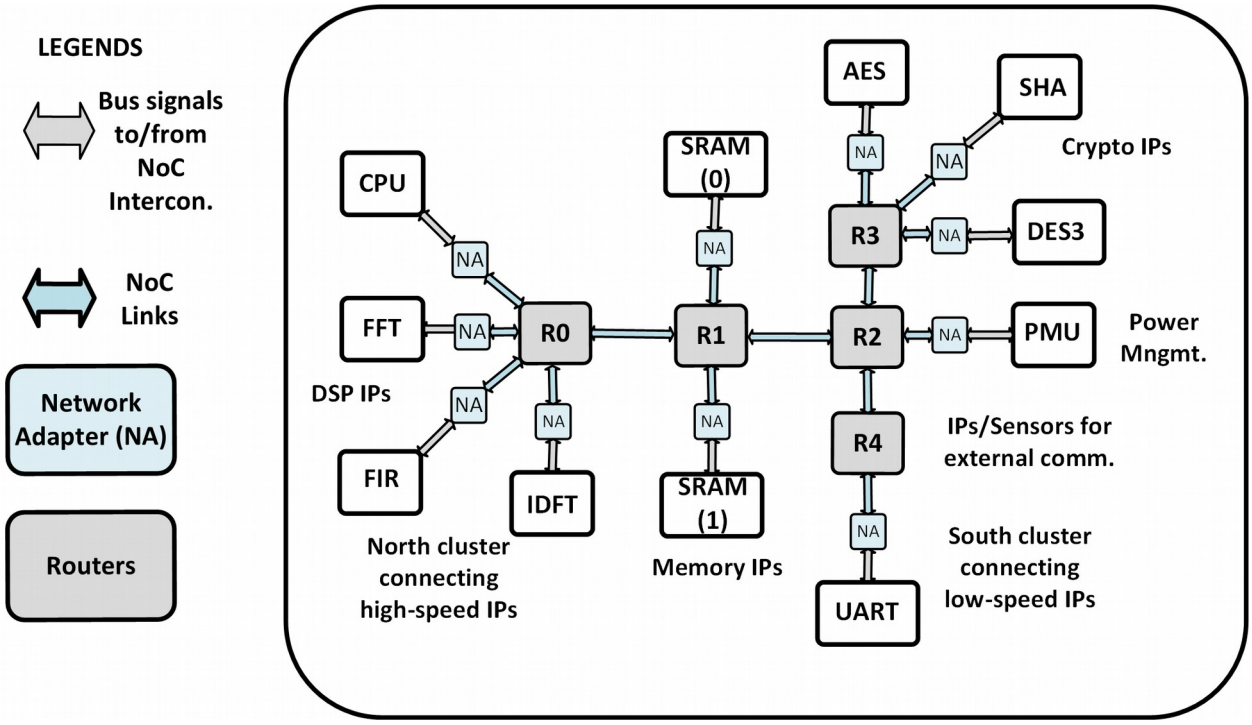
# ClusterSoC: SoC Benchmark with NoC Interconnect




## System Overview:

The ClusterSoC design includes a RISC-V CPU, namely PicoRV32 and two dual port SRAMs integrated as memory modules. It features three high speed DSP blocks, including DFT, IDFT, and FIR. Three crypto modules i.e., AES, DES3, and MD5 are included for generic cryptographic operations. An UART module is integrated with the SoC model for external communication. To facilitates the study of different system use case scenario of realistic industrial SoC designs, the IPs are segregated mainly into two clusters. The North cluster includes the high-speed IPs (e.g., CPU and DSP accelerators) and the South cluster incorporates crypto engines, and external communication IP i.e., UART.

## SoC Components:

### PicoRV32:

PicoRV32 is a CPU core that implements 32-bit RISC-V ISA. It can be configured as RV32E, RV32I, RV32IC, RV32IM, or RV32IMC core, and it optionally contains a built-in interrupt controller. Toolchain (gcc, binutils, etc..) for the core can be obtained from RISC-V Website. Detailed description of the core can be found in [1].

### LiSNoC:

The SoC model incorporates a modular and configurable open-source Network-on-chip, named LiSNoC, as the interconnect fabric. It supports wormhole-based flow control. The basic routing algorithm is dimension-ordered, deadlock-free XY-routing. The network interface permits transfer of one flit per cycle. The packet arbitration employs strict ordering through the router switch. The link multiplexing is

performed by round-robin arbitration. The number of input ports, output ports, and allocation of virtual channels (VCs) to input-output ports are parameterizable. Each VC has a fixed-length queue of 4 flits. The virtual channel flow control is performed in on-off manner with two control signals named valid and ready. The flit transfer occurs through a handshaking protocol implemented with the ready and valid signals of each VC. The transfer occurs when both valid and ready signals are high. The associated network adapters feature DMA engines and message passing functionalities.

