SOBEL FILTER: Software Implementation to RTL using High Level Synthesis

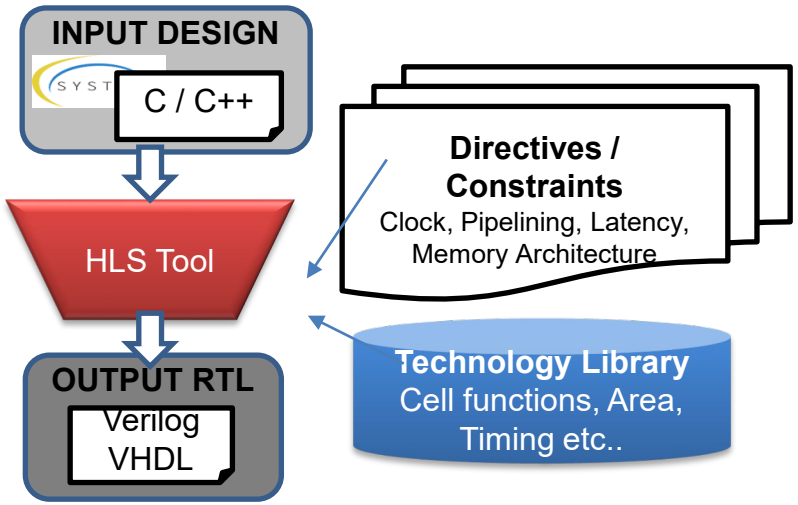
Bhavna Aggarwal, CircuitSutra Technologies, Noida, India
Umesh Sisodia, CircuitSutra Technologies, Noida, India
Snigdha Tyagi, CircuitSutra Technologies, Noida, India

accellera SYSTEMS INITIATIVE
CIRCUIT SUTRA TECHNOLOGIES
2020 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

High-Level Synthesis

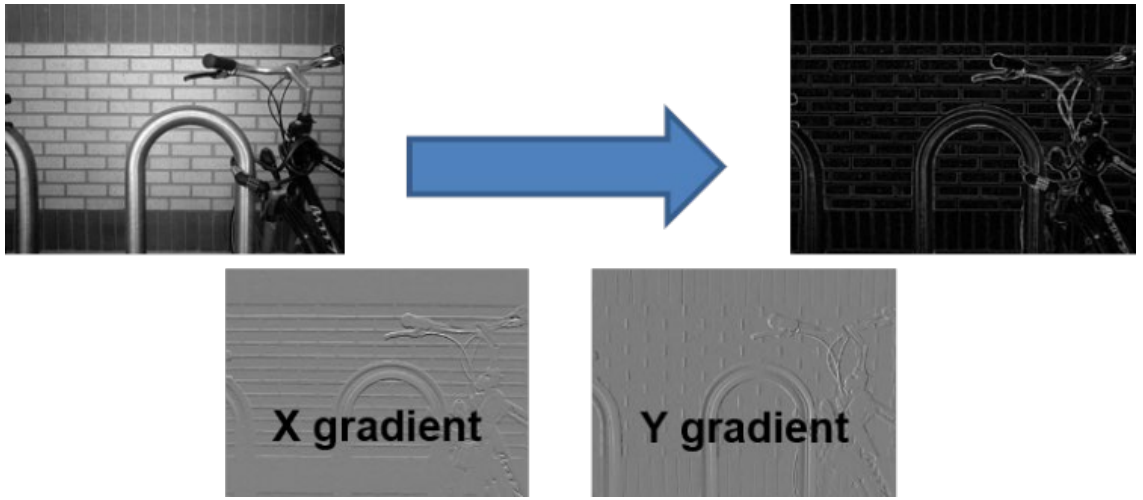
Chip designing at Higher Level of Abstraction

- HLS Tool: RTL Implementation from abstract description
  - 5 – 10x Less Code: Reduce design efforts
  - 10 – 1000x Faster Simulation: Increased productivity
- Bridges Hardware & Software Domain
- Existing C / C++ implementations available to start with
- C/C++ based synthesizable libraries
- SystemC Synthesizable Subset by Accellera
- Re-usability of C/C++ testbench for RTL verification



