POLITECNICO MILANO 1863 | DEPARTMENT OF ELECTRONICS INFORMATION AND BIOENGINEERING

June 21st, 2025

International Symposium on Computer Architecture (ISCA)

Automated Co-Simulation for Functional and System-level Verification of HLS Accelerators

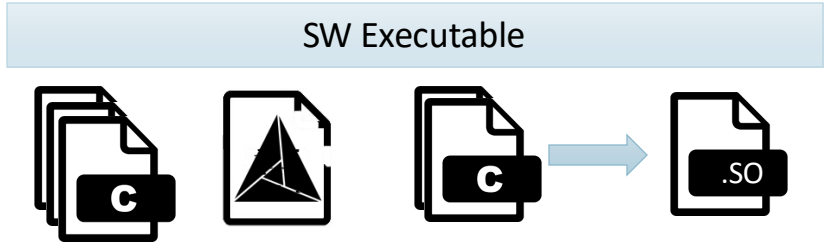
06.21.2025 | Michele Fiorito, Serena Curzel, Fabrizio Ferrandi

Outline

1. Problem Description
2. Proposed Co-simulation Environment
3. HW/SW Memory Interface API
4. Automated Verification
5. Experimental Results

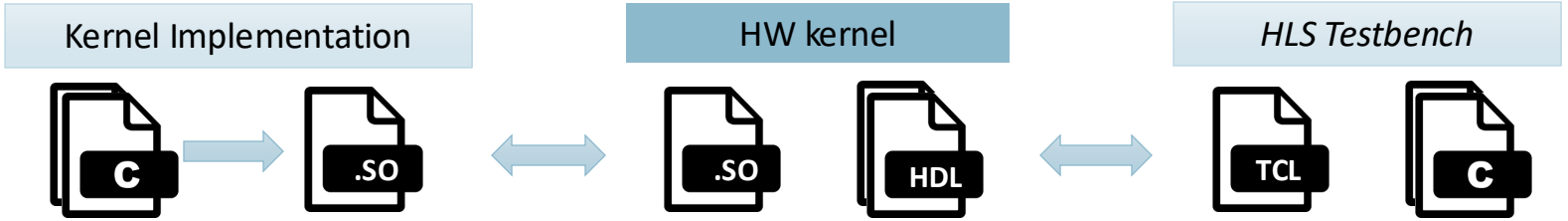
Problem Description

- Starting point
  - Verified software application (source files, build files, external libraries)



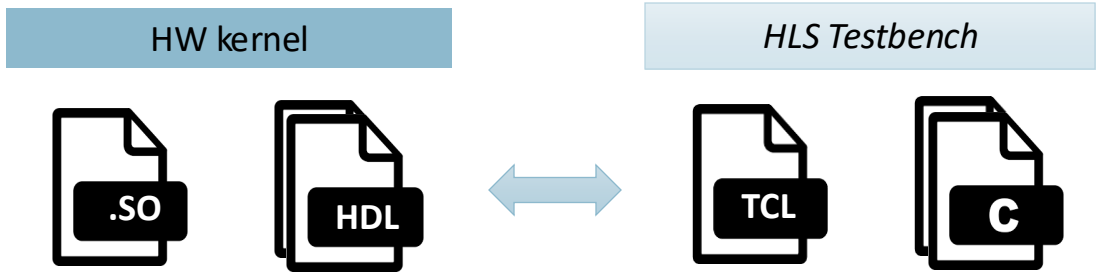
- HLS kernel generation flow
  - Kernel implementation + Testbench implementation

    (data collection, manual verification, no system-level integration)



