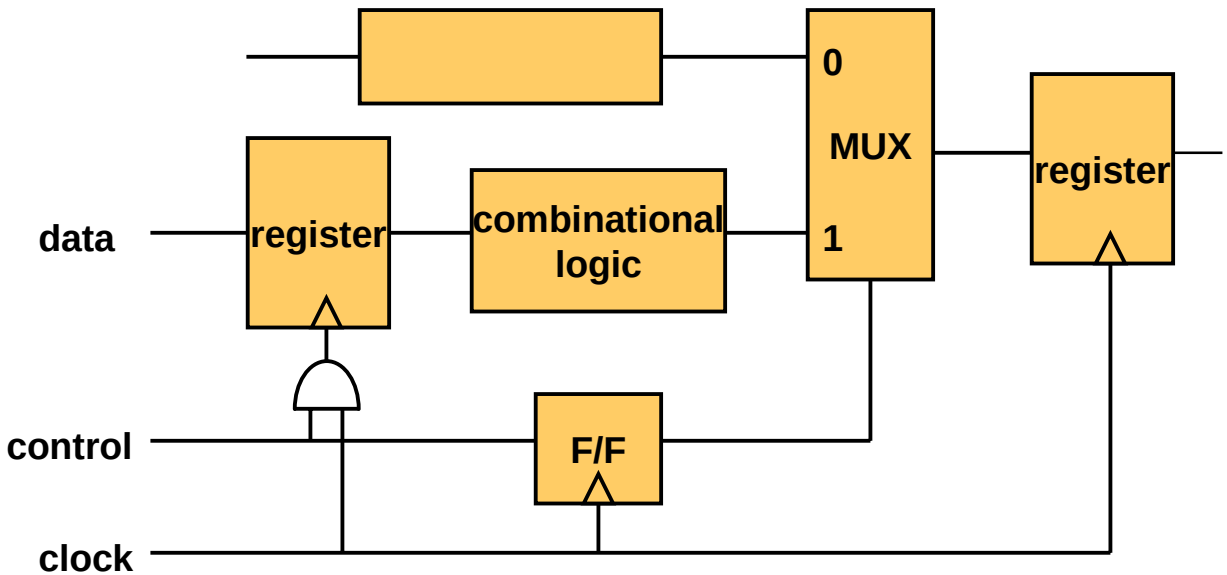
- **De-glitching**
  - **Glitch consumes 10% - 40% of the dynamic power in typical combinational logic circuits**



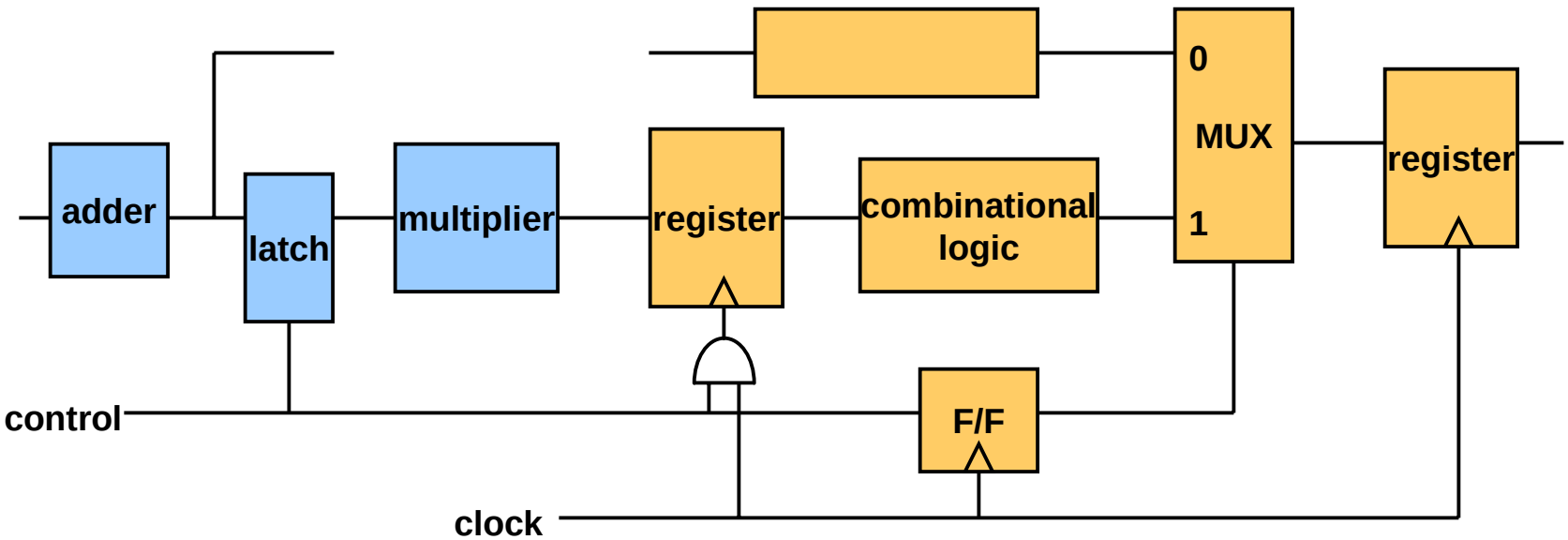
  - **Path balancing**
    - **Add unit-delay buffers selectively such that the delays of all paths can be made equal**

RTL Design

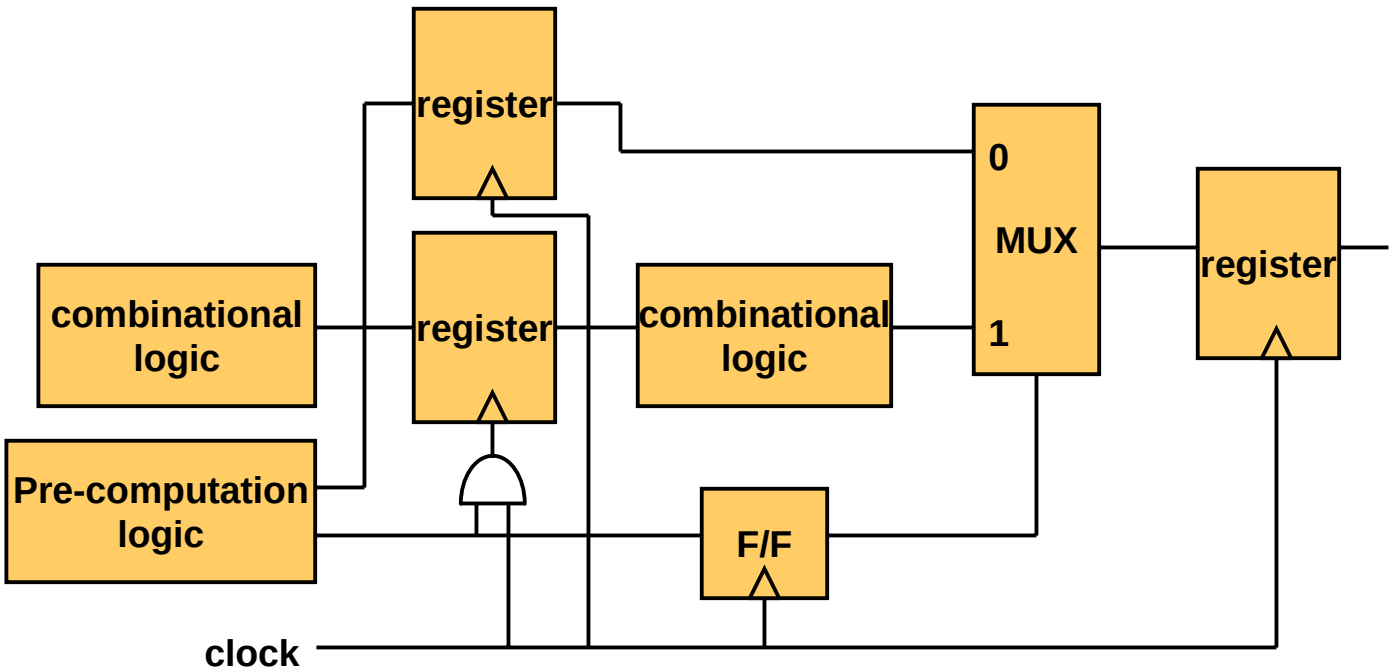
- **Clock gating**
  - **Disable clocks to idle part of the circuit**
  - **Saves clock power and power consumed by registered value change**



