Τμήμα Μηχανικών Η/Υ και Πληροφορικής,
Πανεπιστήμιο Ιωαννίνων

Introduction to low-power microprocessor design – Static power and static power reduction techniques

Vasileios Tenentes

University of Ioannina

E-mail: tenentes@cse.uoi.gr

ARM® University
Worldwide Education Program

Outline

1: Overview of microprocessors manufacturing process

2: Motivation: Why to care about power efficient microrprocessors?

3: Performance of microprocessors

4: Dynamic power and dynamic power reduction design techniques

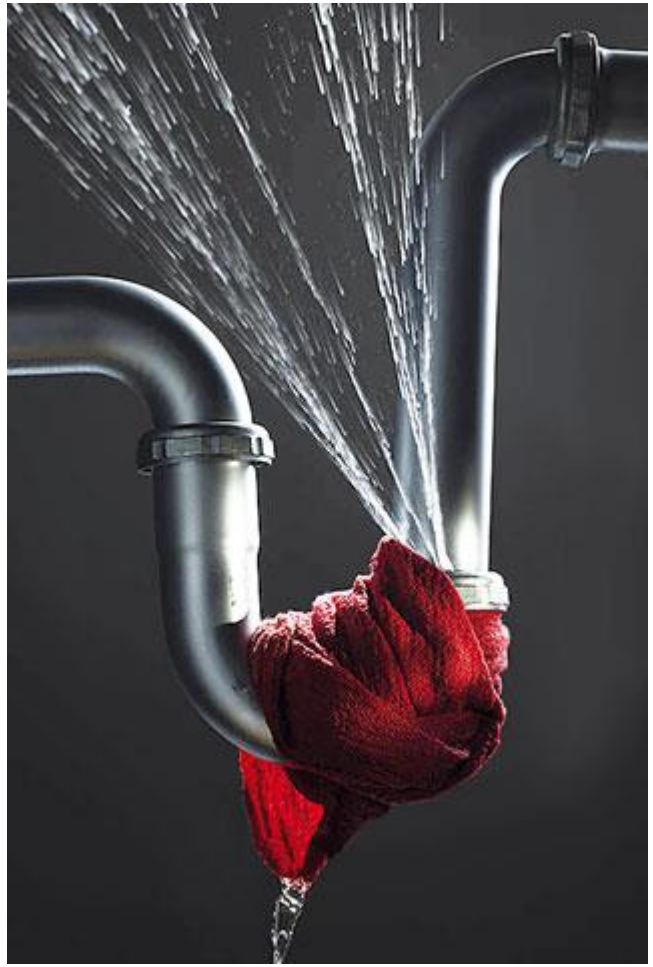
5: Static power and static power reduction techniques

6: The future, Introduction to IoT applications!!!

Static-power

The problem

The solution



so far we spoke only about power when a transistor is ON…
Well.. it still consumes power even when it's OFF!!!

That power is called **static power**

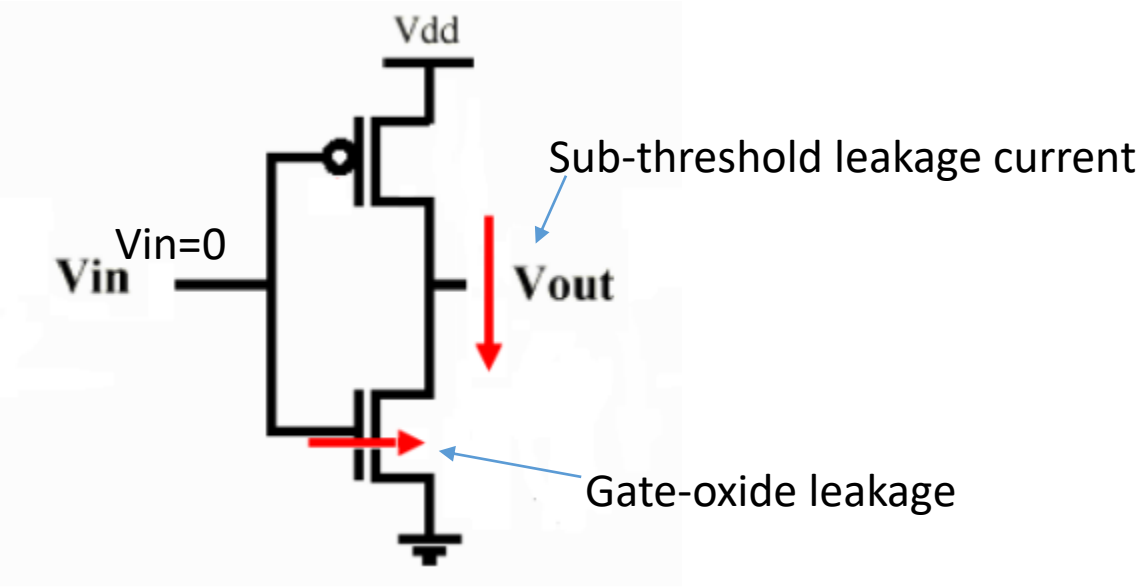
In fact it is:

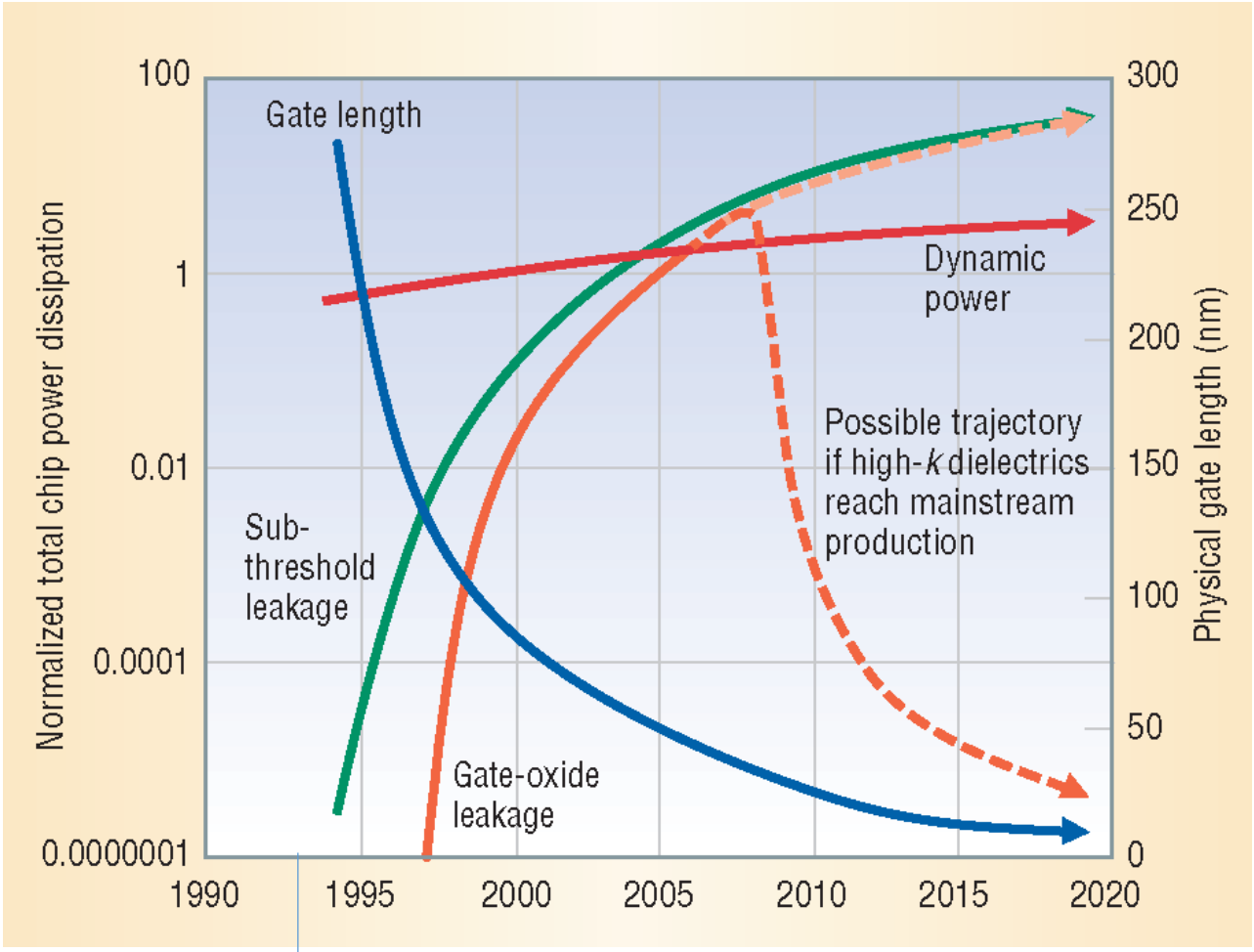
Power=(Dynamic Power) + (Short Circuit Power) + (Static Power)

We will talk about this later

Static power (caused by leakage currents) is typically responsible for 40% of the energy consumption. Thus, increasing the number of transistors increases power dissipation, even if the transistors are always off

Static power (caused by leakage currents)