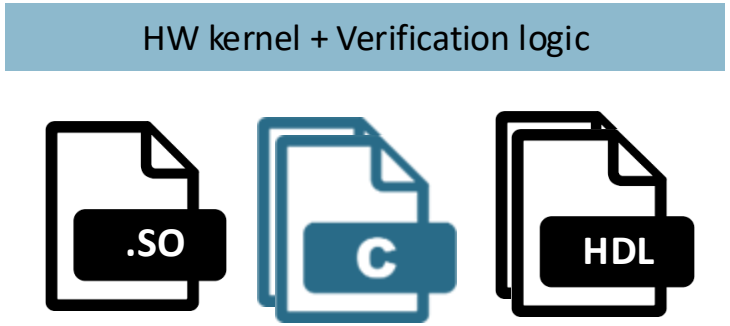
Usual HLS testbench requirements

- Commercial HLS Tools offer limited support for testbench implementation
  - Restricted set of C/C++
  - Non-portable build environment (source must be included in the tool env)
  - Limited I/O interaction
  - Manual implementation required
  - Serial execution
  - Error-prone

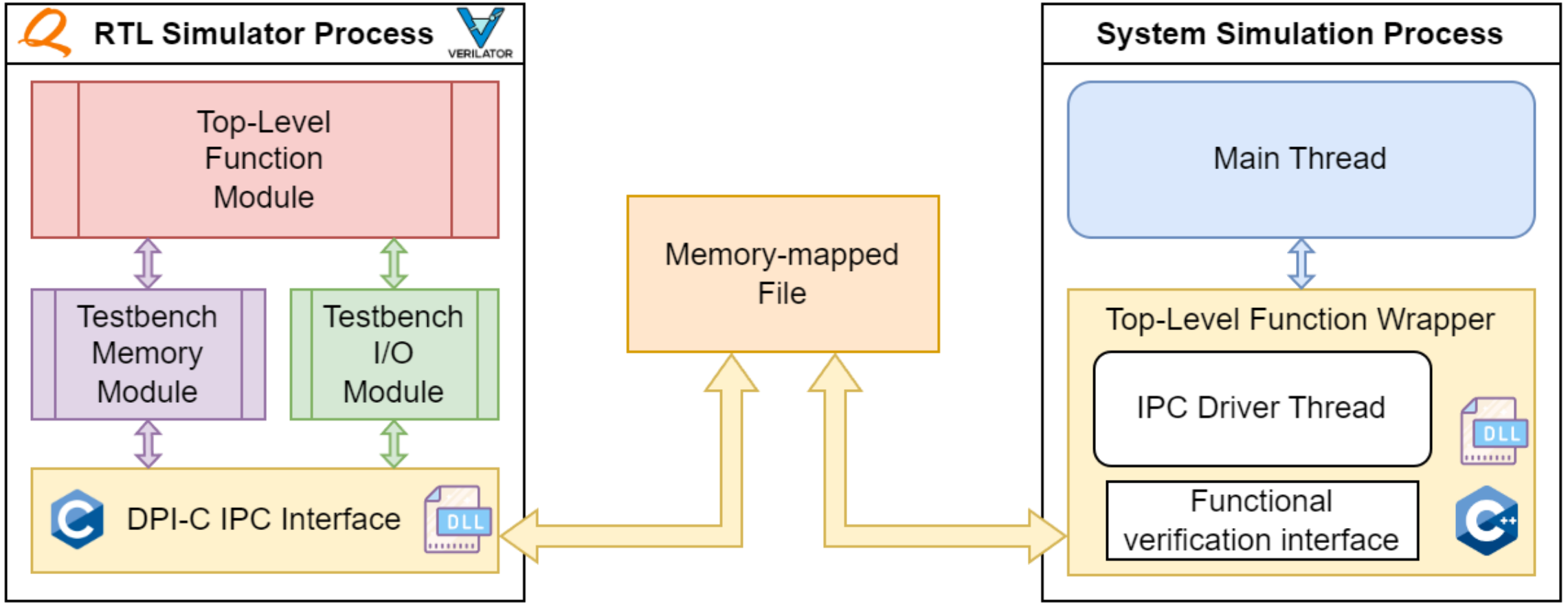


Proposed work

- System-level HW/SW Co-simulation environment
  - HLS tool generates the testbench code automatically
  - Automated simulation verification based on HLS input description
  - Parallel HW/SW execution
  - Integrated memory consistency checks
  - Unconstrained integration within pre-built applications (C, C++, Python)