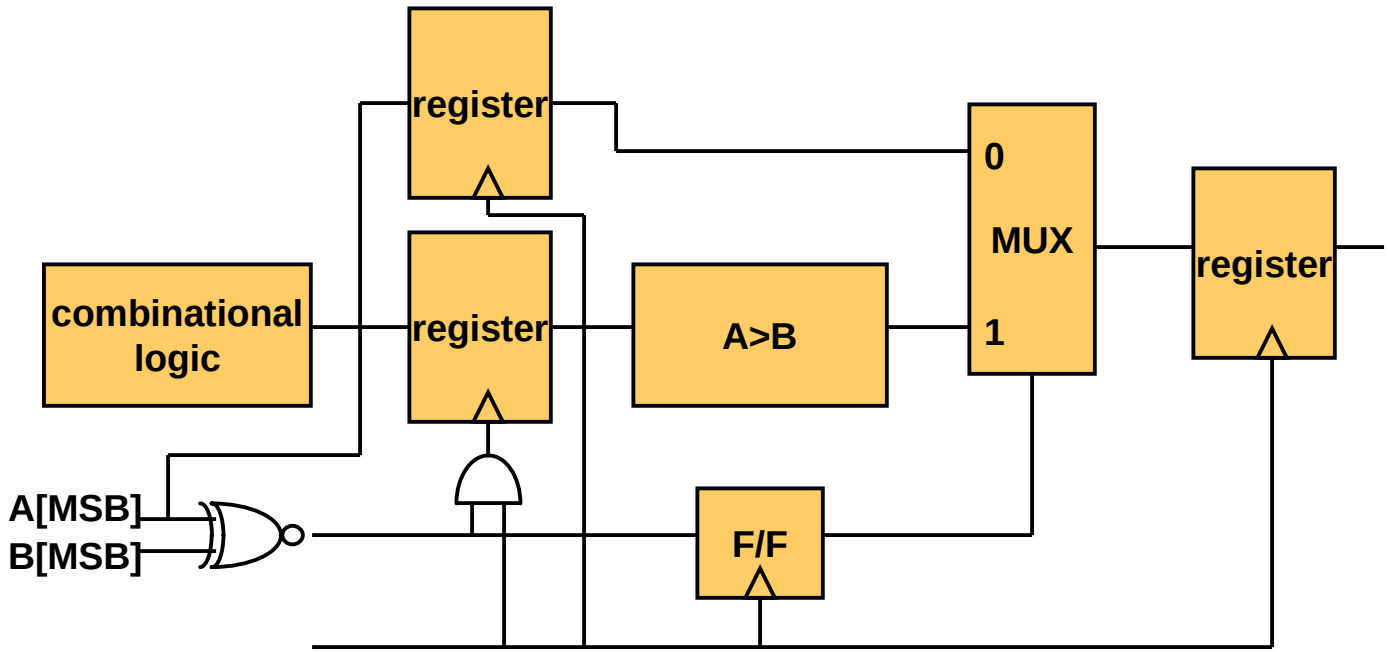
- **Operand isolation**
  - **Exploit output don't cares of large circuit blocks in unused clock cycles**
  - **Insert latches before the circuit blocks to reduce circuit activity**



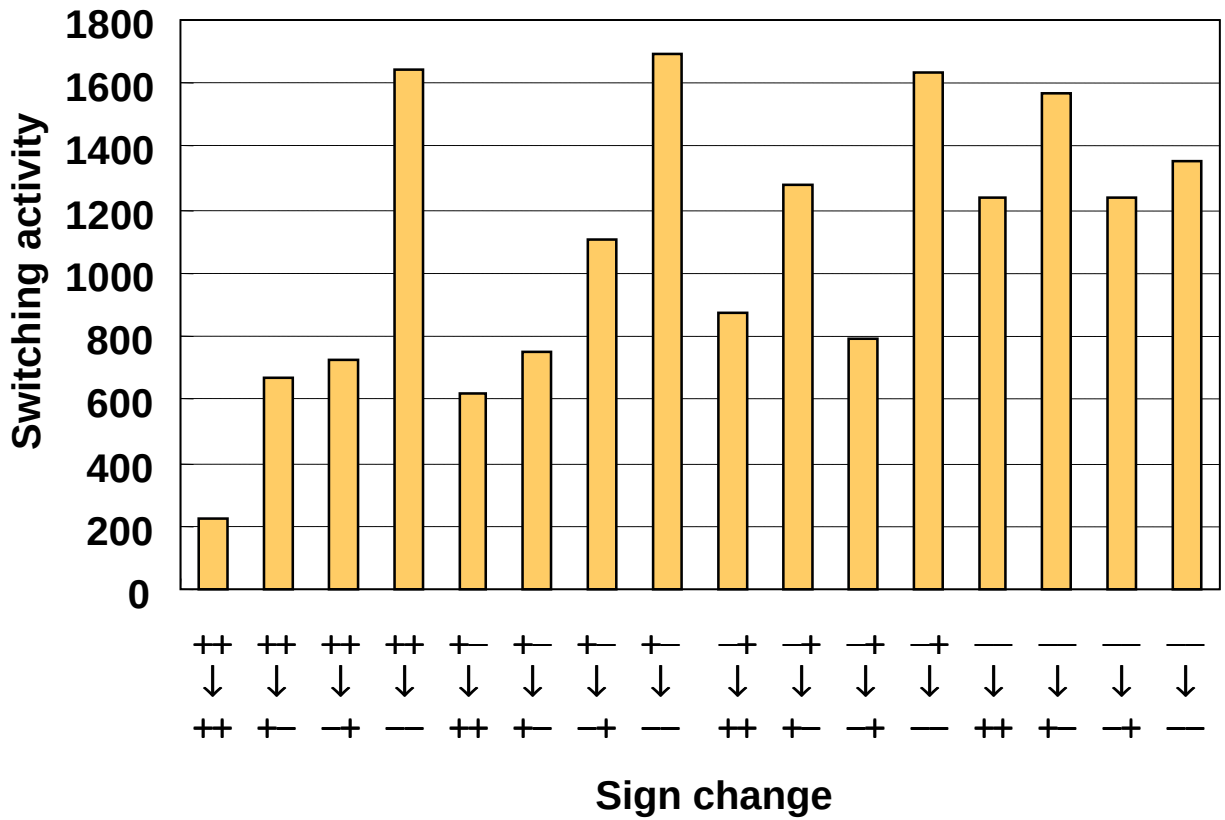
- **Pre-computation**
  - **Pre-compute the results of subsequent pipeline stages**



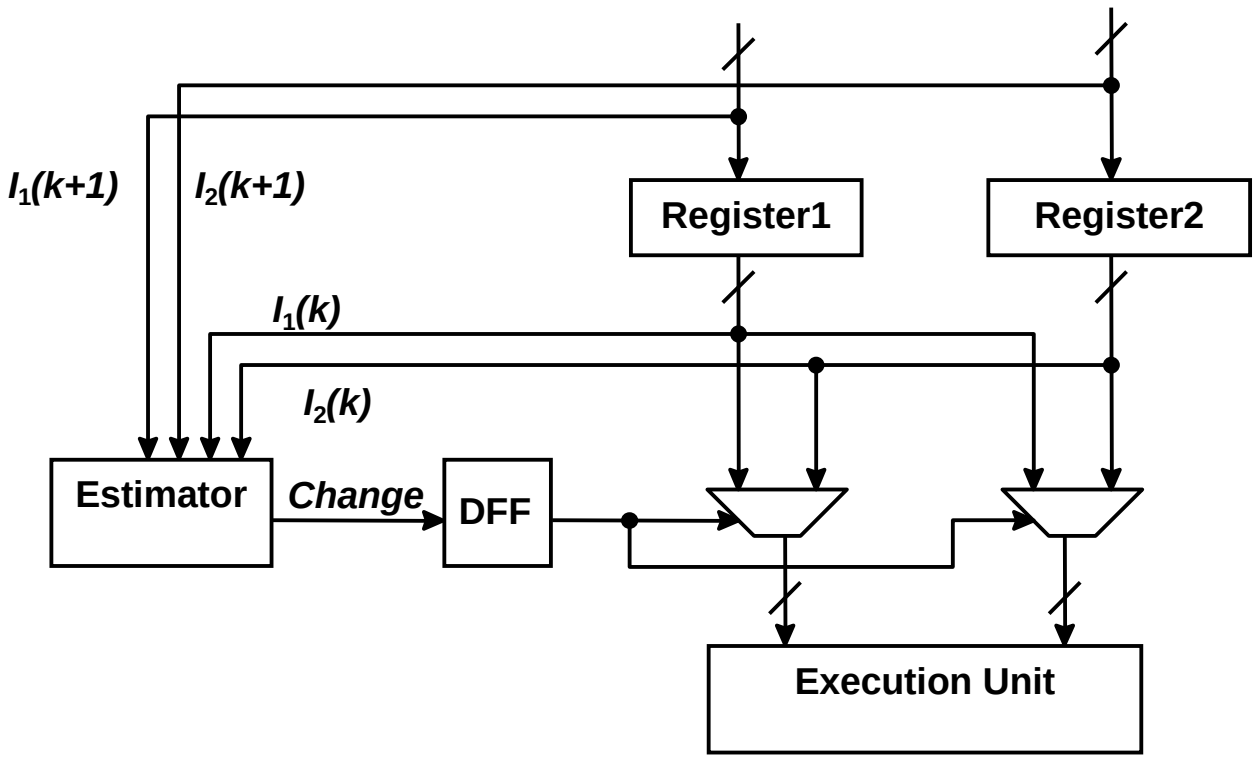
– Comparator example



- **Dynamic operand interchange**
  - **T. Ahn and K. Choi, "Dynamic operand interchange for low power," *Electronics Letters*, pp. 2118-2120, Dec. 1997**
  - **Switching activity of 16-bit array multiplier**

