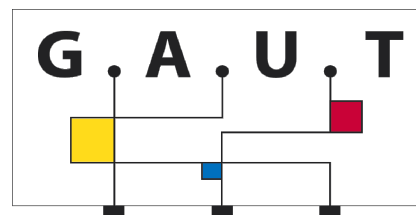


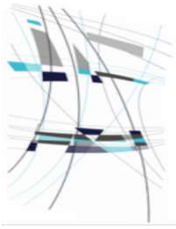
GAUT

A Free and Open-Source High-Level Synthesis tool

Philippe COUSSY

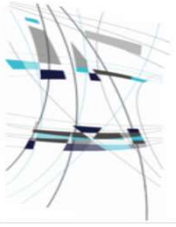
philippe.coussy@univ-ubs.fr





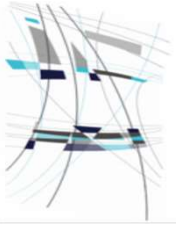
Outline

- Context
- High-Level Synthesis Overview
- **The tool GAUT**
- Conclusion and perspectives



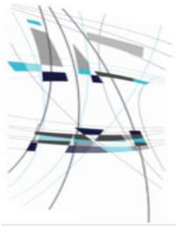
GAUT (1/2)

- An academic, free and open source HLS tool
 - ◇ CECILL B License (GPL-like) / Source code on Github
 - ◇ Eclipse-based + JAVA
- 3rd version (1st dev. started in 90s!)
- Dedicated to any type of applications
 - ◇ data- or control-dominated algorithm
- Input : C/C++ algorithm
- Outputs : RTL Architecture (VHDL) & TLM/CA models (SystemC)
- Automated Test-bench generation
- Automated characterization of operator libraries



GAUT (2/2)

- Synthesis Options
 - ◇ Allocation
 - ◇ Manual, automated (uniform, mean..)
 - ◇ Scheduling
 - ◇ List-based, ASAP, ALAP
 - ◇ Binding
 - ◇ Left-edge, MWBM (Hungarian method)
 - ◇ High-level transformations
 - ◇ Loop unrolling, function in-lining...
 - ◇ Interface synthesis
 - ◇ Memory, handshake...
- Testbench generation
 - ◇ Stimuli: Incremental, from files...



Guided User Interface

The screenshot shows an IDE window titled "DCT64x64.c" with a C program. A guided user interface (GUI) is overlaid on the code, consisting of a vertical bar on the left side of the code editor. This bar contains several colored segments (blue, green, yellow, red) that correspond to different parts of the code, likely providing context-sensitive help or documentation. The code defines a function `dct` that processes an input array `in` into an output array `out` using a discrete cosine transform (DCT) algorithm. The code includes constants for cosine values and a loop for processing the input data.

```
short * dct(const char *in, int nbloc) {
    unsigned short i;
    int x0, x1, x2, x3, x4, x5, x6, x7, x8;
    short *out;
    dct_bloc: for (; nbloc; nbloc--) {

        /* All values are shifted left by 10
        * and rounded off to nearest integer
        */

        static const unsigned short c1 = 1420; /* cos PI/16 * root(2) */
        static const unsigned short c2 = 1338; /* cos PI/8 * root(2) */
        static const unsigned short c3 = 1204; /* cos 3PI/16 * root(2) */
        static const unsigned short c5 = 805; /* cos 5PI/16 * root(2) */
        static const unsigned short c6 = 554; /* cos 3PI/8 * root(2) */
        static const unsigned short c7 = 283; /* cos 7PI/16 * root(2) */

        static const unsigned short s1 = 3;
        static const unsigned short s2 = 10;
        static const unsigned short s3 = 13;

        inner_loop1: for (i = 8; i > 0; i--) {
            x8 = (int) in[0] + (int) in[7]; /* levelshift: - 128 - 128 */
            x7 = (int) in[1] + (int) in[6]; /* levelshift: - 128 - 128 */
            x6 = (int) in[2] + (int) in[5]; /* levelshift: - 128 - 128 */
            x5 = (int) in[3] + (int) in[4]; /* levelshift: - 128 - 128 */

            x0 = (int) in[0] - (int) in[7]; /* levelshift: - 128 + 128 */
            x1 = (int) in[1] - (int) in[6]; /* levelshift: - 128 + 128 */
            x2 = (int) in[2] - (int) in[5]; /* levelshift: - 128 + 128 */
            x3 = (int) in[3] - (int) in[4]; /* levelshift: - 128 + 128 */

            x8 -= 256; /* levelshift: - 128 - 128 */
            x7 -= 256; /* levelshift: - 128 - 128 */
            x6 -= 256; /* levelshift: - 128 - 128 */
            x5 -= 256; /* levelshift: - 128 - 128 */

            x4 = x8 + x5;
            x8 -= x5;

            x5 = x7 + x6;
            x7 -= x6;
        }
    }
}
```



HLS steps and options

The screenshot shows the Xilinx IDE interface. The top part is a tree view of the project settings for 'project.gsettings'. The 'On Chip Monitor' item is selected and highlighted in blue. Below the tree view is a toolbar with options: Selection, Parent, List, Tree, Table, Tree with Columns. Below the toolbar is a tabbed interface with 'Properties' selected. The Properties window shows a table of properties for the selected item.

