A Dual Voltage Dual Frequency Low Power VLSI Chip for Image Watermarking

Saraju P. Mohanty

Dept. of CSE, University of North Texas

smohanty@cs.unt.edu

http://www.cs.unt.edu/~smohanty/

Outline of the Talk

- Introduction
- Why Low Power ?
- Related Works
- Watermarking Algorithms
- Proposed Architecture
- Prototype Chip Implementation
- Conclusions

Why Low-Power ?

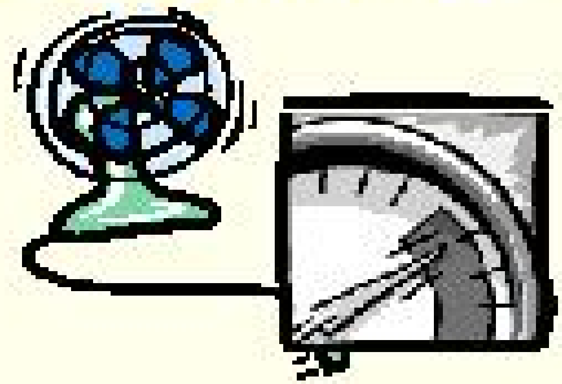
Major Motivation

Extending battery life for portable applications

Why Low-Power ? ……..

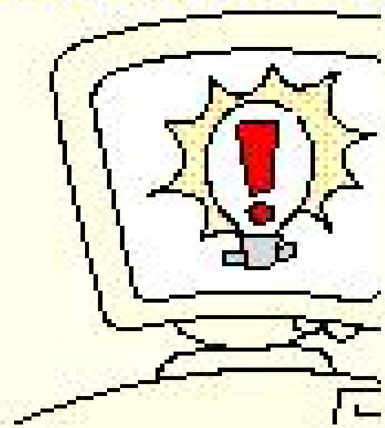
Battery lifetime

Cooling and energy costs

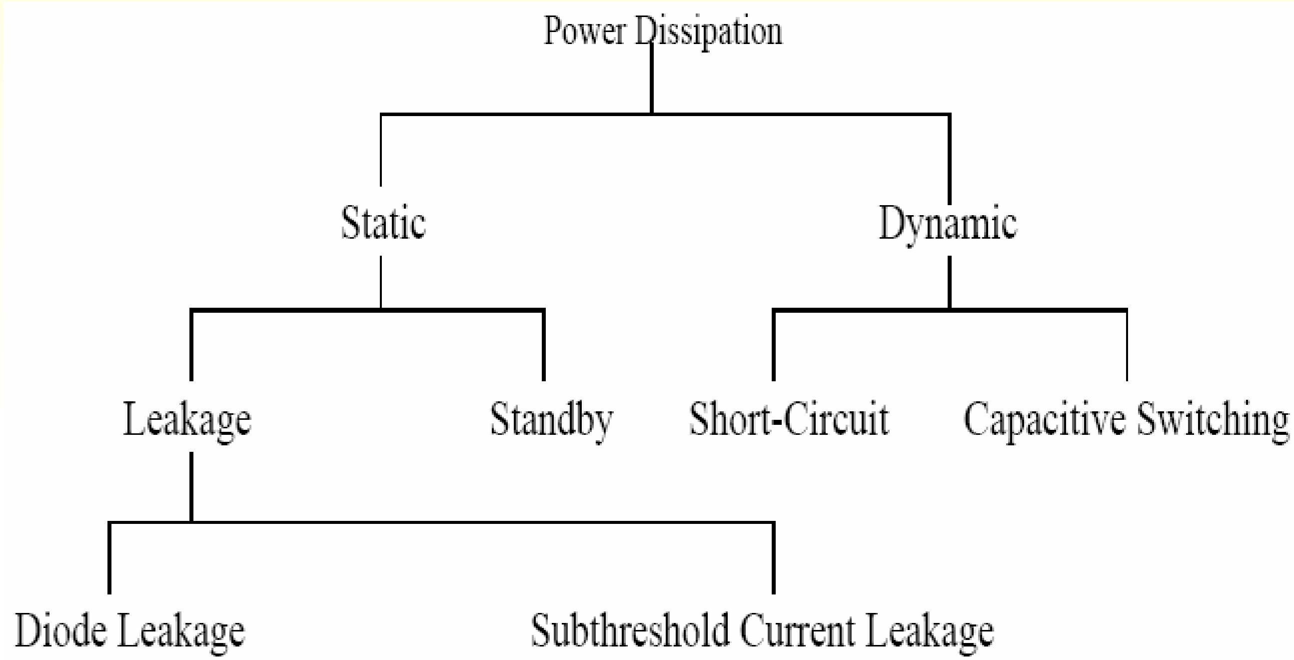


Environmental concerns

System reliability



Power Consumption in CMOS Circuits



Leakage Current: Reverse biased current in the parasitic diode and subthreshold current due to charge inversion existing at gate below V_T.

Standby Current: Continuous DC current from V_{dd} to ground

Short-Circuit Current: DC current from V_{dd} to ground during output transition

Capacitive Current: Flows to charge discharge capacitive loads.

Dynamic Power Consumption

Let, C_L = load capacitor, V_{dd} = supply voltage, N = average number of transitions/clock cycle = E(sw) = α and f = clock frequency. The dynamic power consumption for CMOS:

$$P_{dynamic} = \frac{1}{2} C_L V^2_{dd} N f$$

- Veendrick Observation: In a well designed circuit, short-circuit power dissipation is less than 20% of the dynamic power dissipation.
- Sylvester and Kaul: At larger switching activity the static power is negligible compared to the dynamic power.

We focus on dynamic power reduction !!

Dynamic Power Reduction ?

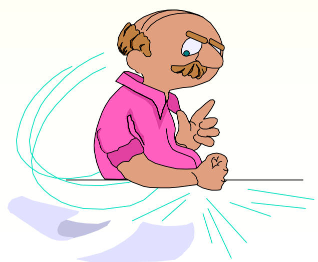
- Reduce Supply Voltage (V_{dd}): delay increases; performance degradation
- Reduce Clock Frequency (f): only power saving no energy savings; performance degradation
- Reduce Switching Activity (N or E(sw)): no switching no power loss !!! Not fully under designers control. Switching activity depends on the logic function and correlations are difficult to handle.
- Reduce Physical Capacitance: done by reducing device size reduces the current drive of the transistor making the circuit slow

Our Approach ?

Adjust the frequency and supply voltage in a co-coordinated manner to reduce dynamic power while maintaining performance.

Digital Watermarking ?

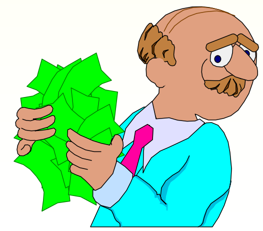
It is mine



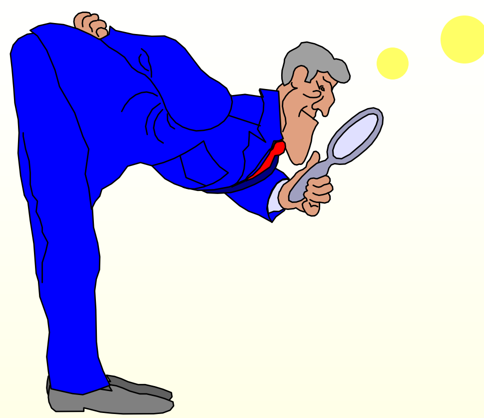
Hacker

Multimedia Object

No, It is mine



Owner



Researcher

Whose is it this ?
How to know ?
What's the solution of this ownership problem?

Solution: "WATERMARKING"

Digital Watermarking ?

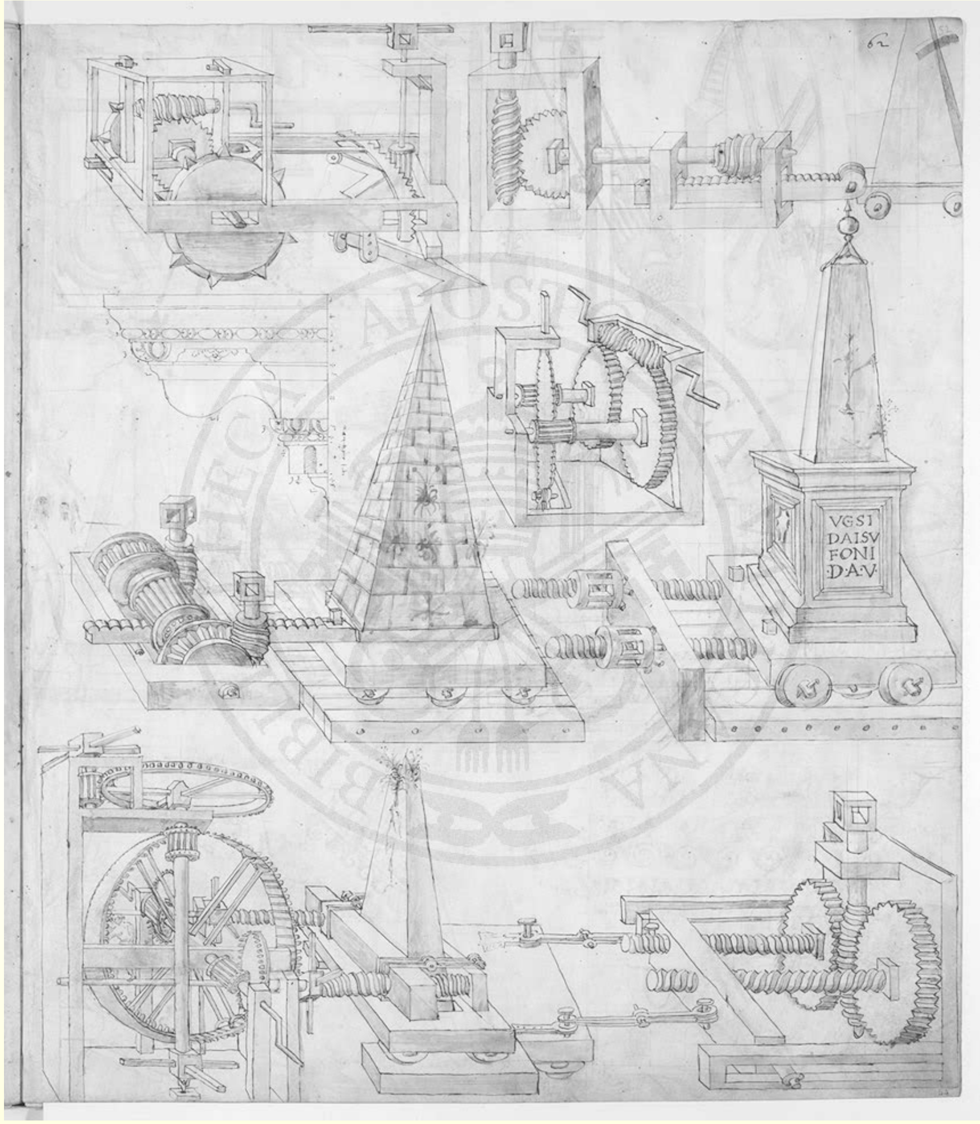
Digital watermarking is a process for embedding data (watermark) into a multimedia object for its copyright protection and authentication.

