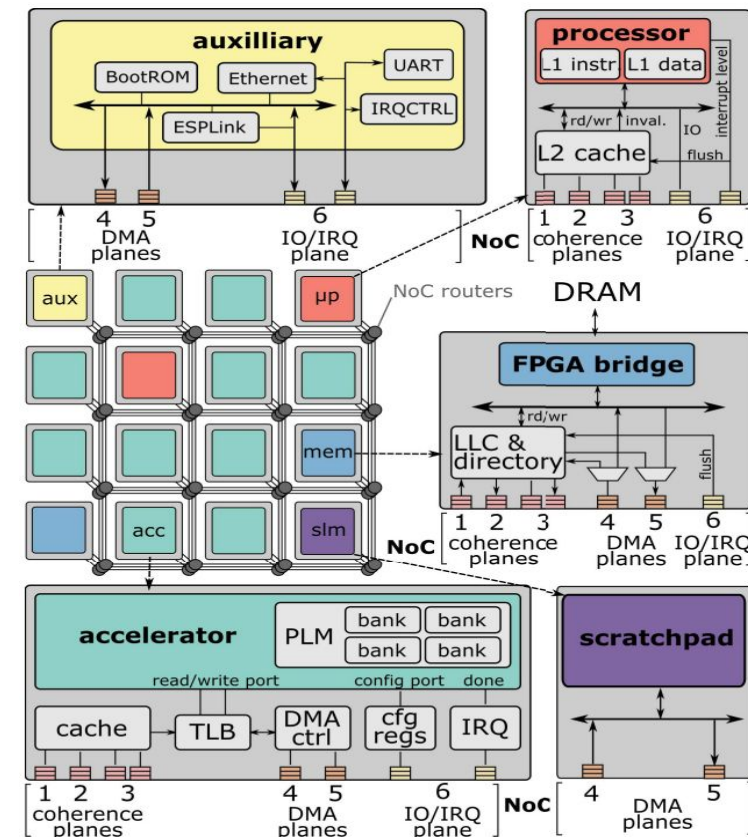


TILY: A CONFIGURABLE TILE-BASED HETEROGENEOUS RISC-V SOC PLATFORM



Describe a chip once. Generate consistent hardware, software and FPGA builds.

Om Maheshwari (B23089)

Guide: Dr. Bikram Paul

Microelectronics & VLSI

A²DSL, Advanced Architecture for Digital Systems Lab

Indian Institute of Technology Mandi

PROBLEM DEFINITION

When the same SoC information is written in different files, one outdated file can cause the system to fail.

Failure mode	Consequence
RTL, linker, device tree and headers repeat the same address facts	One stale file can produce a wrong build that still reports success
New IP depends on shared interconnect and addressing logic	Adding one accelerator forces risky edits across common machinery
Empty or unsupported targets drop requests	The requester waits forever and the fault is difficult to diagnose
Software assumes a fixed tile count and fixed addresses	Each hardware configuration needs a different binary or manual changes

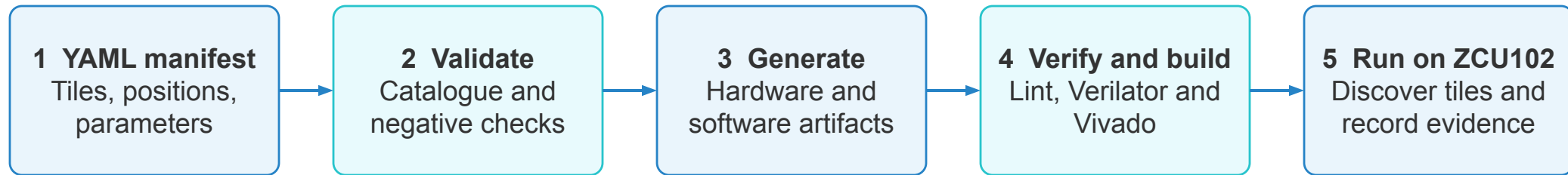
Research question: Can one configuration generate a consistent, extensible FPGA SoC and let software discover the actual hardware at run time?

OBJECTIVES

Make one machine-readable configuration the source of truth for the complete SoC.

- 1 Generate RTL, address maps and software artifacts from one YAML description.
- 2 Modularity: Place CPU, memory, I/O and accelerator tiles behind one standard socket.
- 3 Predictable Communication: Derive routing and base addresses from tile coordinates, with safe error replies for empty positions.
- 4 Validate the generated system through lint, simulation, Vivado implementation and runtime discovery on ZCU102.

PROPOSED METHODOLOGY



Generated artifacts

SoC top RTL and synthesis file list
Address map, C headers and tile table
Linker script and device tree
Configuration-specific boot image

Address contract

$$\text{base}(x,y) = (y \ll 28) + (x \ll 24)$$

**XY routing uses the same
coordinates. Empty positions
receive an error responder.**

Runtime IDs, serial transcripts, mutation tests and implementation reports feed the evidence file.