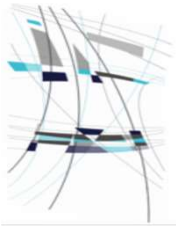
Property	Value
Id	
Monitoring Policy	SPEED
Monitoring Type	ControlAndAssertionCkecking



CDFG

The image displays the GVEdit software interface, which is used for generating and visualizing Control Data Flow Graphs (CDFGs) from source code. The interface is split into several panes:

- Project Explorer:** Shows the project structure for 'DCT64x64', including folders for 'Includes', 'src', 'Debug', and 'src', and various source files like 'project.g', 'xc5v1k11', and 'factorial'.
- CFG View:** Displays a Control Flow Graph with nodes labeled 'ENTRY S0', 'ENTRY S1', 'MAIN S63', 'S64', 'S28', 'S20', 'S30', 'S2', 'S65', and 'EXIT S66'. Each node contains code snippets such as 'g_stat=0', 'abloc_v793 = PHI abloc_v792, abloc_m_v582 = PHI m_v785, m_out_v583 = PHI out_v789, var_5160, v5161 var_5160, v5161 = LSHIFT out_1', and 'D 2419, v356 = LOAD m_v357'.
- CDFG View:** Shows a detailed Control Data Flow Graph with nodes representing operations like 'rshft_0_0', 'add_0_7', 'add_0_6', 'add_0_5', and 'lshft_0_0'. It includes a data flow table with columns for 'Property' and 'Value'.
- Code Editor:** Displays assembly-like code with comments and values, such as 'x8;', 'by 10', 'integer', and a list of values: '= 1420', '= 1338', '= 1204', '= 805;', '= 554;', '= 283;', '= 3;', '= 10;', '= 13;'. It also shows array access like 'i--', '[7];', '[6];', '[5];', '[4];' and memory addresses like '128 -', '128 -', '128 -', '128 -'.
- Properties Window:** A table with 'Property' and 'Value' columns, currently empty.
- Progress Bar:** Shows the progress of the compilation or analysis process.
- Debug Console:** Displays the message 'ation Debug for project DCT64x64 ****'.



Testbench and simulation

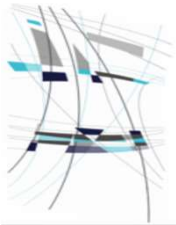
The image displays a software development environment with three main windows:

- Project Explorer:** Shows a project named 'DCT64x64' with sub-directories for 'src' and 'Makefile_DCT64x64.td'. The 'src' directory contains VHDL files: 'dct_carac.vhdl', 'dct_dp.vhdl', 'dct_fsm.vhdl', 'dct_fun.vhdl', 'dct64x64_tb.vhdl', and 'dct64x64_tb.vhdl.bak'.
- Resource Set:** Shows synthesis options for the 'Function dct' component, including Global Synthesis Options, Synthesis Options, Interface Settings, Graph Optimization, Scheduling, Allocation, Binding, and Clock.
- Properties:** A table showing properties for the 'dct' component:

Property	Value
Generate Test Bench	true
Id	
Ignored	false
Interface	CTRL_HandSh
Name	dct