Types

- Visible and Invisible
- Spatial/DCT/ Wavelet
- Robust and Fragile

An Watermarked Image (from IBM)

Another Example ……..



Watermarking: General Framework

- Encoder: Inserts the watermark into the host image
- Decoder: Decodes or extracts the watermark from image
- Comparator: Verifies if extracted watermark matches with the inserted one

Why Hardware Implementation ?

Hardware implementations of watermarking algorithms necessary for various reasons:

- Easy integration with multimedia hardware, such as digital camera, camcorder, etc.
- Low power
- High performance
- Reliable
- Real time applications

Previous Work (Hardware based Watermarking)

Work	Type	Target Object	Domain	Technology	Chip Power
Strycker, 2000	Invisible Robust	Video	Spatial	NA	NA
Tsai and Lu 2001	Invisible Robust	Video	DCT	0.35μ	62.8 mW
Mathai, 2003	Invisible Robust	Image	Wavelet	0.18μ	NA
Garimella, 2003	Invisible Fragile	Image	Spatial	0.13μ	37.6 μW

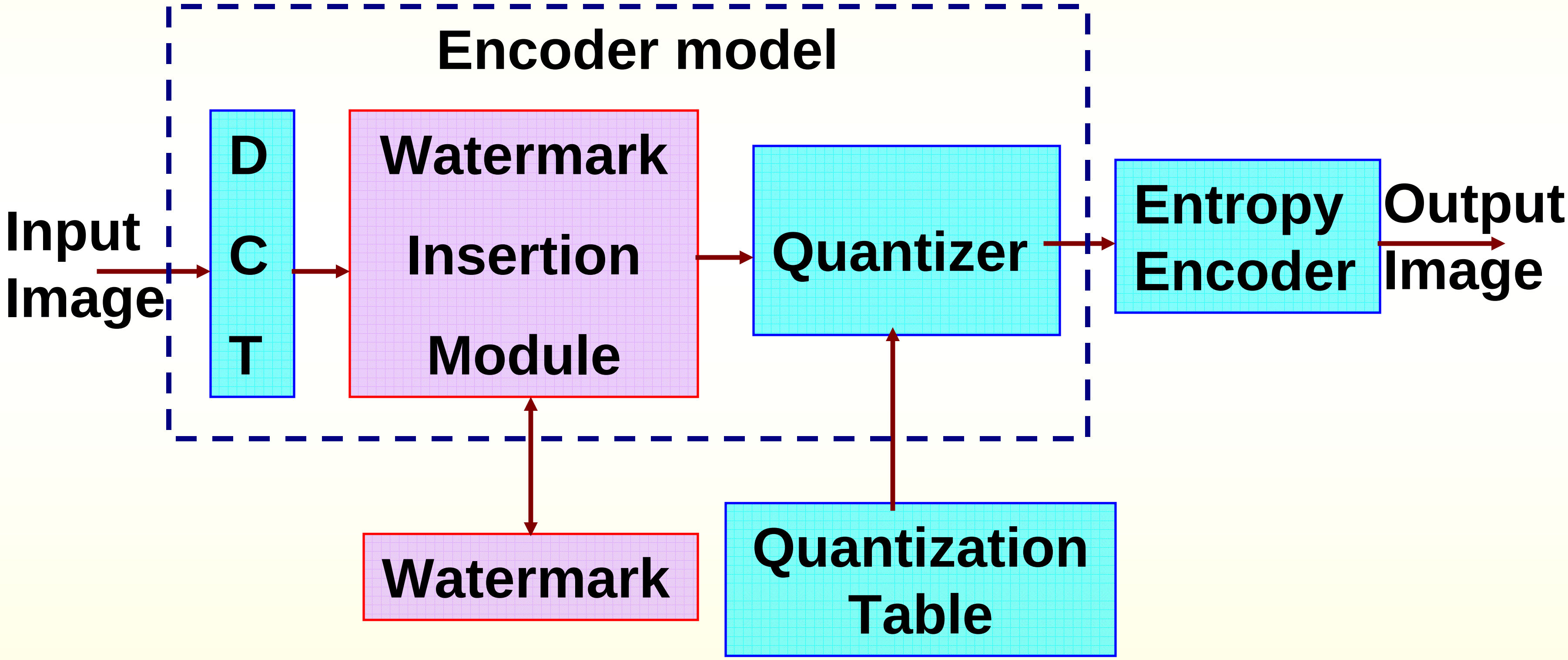
Previous Work: Summary

- Many software implementations of watermarking algorithms.
- Only few hardware implementations.
- Just one hardware implementation in frequency domain which can insert only invisible watermark.
- All other implementations in spatial domain.

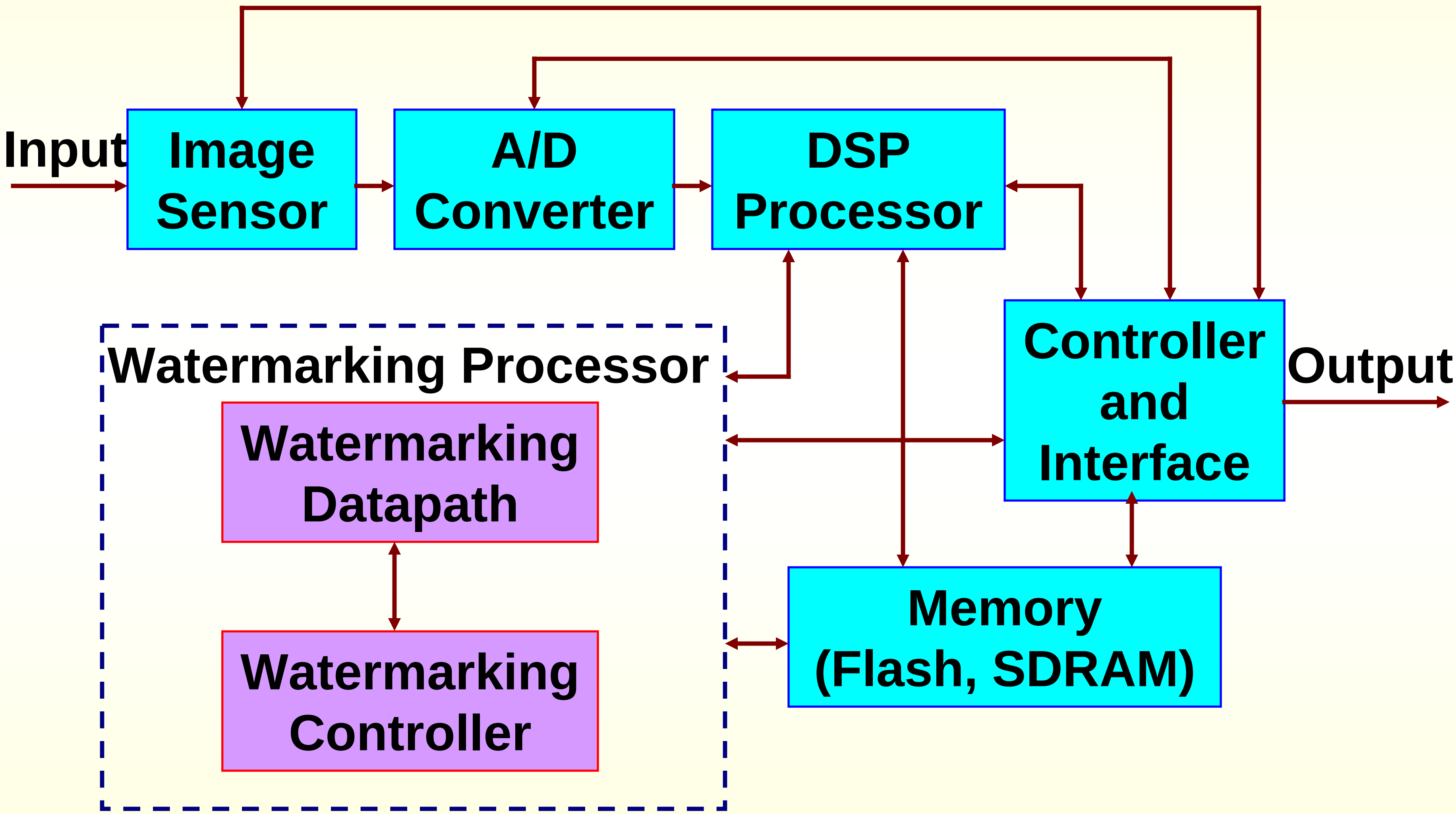
Highlights of our Designed Chip

- DCT domain Implementation
- First to insert both visible and / or invisible watermark
- First Low Power Design for watermarking using dual voltage and dual frequency
- Uses Pipelined / Parallelization for better performance

Watermarking through JPEG Encoder



Watermarking Through Digital Still Camera



Invisible Algorithm Implemented

1. Divide the original image into blocks.
2. Calculate the DCT coefficients of all the image blocks.
3. Generate random numbers to use as watermark.
4. Consider the three largest AC-DCT coefficients of an image block for watermark insertion.

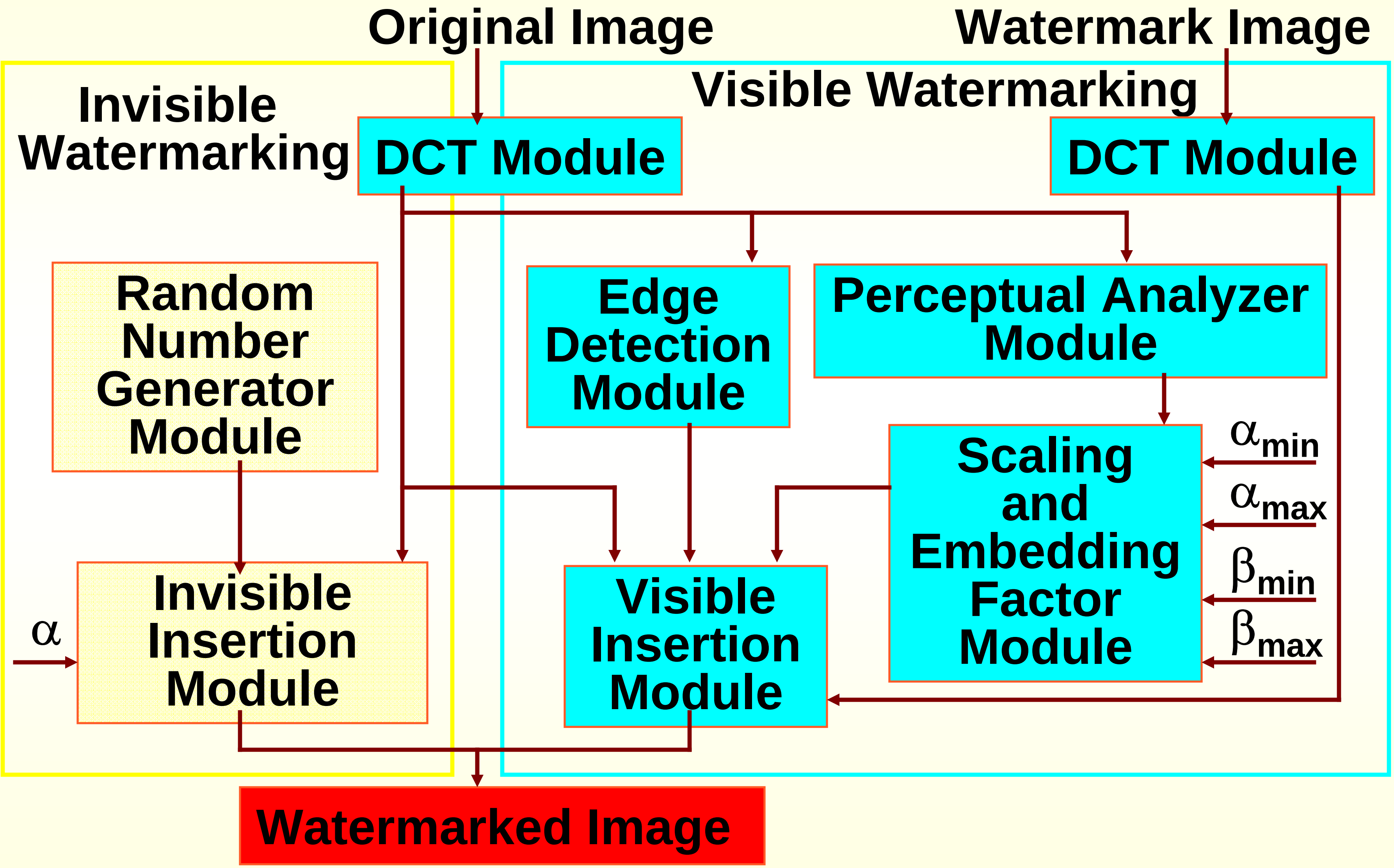
Reference: I.J. Cox, et. al., "Secure Spread Spectrum Watermarking for Multimedia", IEEE transactions on Image Processing, 1997.