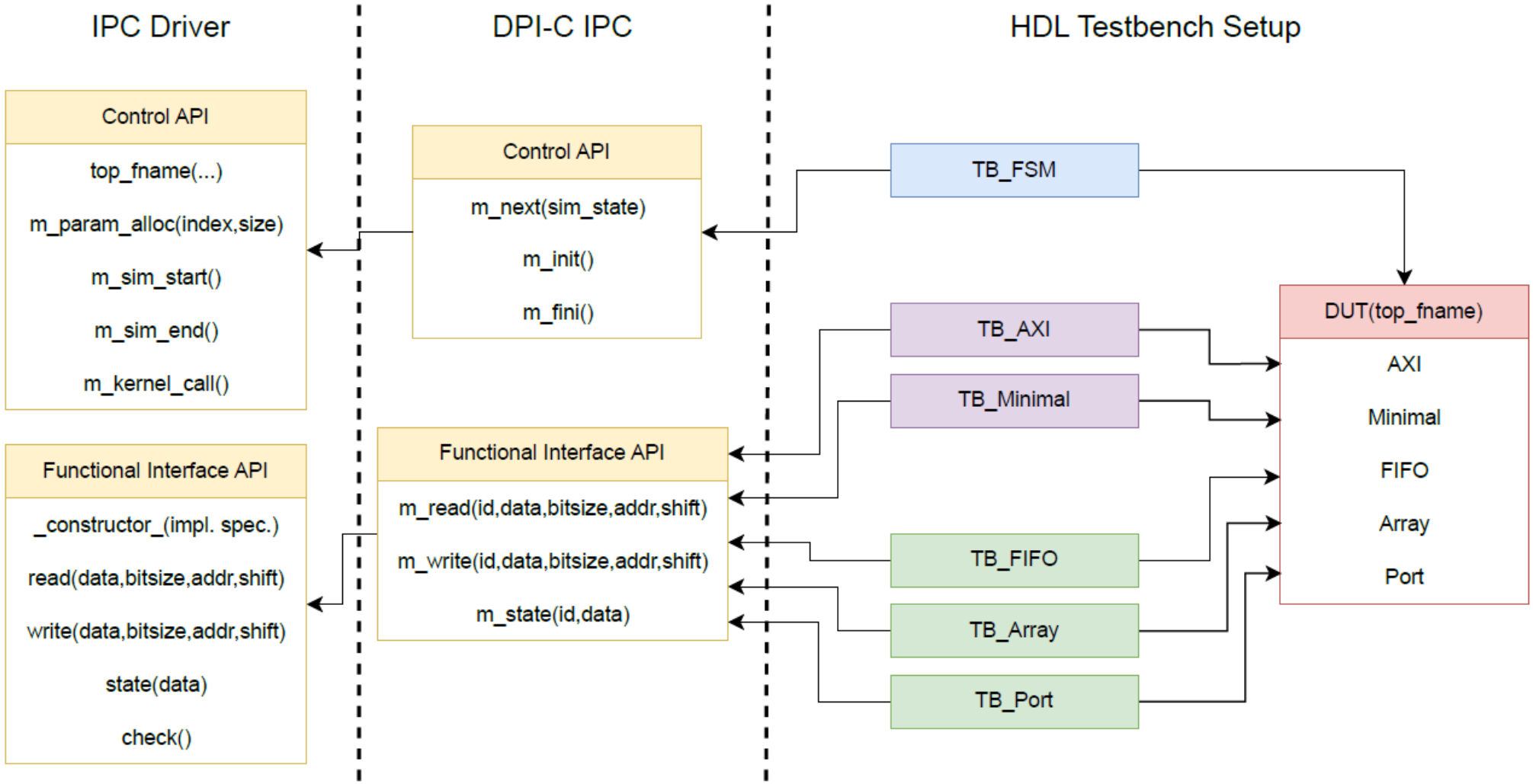
HW/SW Co-Simulation Environment Architecture