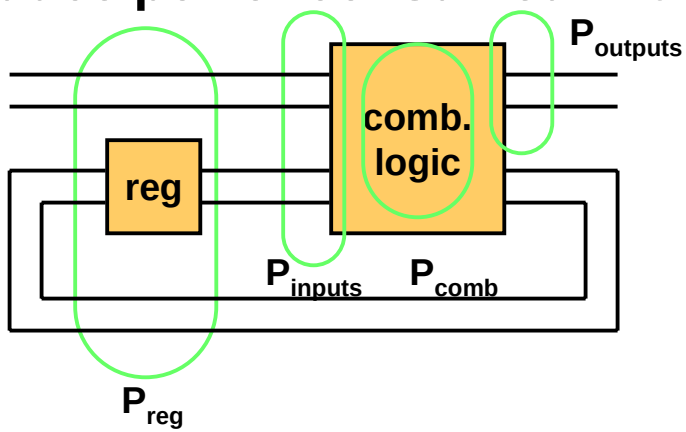


– Architecture

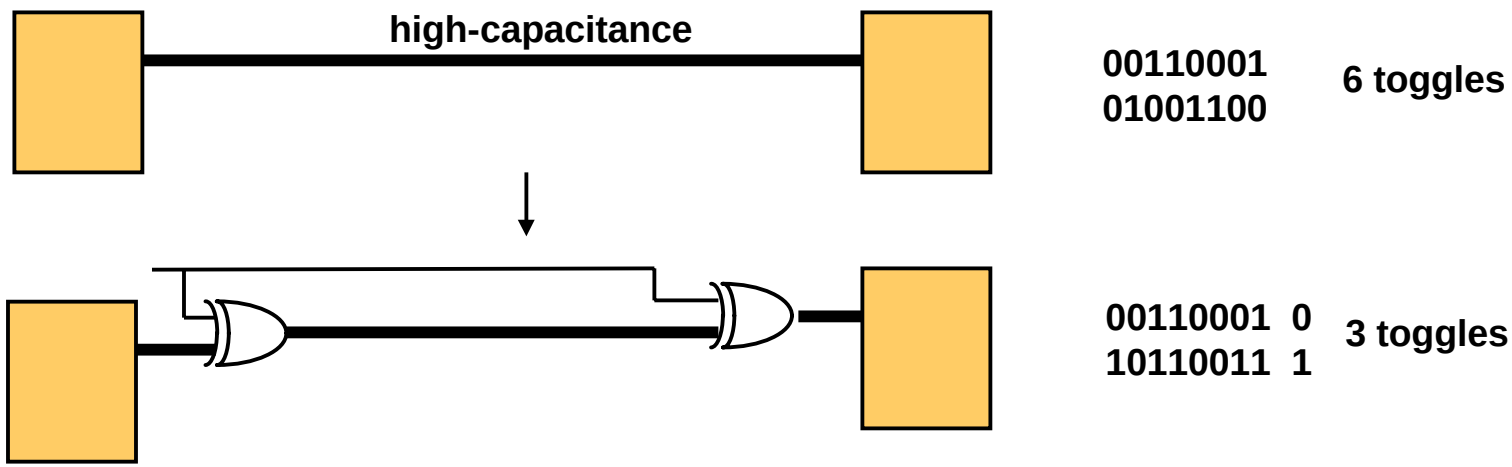


- **FSM encoding**
  - C.-Y. Tsui, M. Pedram, C.-A. Chen, and A.M. Despain, "Low power state assignment targeting two- and multi-level logic implementations," *Proc. of Int'l Conf. on Computer-Aided Design*, pp. 82-87, Nov. 1994
  - Low power state encoding of FSM
  - Reduce switching activity on state bit lines
    - Cost function: $\sum_{S_i S_j \in S} p_{ij} H(S_i, S_j)$

      where p_{ij} is the transition probability from state S_i to state S_j and H(S_i to state S_j) is the Hamming distance between the encodings of the two states
  - Also reduce power consumed in the combinational logic



- **Bus encoding**
  - **Reduce number of transitions on high-capacitance, multi-bit buses by encoding the signals**
  - **Examples**
    - **Bus-invert coding**
      - **M.R. Stan, W.P. Burleson, "Bus-invert coding for low-power I/O," *IEEE Trans. on VLSI Systems*, Vol. 3, No. 1, pp. 49-58, Mar. 1995**
    - **Gray coding**
      - **C. L. Su, "Saving power in the control path of embedded processors," *IEEE Design and Test of Computers*, Vol. 11, No. 4, pp. 24-30, Winter 1994**



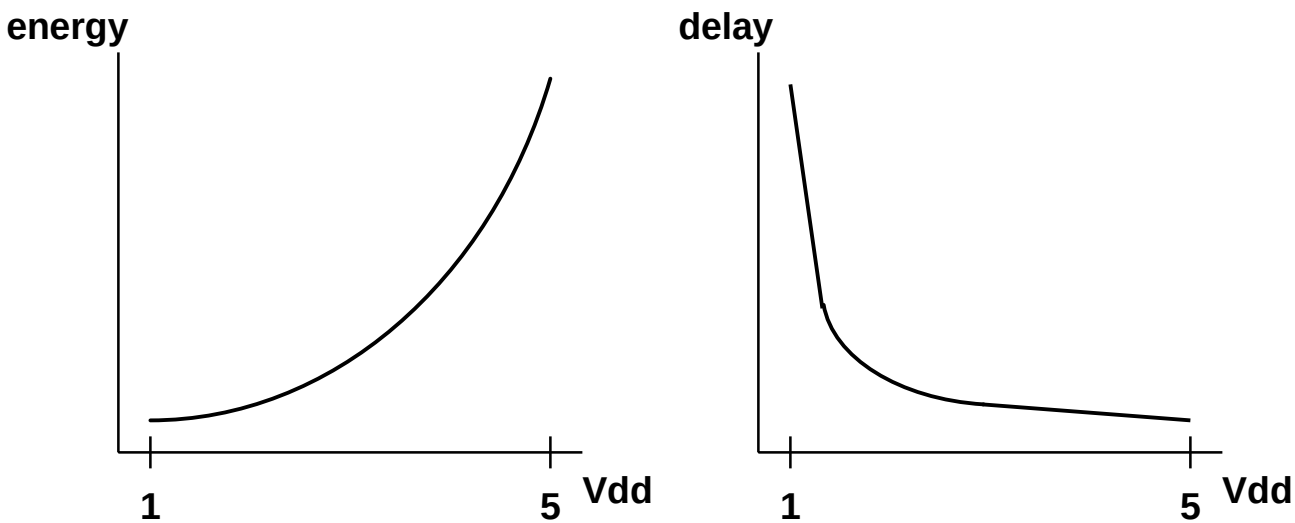
Architecture-Level Design

- **Supply voltage reduction**
  - **Quadratic effect of voltage scaling on power**