Visible Algorithm Implemented

1. Divide Original and watermark image into blocks.
2. Calculate DCT coefficients of all the blocks.
3. Find the edge blocks in the original image.
4. Find the local and global statistics of original image using DC-DCT and AC-DCT coefficients.
5. The mean of DC-DCT coefficients and mean and the variance of AC-DCT coefficients are useful.
6. Calculate the Scaling and embedding factors.
7. Add the original image DCT coefficients and the watermark DCT coefficients block by block.

Reference: S. P. Mohanty, and et. al., "A DCT Domain Visible Watermarking Technique for Images", *Proc. of the IEEE ICME* 2000.

The Proposed Architecture



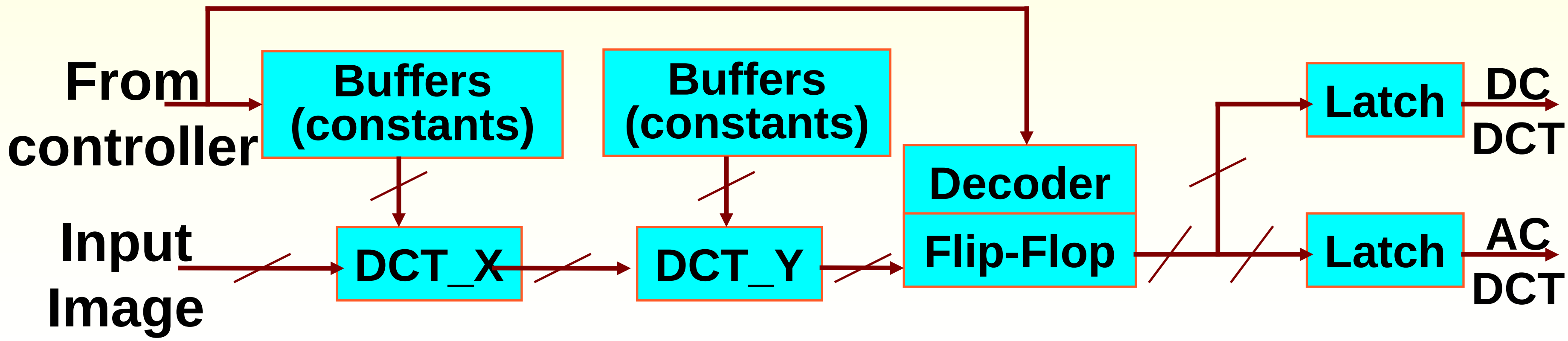
Highlights of the Proposed Architecture

- Hierarchical architecture.
- Decentralized controller scheme.
- Parallelism and Pipelining exploited.
- Dual Voltage and dual frequency mode operation

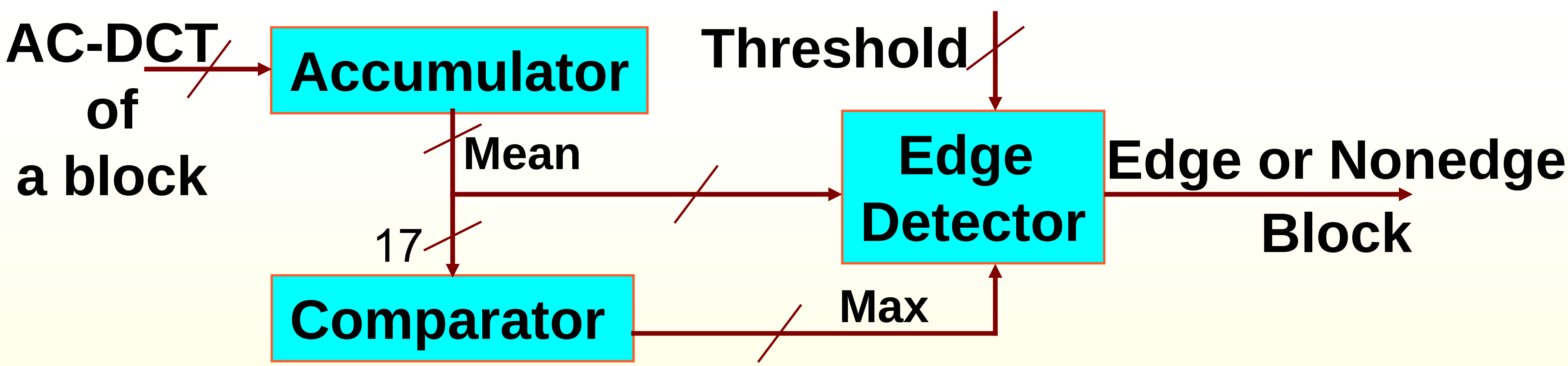
Modules in Proposed Architecture

- DCT Module: Calculates the DCT coefficients.
- Edge Detection Module: Determines edge blocks.
- Perceptual Analyzer Module: Determines perceptually significant regions using original image statistics.
- Scaling and Embedding Factor Module: Determines the scaling and embedding factors for visible watermark insertion.
- Watermark Insertion Module: Inserts the watermark
- Random Number Generator Module: Generates random numbers.