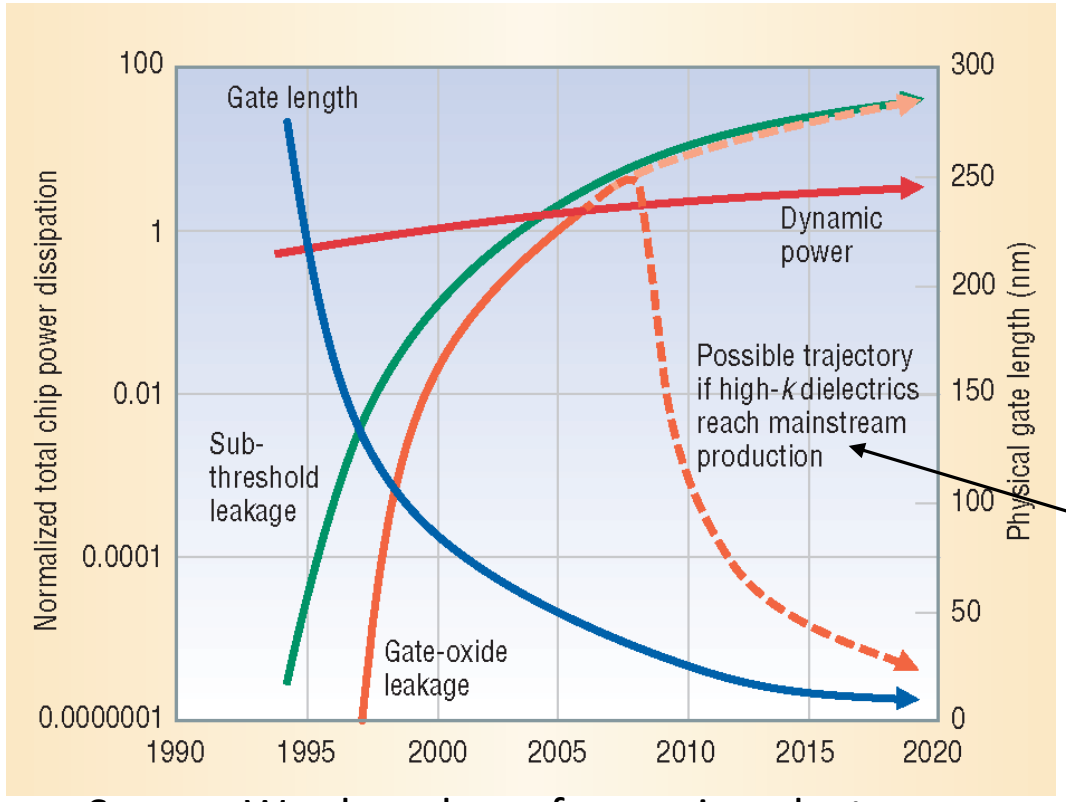




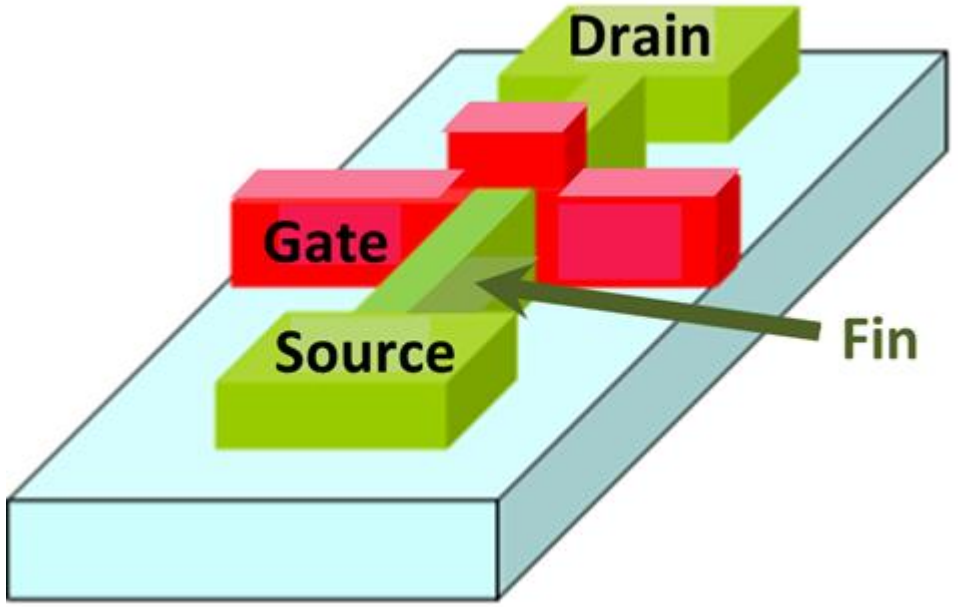
- Static power is as large as dynamic power for next generation ICs, unless specific countermeasures are taken
- Main components
  - Sub-threshold leakage current
  - Gate-oxide leakage current

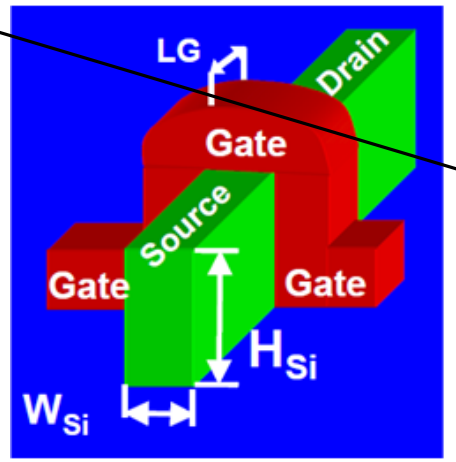
$$P_{static} = V_{dd} * I_{leakage}$$

FINFET and High-k materials have tackled efficiently gate-oxide leakage current

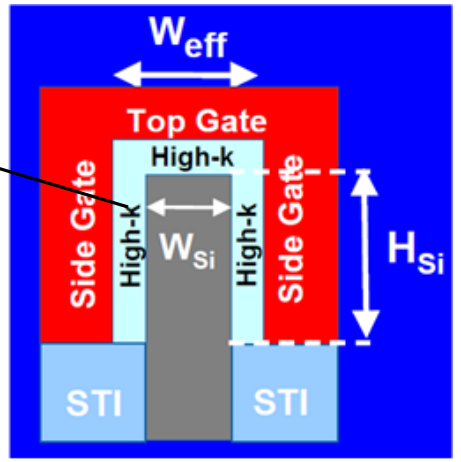


Source: Word roadmap for semiconductors





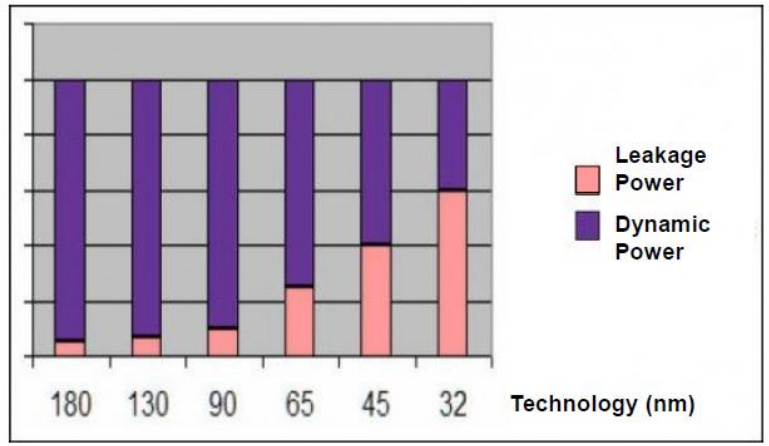
Top View



Cross View

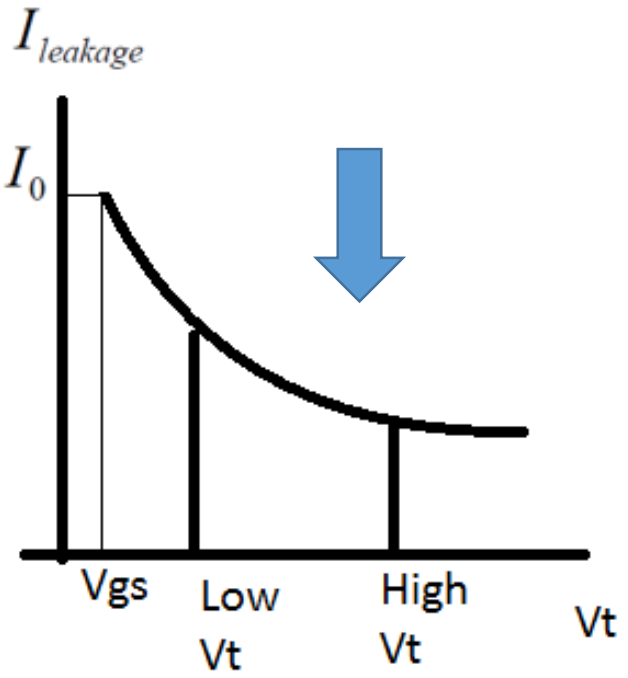
Static power and threshold voltage background

- Leakage Power consumption occurs as long as the circuit is powered on
- Sub-threshold current between source and drain in MOS transistor occurs when gate voltage (V_{gs}) is below transistor threshold voltage V_t
- sub-threshold current increases by 3x with each technology generation due t scaling of the transistor threshold voltage V_t
- At 45nm, total power (60% dynamic and 40% leakage)
- V_{dd} reduces with technology scaling (15% lower operating voltage, 30% smaller transistor), V_t must scale to deliver transistor performance.



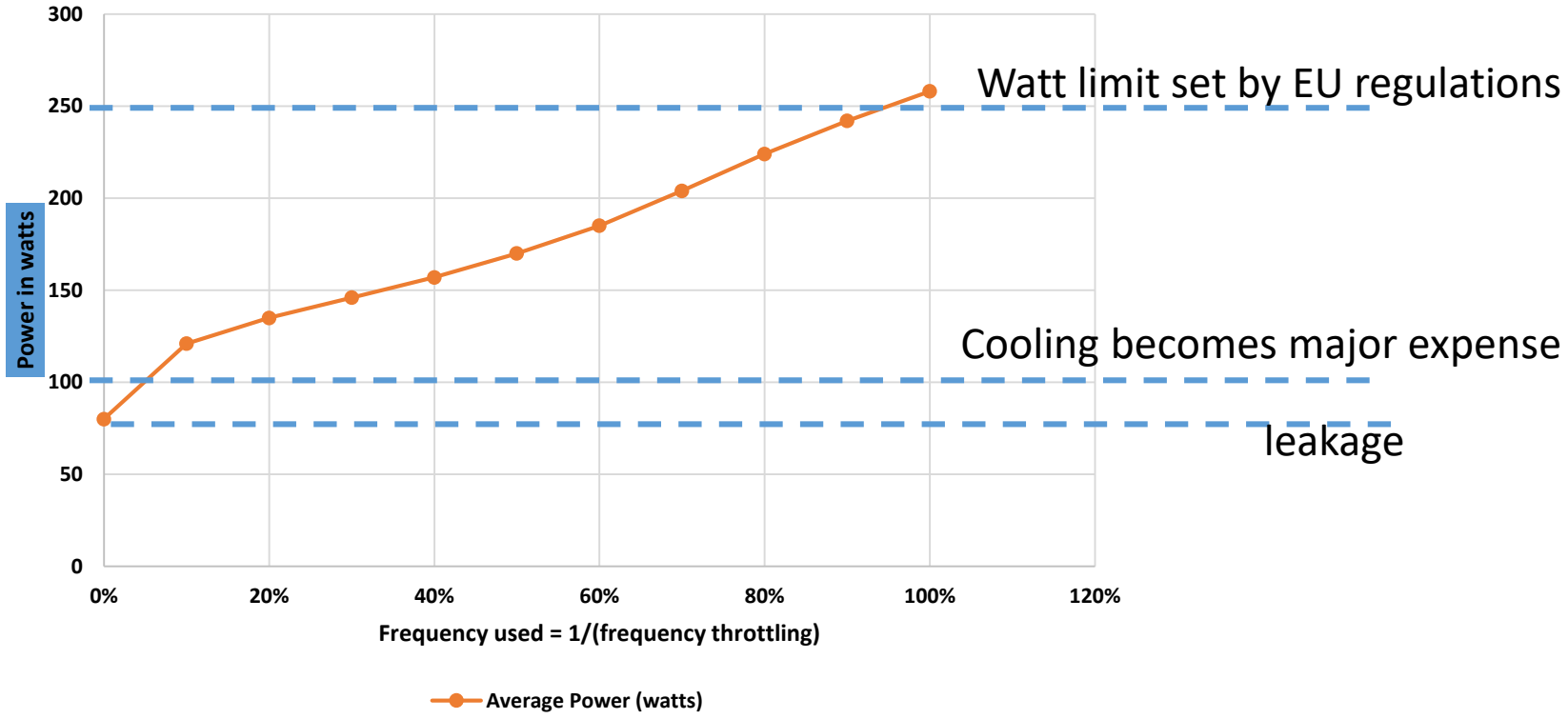
$$P_{leakage} \approx V_{dd} . I_{leakage}$$

$$I_{leakage} \approx I_0 e^{\left(\frac{V_{gs}-V_t}{nV_T}\right)}, I_0 = \mu_0 C_{ox} \frac{W}{L}(n-1)V_T^2$$



Is that a real problem?