## AES:

The AES core implements a symmetric-key algorithm, in which the same key is used for both encrypting and decrypting the data. The block size is restricted to 128 bits. The key size can be 128, 192, or 256 bits. It operates on a 4×4 matrix of bytes, called the state. Some rounds of transformation convert the plaintext into the final cipher-text. The number of rounds is six plus the key size divided by 32. One round reads the state into four 4-byte variables $y_0, y_1, y_2, y_3$; transforms the variables; xor's them by a 16-byte round key; and puts the result into $z_0, z_1, z_2, z_3$.

When targeting a variable-length plaintext, the plaintext must first be partitioned into separate cipher blocks, and then be encrypted under some mode of operation, generally using randomization based on an additional initialization vector. The cipher feedback (CFB) mode, output feedback (OFB) mode are specified in FIPS 81. The counter (CTR) mode is specified by NIST in SP800-38A. The advantage of these modes is only using encryption algorithm for both encryption and decryption. So the AES hardware price may be reduced by 50% (does not need decryption hardware).

## DES3:

DES3 is an implementation of Triple DES algorithm in verilog. It takes 56-bit keys and 64 bits of data as input and generates a 64-bit encrypted/decrypted result. The following variants of the DES3 crypto block can be implemented:

***1) Area Optimized (CBC Mode):*** This is a sequential implementation and needs 48 cycles to complete a full encryption/decryption cycle.

***2) Performance Optimized (EBC Mode):*** This is a pipelined implementation that has a 48 cycle pipeline (plus 1 input and 1 output register). It can perform a complete encryption/decryption every cycle.

## SHA-256:

This IP core is a hardware implementation of SHA-256 cryptographic hash function with support for both SHA-256 and SHA-224. The implementation is written in Verilog 2001 compliant code. The implementation includes the main core as well as wrappers that provides interfaces for simple integration. This is a low area implementation that iterates over the rounds but there is no sharing of operations such as adders.

## MD5:

This IP core that implements the MD5 message digest algorithm. Some of the features of the core are as following:

- Up to 128-bit input message
- Parameterized 'salts' that default to the MD5-prescribed values
- MD5 compliant 128-bit output

- Verilog entry
- Very minimal amount of control signals (3)
- Synchronous logic

## IIR:

This core is a Transposed Direct Form implementation of an Infinite Impulse Response (IIR) filter, from a standard difference equation. The multiplier blocks within the design are generated from the Spiral Multiplier Block Generator. An FIR can be generated by simply setting $a_0$ to one and all other $a_k$ constants

$$a_0 y[n] = \sum_{k=0}^{N-1} b_k x[n-k] - \sum_{k=1}^{N-1} a_k y[n-k]$$

**Standard Difference Equation**

to zero. The values of the $b_k$ constants form the notches of the FIR. In the Transposed Direct Form I implementation, the IIR is generated from two separate FIR filters, connected by an adder. In the Transposed Direct Form II implementation, the FIR filter tool is not used at all, and the two multiplier block outputs are taken directly through adders and input into the flip flop array.

## FIR:

This core is a Transposed Direct Form implementation of a Finite Impulse Response (FIR) filter, from a standard difference equation. The multiplier blocks within the design are generated from the Spiral Multiplier Block Generator. An FIR can be generated by simply setting $a_0$ to one and all other $a_k$ constants

$$a_0 y[n] = \sum_{k=0}^{N-1} b_k x[n-k] - \sum_{k=1}^{N-1} a_k y[n-k]$$

**Standard Difference Equation**

to zero. The values of the $b_k$ constants form the notches of the FIR. In the Transposed Direct Form I implementation, the IIR is generated from two separate FIR filters, connected by an adder. In the Transposed Direct Form II implementation, the FIR filter tool is not used at all, and the two multiplier block outputs are taken directly through adders and input into the flip flop array.

## DFT:

This is a customized Discrete Fourier Transform (DFT) soft IP core in synthesizable RTL Verilog. It uses fast Fourier transform algorithms (FFTs). The user has control over a variety of parameters that control the functionality and cost/performance tradeoffs such as area and throughput. The salient features of the design are as following:

- DFT Size = 64
- ***direction = forward***
- data type = 16-bit fixed point, unscaled

- architecture = fully streaming
- radix = 2
- streaming width = 2
- data ordering = natural input / natural output
- BRAM budget = 1000
- Input/output stream: 2 complex words per cycle
- Throughput: one transform every 32 cycles
- Latency: 145 cycles

### IDFT:

This is a customized Discrete Fourier Transform (DFT) soft IP core in synthesizable RTL Verilog. It uses fast Fourier transform algorithms (FFTs). The user has control over a variety of parameters that control the functionality and cost/performance tradeoffs such as area and throughput. The salient features of the design are as following:

- DFT Size = 64
- ***direction = inverse***
- data type = 16-bit fixed point, unscaled
- architecture = fully streaming
- radix = 2
- streaming width = 2
- data ordering = natural input / natural output
- BRAM budget = 1000
- Input/output stream: 2 complex words per cycle
- Throughput: one transform every 32 cycles
- Latency: 145 cycles

### UART:

The UART (Universal Asynchronous Receiver/Transmitter) core provides serial communication capabilities, which allow communication with modem or other external devices, like another computer using a serial cable and RS232 protocol. This core is designed to be maximally compatible with the industry standard National Semiconductors' 16550A device.