Modules in Proposed Architecture

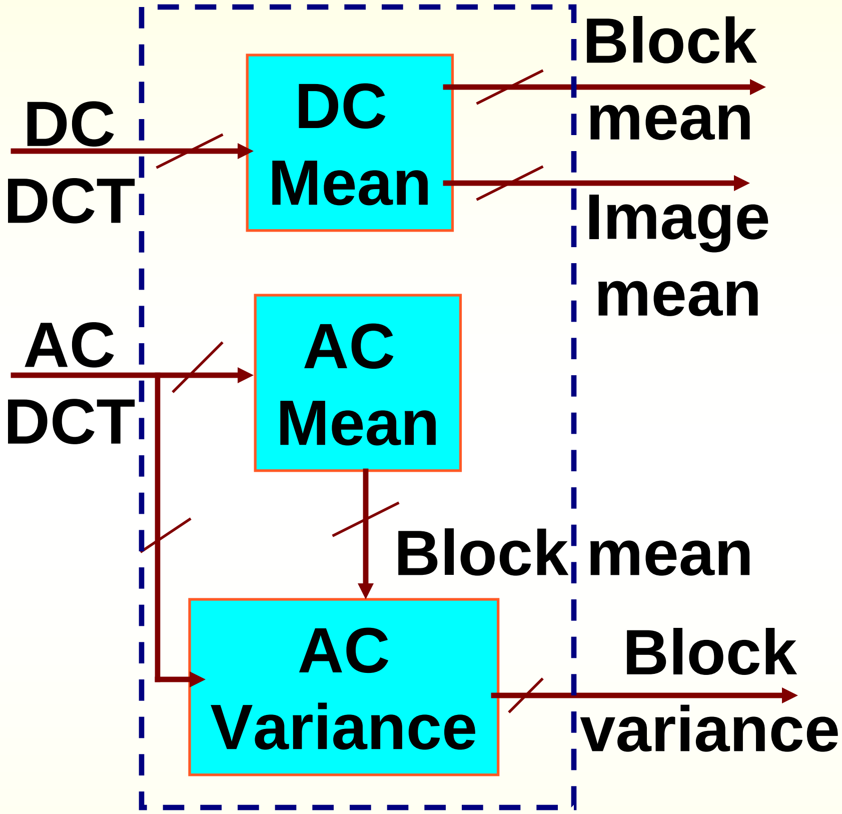


DCT Module

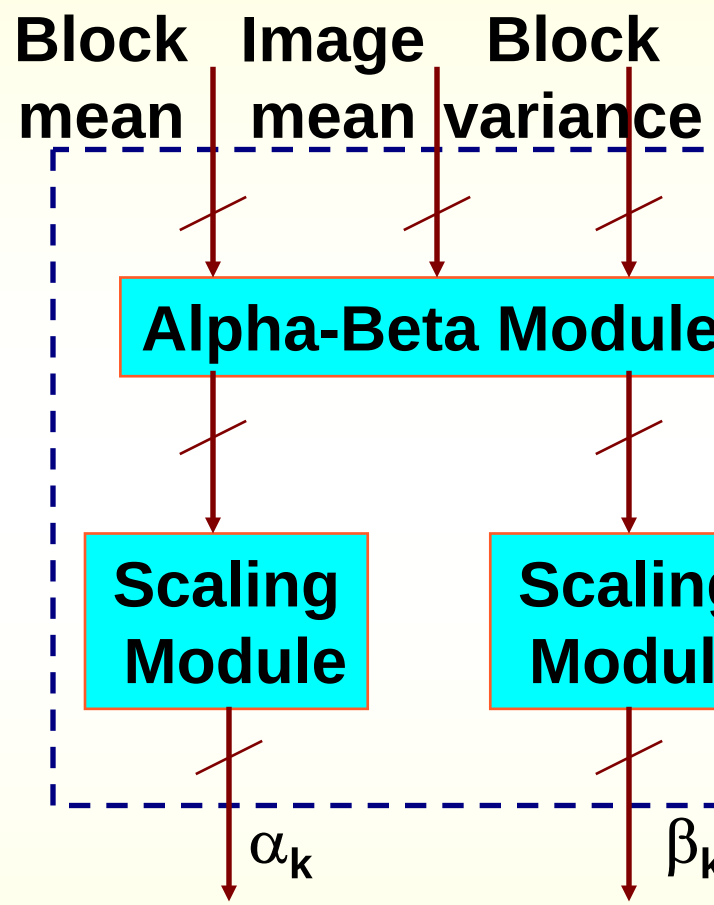


Edge Detection Module

Modules in Proposed Architecture



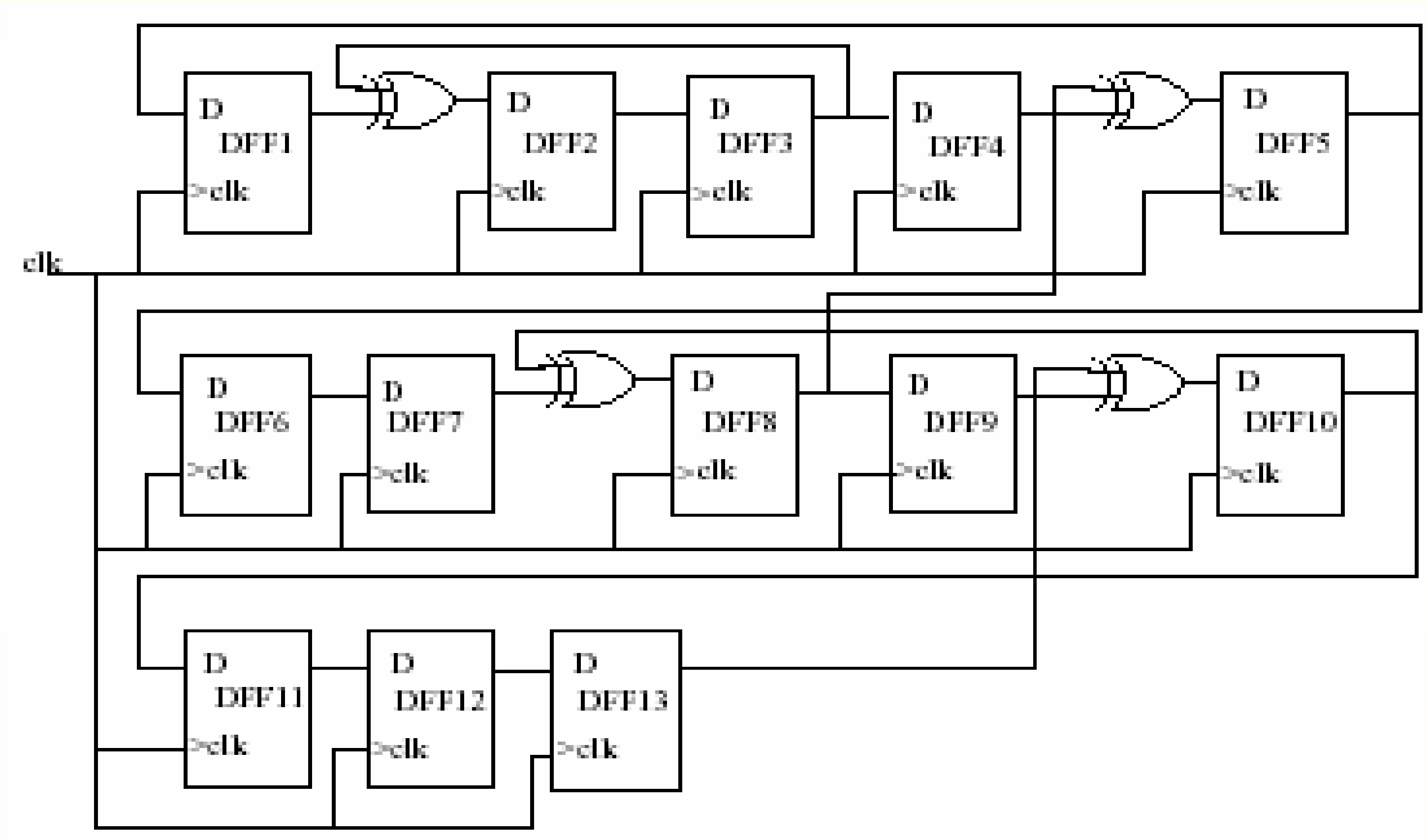
Perceptual Analyzer Module



Scaling and Embedding Factor Module

Modules in Proposed Architecture

Pseudorandom numbers generated using LFSR.



Modules in more Detail: DCT Module

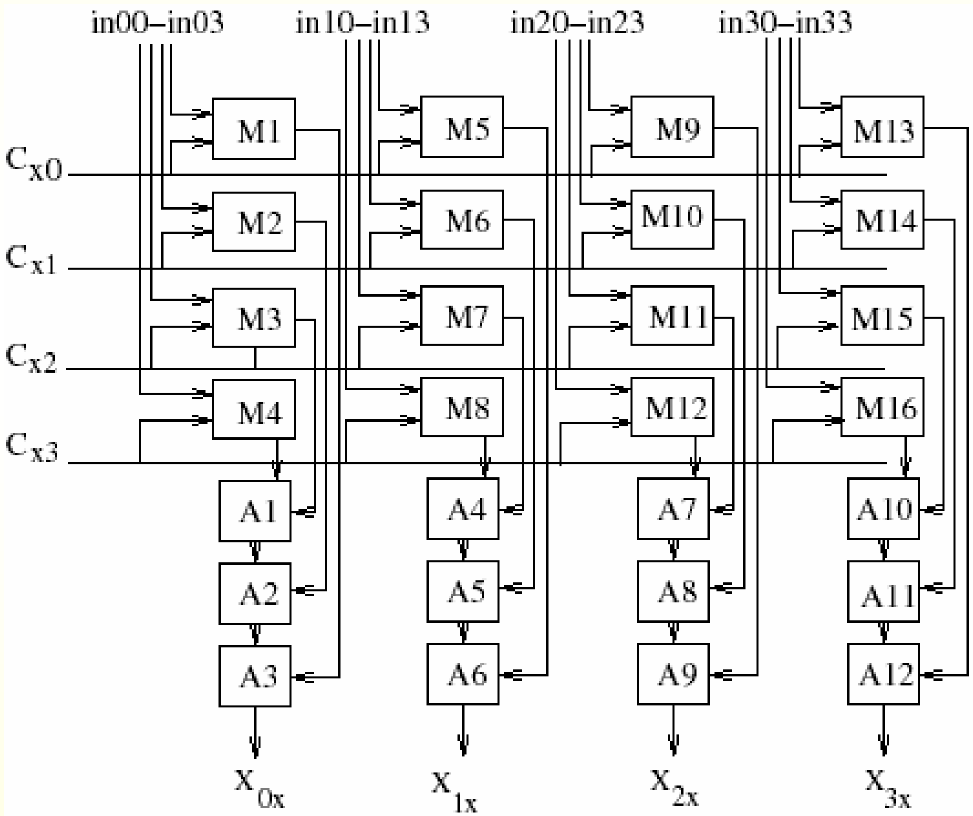
DCT module implements the following set of equations.

x00=((in00*c00) + (in01*c01) + (in02*c02) + (in03*c03))
x10=((in10*c00) + (in11*c01) + (in12*c02) + (in13*c03))
x20=((in20*c00) + (in21*c01) + (in22*c02) + (in23*c03))
x30=((in30*c00) + (in31*c01) + (in32*c02) + (in33*c03))

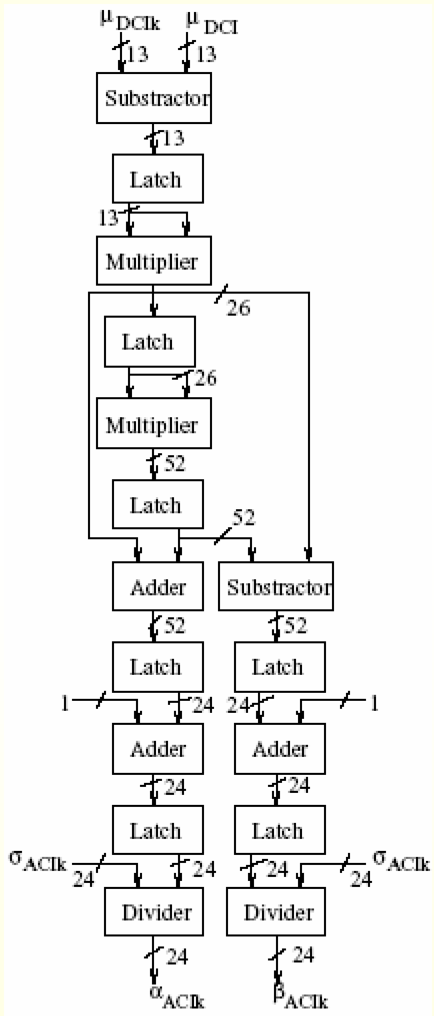
y00=((x00*c00) + (x10*c01) + (x20*c02) + (x30*c03))
y01=((x00*c10) + (x10*c11) + (x20*c12) + (x30*c13))
y02=((x00*c20) + (x10*c21) + (x20*c22) + (x30*c23))
y03=((x00*c30) + (x10*c31) + (x20*c32) + (x30*c33))

Modules in more Detail: DCT Module

DCT module implemented as arrays of multipliers and adders.