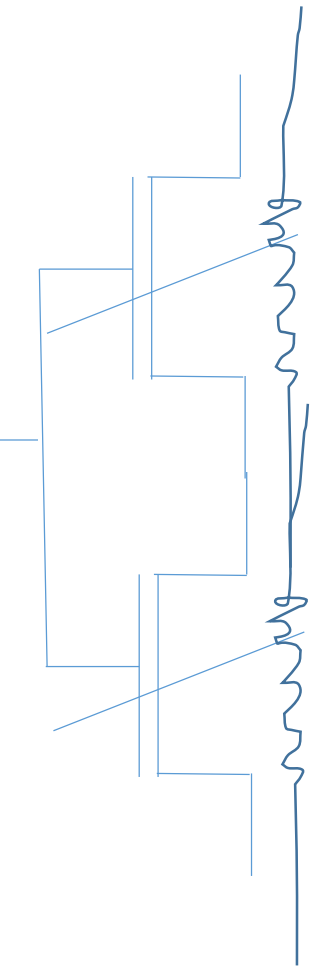
Data from Intel i7 2017:



100W is a limit for cooling expenses in enterprise servers

There is also regulation for per year energy for each computer so this is another problem to consider for enterprise servers

Static-power reduction techniques



Leakage Power Reduction

1. Technology Level
2. Stacking Effect
3. Body Bias
4. Adaptive Body Bias
5. Sleep Transistor (Power Gating)

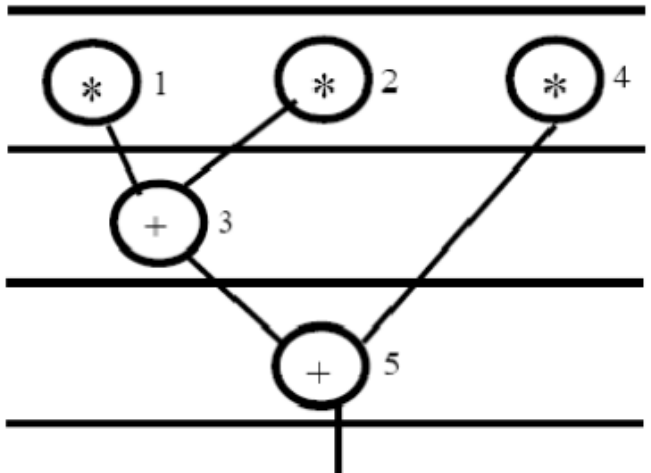
1.Technology level

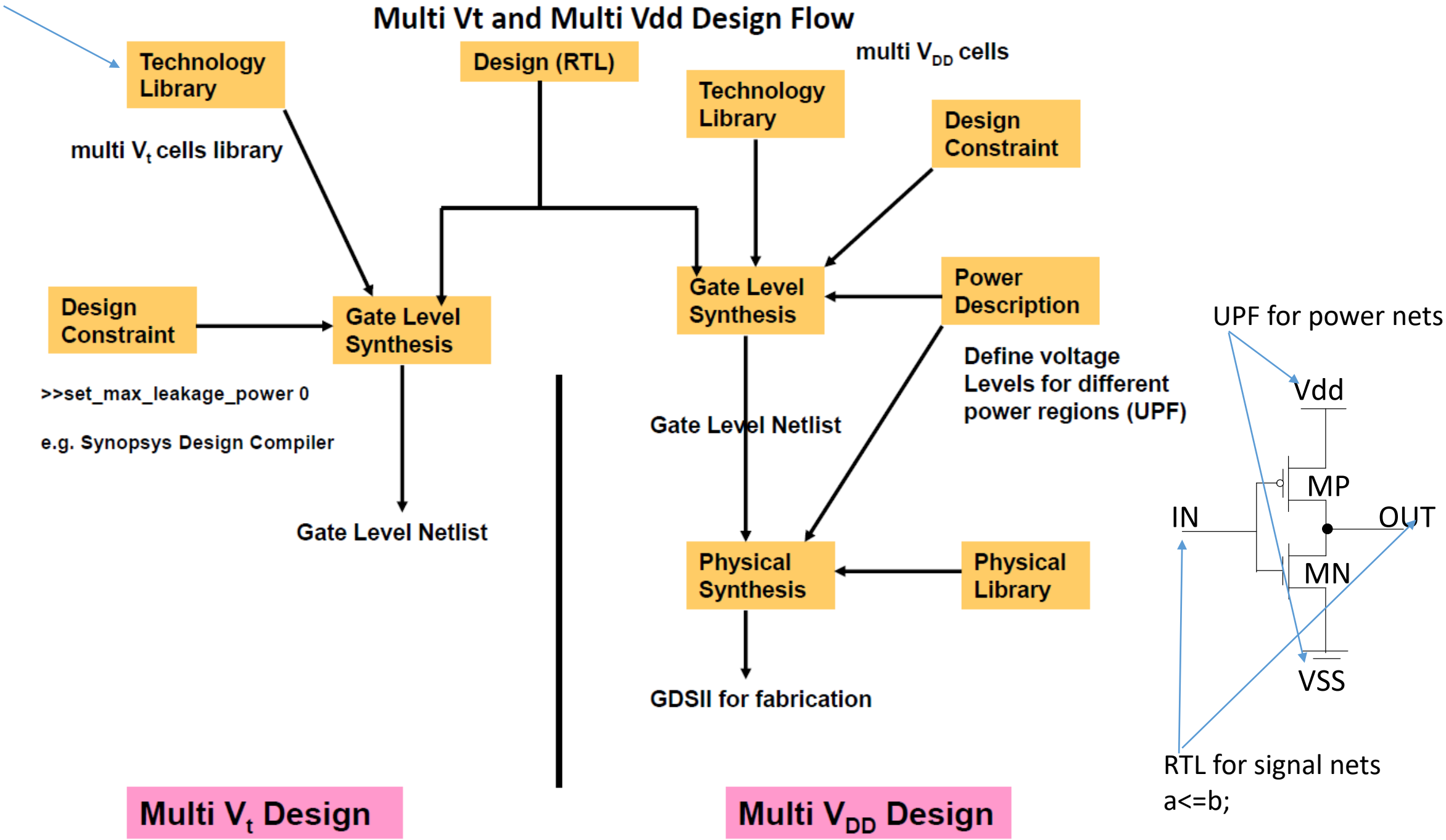
Multiple V_t technology where fabrication process provides both high and low threshold transistor V_t (low: fast and leaky and high: slow and less leaky) for N and P transistors. Default design uses high V_t and careful replacing high V_t with low V_t, one can get low V_t performance and yet significantly lower leakage power.

Synopsys Galaxy Design Platform provides an automated way of embedding *multi-V_t* to reduce leakage power

Example

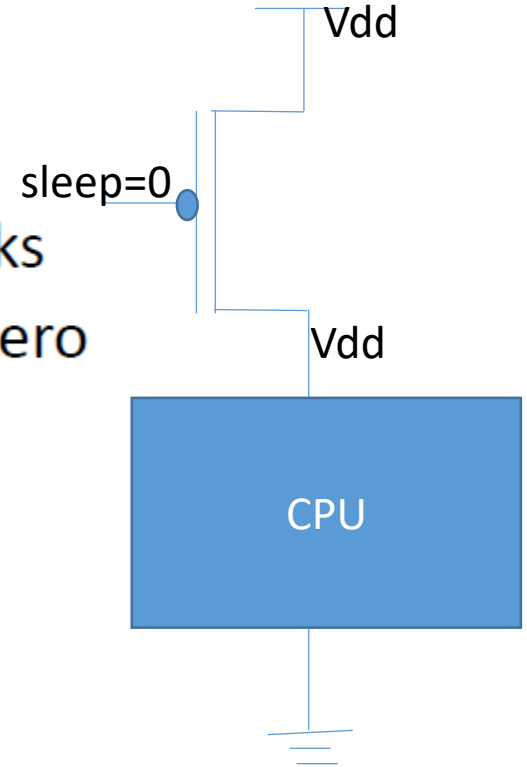
critical path delay determines performance of the design, all other paths have lower delay. Use low V_t transistors in modules on the critical path, and high V_t transistors in modules on non-critical paths.

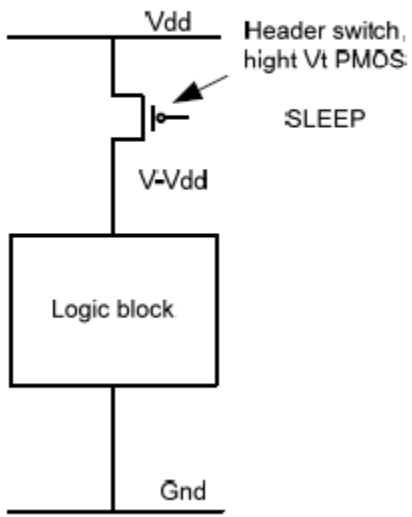


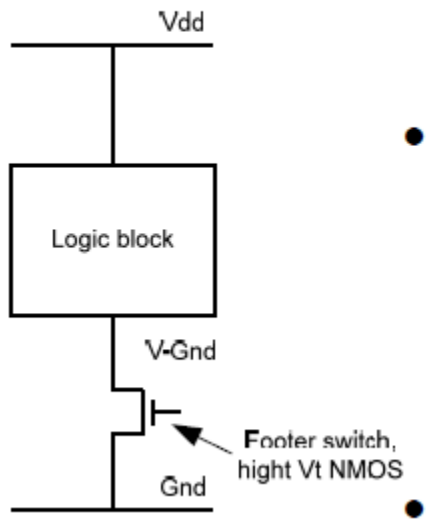

Multi Vt and Multi Vdd Design Flow
Technology Library
multi V_t cells library
Design (RTL)
Design Constraint
Gate Level Synthesis
>>set_max_leakage_power 0
e.g. Synopsys Design Compiler
Gate Level Netlist
Multi V_t Design
multi V_{DD} cells
Technology Library
Design Constraint
Gate Level Synthesis
Power Description
Define voltage Levels for different power regions (UPF)
Gate Level Netlist
Physical Synthesis
Physical Library
GDSII for fabrication
Multi V_{DD} Design
UPF for power nets
Vdd
MP
IN
OUT
MN
VSS
RTL for signal nets
a<=b;

5. Power Gating

- Sleep transistors
  - Insert high V_t MOSFET between power rail and logic blocks
  - Header transistor off lead to logic block floating to near zero
  - Footer transistor off lead to logic block floating to near 1







- Header switches preferred if multiple power rails since gated block logics float to zero irrespective of supply voltages
- See ref [8] for more details