LITERATURE SURVEY

Work	Year	Venue	Most relevant idea
Chisel	2012	DAC	Parameterised hardware construction and reusable RTL generators.
Rocket Chip Generator	2016	UCB TR	A configurable RISC-V SoC generator built around the Chisel ecosystem.
OpenPiton	2016	ASPLOS	A scalable tiled manycore with NoC, verification and FPGA support.
Chipyard	2020	IEEE Micro	An integrated framework spanning SoC design, simulation and implementation.
ESP	2022	ICCAD	Tile sockets and hardware-software co-generation; the closest architectural precedent.

TILY focuses on a smaller FPGA-first flow with one YAML source, coordinate-derived addresses, runtime discovery and evidence tied to real board runs.

TILY ARCHITECTURE AND TARGET

Example 3 × 3 generated mesh

CVA6 core (0,0) 0x00000000	UART (1,0) 0x01000000	CLINT (2,0) 0x02000000
On-chip RAM (0,1) 0x10000000	MAC accelerator (1,1) 0x11000000	MAX accelerator (2,1) 0x12000000
Error responder (0,2) 0x20000000	POPCOUNT (1,2) 0x21000000	Error responder (2,2) 0x22000000

Target platform

Board: Xilinx ZCU102

Part: xczu9eg-ffvb1156-2-e

Fabric clock: 50 MHz

Cores: CVA6 RV64 and PicoRV32 RV32

Software: FreeRTOS and bare metal

Position is the address

31:28 tile_y	27:24 tile_x	23:0 local offset
-------------------------------	-------------------------------	------------------------------------

base(x,y) = (y << 28) + (x << 24)
16 × 16 positions, 16 MiB per tile

MILESTONES AND NEXT STEPS

Validated in MTP-I

- 1 Generated PicoRV32 SoC ran on the ZCU102
- 2 One config line switched the platform to CVA6 RV64
- 3 Popcount accelerator integrated end to end
- 4 CLINT enabled timer interrupts
- 5 FreeRTOS ran one worker per discovered tile
- 6 UART receive path enabled an interactive shell
- 7 CVA6 and PicoRV32 communicated on one mesh

Next phase

Multi-core memory

Define a safe locking or message-passing contract.

External memory

Extend beyond the on-chip BRAM capacity.

Scalability study

Measure latency, resources and generation time as the mesh grows.

console: press the board's CPU_RESET to see the boot banner again.

```
RTOS BOOT
CONFIG=config_zcu102_shell3x3
TIMEBASE=1000000
TASKS=6
task[0] io tile=1,0
task[1] system tile=2,0
task[2] memory tile=0,1
task[3] accelerator tile=1,1
task[4] accelerator tile=2,1
task[5] accelerator tile=1,2
TICK_TEST = PASS
SCHEDULER_TEST = PASS
WORKER_TEST = PASS
RTOS_TEST = PASS
```

MTP1 shell on config_zcu102_shell3x3. Type help.
The workers are still running underneath; 'tasks' shows them.

```
mtp1> tiles
#  tile      x,y  base      device ip-version answers
0  core.cva6  0,0  0x00000000 0x000000C6 0x00010000 yes
1  io.uart    1,0  0x01000000 0x00000011 0x00010000 yes
2  system.clint 2,0  0x02000000 0x00000012 0x00010000 yes
3  memory.onchip 0,1  0x10000000 0x00000010 0x00010000 yes
4  accelerator.mac 1,1  0x11000000 0x000000A1 0x00010000 yes
5  accelerator.max 2,1  0x12000000 0x000000A2 0x00010000 yes
6  accelerator.pop 1,2  0x21000000 0x000000A3 0x00010000 yes
```

mtp1>

```
TICK_TEST = PASS
SCHEDULER_TEST = PASS
WORKER_TEST = PASS
RTOS_TEST = PASS
```

MTP1 shell on config_zcu102_amp. Type help.
The workers are still running underneath;

```
mtp1> tiles
#  tile      x,y  base      dev
0  core.cva6  0,0  0x00000000 0x0
1  io.uart    1,0  0x01000000 0x000000
2  system.clint 2,0  0x02000000 0x000000
3  memory.onchip 0,1  0x10000000 0x000000
4  core.picorv32 1,1  0x11000000 0x000000
5  accelerator.mac 2,1  0x12000000 0x000000
```

mtp1>

RTOS_TEST = PASS

MTP1 shell on config_zcu102_freertos. Type help.
The workers are still running underneath; 'tasks' shows them.

```
mtp1> tiles
#  tile      x,y  base      device ip-version answers
0  core.cva6  0,0  0x00000000 0x000000C6 0x00010000 yes
1  io.uart    1,0  0x01000000 0x00000011 0x00010000 yes
2  system.clint 2,0  0x02000000 0x00000012 0x00010000 yes
3  memory.onchip 0,1  0x10000000 0x00000010 0x00010000 yes
4  accelerator.mac 1,1  0x11000000 0x000000A1 0x00010000 yes
```