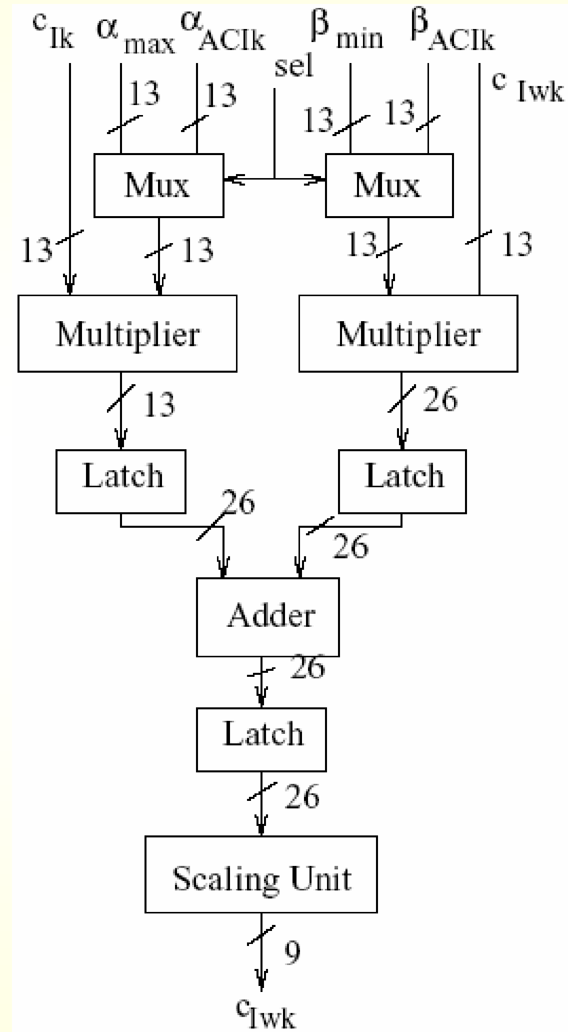


Modules in more Detail:
Scaling and Embedding Factor, Visible Insertion



Scaling and Embedding Factor Module ←

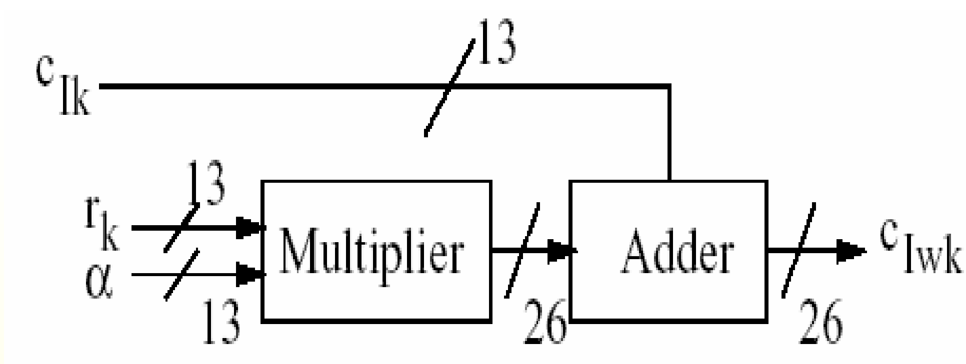
→ **Visible Insertion Module**



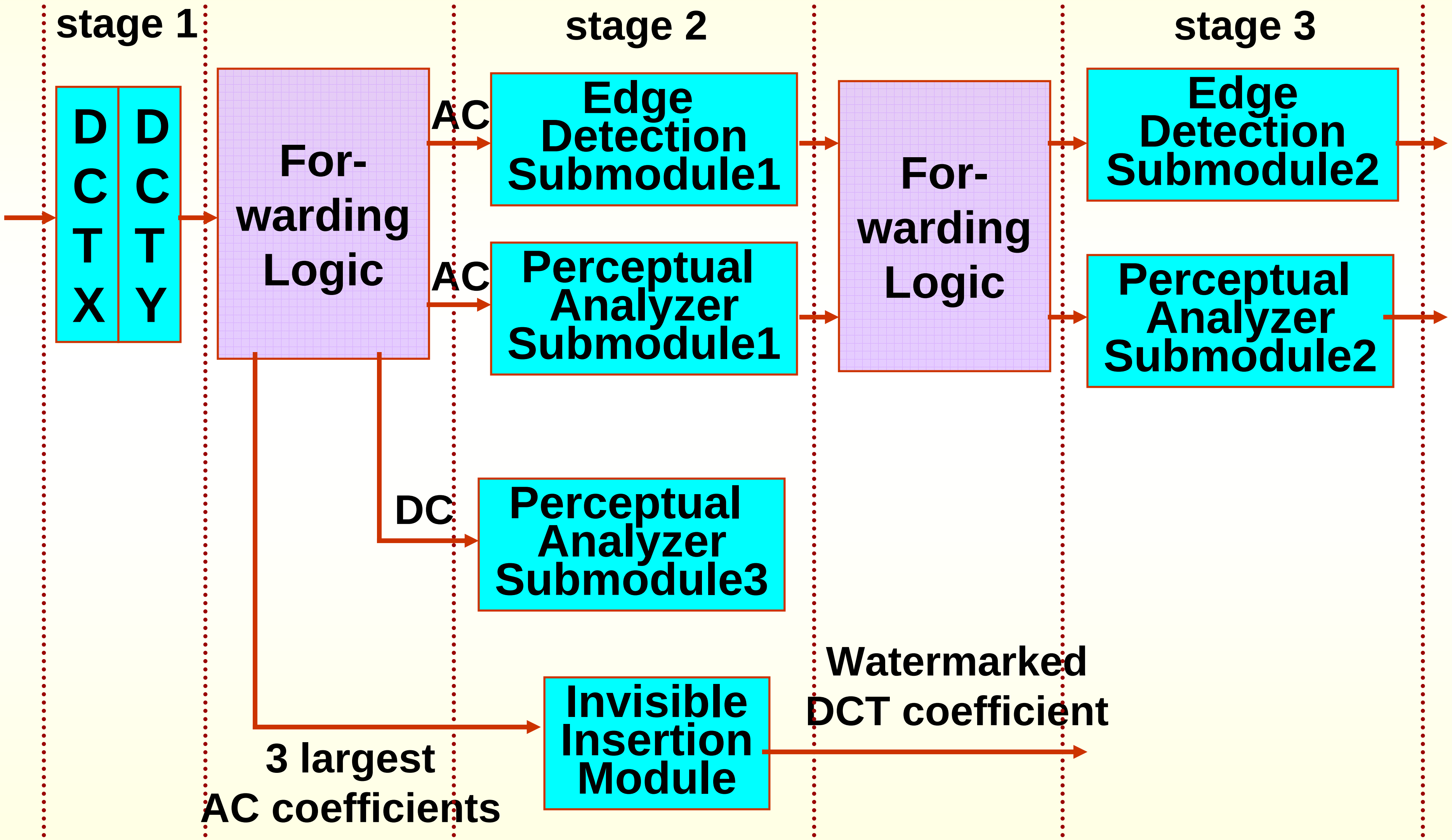
Modules in more Detail: Invisible Insertion

- Insertion module implemented with a multiplier and an adder.

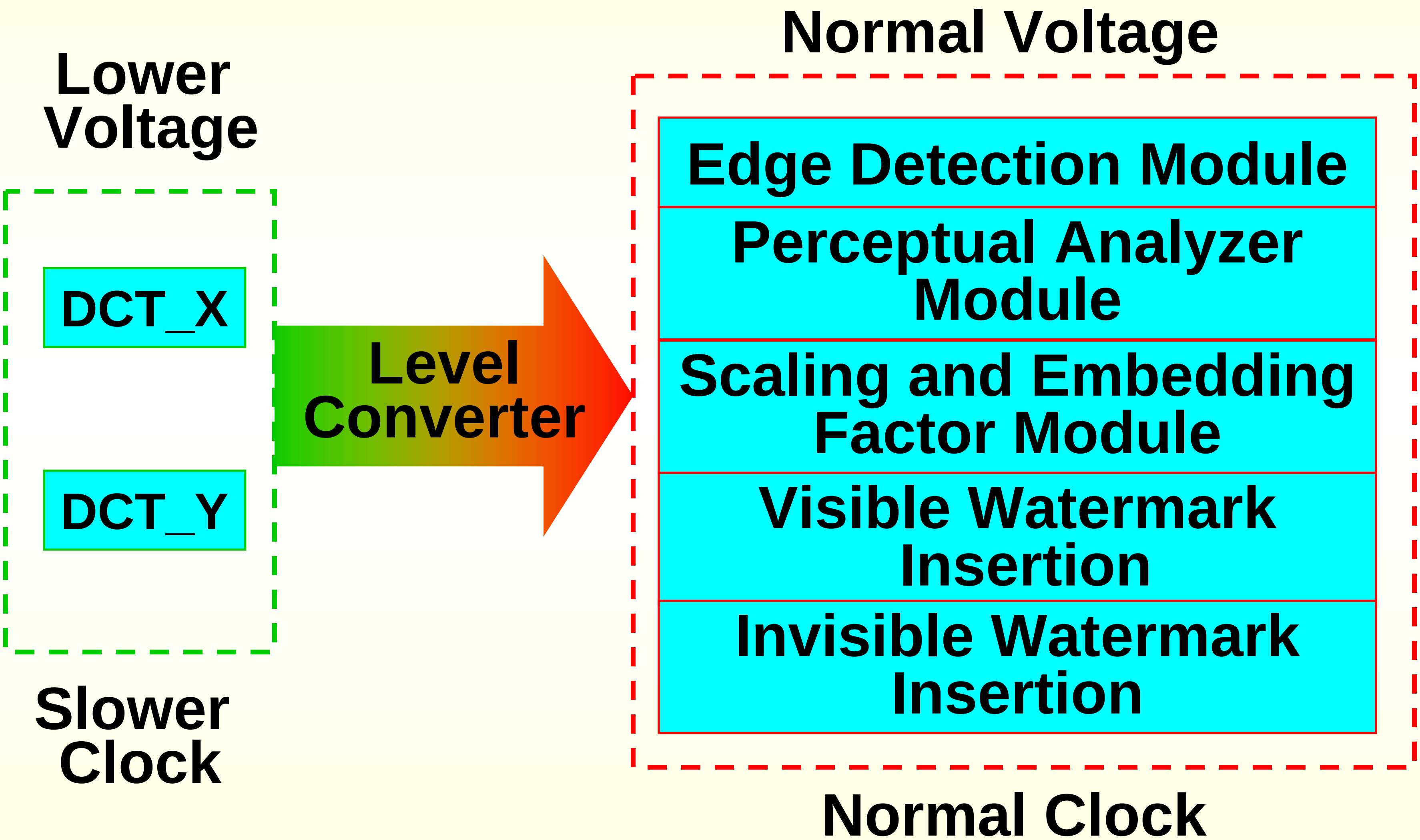
$$C_{IWk}=C_{Ik}+\alpha r_{IK}$$



Pipeline and Parallelism



Dual Voltage and Dual Frequency

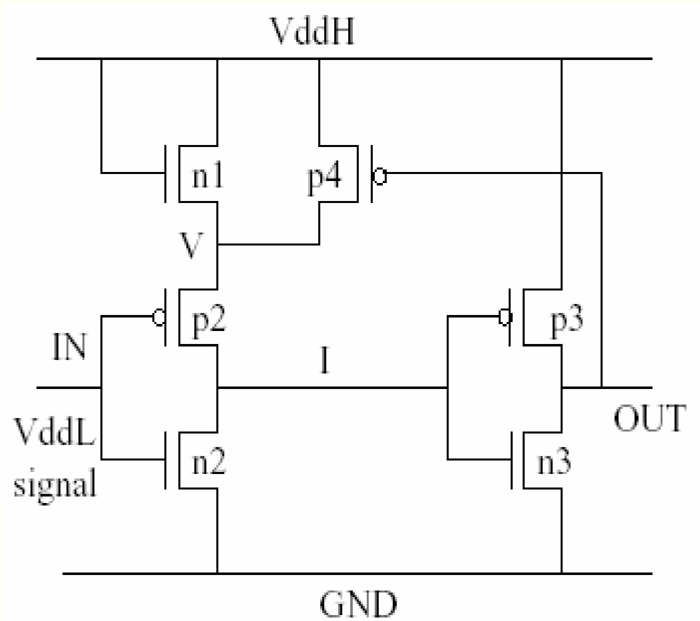


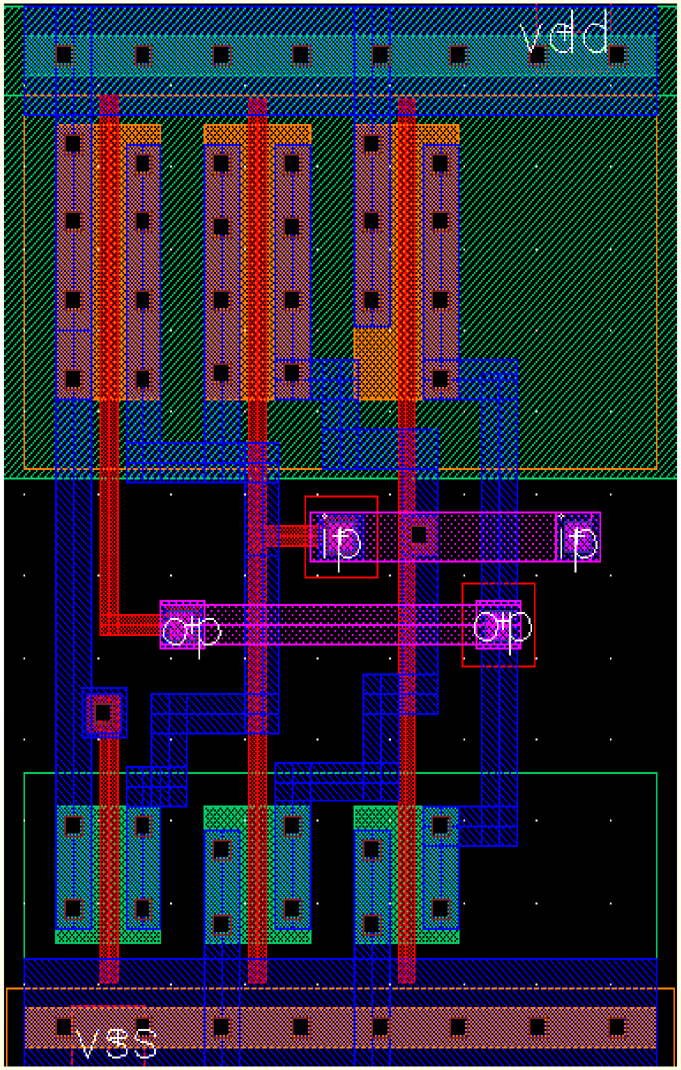
Dual Voltage: Level Converters

- Level converters required to step up the low voltage to high voltage.
- Traditional level converter: Differential Cascode Voltage Switch (DCVS).
- In this work: Single Supply Level Converters – faster, better power consumption, needs single voltage supply only.