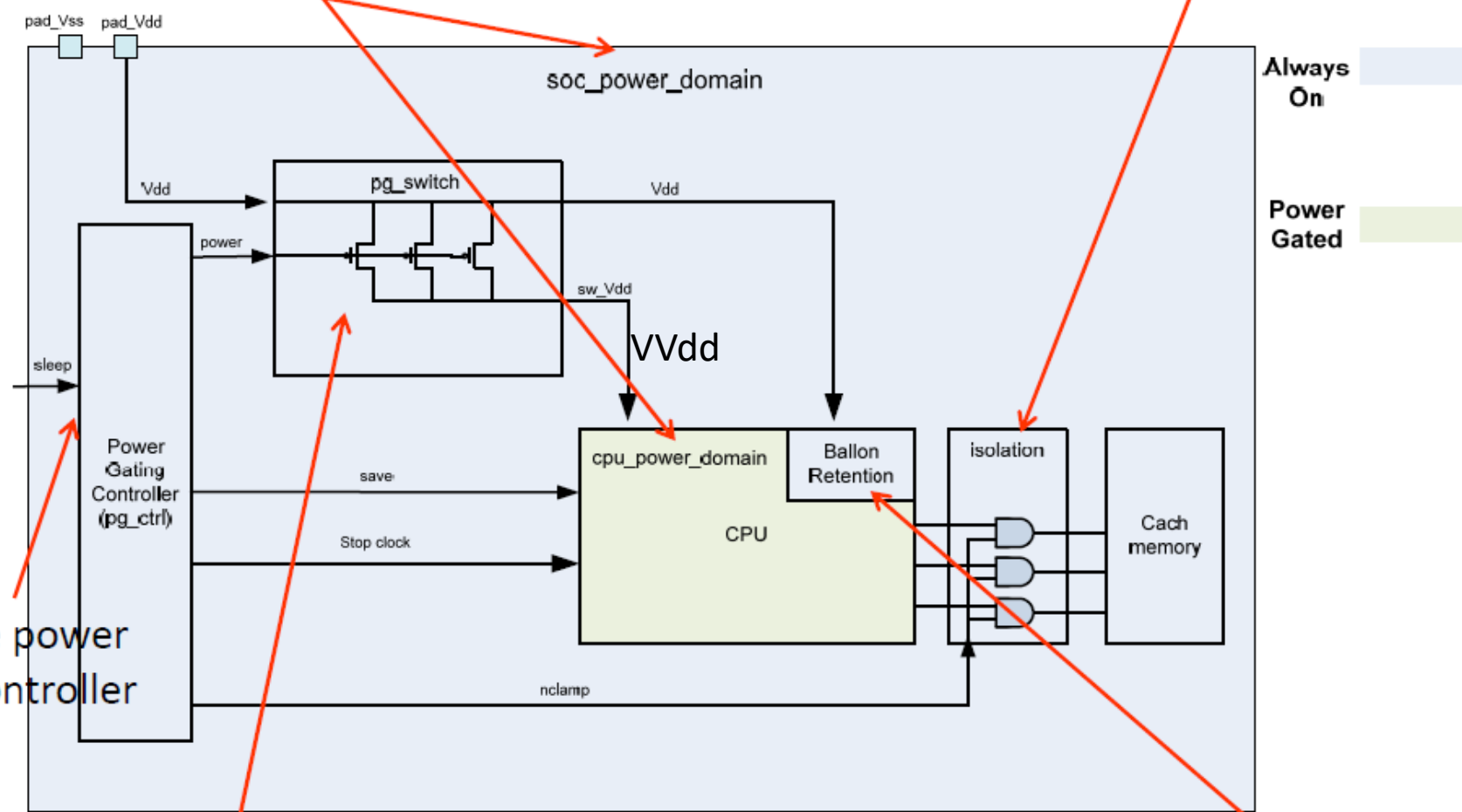
Example: SoC with Power Gated CPU

Fast latch

Slow latch

1. Define power domain and create supply net for each power domain

3. Create output isolation



5. Create power gating controller

2. Create power switches

4. Create state retention

UPF (Unified Power Format)

1. Define power domain and create supply net for each power domain

```
create_power_domain soc_power_domain
create_power_domain cpu_power_domain -elements {CPU}

set_domain_supply_net soc_power_domain \
                      -primary_power_net Vdd \
                      -primary_ground_net Vss

set_domain_supply_net cpu_power_domain \
                      -primary_power_net sw_Vdd \
                      -primary_ground_net Vss
```

2. Create power switches

```
create_power_switch pg_switch -domain cpu_power_domain \
                      -input_supply_port {Vdd Vdd} \
                      -output_supply_port {sw_Vdd sw_Vdd} \
                      -control_port {power pg_ctrl/power} \
                      -on_state {on_state Vdd {power}} \
                      -off_state {off_state {!power}}
```

3. Create output isolation

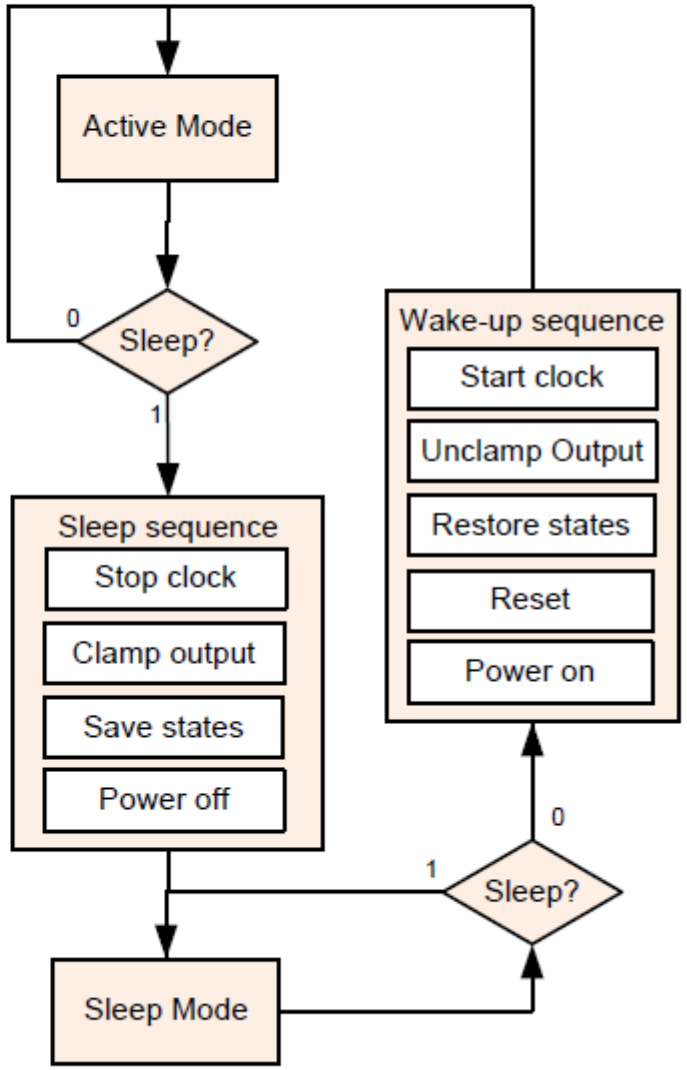
```
set_isolation iso1 -domain cpu_power_domain \
                   -isolation_power_net Vdd \
                   -isolation_ground_net Vss \
                   -clamp_value 0 \
                   -applies_to outputs

set_isolation_control iso1 -domain cpu_power_domain \
                           -isolation_signal pg_ctrl/nclamp \
                           -isolation_sense low \
                           -location parent
```

4. Create state retention

```
set_retention ret1 -domain cpu_power_domain \
                   -retention_power_net Vdd \
                   -retention_ground_net Vss

set_retention_control ret1 -domain cpu_power_domain \
                           -save_signal {pg_ctrl/retain high} \
                           -restore_signal {pg_ctrl/retain low}
```

5. Create power gating controller FSM (RTL model)

- Require a controller to:
  - clamp the power gated block output and save the states before power off
  - Restore the states and unclamp the output after power on



Power Gating Design Synthesis Flow

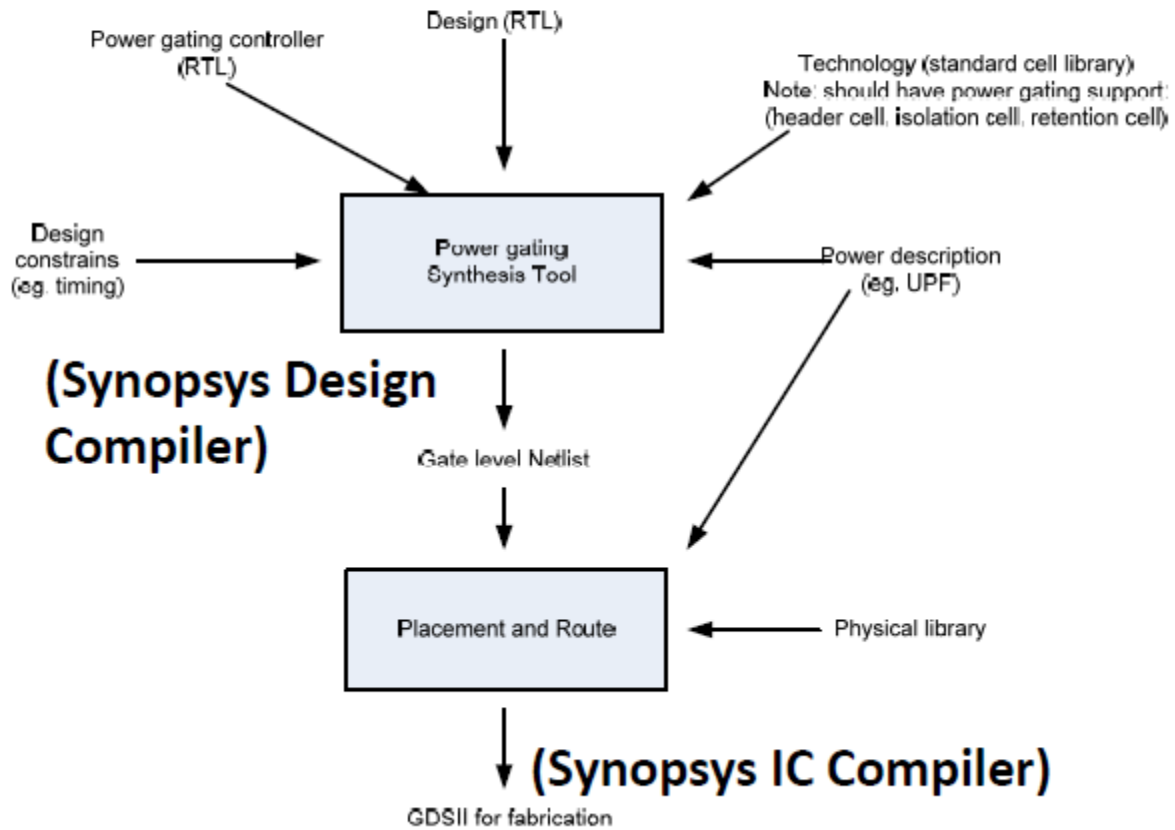
Power gating design synthesis requires:

1. Power descriptions (upf file)

2. Power gating controller model (RTL)

3. Cell library with power gating support

4. Synthesis Tool with power gating capability (e.g. latest version of Synopsys Design Compiler)



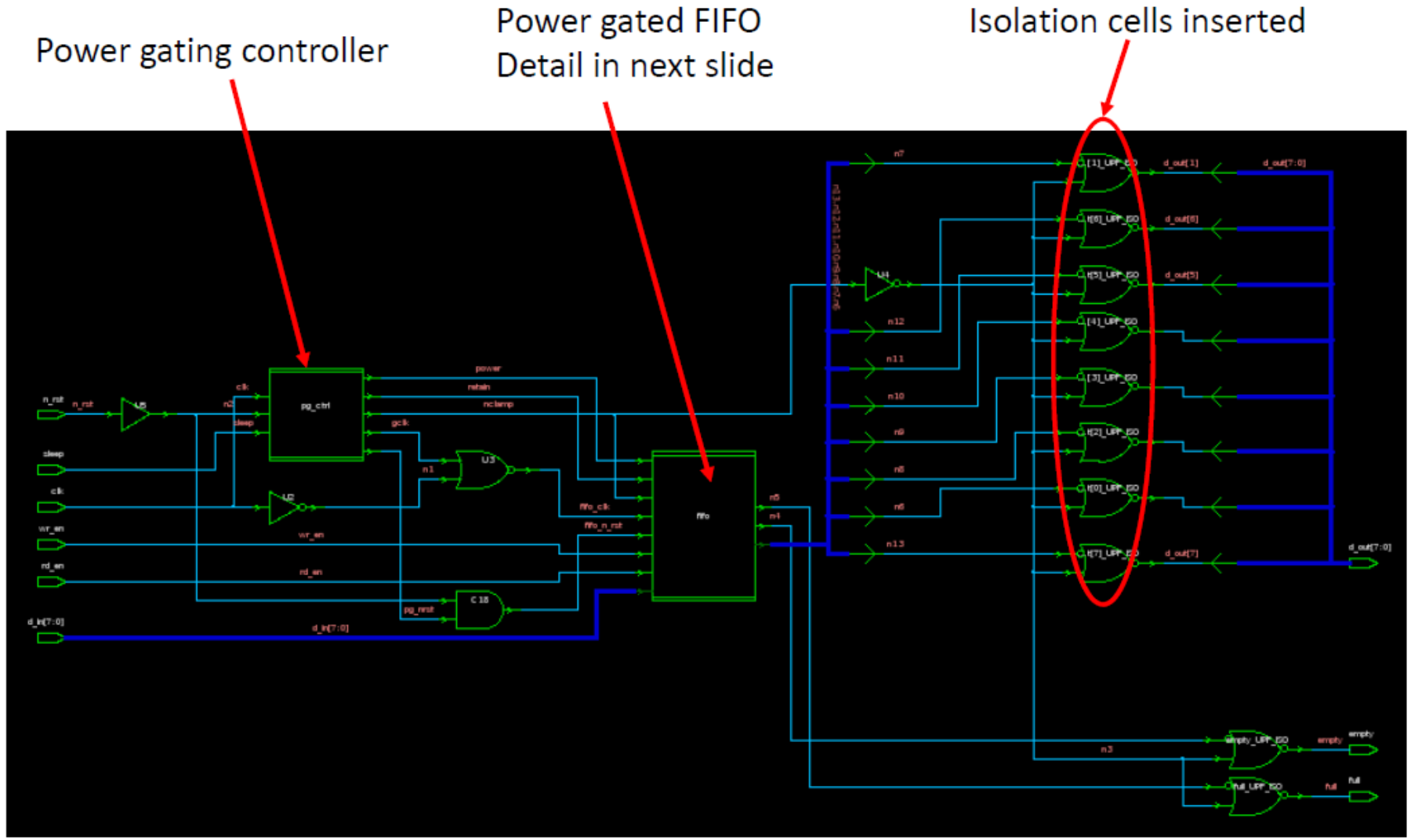
FIFO (First In First Out)
A design to be power gated

```
module fifo(power, retain, nclamp
        ,clk ,n_rst ,wr_en ,rd_en ,d_in
        ,d_out ,full ,empty
);

    parameter DATA_WIDTH = 8;
    parameter ADDR_WIDTH = 3;
    parameter DEPTH = 8;

    input power, retain, nclamp;
    input clk, n_rst, wr_en, rd_en;
    input [DATA_WIDTH-1 : 0] d_in;
    output full, empty;
    output [DATA_WIDTH-1 : 0]   d_out;
    reg [ADDR_WIDTH : 0]        distance;
    reg [DATA_WIDTH-1 : 0]  buffer [DEPTH-1 : 0];
    reg [ADDR_WIDTH-1 : 0]  wr_p, rd_p;
    integer  i;

    always @ (posedge clk) begin
      if(~n_rst) wr_p <= {(ADDR_WIDTH){1'b0}};
      else if(wr_en&&!full) wr_p <= wr_p + 1;
    end

    always @ (posedge clk) begin
      if(~n_rst) rd_p <= {(ADDR_WIDTH){1'b0}};
      else if(rd_en&&!empty) rd_p <= rd_p + 1;
    end

    always @ (posedge clk) begin
      if(~n_rst) distance <= {(ADDR_WIDTH+1){1'b0}};
      //if only read, decrease distance
      else if(rd_en&&!empty&&(!wr_en||full)) distance <= distance - 1;
      //if only write, increase distance
      else if(wr_en&&!full&&(!rd_en||empty)) distance <= distance + 1;
      //otherwise keep distance the same
    end

     always @ (posedge clk) begin
        if(~n_rst) begin
            for (i=0; i<DEPTH; i=i+1) buffer[i] <= {(DATA_WIDTH){1'b0}};
        end
        else if(wr_en&&!full) begin
            buffer[wr_p] <= d_in;
        end
     end

     assign full = (distance == DEPTH)? 1'b1 : 1'b0;
     assign empty = (distance == 0)? 1'b1 : 1'b0;
     assign d_out = buffer[rd_p];

endmodule // fifo
```

Power gating controller

```
module pg_ctrl(clk, n_rst, sleep, power, retain, nclamp, gclk, pg_nrst);

    //power control states talbe
    parameter pcs_active    = 7'b101_0100;
    parameter pcs_stopclk   = 7'b101_1110;
    parameter pcs_clamp     = 7'b100_1110;
    parameter pcs_save      = 7'b110_1110;
    parameter pcs_reset     = 7'b110_1010;
    parameter pcs_poweroff  = 7'b010_1010;
    parameter pcs_sleep     = 7'b010_1000;
    parameter pcs_poweron   = 7'b110_1001;
    parameter pcs_resetoff  = 7'b110_1101;
    parameter pcs_restore   = 7'b100_1101;
    parameter pcs_unclamp   = 7'b101_1101;

    input   clk, n_rst, sleep;
    output  power, retain, nclamp, gclk, pg_nrst;
    reg [6:0]   pcs;

    //control power state transistions
    always @ (posedge clk, negedge n_rst) begin
        if(~n_rst) pcs <= pcs_active;
        else begin
            if ((pcs == pcs_active) && (sleep))     pcs <= pcs_stopclk;
            if (pcs == pcs_stopclk)     pcs <= pcs_clamp;
            if (pcs == pcs_clamp)       pcs <= pcs_save;
            if (pcs == pcs_save)        pcs <= pcs_reset;
            if (pcs == pcs_reset)       pcs <= pcs_poweroff;
            if (pcs == pcs_poweroff)    pcs <= pcs_sleep;
            if ((pcs == pcs_sleep) && (~sleep))         pcs <= pcs_poweron;
            if (pcs == pcs_poweron)     pcs <= pcs_resetoff;
            if (pcs == pcs_resetoff)    pcs <= pcs_restore;
            if (pcs == pcs_restore)     pcs <= pcs_unclamp;
            if (pcs == pcs_unclamp)     pcs <= pcs_active;
        end
    end

    assign power    = pcs[6];
    assign retain   = pcs[5];
    assign nclamp   = pcs[4];
    assign gclk     = pcs[3];
    assign pg_nrst  = pcs[2];

endmodule
```

Schematic of Synthesized Gate Level Netlist of the above Example

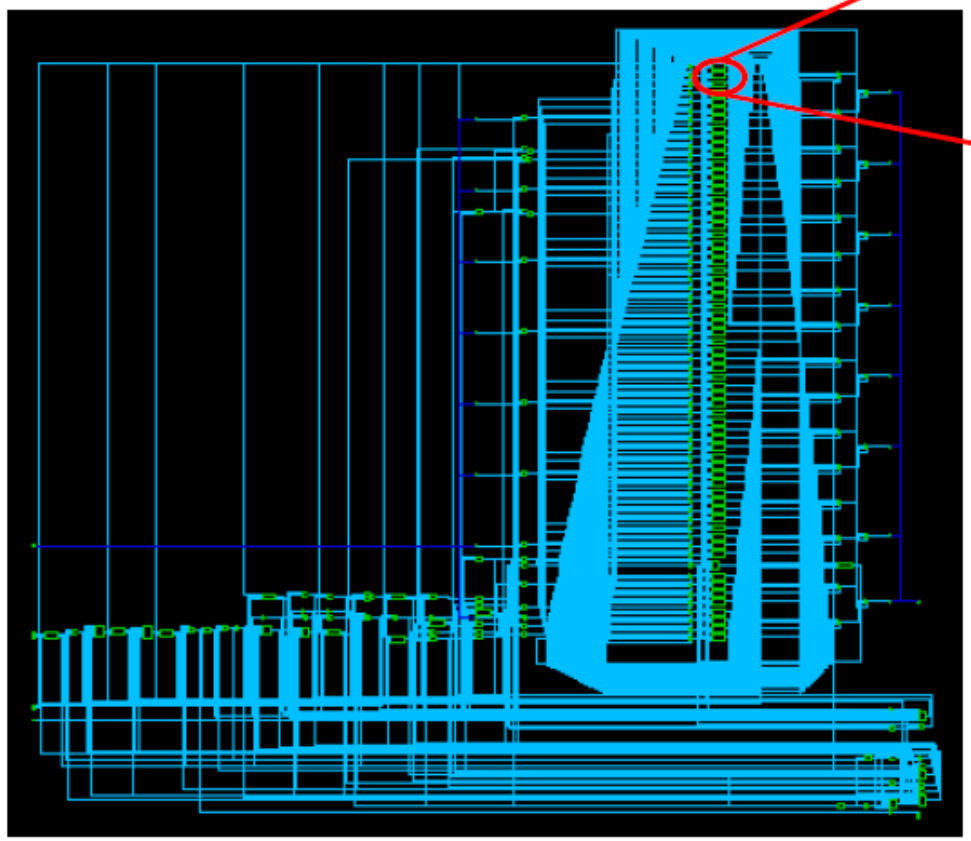


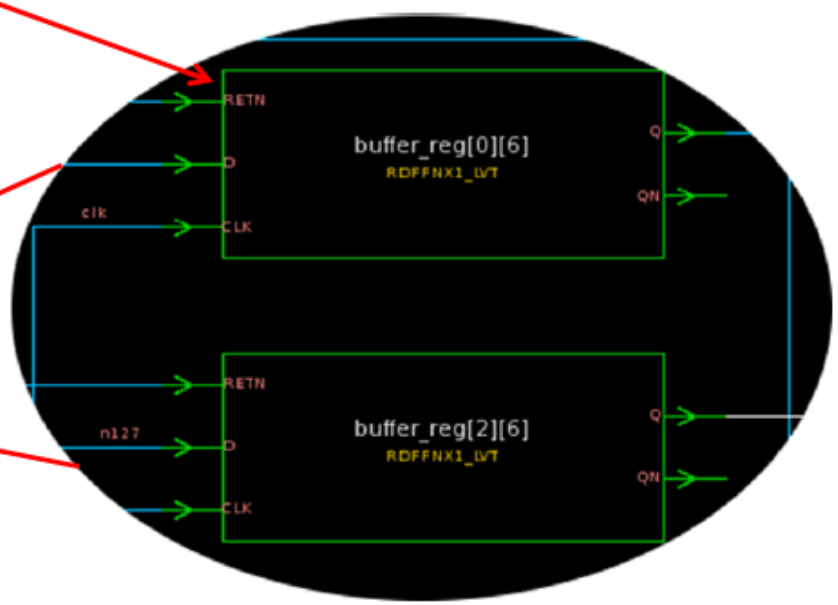
Note: (i) Power Domain is not created just after logic synthesis