Memory DPI-C Inter-Process Communication API

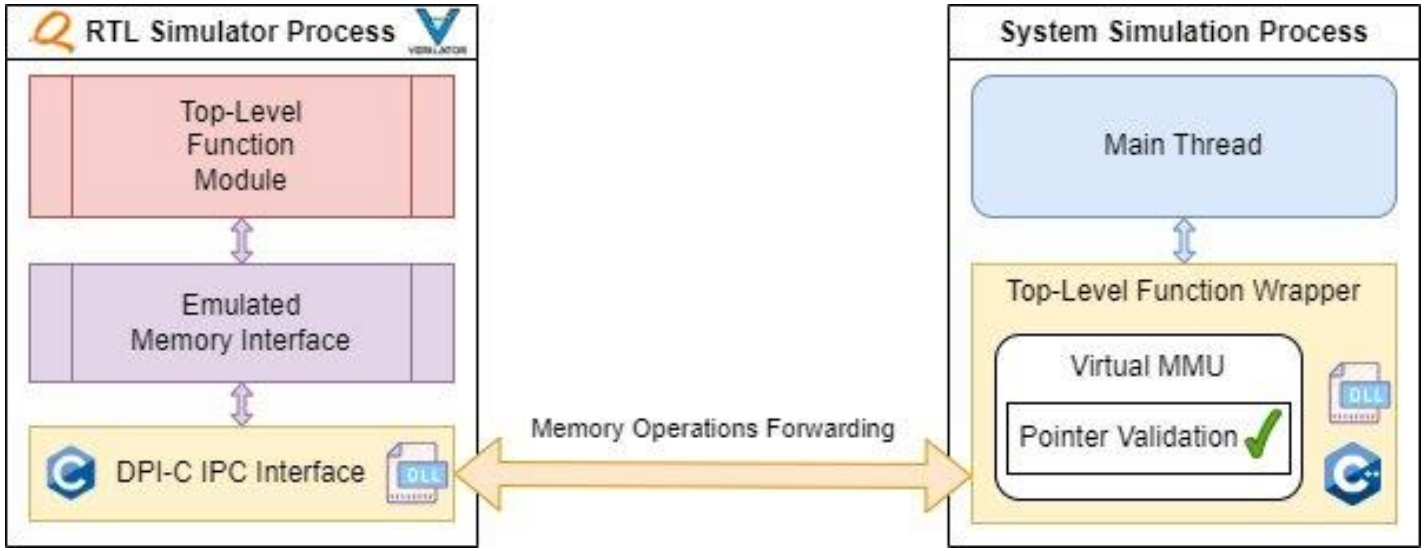
- Inter-process communication between HDL simulator and system application:
  - Realistic HW/SW interaction simulation
  - Address space separation emulation through virtual Memory Mapping Unit (vMMU)
  - Integrated hardware protocol consistency verification
  - Integrated memory operations consistency verification
  - Parallel verification