Reference: R.Puri et. al., "Pushing ASIC performance in a power envelope" in the Proceedings of the Design Automation Conference, 2003, pp. 788-793

Layout and Schematic of SSLV



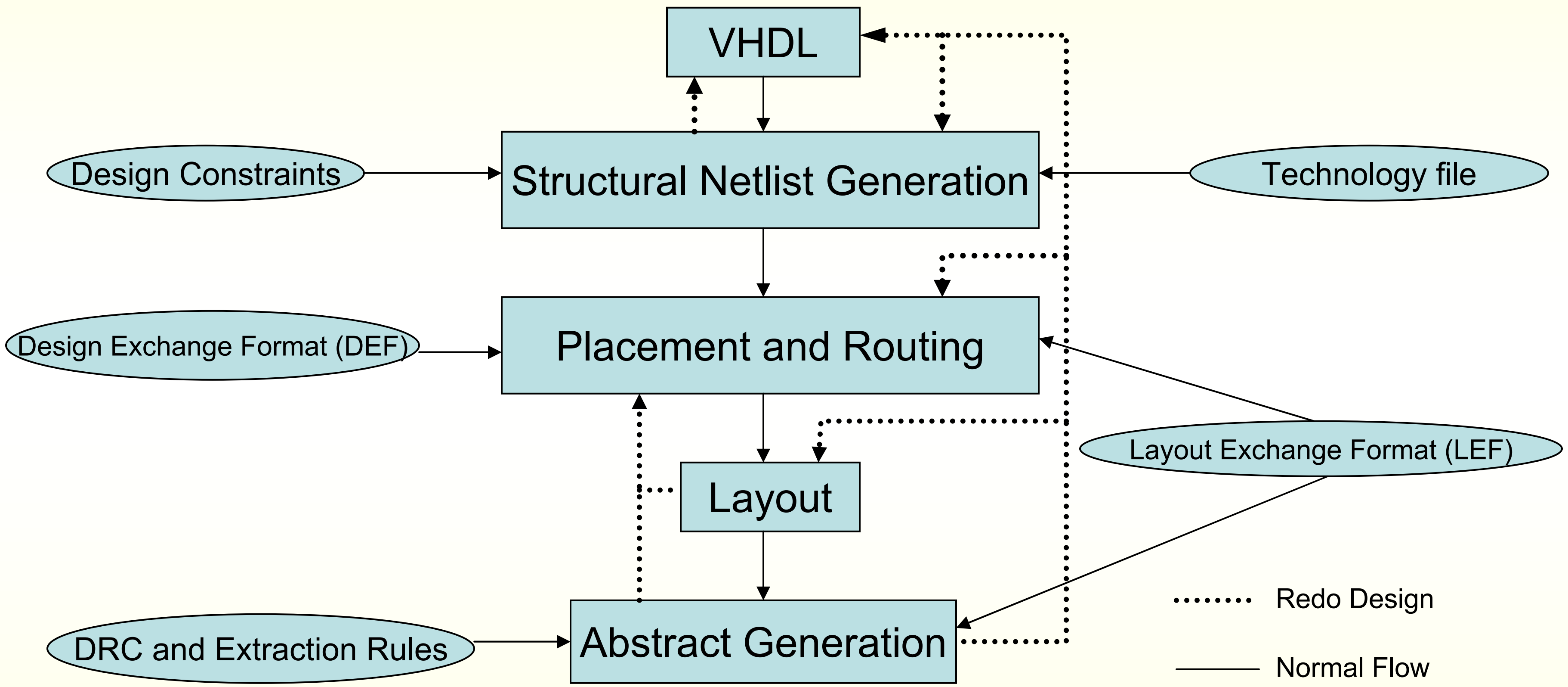


Prototype Chip Implementation: Tools Used

Tools	Purpose
Cadence NClaunch	VHDL simulator
Synopsys Design Analyzer	Verilog netlist generation
Cadence Silicon Ensemble	Layout, Placement and routing
Cadence Virtuose tool	Layout Editing
Cadence Abstract Generator	Abstract generation
Synopsys Nanosim	Power and delay calculations

Standard Cell Design Style adopted. Standard Cells obtained from Virginia Tech. Technology: TSMC 0.25 μm

Prototype Chip Implementation: Design Flow



Design Flow Example: VHDL Code

File Edit Window Tools Syntax Help

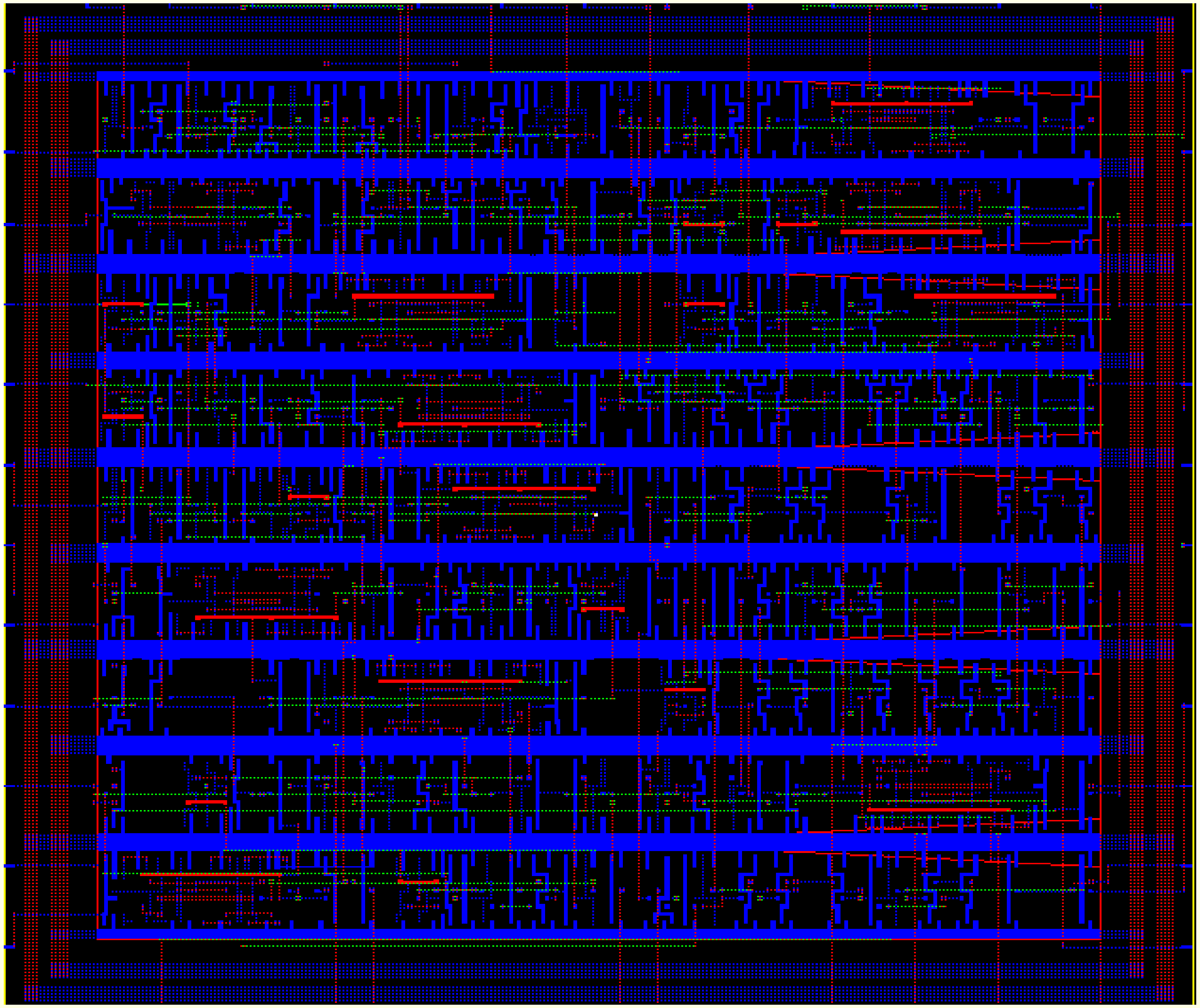
```
entity edm3 is
port (clk, reset, vdd1, enable, vss1 : in std_logic;
      AnMax, An : in std_logic_vector(16 downto 0);
      edge_block, done, write_edm3 : out std_logic;
      countout : out std_logic_vector(7 downto 0)
      );
end entity edm3;
-----------------------------------------------------------------------------
-

architecture behav of edm3 is

component counter8 is
port (clk : in std_logic;
      reset, vdd1, enable : in std_logic;
      q  : inout std_logic_vector(7 downto 0)
      );
end component counter8;

signal An_max, AnMax_by_2 : std_logic_vector(16 downto 0);
signal count_out : std_logic_vector(7 downto 0);
signal At, A, B, Bt, count, edgeblock, en_count, write, proces,
tempedgeblock : std_logic;

begin

COUNTER: counter8 port map (clk=>clk, reset=>reset, enable=>write, vdd1=>vdd1,
 q=>count_out);

countout<=count_out;

counting: process(count_out) is
          begin
          if (count_out="11111111") then
          count<='1';
          else
          count<='0';
          end if;
          end process;


-----------------------------------------------------------------------------
-
```