(ii) Physical synthesis is necessary for that

retention registers

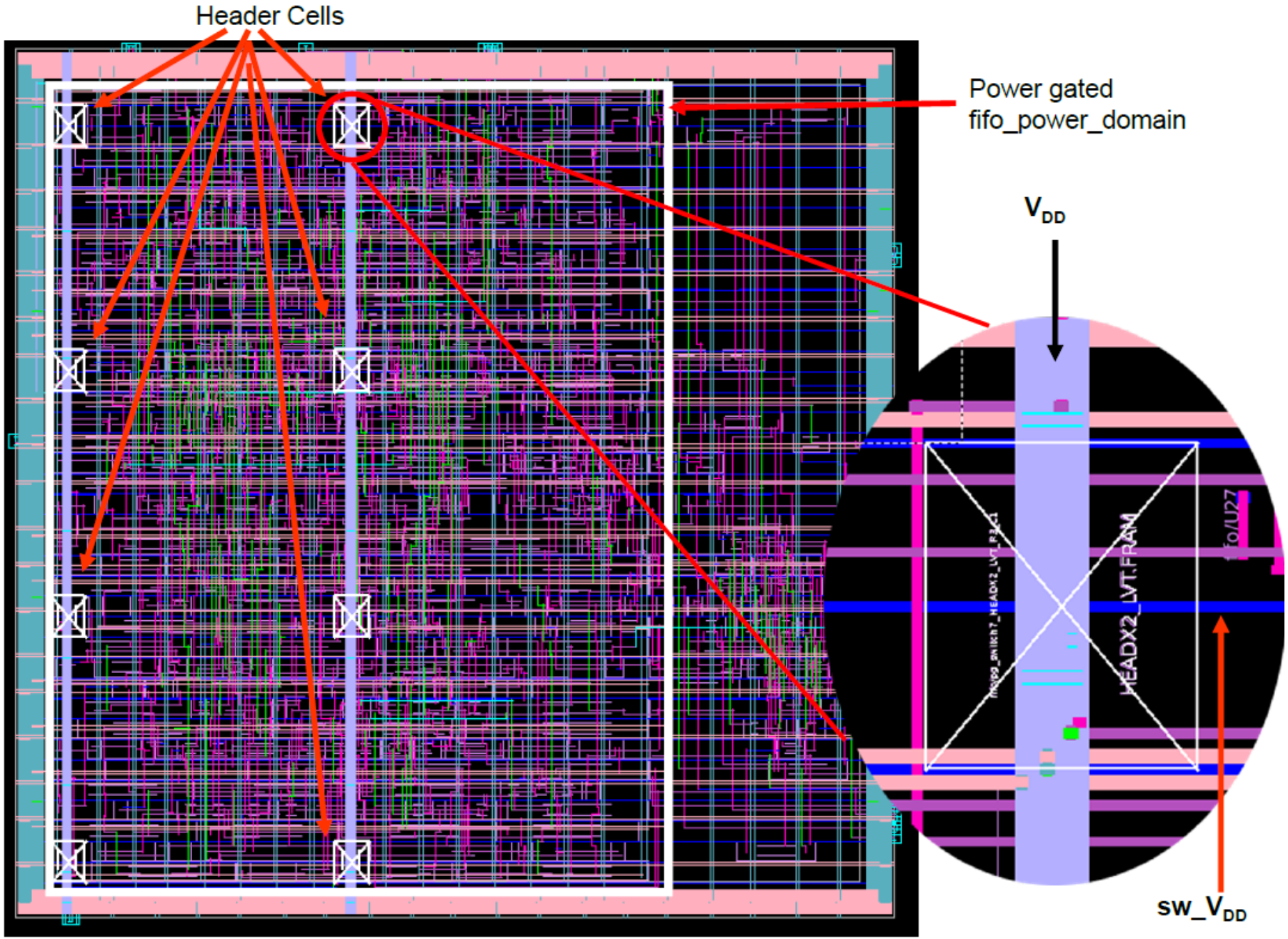
Power gated FIFO with normal registers replaced with retention registers





Note: The sleep transistors are not created in logic synthesis, they will be created after physical synthesis

Schematic of chip layout after physical synthesis (Place and Routed Design)



References

1. http://www.synopsys.com/products/power/multivoltage_bkgrd.pdf

2. R.X. Gu, et al , Power dissipation analysis and optimisation of DSM CMOS digital circuits, IEEE J. Solid State Circuits, 31, pp.707, 1996

3. K.S. Khouri, et al, Leakage power analysis and reduction during behavioural synthesis, IEEE Transactions on CAD, 10(6), pp.876, 2002

4. J. Tschanz et all, Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage", IEEE Journal of Solid State Circuits, 2002, pp:1396

5. D., Lee, et al, Gate oxide leakage current analysis and reduction in VLSI circuits, IEEE Transactions on VLSI system, 12(2), pp. 155, 2004

6. K. Roy, et al, Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-submicrometer CMOS Circuits, Proceedings of the IEEE, Vol 91, No. 2, 2003

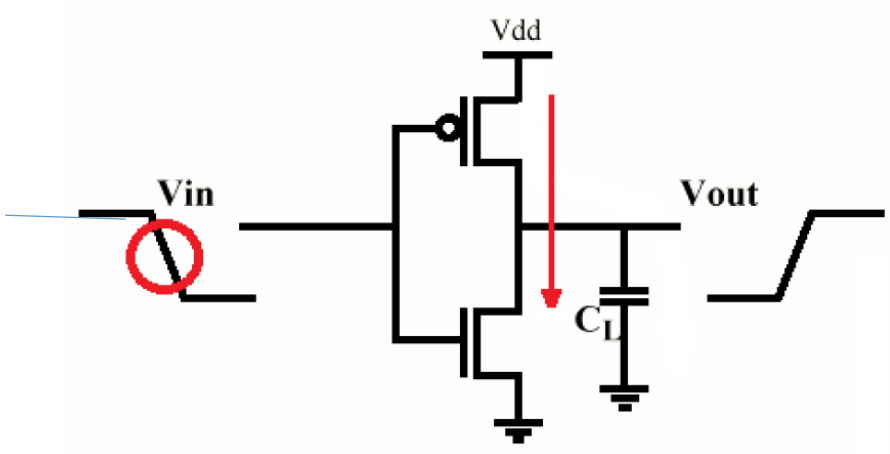
7. Bashir Al. Hashimi, System on Chip: Next Generation Electronics, chapter 13, IEE Circuits, Devices and Systems Series 18.

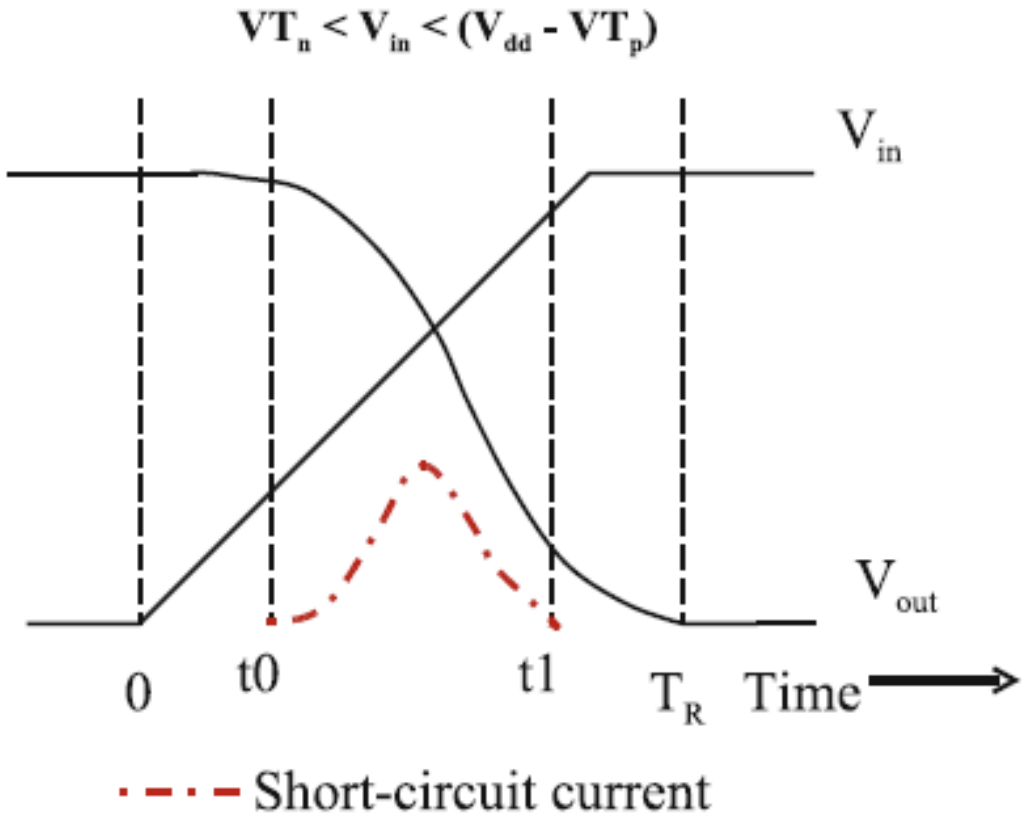
8. Michael Keating *et al*, Low Power Methodology Manual, chapter 5 and 14, Springer.