Memory DPI-C Inter-Process Communication API

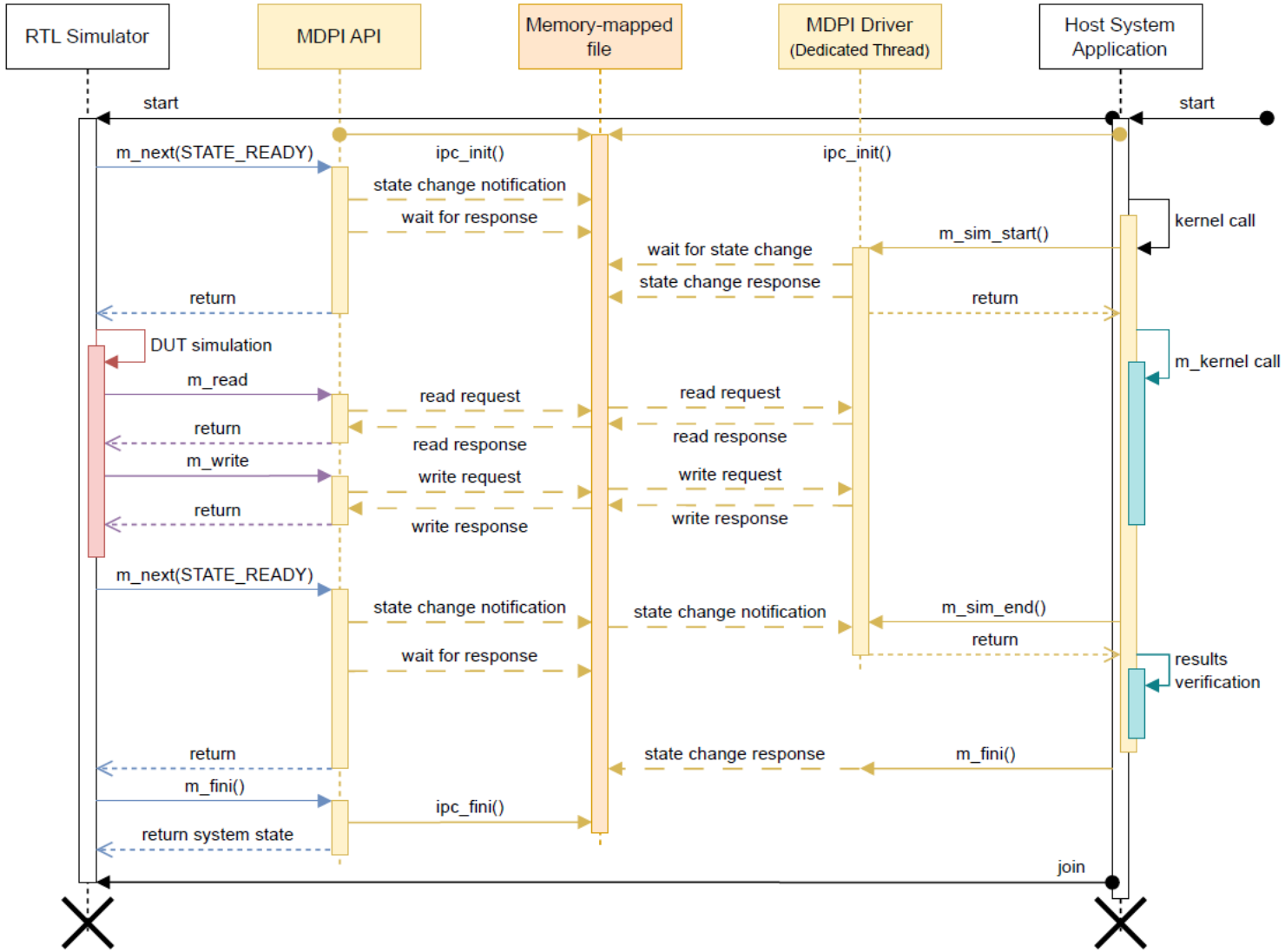


Integrated Hardware Protocol Consistency Checks

- Inter-process communication middleware enables many additional verifications:
  - Hardware protocol functional verification
  - Memory operations verification (read-only memory, out of bounds)
  - Detailed operations profiling