$$P_{dynamic} = C_{eff} V_{dd}^2 f_{clk}$$

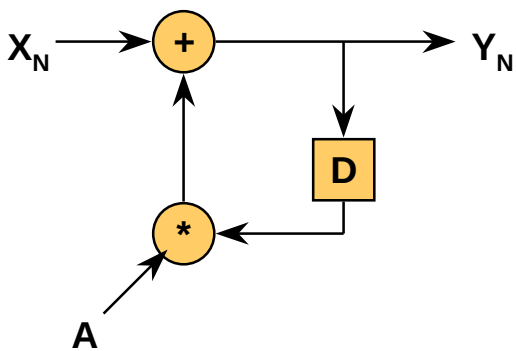
5V --> 3.3V => 60% power reduction

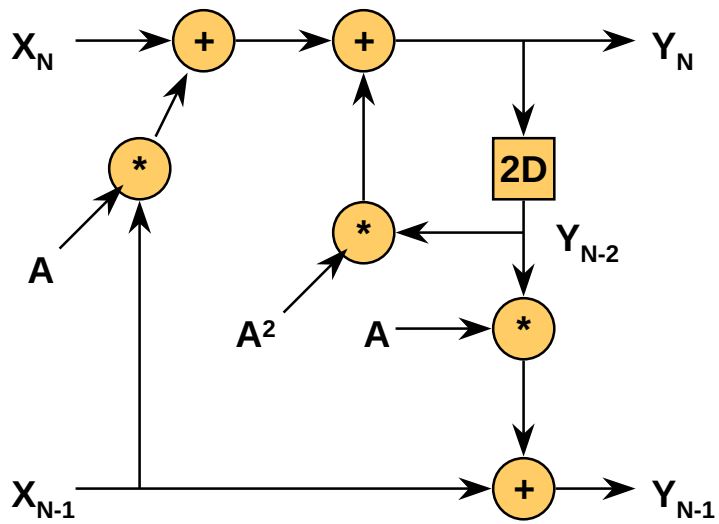
  - **Supply voltage reduction => increased latency**

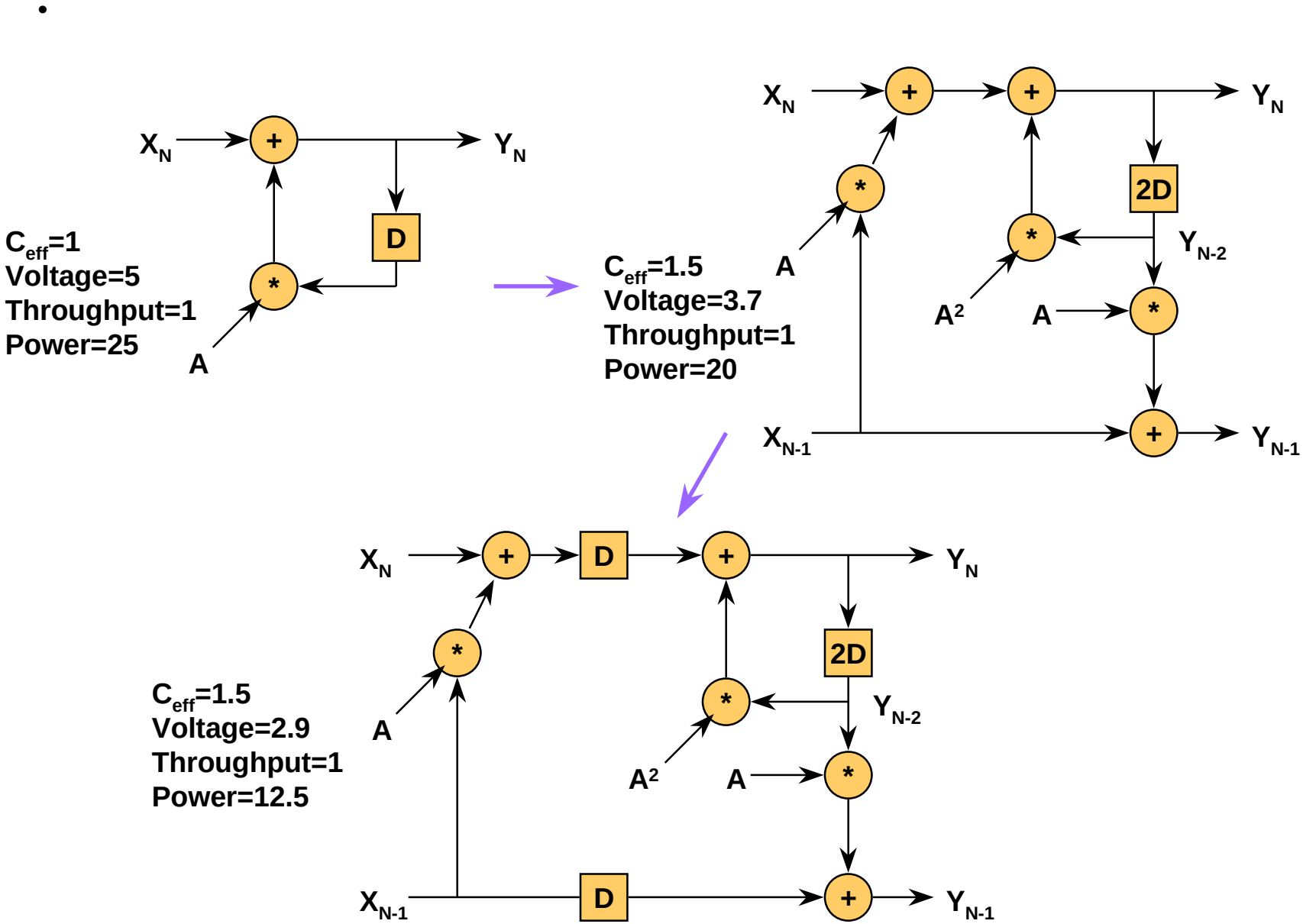


- Use of optimizing transformation for meeting throughput constraint even with the voltage reduction
- Concurrency increasing transformation (increased hardware cost) => critical path reduction
- Loop unrolling, pipelining, retiming, algebraic transformation, module selection
  - A.P. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, and R.W. Brodersen, “Optimizing power using transformation,” *IEEE Tr. on CAD/ICAS*, pp. 12-31, Jan. 1995
- $Y_N=AY_{N-1}+X_N$ --> $Y_N=A^2Y_{N-2}+AX_{N-1}+X_N$

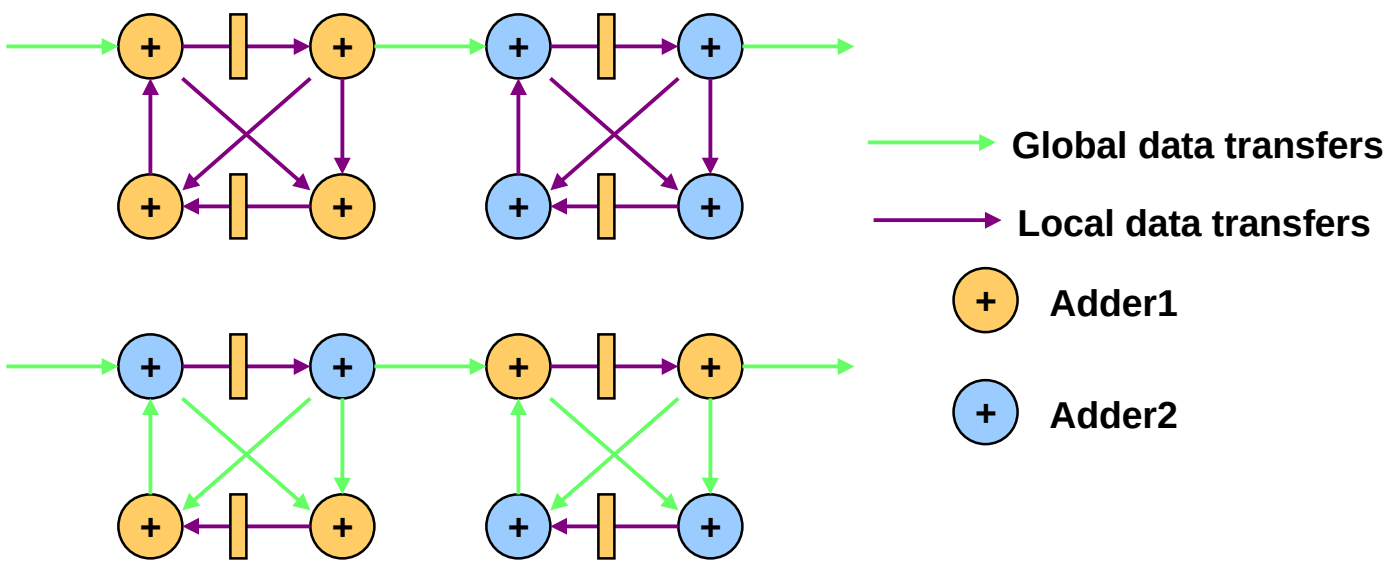
  $Y_{N-1}=AY_{N-2}+X_{N-1}$ $Y_{N-1}=AY_{N-2}+X_{N-1}$