***Features:***

- WISHBONE interface in 32-bit or 8-bit data bus modes (selectable)
- FIFO only operation
- Register level and functionality compatibility with NS16550A (but not 16450).
- Debug Interface in 32-bit data bus mode.

## SoC Simulation & Emulation:

This SoC can be simulated with directed and random testbenches written in HDL. IP level, bus-level, and system level testbenches for the hardware accelerators are provided. IP level and bus-level simulation can be performed via Verilator and/or ModelSim. FPGA mapping is required for system level emulation. RISC-V tool chain can be used to compile programs (c code compiled to binaries using RISC-V tool chain)

for the RV32 processor core and the generated firmware HEX / MEM file can be stored in the RAM block of the SoC. Generated SoC can be synthesize the design to get area, power, and timing results.

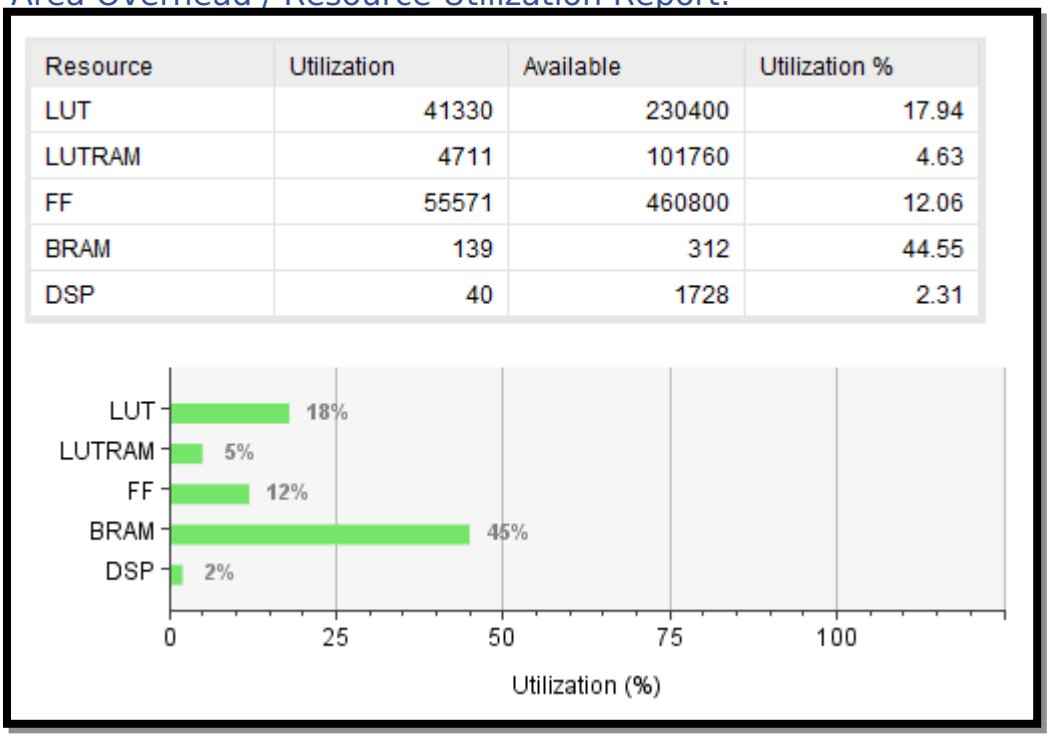
## Implementation Results:

The SoC model is synthesized on a Xilinx FPGA using Vivado Design Suite. The implementation results are shown below:

***Product Family:*** Zynq UltraScale+
***Part:*** xczu7ev-ffvf1517-1LV-i

### Area Overhead / Resource Utilization Report:

| Resource | Utilization | Available | Utilization % |
|---|---|---|---|
| LUT | 41330 | 230400 | 17.94 |
| LUTRAM | 4711 | 101760 | 4.63 |
| FF | 55571 | 460800 | 12.06 |
| BRAM | 139 | 312 | 44.55 |
| DSP | 40 | 1728 | 2.31 |



## Power Overhead / On-chip Power Report:

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

| | |
|---|---|
| **Total On-Chip Power:** | **18.166 W** |
| **Design Power Budget:** | **Not Specified** |
| **Power Budget Margin:** | **N/A** |
| **Junction Temperature:** | **40.4°C** |
| Thermal Margin: | 59.6°C (69.2 W) |
| Effective ϑJA: | 0.8°C/W |
| Power supplied to off-chip devices: | 0 W |
| Confidence level: | High |

Launch Power Constraint Advisor to find and fix invalid switching activity