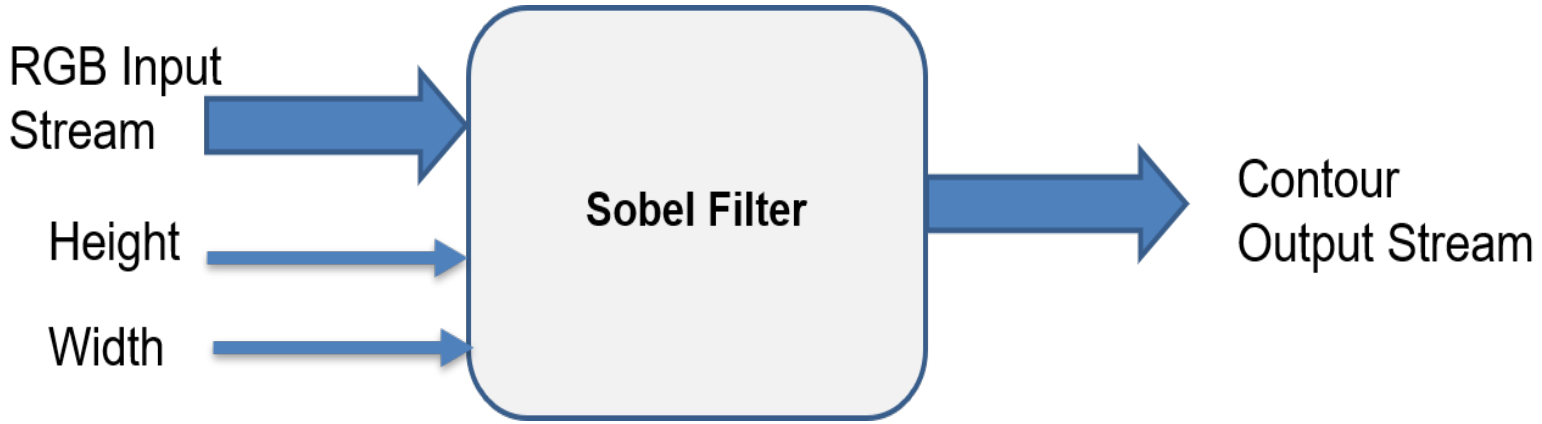
Sobel Filter – Taking through HLS

- Used in image processing and computer vision
- Used for edge detection
- 2D filtering operation
- Generates 2D Map of the gradient, X & Y Gradient of image intensity at each pixel
- Open Source Implementation
  - Contributors: Pedro Melgueira, Alessandro Capotondi



Sobel Filter
Architectural Interface : Open Source C

- **int sobelFilter (byte* rgb, byte* contour_img, int width, int height);**



HLS Implementation 1: Synthesizable C

We modified the C code to make it compliant with **Synthesizable Subset**.
Used **Mentor Catapult** to synthesize C code and generate Verilog

- Created Hierarchical Design with SobelFilter as “Top”
- Sub-function calls marked as BLOCK
- Contains few inline functions
- Used AC_Channels as inter-connect between two blocks

HLS Implementation 1: Synthesizable C

NOT SUPPORTED

- Pointers-to-pointer
- Dynamic Memory allocations
- Float/Double
- C/C++ Math library functions
  - abs
  - pow
  - sqrt
- Unbounded Loops (i.e. loops with termination condition dependent on some variable) are expected to give bad results.
- Multiple same function calls

CHANGED TO

- STATIC ARRAY or AC_CHANNEL
- Bounded Arrays
- AC_FIXED
- Include <ac_math.h> library
  - ac_abs
  - ac_pow_pwl
  - ac_sqrt_pwl
- Bounded Loops using constant integers
- Templated for ID to differentiate the call instance