Short-circuit power

Short-circuit power

Short circuit power is consumed in a circuit when both the nMOS and pMOS transistors are “on”

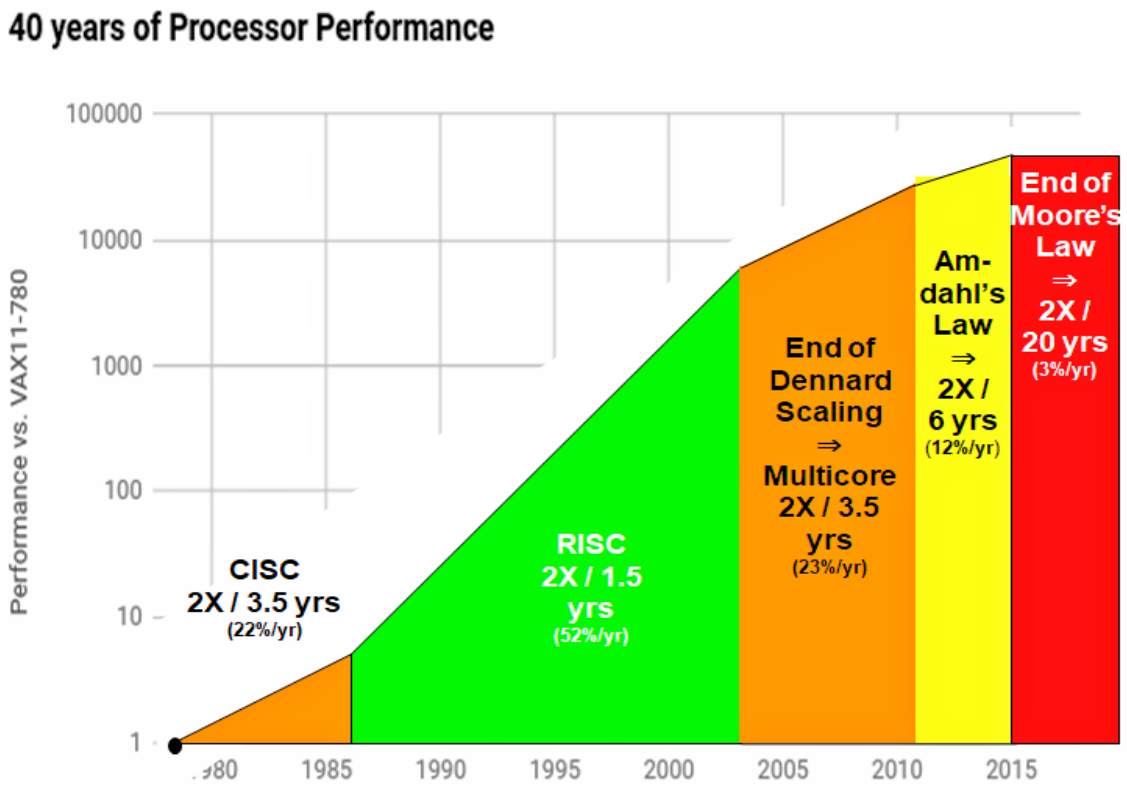




$$P_{short-circuit} = 1/12.k.\tau.F_{clk}.(V_{dd} - 2V_t)^3$$

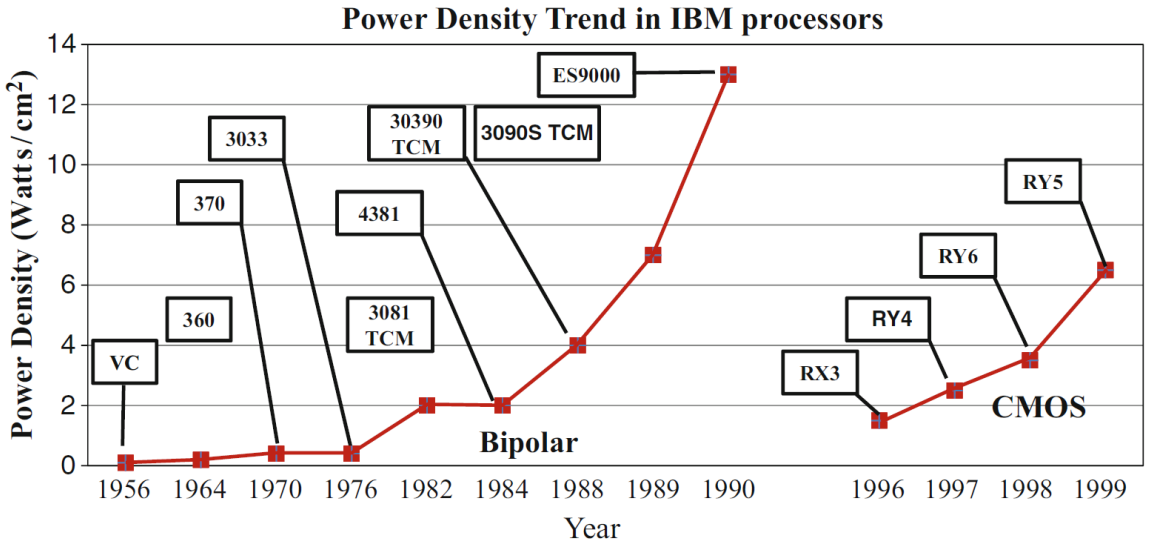
where τ is the rise time and fall time (assumed equal) and *k* is the gain factor of the transistor

The future…



What about the future? is it that bad?

Well we need again to check the past:

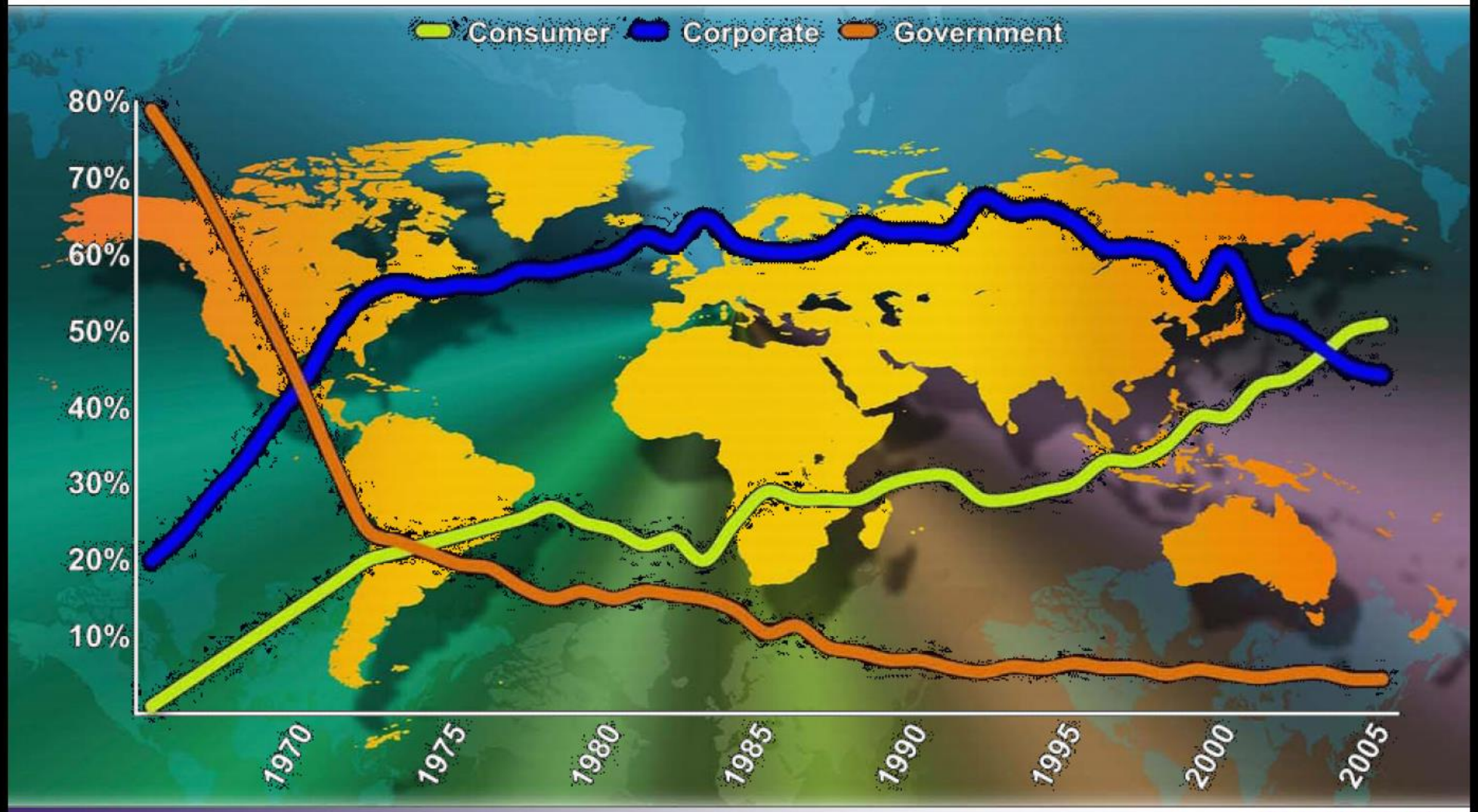


a similar scaling problem occurred in the 80's with CMOS technology replacing Bipolar. There is research on new materials, computing technologies and of course low-power design techniques to tackle leakage! We need also to adapt our markets…

Markets??? What about markets?

The Era Of The Consumer is it over?

Households, Semiconductor Industry's #1 Customer

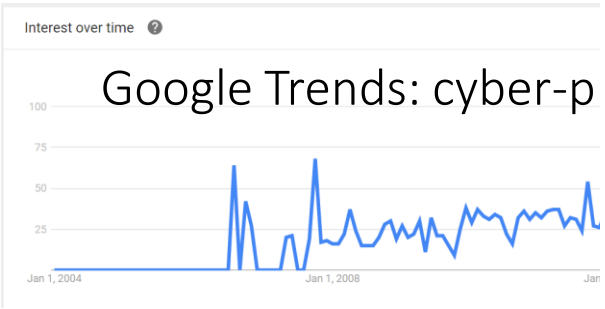


Source: J.-H. Huang, SIA 2004

Smart applications of embedded systems (Cyber-Physical Systems and Internet of Things (IoT))

1926 Nikola Tesla

"When wireless is perfec
converted into a huge br
instruments through wh
be amazingly simple con
A man will be able to car



Source: everything 4.0? Sabina Jesch
2013, invited talk Wuppertal

Back to: The earth converted into a huge "brain"... (Tesla 1926)

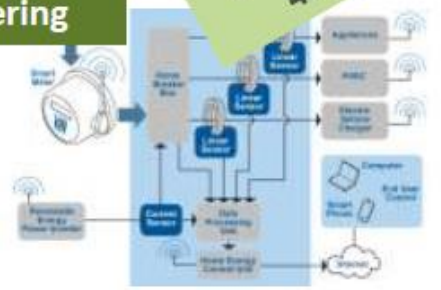
Integrating complex information from multiple heterogenous sources opens multiple possibilities of optimization: e.g. energy consumption, security services, rescue services as well as increasing the quality of life

Building automation

„house 4.0"

Smart metering

„metering 4.0"



Smart grid

„power grid 4.0"

Room automation

„room 4.0"