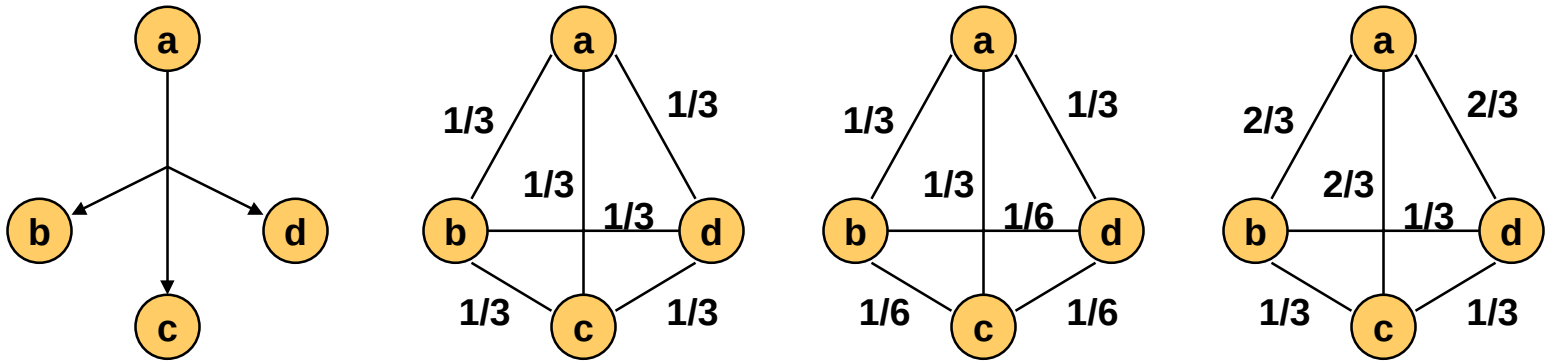



C_{eff}=1
Voltage=5
Throughput=1
Power=25
X_N
Y_N
D
A
C_{eff}=1.5
Voltage=3.7
Throughput=1
Power=20
2D
Y_{N-2}
A^2
X_{N-1}
Y_{N-1}
C_{eff}=1.5
Voltage=2.9
Throughput=1
Power=12.5

- **Reduction of effective capacitance**
  - **Physical capacitance reduction**
    - **Buses may consume 5-40% of the total power**
    - **Reducing access to global resource thru clustering**
      - **R. Mehra, L.M. Guerra, and J.M. Rabaey, "Low power architectural synthesis and the impact of exploiting locality," *Journal of VLSI Signal Processing*, 1996**

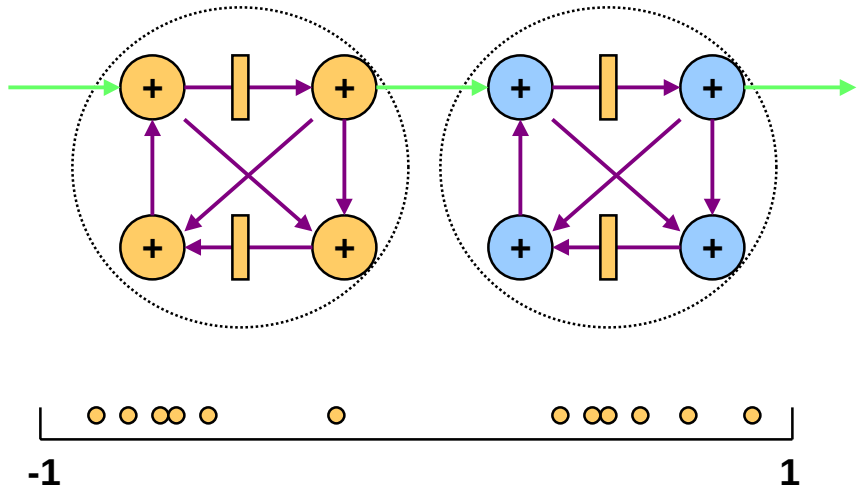


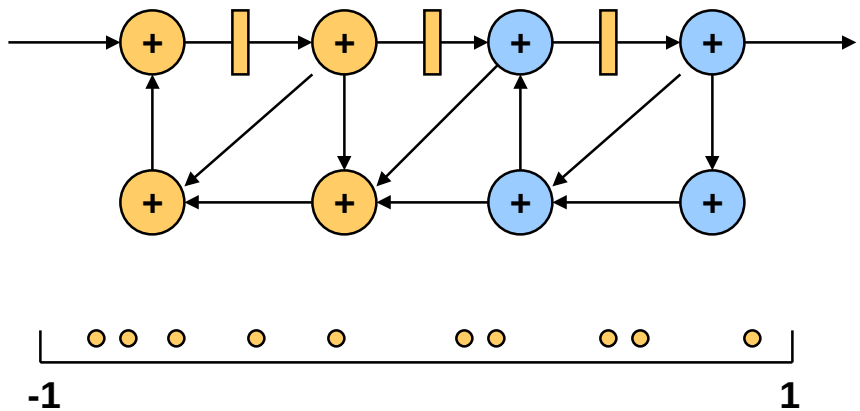
- **Hyperedge models**



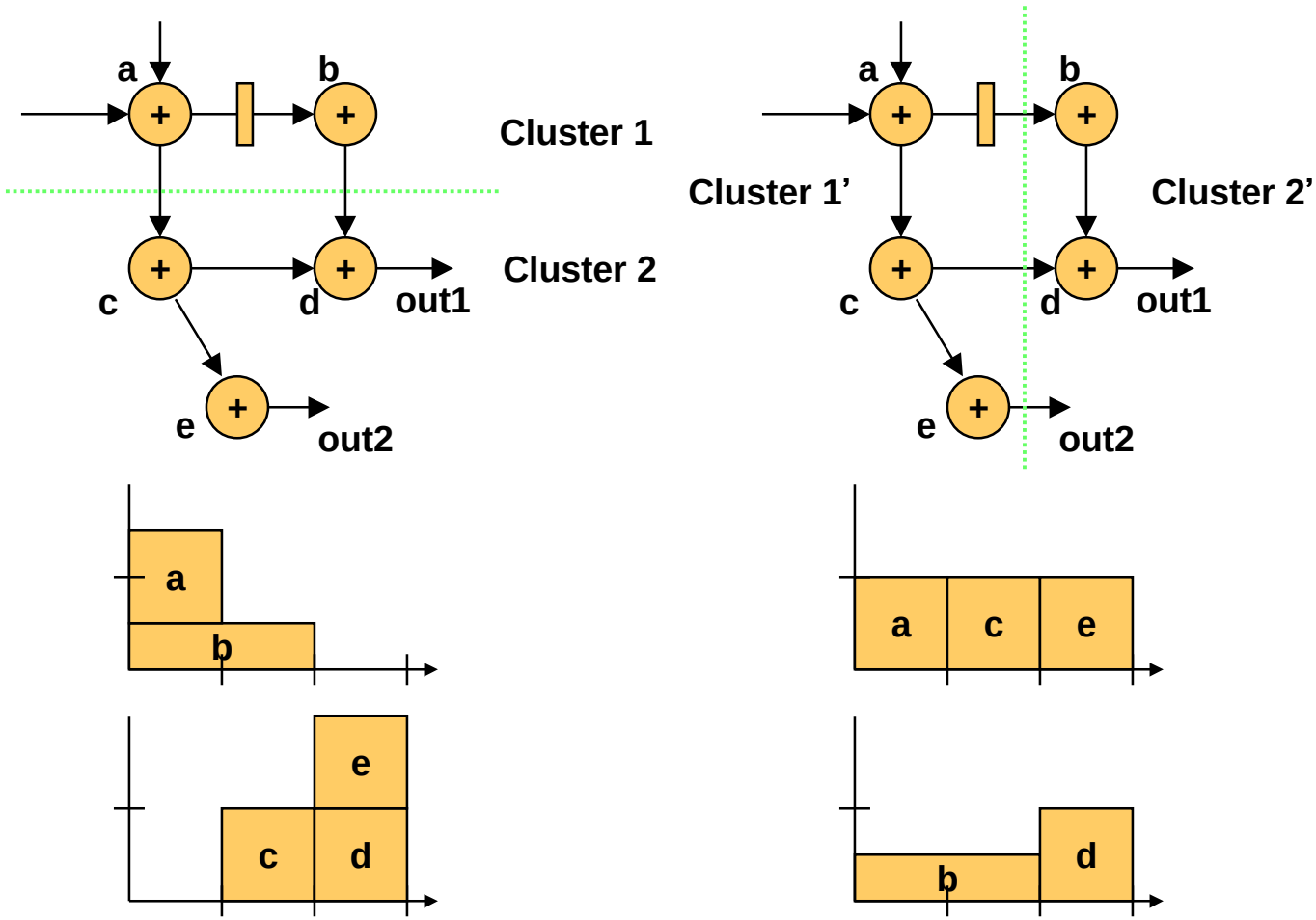
- **partitioning based on spectral method**
  - **minimize $z=1/2\Sigma\Sigma(x_i - x_j)^2A_{ij}$ subject to $x^Tx=1$**
    => non-trivial solution is the 2nd smallest eigenvector of the Laplacian of the graph
    Q=D-A