HW/SW Co-Simulation Runtime Execution Diagram

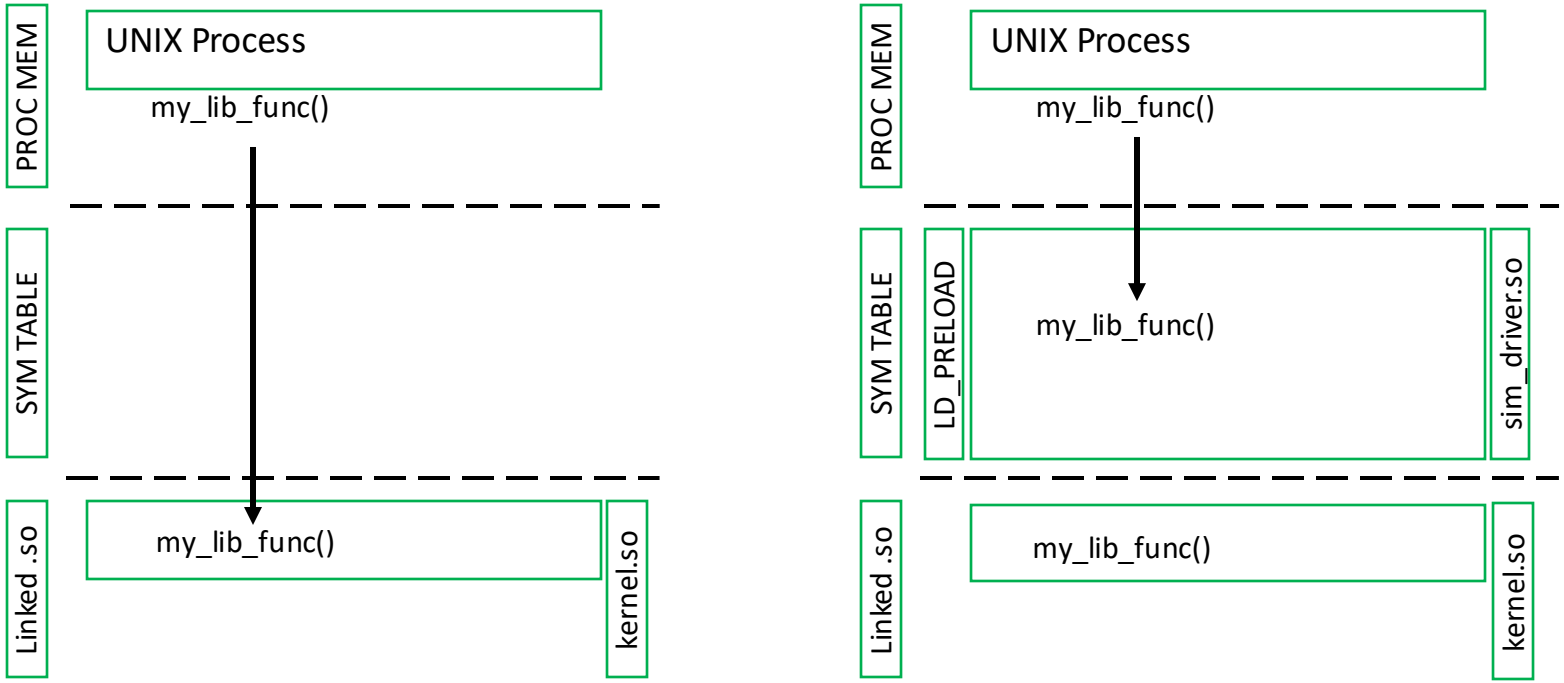


System-Level HW/SW Co-Simulation Integration

- Pre-built application may interact directly with the simulated hardware environment
  - Only requirement is the application must load the top-level function symbol dynamically (or link to the generated library at compile-time)
  - System application may be written in **ANY** language (even Python) as long as the top-level symbol is linked in C

Dynamic Symbol Loading at Runtime

- Co-Simulation symbol injection is also possible in dynamically linked application
  - Dynamically linked symbol are loaded at runtime by the OS (left)
  - When a required symbol is not found in the symbol table it is loaded
  - The symbol table can be pre-loaded setting the LD_PRELOAD variable (right)