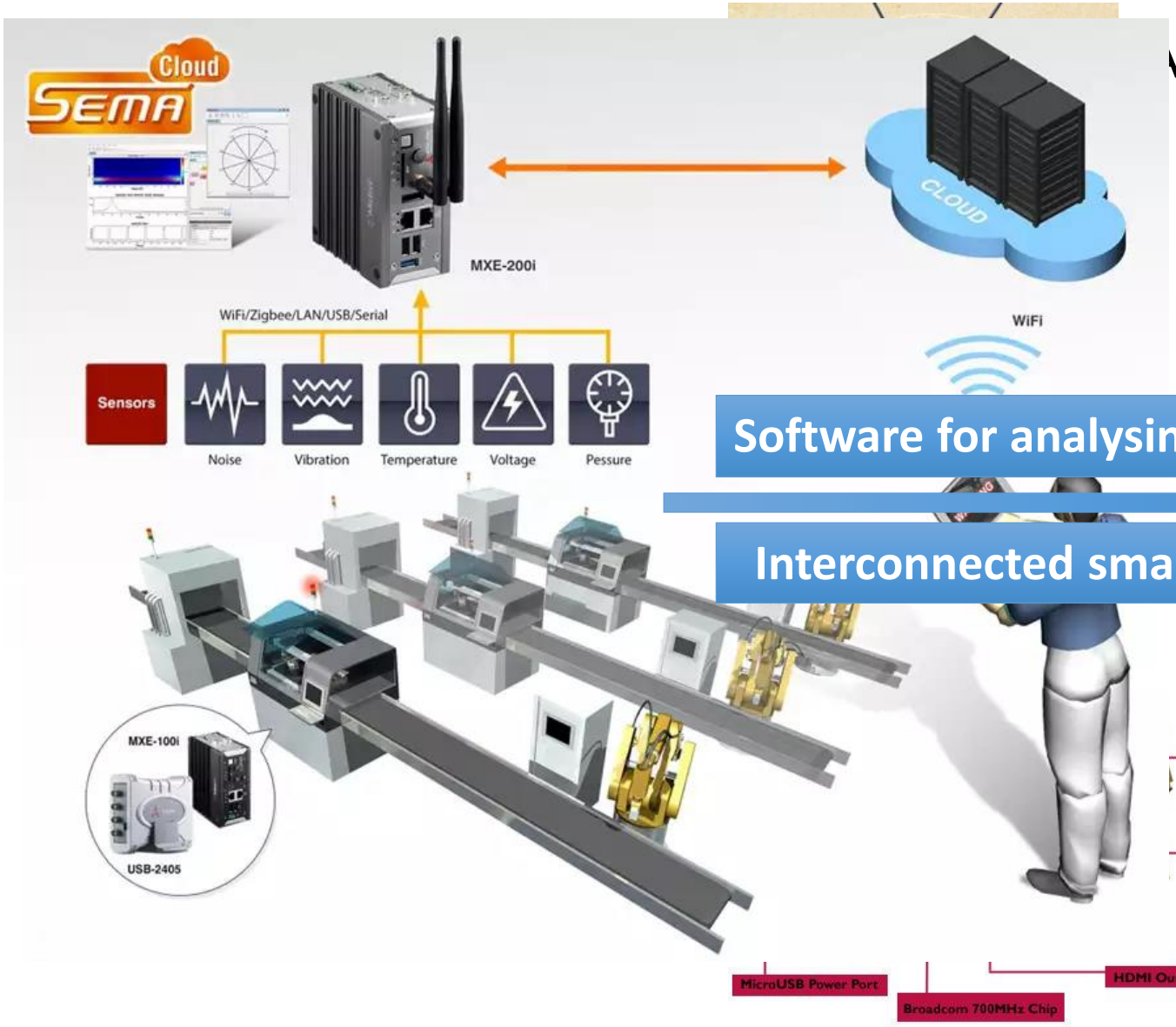
Smart environment

„mobility 4.0"

„...th 4.0"

... and more

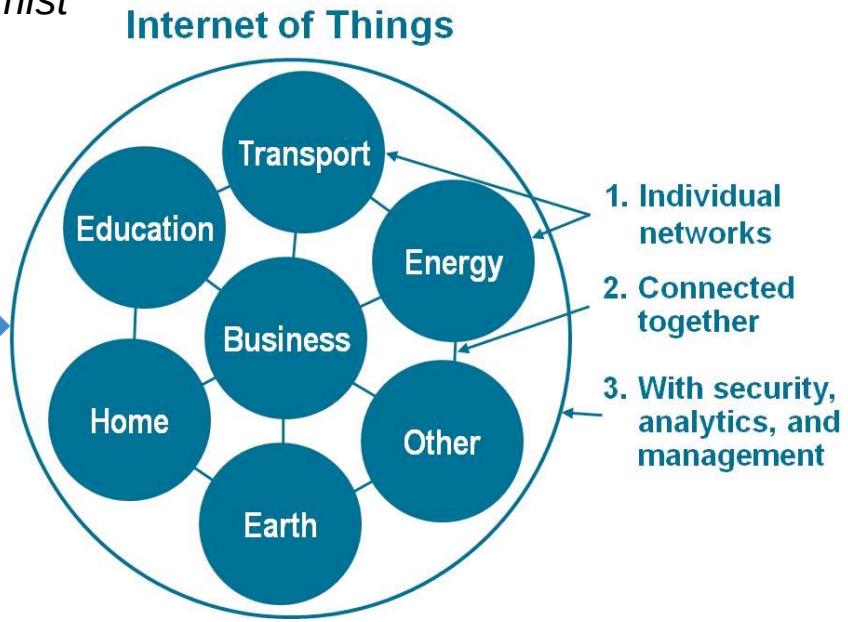
Drivers for CPS/IoT applications



Holy Cow!
A cow generates …
200 MB a year
Source: Augmented business, economist

Software for analysing big data

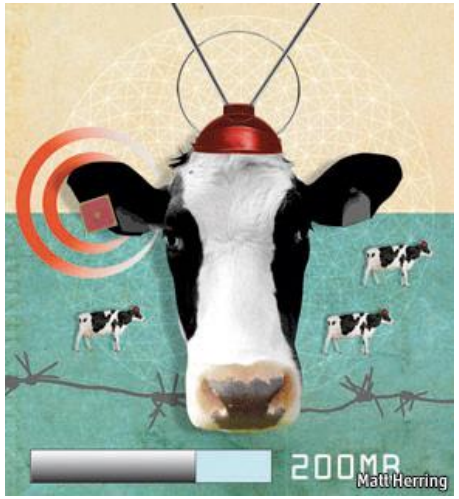
Interconnected small devices

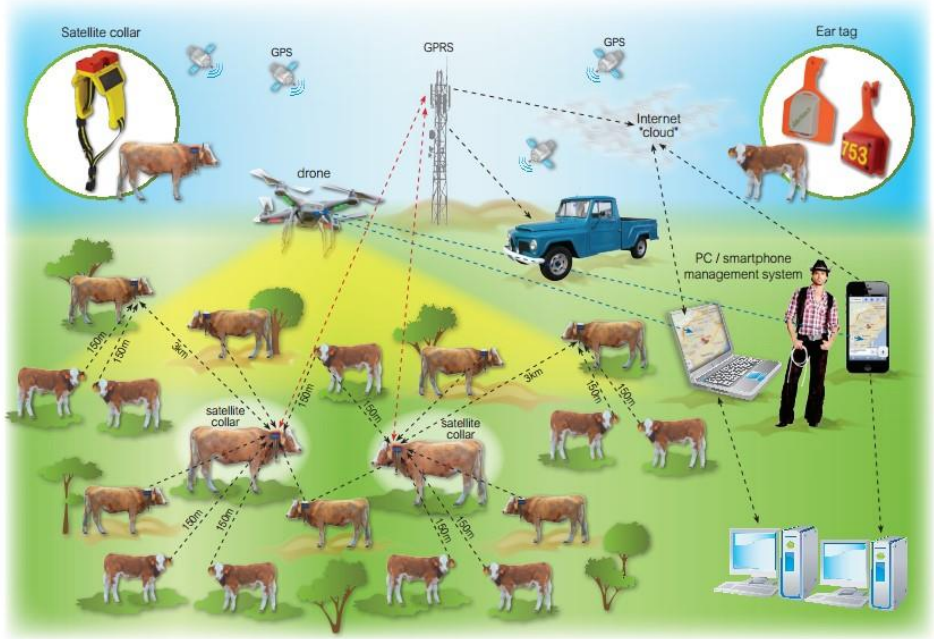


Wearable embedded systems on cattle

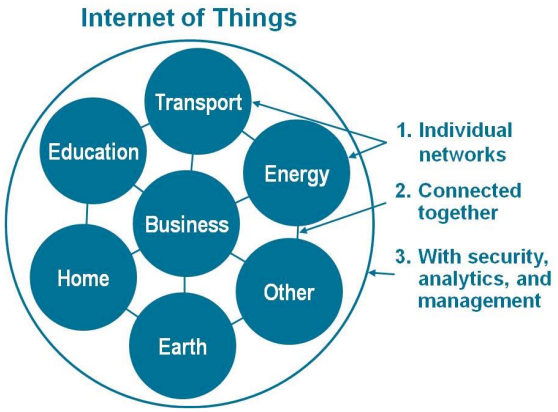
For monitoring their position and status…

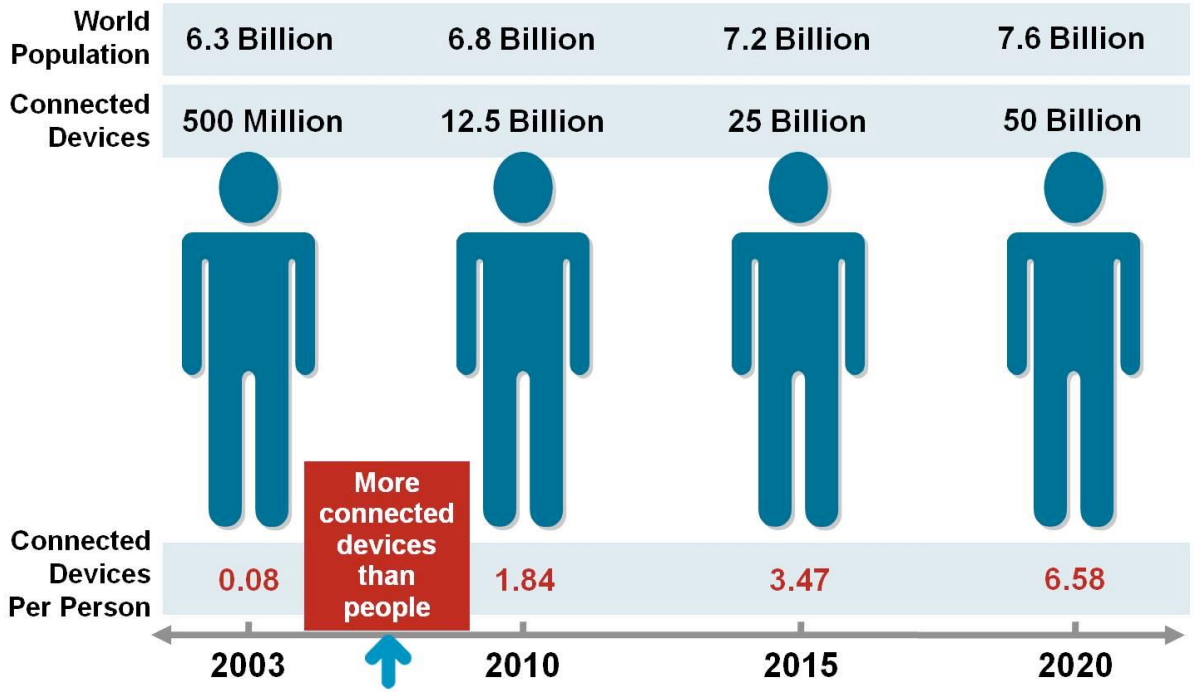
Holy Cow!
A cow generates …
200 MB a year
Source: Augmented business, economist





Energy efficiency: a possible bottleneck for IoT





- There is an exponential growth of the demand for connected devices
- There is a bound of 50 Billion connected devices due to **energy constraints** at the networking infrastructure (ARM white paper)
  - Low power digital hardware is required especially for IoT networking stacking (see mbed SoCs) **https://www.openfogconsortium.org**

D. Evans, "The internet of things: How the next evolution of the internet is changing everything," CISCO White paper, vol. 1, p. 14, 2011.
"The Intelligent Flexible Cloud," https://community.arm.com/docs/ DOC-9981, ARM Ltd, White Paper, 2015.
"Intelligent Flexible IoT Nodes," https://community.arm.com/docs/ DOC-10861, ARM Ltd, White Paper, 2015.