- **Finding good partitions**





- **Evaluation of the partitions**
  - **area : distribution graph**
  - **power : (number of data transfers) x (area)**



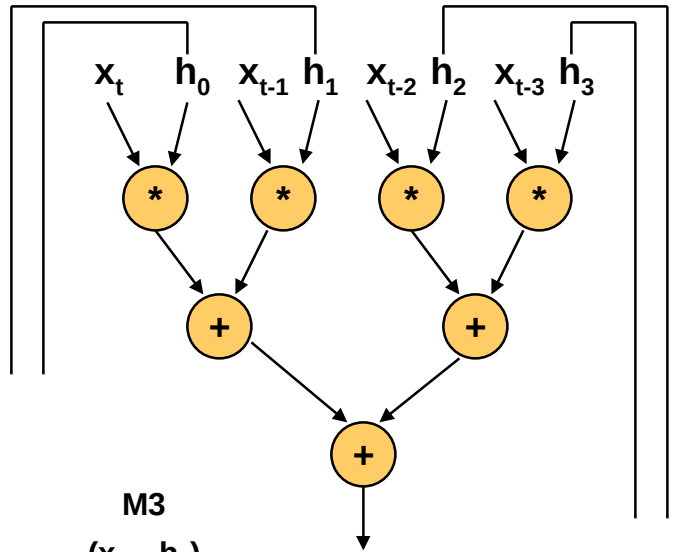
- **Switching activity reduction**
  - **Increasing data correlation thru operand sharing**
    - **Operations sharing an operand also share resource**
    - **Actively increase the chance of operand sharing thru loop interchange, operand reordering, loop unrolling, loop folding**
  - **Loop interchange**

```
for i
   for j
      for k
         for l
            a=f(k, l)
            b=f(i, j, k, l)
            c(i, j) = a - b
```

```
for k
   for l
      a=f(k, l)
      for i
         for j
            b=f(i, j, k, l)
            c(i, j) = a - b
```

– **Operand reordering**

- **4th order LMS adaptive filter**

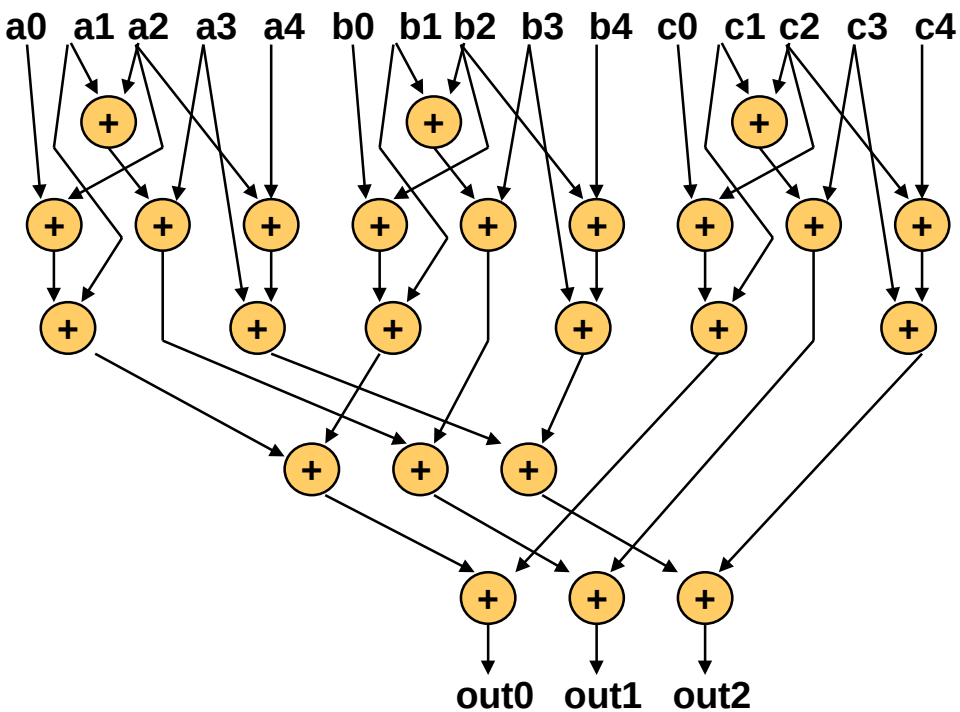


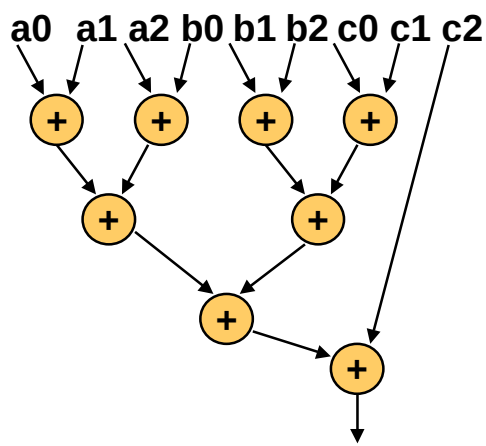
Iter	Reordering A			
	M0	M1	M2	M3
i	(x_t, h_0)	(x_{t-1}, h_1)	(x_{t-2}, h_2)	(x_{t-3}, h_3)
i+1	(x_{t+1}, h_0)	(x_t, h_1)	(x_{t-1}, h_2)	(x_{t-2}, h_3)
i+2	(x_{t+2}, h_0)	(x_{t+1}, h_1)	(x_t, h_2)	(x_{t-1}, h_3)
i+3	(x_{t+3}, h_0)	(x_{t+2}, h_1)	(x_{t+1}, h_2)	(x_t, h_3)

Iter	Reordering B			
	M0	M1	M2	M3
i	(x_t, h_0)	(x_{t-1}, h_1)	(x_{t-2}, h_2)	(x_{t-3}, h_3)
i+1	(x_t, h_1)	(x_{t-1}, h_2)	(x_{t-2}, h_3)	(x_{t+1}, h_0)
i+2	(x_t, h_2)	(x_{t-1}, h_3)	(x_{t+2}, h_0)	(x_{t+1}, h_1)
i+3	(x_t, h_3)	(x_{t+3}, h_0)	(x_{t+2}, h_1)	(x_{t+1}, h_2)

- **Loop unrolling**
  - **E. Musoll and J. Cortadella, "High-level synthesis techniques for reducing the activity of functional units," *Proc. of Int'l Symp. on Low Power Design*, pp. 99-104, Nov. 1995**
  - **Low-pass image filter**

```
for i=0 to M
   for j=0 to N
      out=a[i-1][j-1]+      /* a0 */
          a[i-1][j]+        /* a1 */
          a[i-1][j+1]+      /* a2 */
          a[i][j-1]+        /* b0 */
          a[i][j]+          /* b1 */
          a[i][j+1]+        /* b2 */
          a[i+1][j-1]+      /* c0 */
          a[i+1][j]+        /* c1 */
          a[i+1][j+1]       /* c2 */
```