Design Flow Example: Synthesized Verilog Netlist

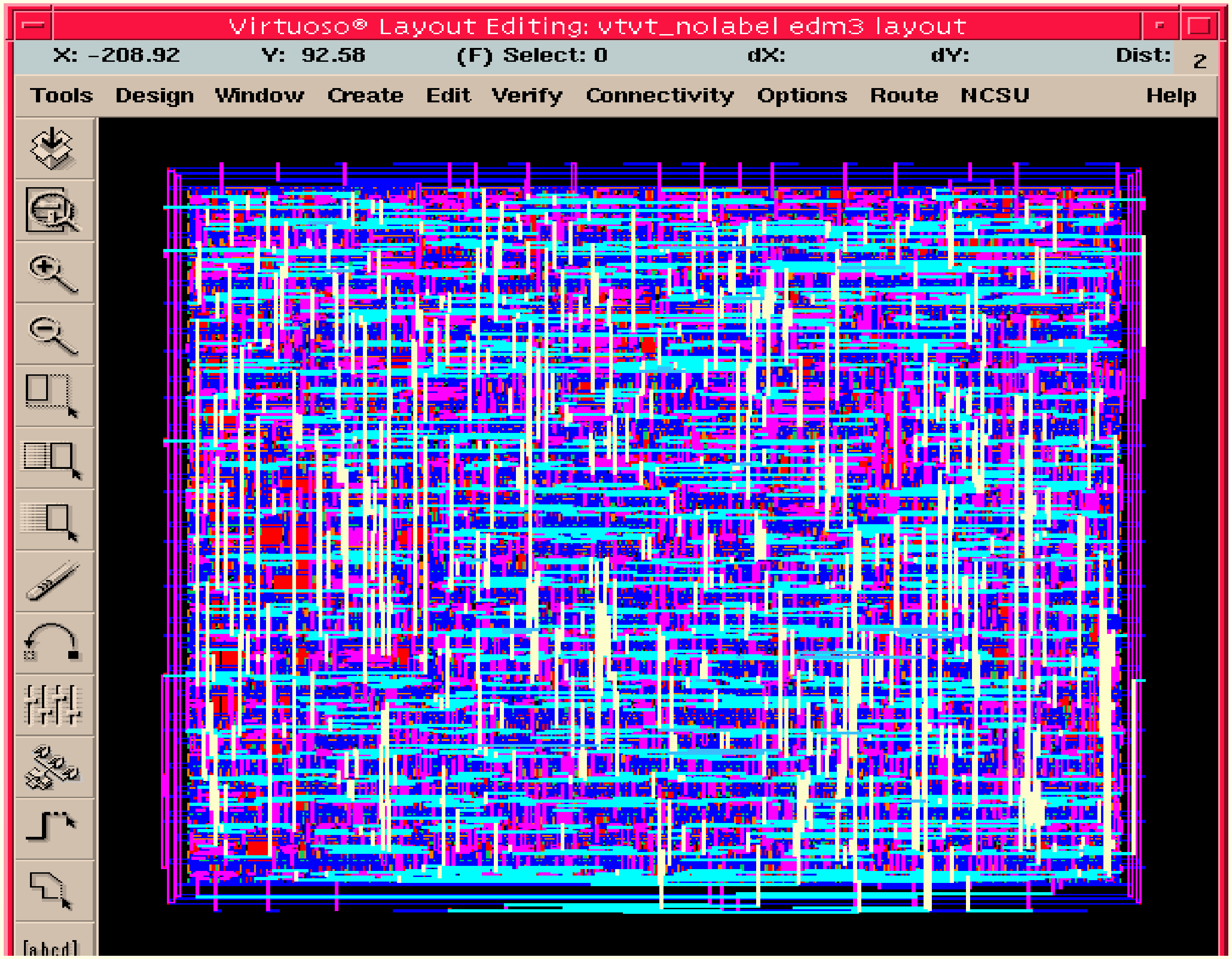
File Edit Window Tools Syntax Help

```
module edm3 ( clk, reset, vdd1, enable, vss1, AnMax, An, edge_block, done,
    write_edm3, countout );
input  [16:0] An;
input  [16:0] AnMax;
output [7:0] countout;
input  clk, reset, vdd1, enable, vss1;
output edge_block, done, write_edm3;
    wire Bt, n_133, At88, n_134, At, \"<"-return148 , count, Bt100, n195, n196,
        n197, n198, n199, n200, n201, n202, n203, n204, n205, n206,
        \*cell*78/U5/Z_0 ;
    counter8 COUNTER ( .clk(clk), .reset(reset), .vdd1(vdd1), .enable(
        write_edm3), .q(countout) );
    and3_1 U39 ( .ip1(n195), .ip2(n_133), .ip3(n196), .op(\*cell*78/U5/Z_0 )
         );
    or2_1 U40 ( .ip1(n197), .ip2(n198), .op(At88) );
    and2_1 U41 ( .ip1(\"<"-return148 ), .ip2(n_133), .op(n_134) );
    inv_1 U42 ( .ip(reset), .op(n_133) );
    nand2_1 U43 ( .ip1(n199), .ip2(n200), .op(Bt100) );
    nor2_1 U44 ( .ip1(n201), .ip2(n199), .op(n197) );
    nor2_1 U45 ( .ip1(n198), .ip2(n201), .op(n202) );
    nor3_1 U46 ( .ip1(n203), .ip2(n204), .ip3(n205), .op(count) );
    nand2_1 U47 ( .ip1(At), .ip2(n_133), .op(n199) );
    mux2_2 U48 ( .ip1(n202), .ip2(n198), .s(n199), .op(write_edm3) );
    mux2_2 U49 ( .ip1(n201), .ip2(enable), .s(n199), .op(n195) );
    and2_1 U50 ( .ip1(countout[1]), .ip2(countout[3]), .op(n206) );
    nand3_1 U51 ( .ip1(countout[0]), .ip2(countout[2]), .ip3(n206), .op(n205)
         );
    nand2_1 U52 ( .ip1(countout[5]), .ip2(countout[4]), .op(n203) );
    nand2_1 U53 ( .ip1(countout[7]), .ip2(countout[6]), .op(n204) );
    inv_1 U54 ( .ip(count), .op(n201) );
    nand2_1 U55 ( .ip1(Bt), .ip2(n_133), .op(n196) );
    inv_1 U56 ( .ip(n196), .op(n198) );
    nand2_1 U57 ( .ip1(enable), .ip2(n196), .op(n200) );
    drp_2 At_reg ( .ck(clk), .ip(At88), .rb(n_133), .q(At) );
    lp_2 edgeblock_reg ( .ck(\*cell*78/U5/Z_0 ), .ip(n_134), .q(edge_block) );
    dp_2 done_reg ( .ck(clk), .ip(count), .q(done) );
    drp_2 Bt_reg ( .ck(clk), .ip(Bt100), .rb(n_133), .q(Bt) );
    edm3_DW01_cmp2_17_0 \lt_100/lt/lt  ( .A(An), .B({vss1, AnMax[16],
        AnMax[15], AnMax[14], AnMax[13], AnMax[12], AnMax[11], AnMax[10],
        AnMax[9], AnMax[8], AnMax[7], AnMax[6], AnMax[5], AnMax[4], AnMax[3],
        AnMax[2], AnMax[1]}), .LEQ(1'b0), .TC(1'b0), .LT_LE(\"<"-return148 )
         );
endmodule
```

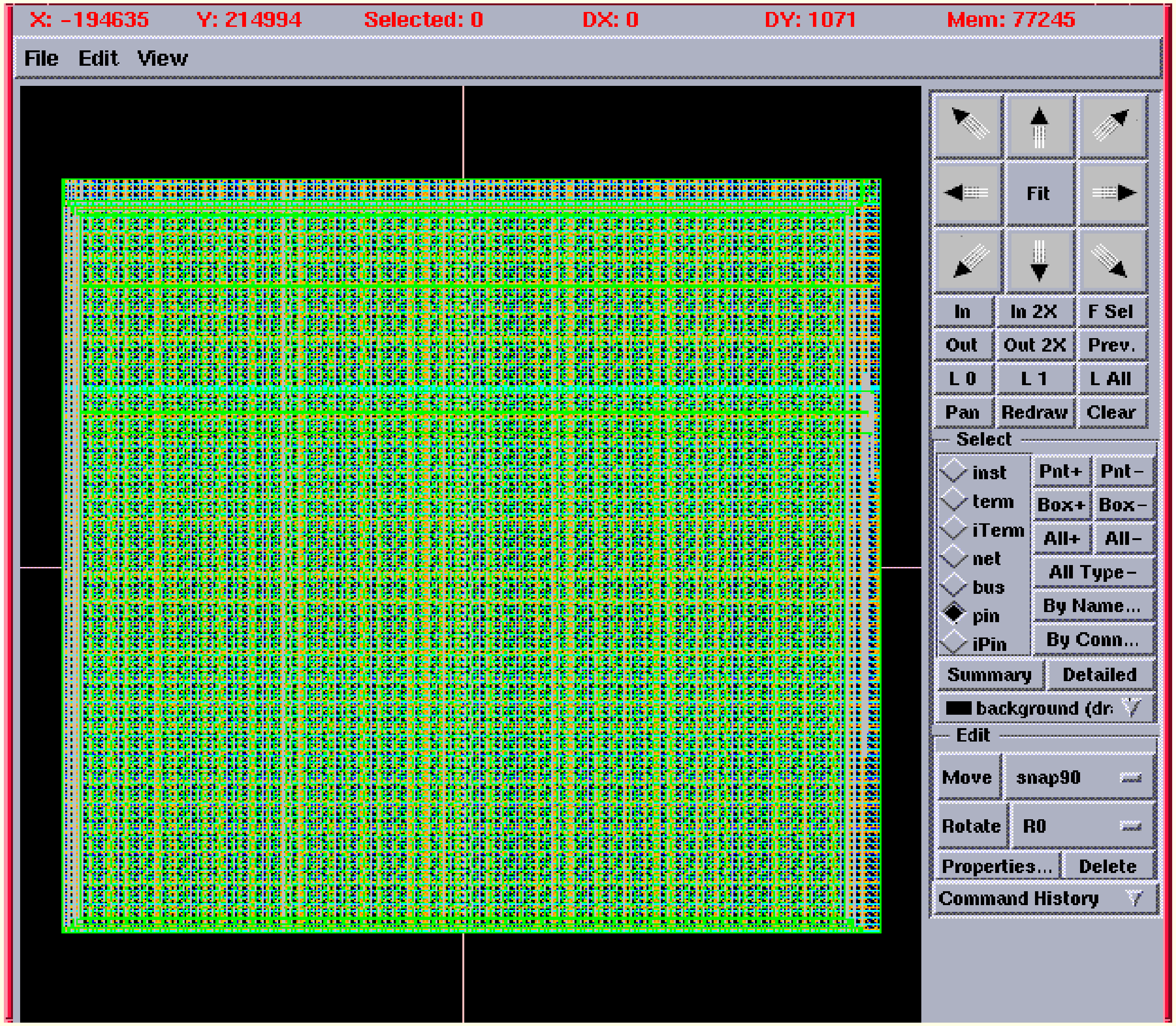
Design Flow Example: Placement and Routing