HLS Implementation 2: Synthesizable SystemC

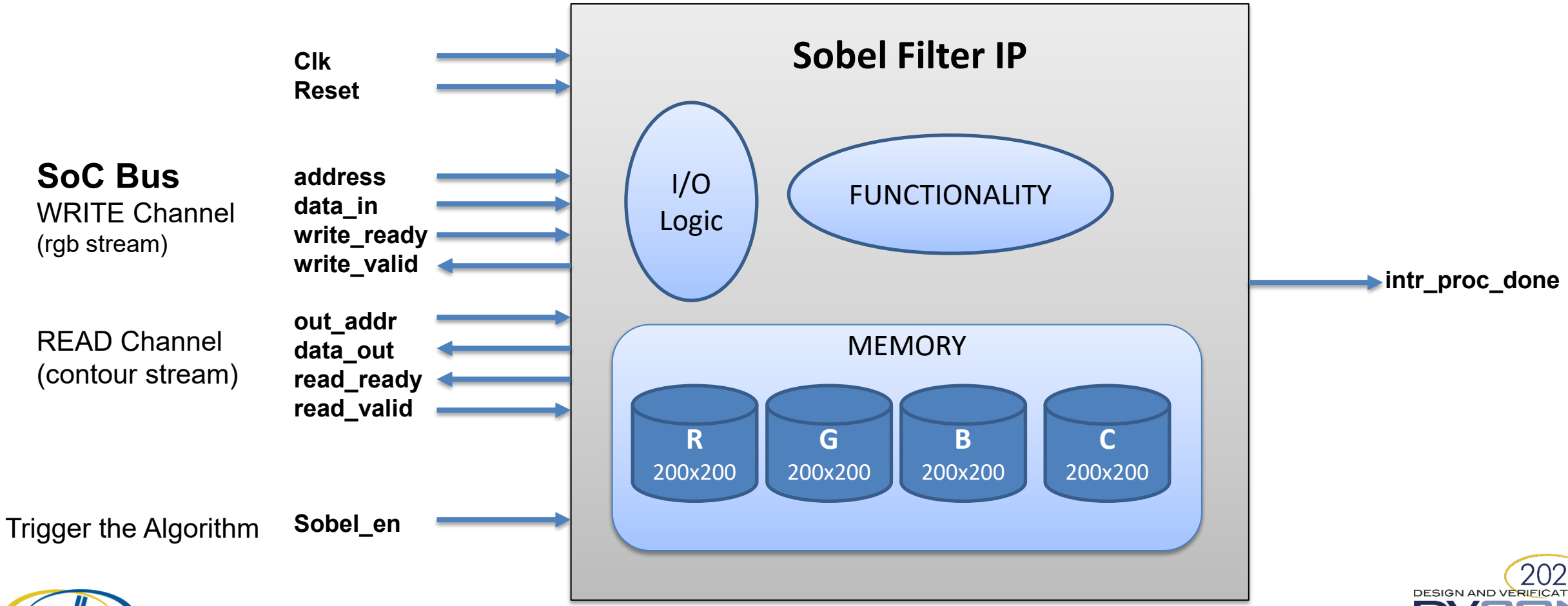
We developed a SystemC IP that would encapsulate the functionality.
Used **Mentor Catapult** to synthesize SystemC code and generate Verilog

- **Top Level Design:**
  - SystemC Module (DUT)
  - C++ class should be instantiated in top level C function.
  - Identified using CCS_DESIGN construct
- **Threads/Methods:**
  - One clock and reset
  - Output ports must be part of RESET block in the THREAD
  - Bounded Loops