– Loop folding

- D. Kim and K. Choi, "Power-conscious high level synthesis using loop folding," *Proc. of Design Automation Conference*, pp. 441-445, June 1997
- Fold two consecutive iterations in such a way that h(i) * x[n-i] for y[n] and h(i+1) * x[(n+1)-(i+1)] for y[n+1] are computed consecutively in one shared multiplier

$$y[n] = \sum_i h_i x[n-i]$$

Significant effects on DSP applications such as filters

$$y[n] = \cdots + h_i \times x[n-i] + \cdots$$
$$y[n+1] = \cdots + h_{(i+1)} \times x[(n+1)-(i+1)] + \cdots$$
$$\vdots \qquad\qquad \vdots$$

```
(1...N-1)
m0[n-1] = h0x[n]
m1[n-1] = h1x[n-1]
out[n-1] = m0[n-1]+m1[n-1]
```

loop folding

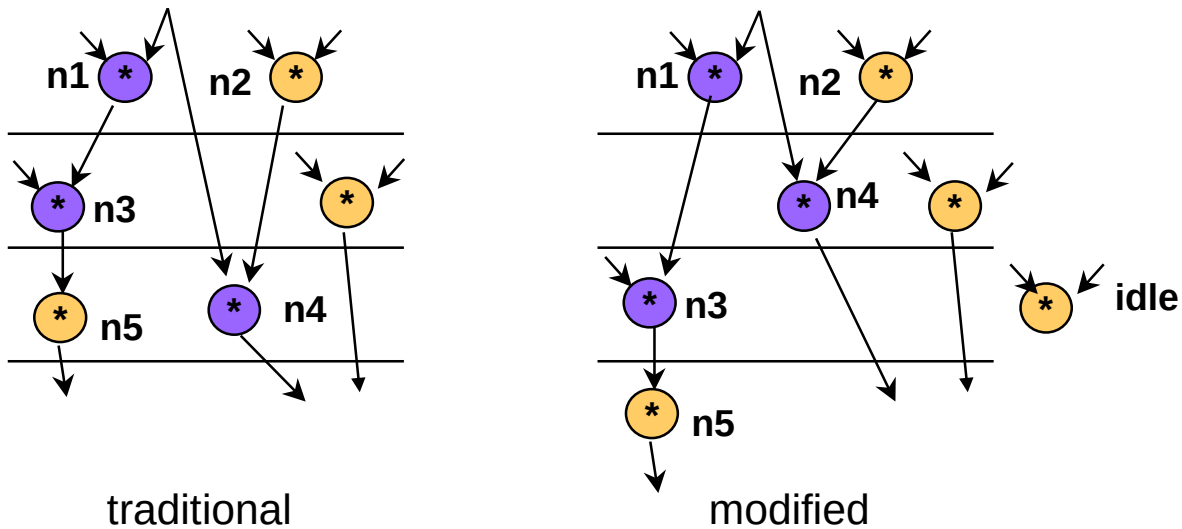
```
m1[0] = h1x[0]
```

```
(1...N-2)
m0[n-1] = h0x[n]
m1[n] = h1x[n]
out[n-1] =m0[n-1]+m1[n-1]
```

```
m0[N-2] = h0x[N-1]
out[N-2] =m0[N-2]+m1[N-2]
```

- **Binding**
  - **A. Raghunathan and N. K. Jha, "Behavioral synthesis for low power," *Proc. of Int'l Conf. on Computer Design*, pp. 318-322, Oct. 1994**
  - **Binding based on edge weighted compatibility graph**
    - **weight = $(1-W_t)W_c$**
      where W_t is transition activity and W_c is capacitance weight
  - **Functional unit and register sharing**
  - **Controller optimization to reduce power consumed during idle time of functional units**
    - **use don't cares**
    - **select the mux port with least transition activity**
    - **disable loading into registers**

- **Scheduling and binding**
  - **E. Musoll and J. Cortadella, "Scheduling and resource binding for low power," *Proc. of Int'l Symp. on System Synthesis*, pp. 104-109, Apr. 1995**
  - **Resource sharing by sibling operations**
  - **List scheduling is used**
  - **Operations sharing the same operand (operations in an operand sharing set) are scheduled in control steps as close as possible (higher priority is given)**



  - **After functional unit binding, bind registers such that useless power is reduced (no change of inputs to idle functional unit)**
  - **A few sibling operations available in normal circuits**

- **Scheduling and binding**
  - **A. Raghunathan and N. K. Jha, "An iterative improvement algorithm for low power data path synthesis," *Proc. of Int'l Conf. on Computer-Aided Design*, pp. 597-602, Nov. 1995**
  - **Thorough power minimization including voltage scaling, clock selection, and module selection as well as scheduling and binding**
  - **Iterative improvement**
  - **Pruning for efficiency of the algorithm**
    - **supply voltage pruning:**

      prune V_{dd} if the lower bound of power at V_{dd} is greater than the best solution seen
    - **clock period pruning:**

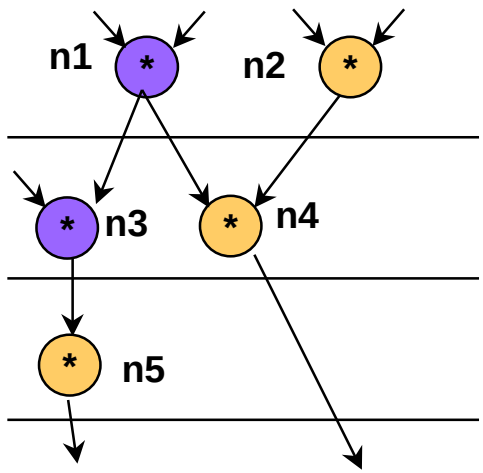
      $T_{clk} \times i = T_s$ for some integer i => prune other T_{clk}

      $T_{clk1} < T_{clk2}$ and $\lceil delay_t/T_{clk1} \rceil = \lceil delay_t/T_{clk2} \rceil$ for all functional unit template t => prune T_{clk2}

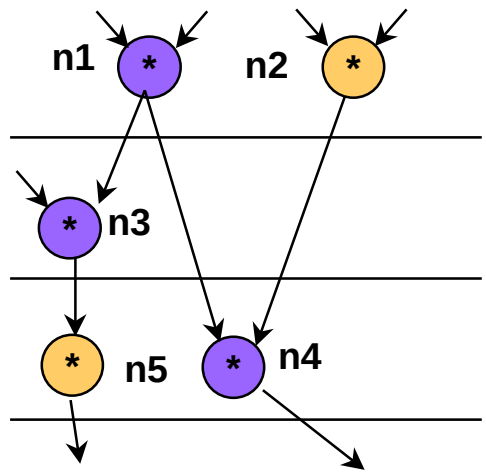
```
SCALP (CDFG G, Sample Period Ts, Library L) {
   Vmin=estimate_min_volt(G, Ts, L);
   Vmax=5V;
   best_dp=null;
   cur_dp=null;
   for(Vdd=Vmin; Vdd≤Vmax; Vdd=Vdd+ΔV) {
      if(Vdd_prune(G,cur_dp,Vdd)) continue;
      for(csteps=max_csteps; csteps≥min_csteps;
         csteps=csteps-1){
         if(clk_prune(G, L, csteps)) continue;
         cur_dp=initial_solution(G, L, Vdd, csteps);
         iterative_improvement(G, L, cur_dp);
         if(power_est(cur_dp) < power_est(best_dp))
              best_dp=cur_dp;
         }
      }
   }
}
```

```
iterative_improvement(G, L, DP) {
   do {
      for(i=1; i ≤ max_moves; i=i+1) {
         gaini = generate_moves(G, L, DP);
         append gaini to gain_list;
      }
      find subsequence, gain1 ... gaink in
         gain_list so that G=Σgaini is maximized;
      if(G>0) {
         accept moves 1…k;
      }
   }
   until(G<0);
}
```

- **Scheduling and binding**
  - **D. Shin and K. Choi, "Low power high level synthesis by increasing data correlation," *Proc. of Int'l Symp. on Low Power Electronics and Design*, pp. 62-67, Aug. 1997**
  - **Simultaneous scheduling and binding in such a way that input data correlation between consecutive inputs increase**
  - **(Modified) list scheduling is used for efficiency**
  - **DBT (Dual Bit Type) method for estimating switched capacitance in execution units**
    - **P.E. Landman and J.M. Rabaey, "Architectural power analysis: the dual bit type method," *IEEE Tr. on VLSI Systems*, pp. 173-187, June 1995**