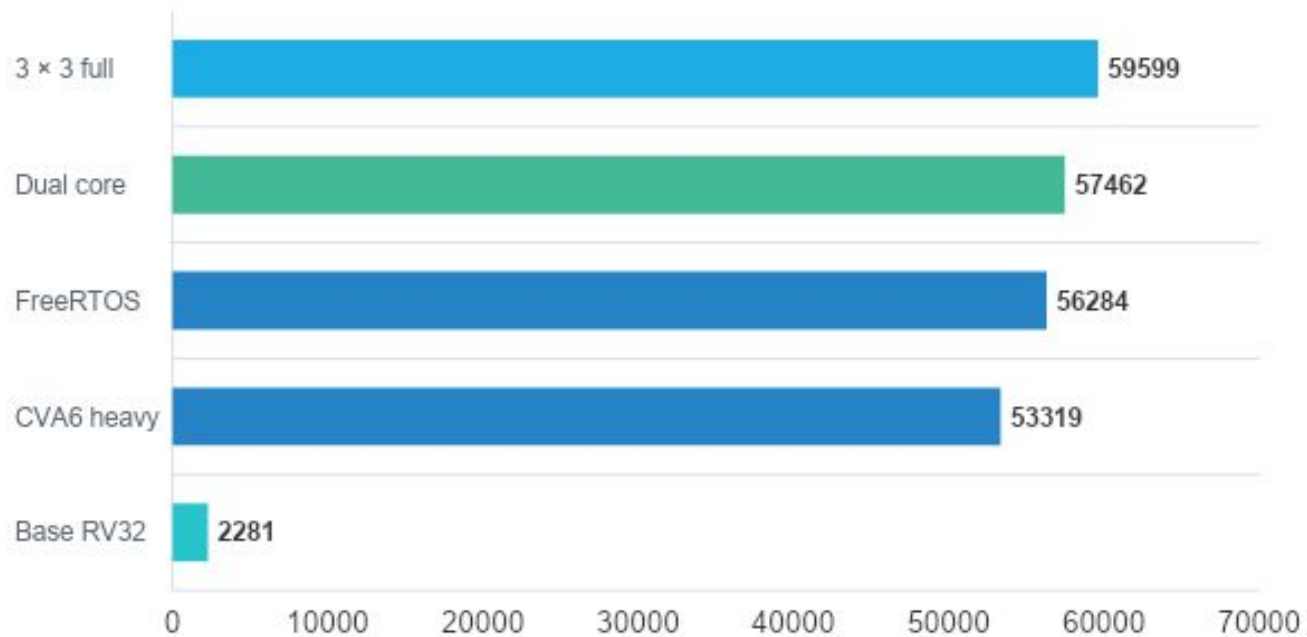
```
mtp1> help
tiles      every tile this SoC has, read from its own registers
tasks      the tasks the scheduler is running, and their stacks
peek ADDR  read a 32-bit word from anywhere in the mesh
poke ADDR VAL write one
mem         write a pattern to the memory tile and read it back
mac A B     run the accelerator, if this SoC has one
uptime      ticks since boot
echo TEXT   prove the console round-trips
help
```

```
mtp1> uptime
37733 ticks at 1000 Hz = 37 s
```

```
mtp1> mac A B
mac A B (try: mac 7 6)
```

MEASURED IMPLEMENTATION RESULTS

Post-route LUT usage on ZCU102



Board evidence

21.8%

Device LUTs used by the full 3 × 3 CVA6 system

+1,178 LUT

Added an entire second PicoRV32 processor

5 and 7 tiles

One program discovered both configurations at run time

11 of 11

UART receive mutations were caught by the tests

All listed LUT values come from utilisation_impl.rpt after place and route. The evidence file reports all 11 benchmarked configurations meeting timing.

REFERENCES

- [1] J. Bachrach et al., ‘Chisel: Constructing Hardware in a Scala Embedded Language,’ DAC, 2012, doi: 10.1145/2228360.2228584.

- [2] K. Asanović et al., ‘The Rocket Chip Generator,’ UCB/EECS-2016-17, University of California, Berkeley, 2016.

- [3] J. Balkind et al., ‘OpenPiton: An Open Source Manycore Research Framework,’ ASPLOS, pp. 217-232, 2016, doi: 10.1145/2872362.2872414.

- [4] A. Amid et al., ‘Chipyard: Integrated Design, Simulation, and Implementation Framework for Custom SoCs,’ IEEE Micro, vol. 40, no. 4, pp. 10-21, 2020, doi: 10.1109/MM.2020.2996616.

- [5] M. C. dos Santos et al., ‘A Scalable Methodology for Agile Chip Development with Open-Source Hardware Components,’ ICCAD, 2022, doi: 10.1145/3508352.3561102.

- [6] O. Maheshwari, ‘MTP1/TILY Complete Reference and Evidence File,’ A²DSL, IIT Mandi, Sep. 2026.

THANK YOU.

TILY: Describe a chip in one file.

Om Maheshwari (B23089)
A²DSL
Advanced Architecture for Digital Systems Lab
Microelectronics & VLSI, IIT Mandi
Guide: Dr. Bikram Paul