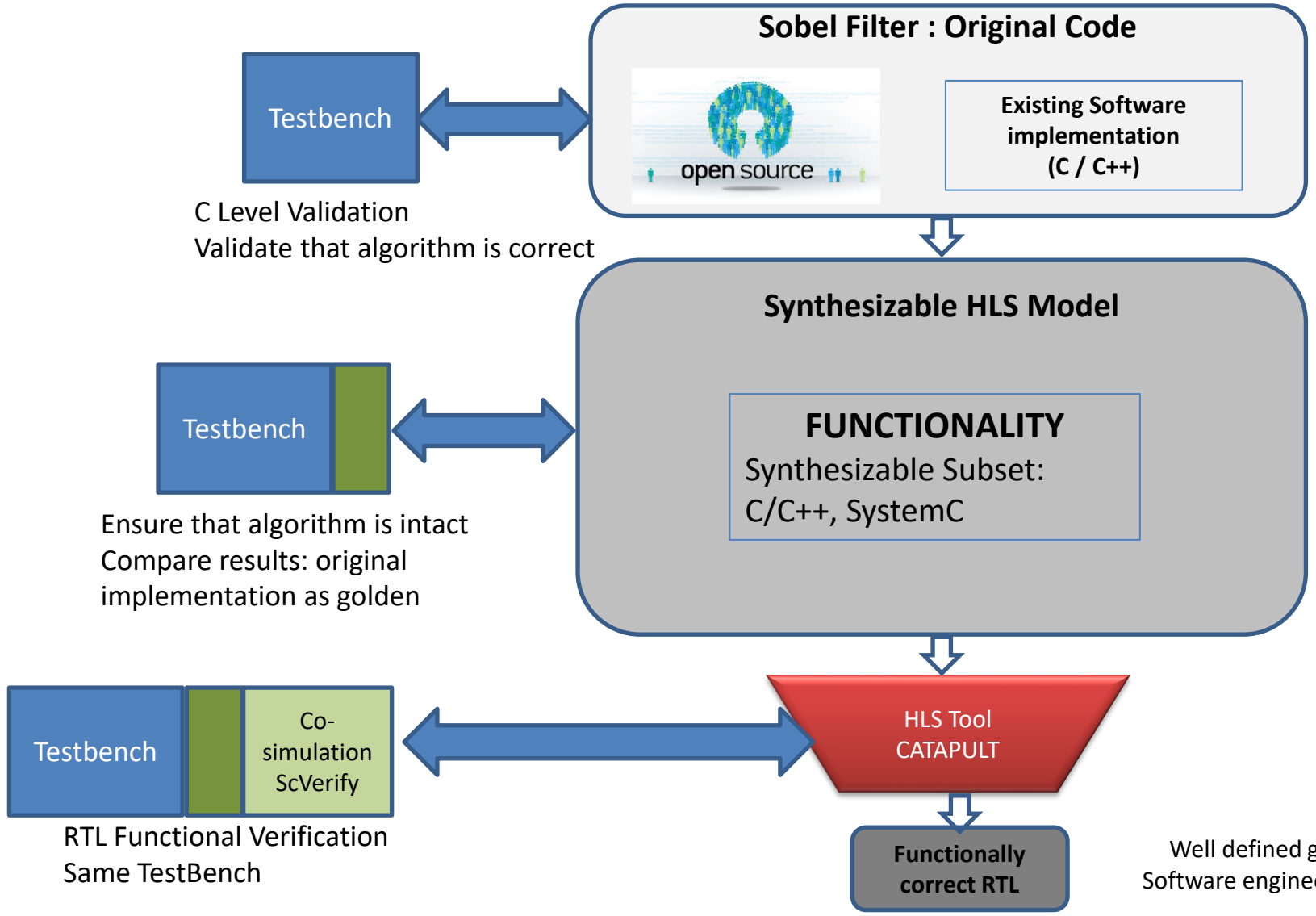
HLS Implementation 2: Synthesizable SystemC (Cont.)

- **Arrays:**
  - Static Array
  - Arrays synthesize to Memory: based on MEM_MAP_THRESHOLD configured
- **SC Module Data Members:**
  - Cannot be shared across Threads/Methods
  - Use shared memories, for sharing data.
  - **Used CCS sample memory : mem_1r1w**
- **Data Types**
  - ac_fixed
  - sc_uint
- **Math Operations**
  - Include <ac_math.h> library
    - ac_abs
    - ac_pow_pwl
    - ac_sqrt_pwl

Encapsulated the Algorithm in a Semiconductor IP



Synthesis & Verification



Software Simulation REAL Time

SW version	REAL Time
Open Source C design	0m0.011s
Synthesizable C design	0m0.351s
Synthesizable SystemC design	0m0.970s

HW vs SW Simulation of SystemC Design

Simulation	Time
Software (using GCC)	1560006 NS
Generated RTL (using QuestaSim)	4439999 NS