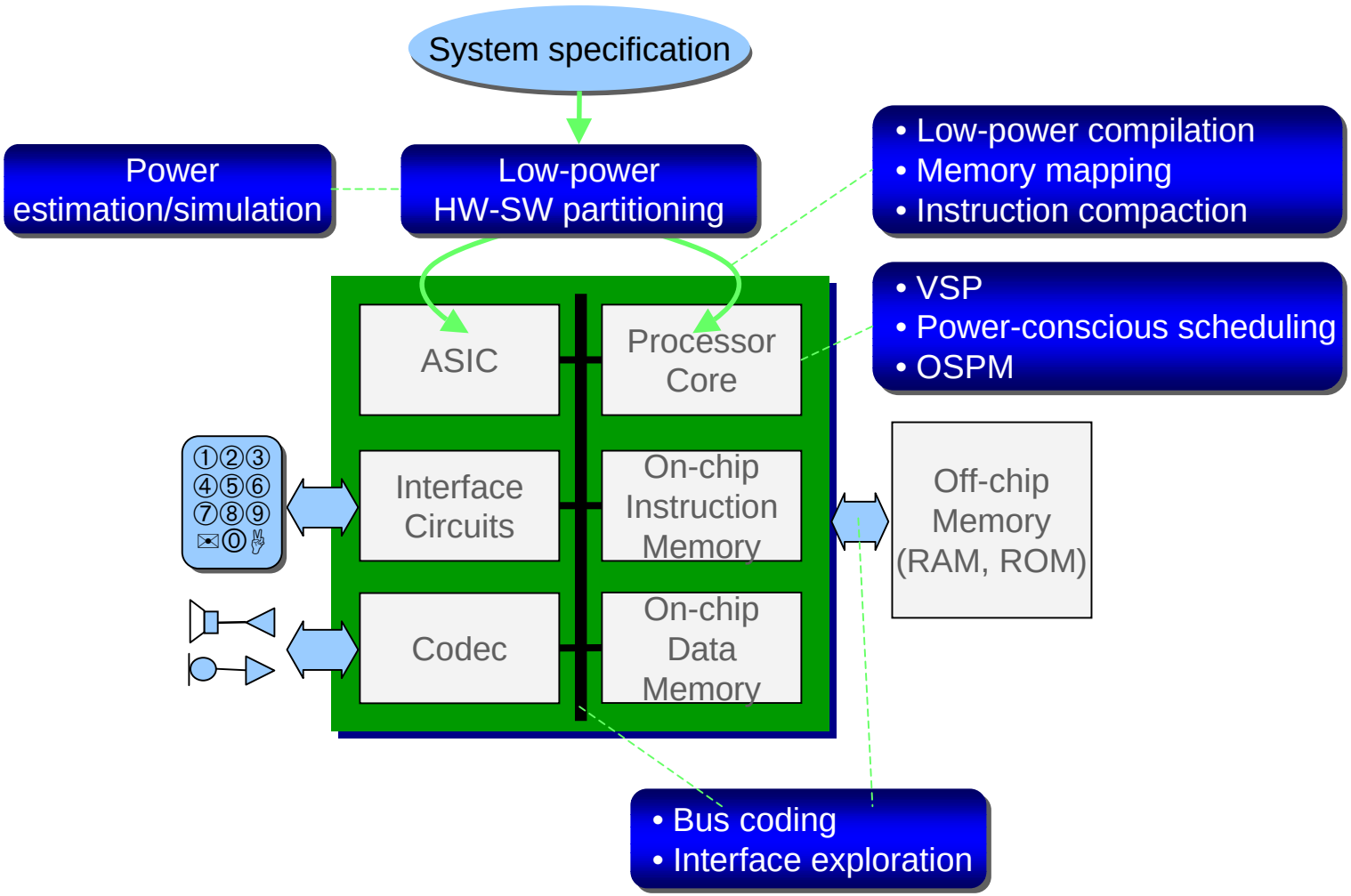
traditional list scheduling



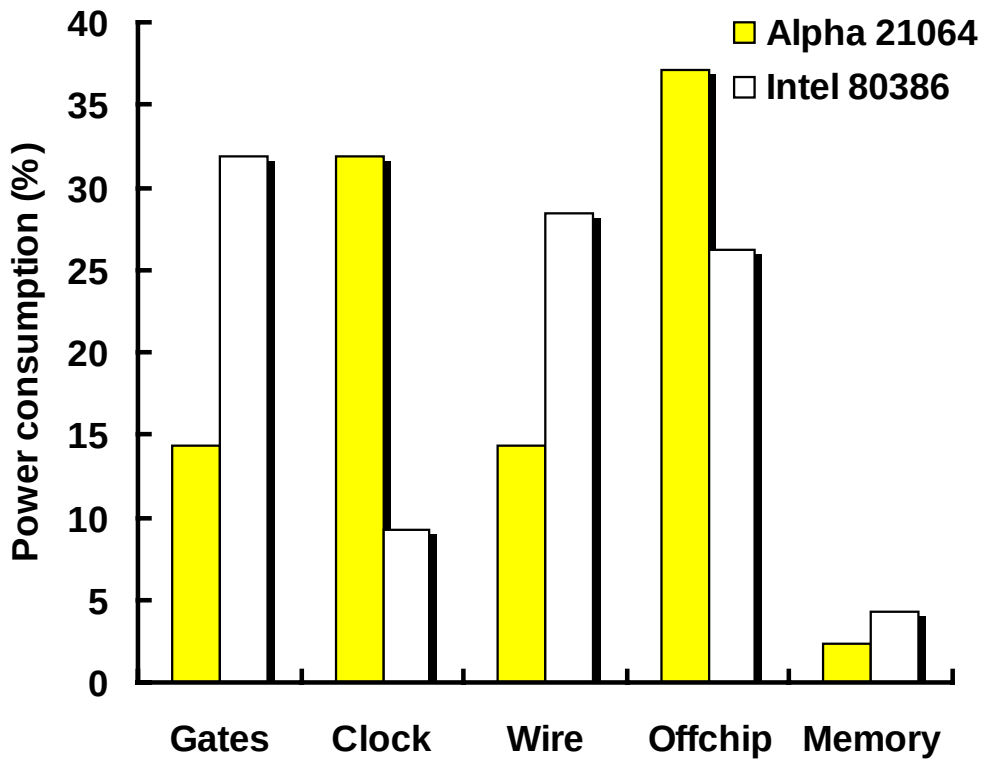
modified list scheduling

System-Level Design

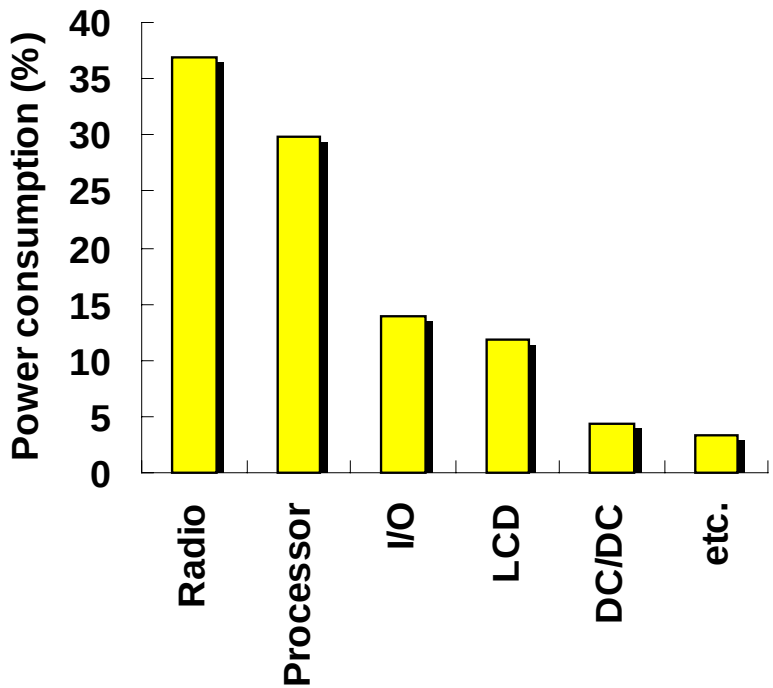
- **System-level power optimization**



- **Power consumption in processors**
  - **Buses consume significant power**
    - **Capacitive load at I/O of a chip is three orders of magnitude larger than that of internal nodes**
  - **Example**
    - **D. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips," *IEEE JSSC*, pp. 663-670, June 1994**



- **Power consumption in portable embedded systems**
  - **Power consumption in processors becomes more significant as increasing amount of functionality is realized through software**
  - **Example**
    - **T. Truman, T. Pering, R. Doering, and R. Brodersen, "The InfoPad multimedia terminal: a portable device for wireless information access," *IEEE Transactions on Computers*, pp. 1073-1087, October 1998**



- **Low power design issues**
  - **L. Benini and G. De Micheli, "System-level power optimization: techniques and tools," *Proc. of Int'l Symp. on Low Power Electronics and Design*, pp. 288-293, Aug. 1999**
  - **Memory optimization**
    - **Memory hierarchy, cache size, memory size (related with software transformation), data transfer and placement**
    - **E.g. large cache size → low cache miss → high speed and low power, but large capacitance**
  - **Hardware-software partitioning**
    - **Power consumption in hardware, software, and interface**
  - **Instruction-level power optimization**
    - **Dedicated low-power instruction set, instruction transformation,**
  - **Variable-voltage**
    - **Dynamically variable voltage supply**
    - **Effective**
  - **Dynamic power management**
    - **Low-power sleep state**
    - **Predictive, stochastic**
    - **Standard (OnNow, ACPI)**
  - **Interface power minimization**
    - **Bus encoding**