Experimental Results

❑ Comparison with commercial HLS tool simulation environment on MachSuite**

- ➢ 6.1x faster simulation
- ➢ 29.5x cycles/s
- ➢ 14.8x AC/s*

Benchmark	Simulation Time (s)			Cycle/s			AC/s		
	Tool 1	Bambu	Speedup	Tool 1	Bambu	Speedup	Tool 1	Bambu	Speedup
aes	32.82	12.28	2.67x	41	247	6.05x	0.82	1.04	1.26x
backprop	23727.90	2812.11	8.44x	189	8081	42.86x	56.89	866.08	15.22x
bfs_bulk	31.00	10.97	2.83x	386	1838	4.76x	0.72	3.59	4.99x
bfs_queue	30.64	11.91	2.57x	447	1533	3.43x	0.51	2.24	4.39x
fft_strided	153.89	16.73	9.20x	632	4593	7.26x	6.54	33.99	5.20x
fft_transpose	120.34	36.47	3.30x	103	2960	28.87x	14.42	892.10	61.87x
gemm_blocked	600.37	46.54	12.90x	437	35295	80.83x	3.03	142.59	47.05x
gemm_ncubed	712.70	39.12	18.22x	184	14033	76.13x	4.65	48.97	10.54x
kmp	32.62	13.14	2.48x	5989	12483	2.08x	8.32	14.36	1.72x
md_grid	808.32	59.58	13.57x	490	11078	22.59x	15.44	427.29	27.67x
md_knn	261.14	26.43	9.88x	9	2073	219.85x	1.30	75.92	58.31x
merge	31.15	11.56	2.69x	2039	7087	3.48x	4.69	21.61	4.61x
nw	30.27	13.23	2.29x	1121	5065	4.52x	3.40	13.83	4.07x
spmv_crs	69.15	12.59	5.49x	189	540	2.85x	0.75	2.06	2.76x
spmv_ellpack	81.34	13.32	6.11x	316	1002	3.17x	1.26	3.85	3.05x
stencil2d	32.10	11.75	2.73x	3164	11314	3.58x	1.30	4.30	3.31x
stencil3d	41.64	11.87	3.51x	546	4651	8.52x	58.39	6.93	0.12x
viterbi	363.63	144.78	2.51x	810	8303	10.25x	75.57	760.55	10.06x