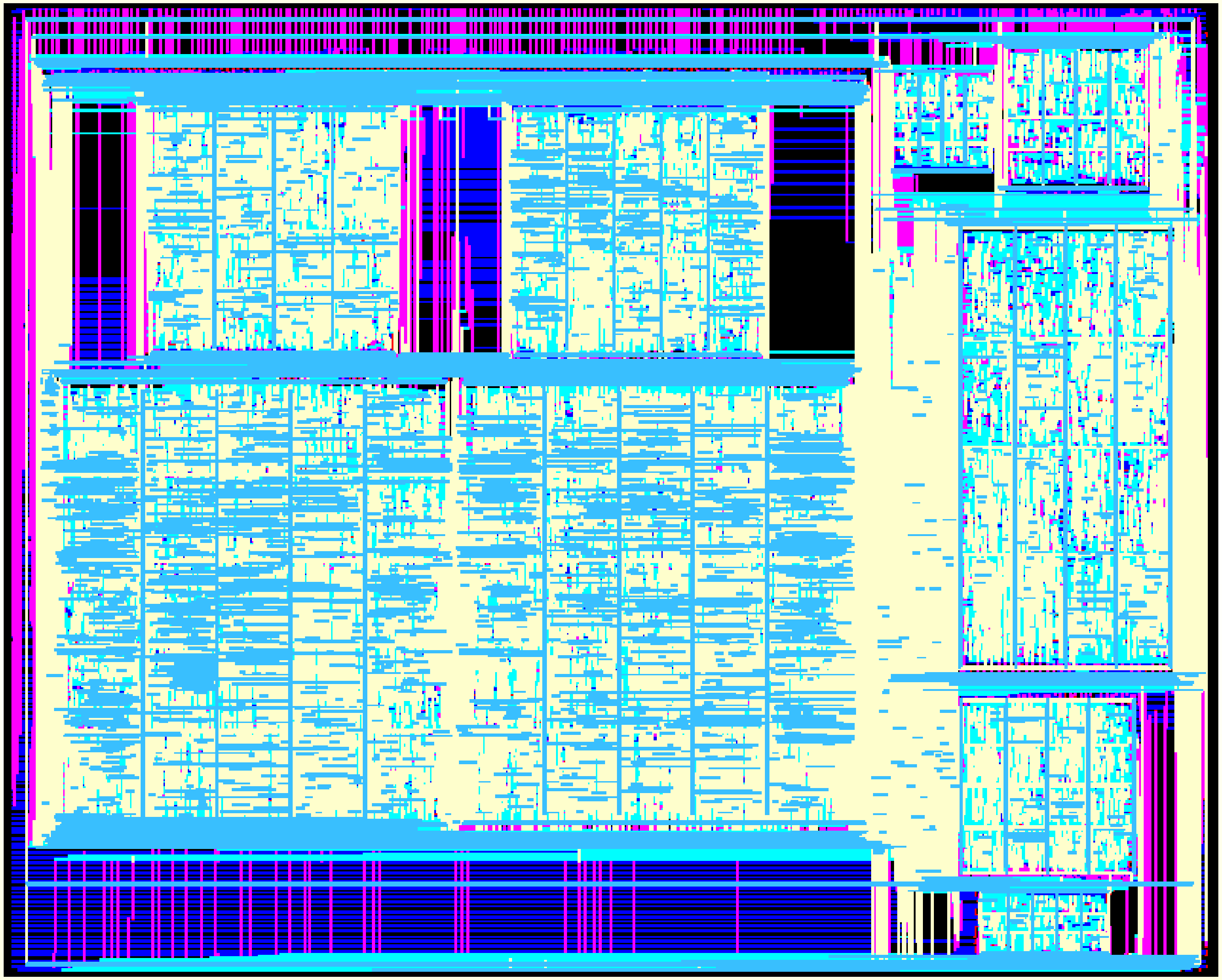
Design Flow Example: Layout



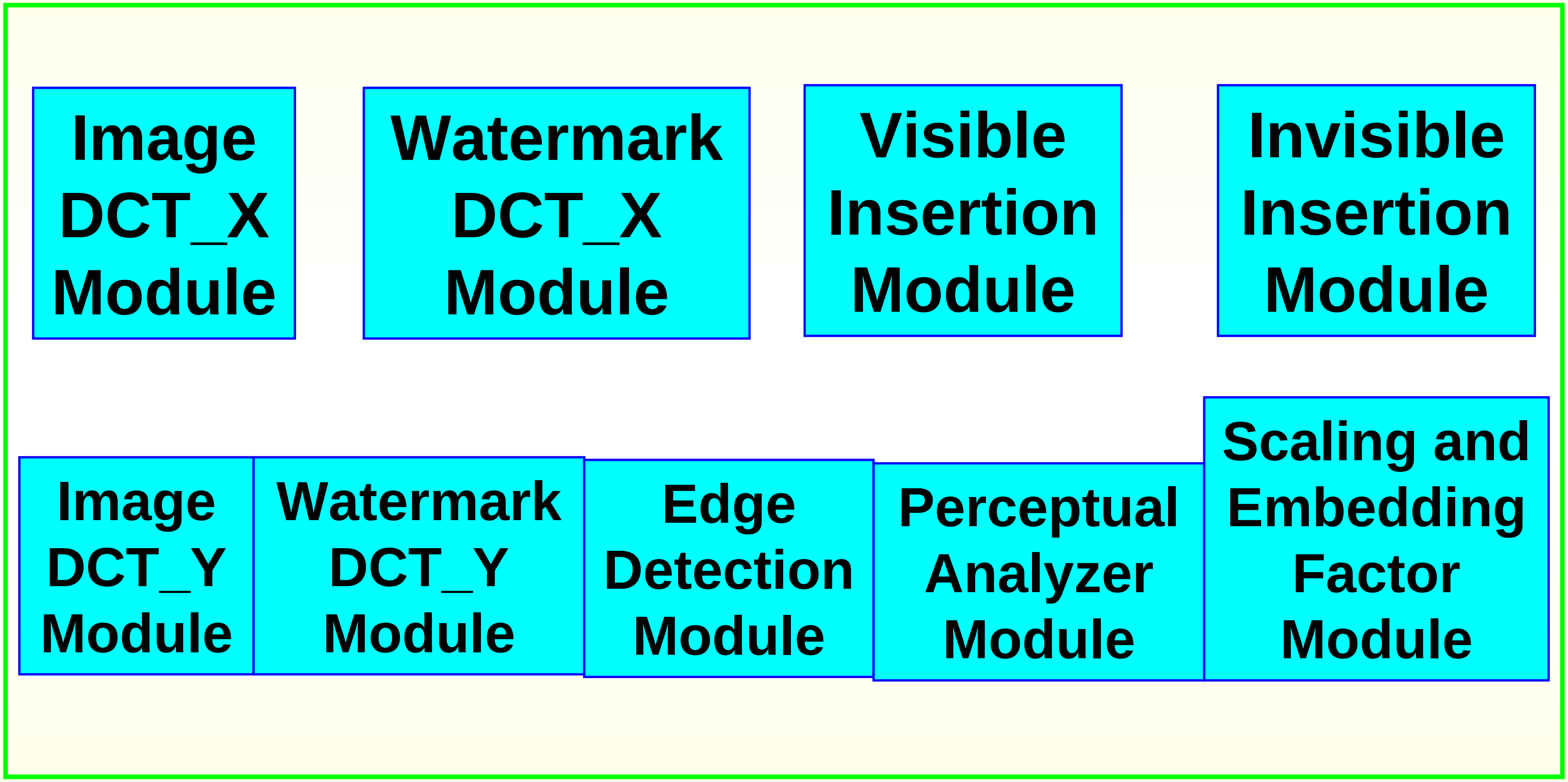
Design Flow Example: Abstract Generation



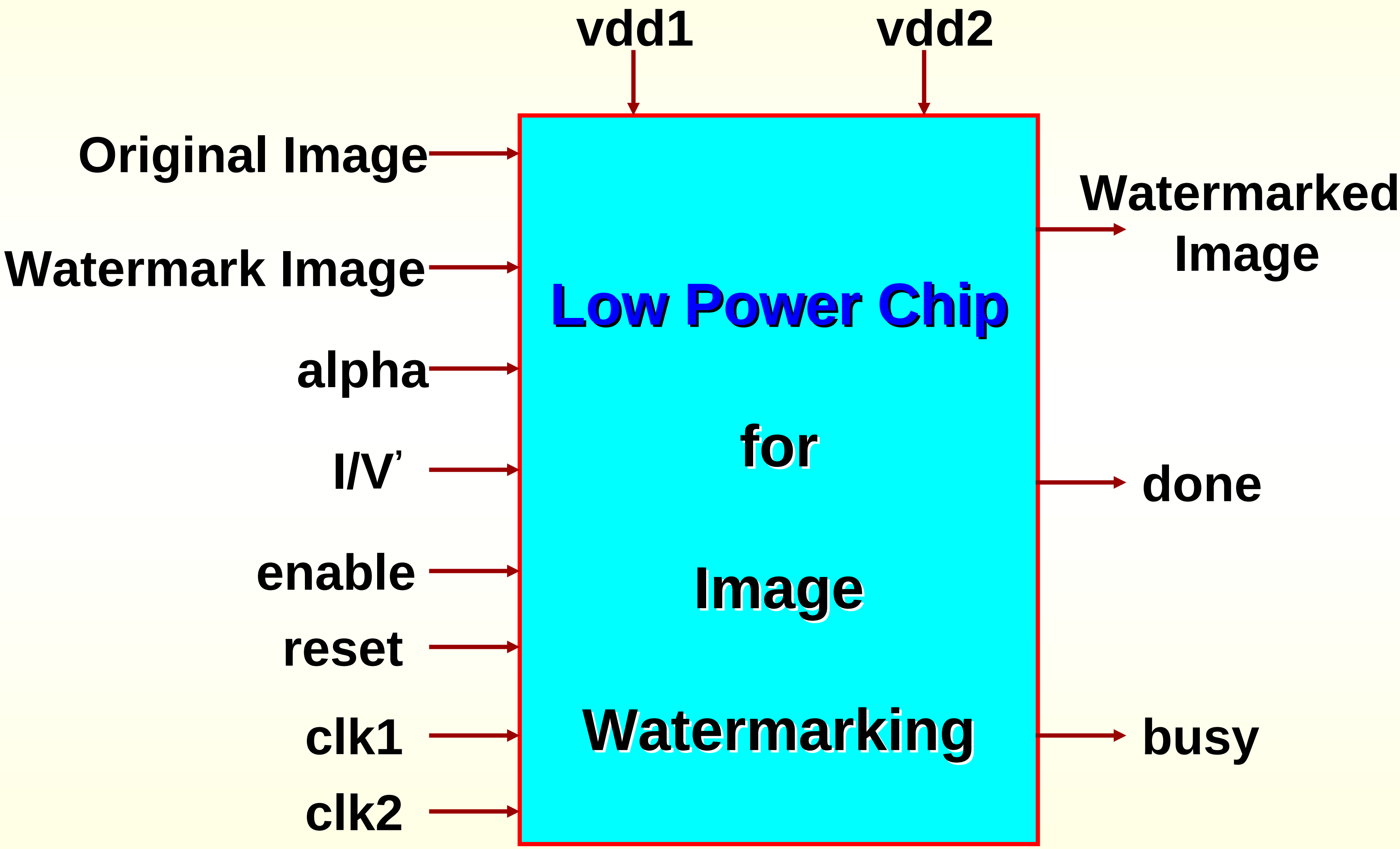
Overall Prototype Chip: Layout



Prototype Chip: Floor plan



Prototype Chip: Pin diagram



Prototype Chip: Statistics

Technology: TSMC 0.25 μ

Total Area : 16.2 sq mm

Dual Clocks: 280 MHz and 70 MHz

Dual Voltages: 2.5V and 1.5V

No. of Transistors: 1.4 million

Power (dual voltage and frequency): 0.3 mW

Chip (single voltage and frequency): 1.9 mW

Conclusion and Future Work

- Dual Voltage, Dual frequency watermarking chip was developed.
- Invisible / Visible insertion
- Pipelined and Parallelized architecture for performance.
- Frequency domain implementation for real time audio and video watermarking.
- Real time watermark extraction.
- Need more robust watermarking algorithms.