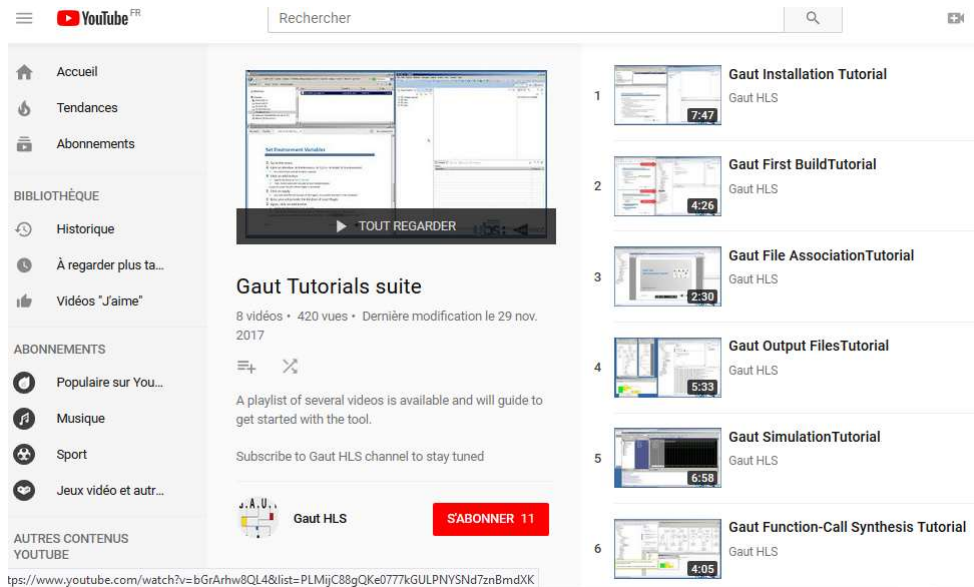
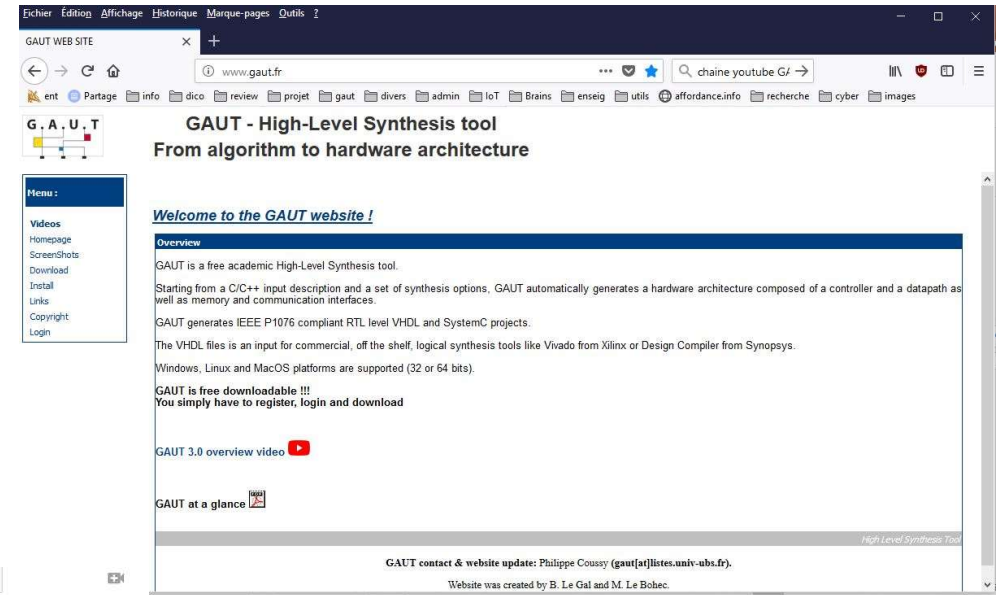
The **ModelSim PE 6.5c** window shows a simulation waveform for the testbench 'dct64x64_tb.vhdl'. The waveform displays various signals over time, with a time scale from 140,000 ns to 160,000 ns. A cursor is positioned at 157 ns. The signals include:

- Control signals: 'dct64x64_tb/dct_p_in', 'dct64x64_tb/dct_nbloc', 'dct64x64_tb/dct_p_out', 'dct64x64_tb/dct_g_return', 'dct64x64_tb/dct_p_in_mem_addr', 'dct64x64_tb/dct_p_in_mem_ce', 'dct64x64_tb/dct_p_in_mem_we', 'dct64x64_tb/dct_p_in_mem_data', 'dct64x64_tb/dct_out_v796_mem_addr', 'dct64x64_tb/dct_out_v796_mem_ce', 'dct64x64_tb/dct_out_v796_mem_we', 'dct64x64_tb/dct_out_v796_mem_data', 'dct64x64_tb/dct_mem_addr', 'dct64x64_tb/dct_mem_ce', 'dct64x64_tb/dct_mem_we', 'dct64x64_tb/dct_mem_in_data', 'dct64x64_tb/dct_mem_out_data', 'dct64x64_tb/dct_g_start', 'dct64x64_tb/dct_g_done', 'dct64x64_tb/dct_g_idle', 'dct64x64_tb/dct_g_enable', 'dct64x64_tb/dct_g_reset', 'dct64x64_tb/dct_clock'.
- Timing: The simulation is currently at 180 ns, with a delta of 9 ns. The time range shown is from 138,672 ns to 163,483 ns.

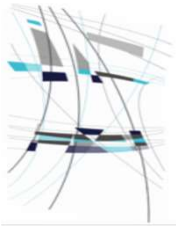


How to...

- Web site : www.gaut.fr
- Youtube chain
 - ◇ Video tutorials



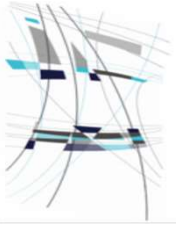
tps://www.youtube.com/watch?v=bGrArhw8QL4&list=PLMijC88gQKe0777kGULPNYSNd7zn8mdXX



GAUT Downloads

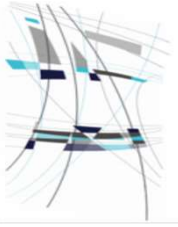


195 downloads from 43 countries during the last 18 months



Outline

- Context
- High-Level Synthesis Overview
- The tool GAUT
- **Conclusion**



Conclusion

- GAUT is an open source and free HLS tool
- Did not focus on synthesis performances
 - ◇ Rather on providing a “good-enough” open environment
- Objective
 - ◇ Introduce/promote HLS to newcomers
 - ◇ Provide an open environment to test new algorithms/ideas (Ph.D and MsC thesis...)
- Collaborations are welcome!

GAUT

A Free and Open-Source High-Level Synthesis tool

Philippe COUSSY

philippe.coussy@univ-ubs.fr