* AC/s = $\frac{Design\ Slices * Simulated\ Cycles}{100.000}$ per second

**Results of synthesis of Virtex 7 (xc7vx690tffg1930-3) at 100MHz simulated with AMD Xilinx XSIM

Experimental Results

- Comparison with commercial HLS tool simulation environment on PolyBenchC**

  - 27.8x faster simulation
  - 13.2x cycles/s
  - 24.4x AC/s*

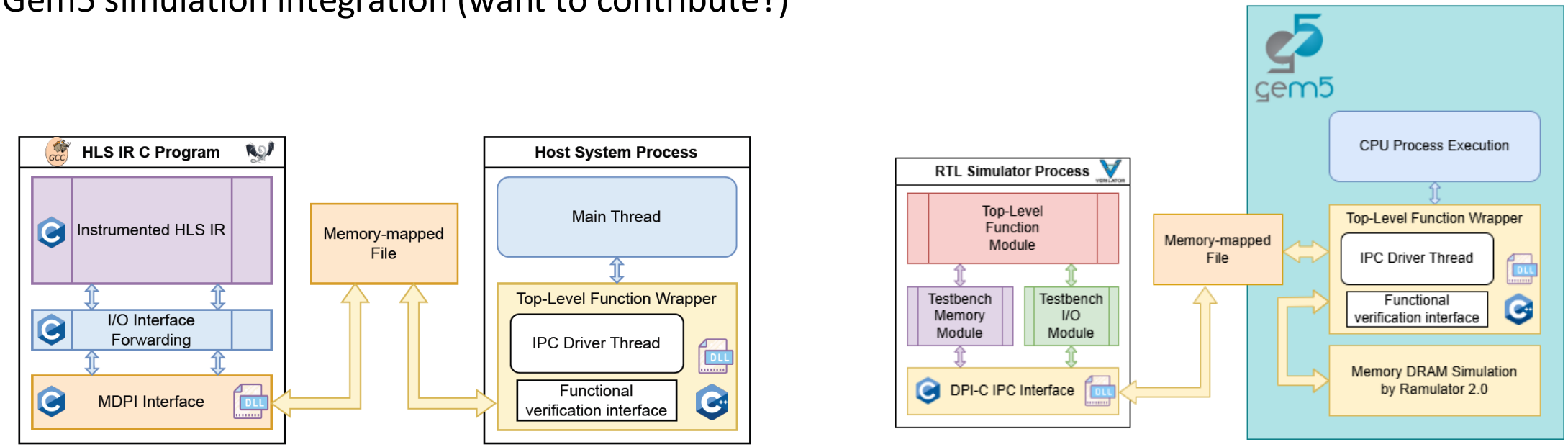
Benchmark	Simulation Time (s)			Cycle/s			AC/s		
	Tool 1	Bambu	Speedup	Tool 1	Bambu	Speedup	Tool 1	Bambu	Speedup
2mm	1104	18.94	58.30x	1165	9303	7.98x	2.48	36.28	14.62x
3mm	1519	21.15	71.83x	1156	14351	12.42x	3.24	79.94	24.70x
atax	119	14.12	8.43x	626	2091	3.34x	0.61	8.13	13.26x
bicg	138	14.77	9.34x	429	1020	2.38x	0.41	2.54	6.24x
covariance	843	25.47	33.10x	1695	18067	10.66x	24.98	223.85	8.96x
doitgen	1126	18.24	61.73x	1746	15381	8.81x	1.96	39.99	20.44x
durbin	92	15.32	6.01x	536	1128	2.11x	1.20	6.57	5.45x
floyd-warshall	175	185.81	0.94x	67394	78820	1.17x	68.74	125.32	1.82x
gemm	575	21.41	26.86x	751	16803	22.37x	1.10	47.72	43.21x
gemver	246	16.13	15.25x	528	2337	4.43x	1.43	15.40	10.73x
gesummv	112	15.11	7.41x	305	286	0.94x	0.25	0.52	2.03x
jacobi-1d	80	15.61	5.12x	252	2778	11.02x	0.92	23.44	25.42x
lu	2209	37.89	58.30x	1871	48064	25.69x	5.50	269.64	49.02x
ludcmp	1756	29.30	59.93x	1507	24406	16.20x	5.94	272.37	45.87x
mvt	134	13.72	9.76x	860	1154	1.34x	1.41	4.34	3.08x
nussinov	40	44.24	0.90x	51812	48669	0.94x	173.57	188.84	1.09x
seidel-2d	9594	314.69	30.49x	2736	67825	24.79x	32.20	346.58	10.76x
symm	908	19.58	46.36x	804	12508	15.56x	1.27	34.02	26.79x
syr2k	1010	22.55	44.78x	310	26662	85.88x	0.46	72.52	156.78x
syrk	401	18.69	21.45x	722	17021	23.57x	0.98	51.40	52.34x
trisolv	135	14.52	9.30x	656	1335	2.03x	0.96	5.19	5.38x
trmm	452	17.10	26.43x	3260	21895	6.72x	5.31	52.33	9.85x

**Results of synthesis of Virtex 7 (xc7vx690tffg1930-3) at 100MHz simulated with AMD Xilinx XSIM

* AC/s = $\frac{Design\ Slices * Simulated\ Cycles}{100.000}$ per second

Current & Future Development

- The proposed co-simulation environment enabled further research on:
  - Faster C-based simulation environment (under review)
  - Timing-accurate RAM simulation (under development)
  - Gem5 simulation integration (want to contribute?)



Questions

Bambu HLS Open-Source Project

https://github.com/ferrandi/PandA-bambu

Bambu HLS Documentation

https://docs.bambuhls.eu

Thank you!

Bambu HLS Releases

https://release.bambuhls.eu

This work has been partially supported by the Spoke 1 "FutureHPC & BigData" of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC) funded by MUR Missione 4 - Next Generation EU (NGEU).