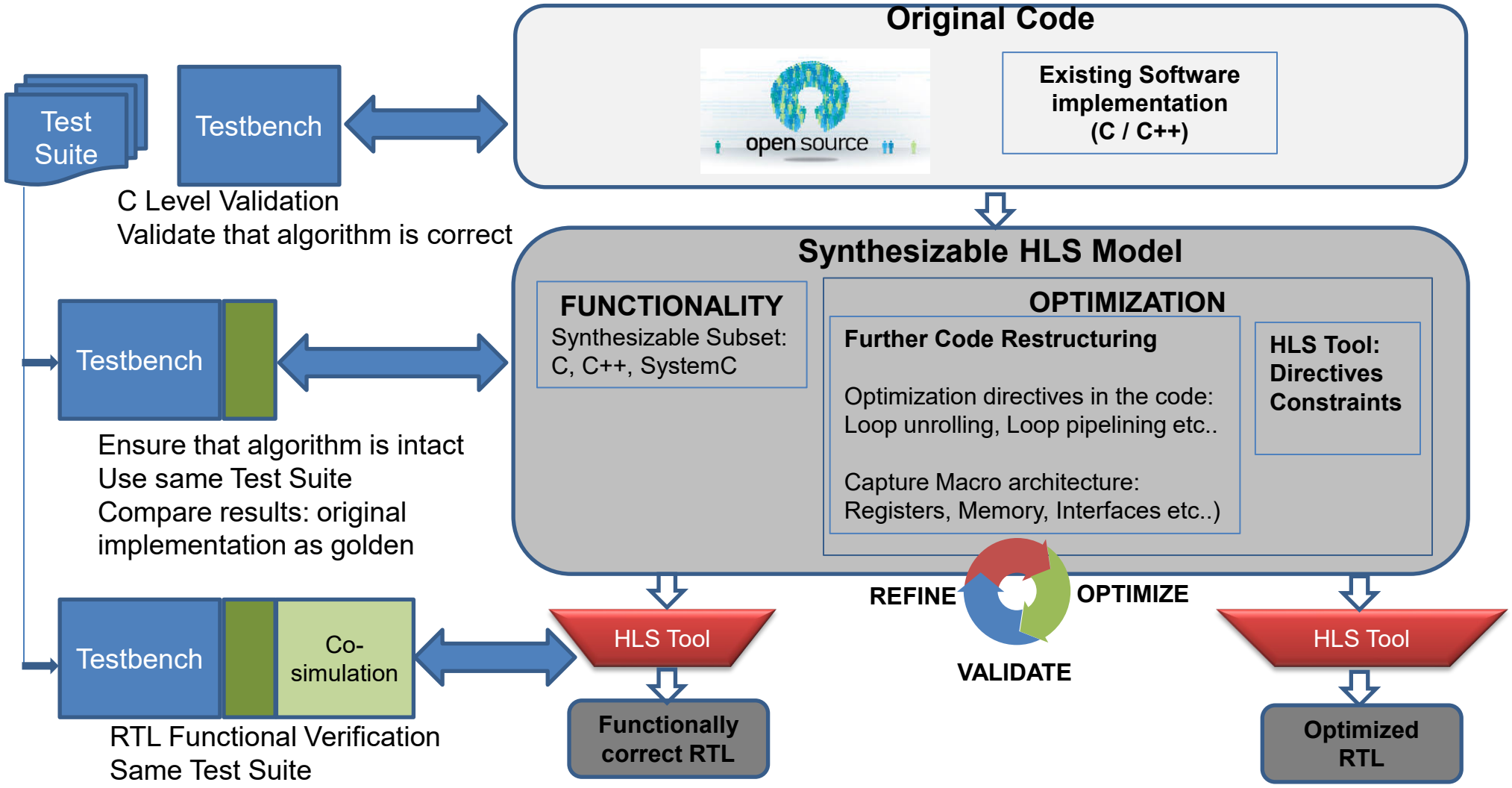
RTL (Verilog) Simulation

HW version	Simulation Time
RTL from Synthesizable C design	12760123 NS
RTL from Synthesizable SystemC design	04439999 NS

Lines of Code (LoC)

Lines of Code	
Open Source C Implementation	RTL (Verilog) Generated
~150	-
Synthesizable C Design	RTL (Verilog) Generated
~210	~10k
Synthesizable SystemC Design	RTL (Verilog) Generated
~300	~5k

Refining the code for HLS ..



C Level validation
Faster Simulation
Less code to verify
Catch bugs early
Reduce efforts for RTL verification

Test Suite
Develop Comprehensive Test Suite for high functional coverage

Reuse:
-Testsuite of original software
-Compliance test suite of protocols

Future Scope to Study

- Optimize & Refine the RTL for Power Performance Analysis, by using HLS Tool directives, constraints, and code re-structuring for macro architecture.
- Expand the Sobel's SystemC design to use synthesizable hardware functions or components in Nvidia's open source HLS library, viz. MatchLib.

Thank you for your time.