

Design and Simulation of Parallel CRC Using Pipelining, Unfolding and Retiming

Vishnu Vardhan Reddy Yerramala
Rajiv Gandhi University of Knowledge Technologies
Nuzvid, Andhra Pradesh, India
Email: n220291@rguktn.ac.in

Abstract—This project presents the design and mixed-signal simulation of a high-throughput parallel CRC-9 architecture based on the generator polynomial $G(y) = 1 + y + y^8 + y^9$. By applying an unfolding factor of $J = 2$ alongside pipelining and retiming, the design processes two data bits per clock cycle while constraining the critical path delay to $1 T_{XOR}$. Implemented in synthesizable Verilog and simulated using eSim via NgVeriNgspice ADC/DAC bridging interfaces, the design processes a 10-bit sequence (1100000011) in 5 clock cycles, yielding the exact remainder 9'b010000001 (0x081) with a 44.4% latency reduction over serial implementations.

Index Terms—Cyclic Redundancy Check (CRC), Parallel CRC-9, eSim

I. INTRODUCTION

Cyclic Redundancy Check (CRC) is a standard error-detection scheme in high-speed digital networks and storage interfaces. Conventional CRC implementations rely on serial Linear Feedback Shift Registers (LFSRs) that process one bit per clock cycle, creating a latency bottleneck in high-throughput channels.

To overcome this throughput limitation, VLSI optimization techniques are applied:

- **Unfolding** ($J = 2$): Transforms the serial single-bit structure into a parallel framework that ingests two data bits ($Y(k)$, $Y(2k)$) simultaneously each cycle.
- **Pipelining & Retiming**: Distribute intermediate registers across the logic network to eliminate feedback bottlenecks and balance propagation delay to a single XOR gate level (T_{XOR}).

This work models and verifies the parallel CRC-9 architecture entirely within the open-source eSim mixed-signal environment, validating digital state transitions and analog voltage levels against published research benchmarks.

II. ARCHITECTURE AND WORKING PRINCIPLE

- **Structural Design**: Based on the generator polynomial $G(y) = 1 + y + y^8 + y^9$, the architecture incorporates an unfolding factor of $J = 2$ with pipelining and retiming. It organizes 9 internal registers (x_0 to x_8) into an optimized pipeline that ingests two concurrent data bits— $Y(k)$ and $Y(2k)$ —simultaneously per clock cycle while limiting critical path delay to a single XOR gate ($1T_{XOR}$).
- **Concurrent Feedback Logic**: Unfolding maps the feedback equations to evaluate parallel transitions:

$$\begin{aligned} - x'_0 &= x_7 \oplus Y(k) \\ - x'_1 &= x_8 \oplus x_7 \oplus Y(k) \oplus Y(2k) \\ - x'_8 &= x_6 \oplus x_7 \oplus Y(k) \\ - \text{Direct shifts: } x'_2 &= x_0, x'_3 = x_1, x'_4 = x_2, x'_5 = x_3, \\ &x'_6 = x_4, x'_7 = x_5 \end{aligned}$$

- **Operational Flow**: Following an active-low reset assertion ($\text{rst_n} = 0$), registers clear to zero. At each rising clock edge, the circuit consumes 2 bits and evaluates next-state values concurrently. For a 10-bit frame (1100000011), computation completes in **5 clock cycles**, shifting across states $0 \times 001 \rightarrow 0 \times 006 \rightarrow 0 \times 018 \rightarrow 0 \times 060$ before settling at the final remainder 9'b010000001 (0x081). DAC bridges translate these digital transitions into 0–5 V SPICE transient voltage waveforms.

III. IMPLEMENTATION IN ESIM

The parallel CRC-9 architecture is modeled in synthesizable Verilog and imported into eSim via the NgVeri interface, which compiles the hardware description into a mixed-signal C++/SPICE-compatible block with a custom schematic symbol. In KiCad Eeschema, four analog voltage pulse sources drive the clock, active-low reset, and parallel input streams (Y_k, Y_{2k}) through an `adc_bridge_4` converter into the Verilog core. The 9-bit digital output is translated back into standard 0–5 V levels via `dac_bridge_1` and `dac_bridge_8` interfaces. Finally, a 120 ns Ngspice transient co-simulation (`.tran`) records and verifies the cycle-by-cycle remainder convergence on stacked voltage plots.

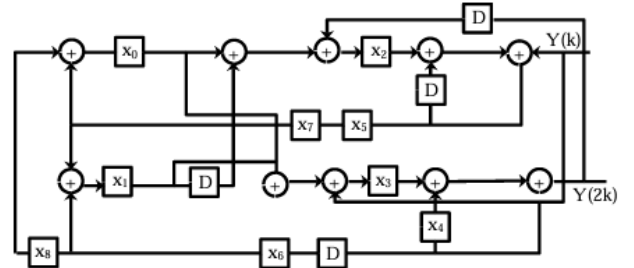


Fig. 1. Circuit diagram of the proposed project

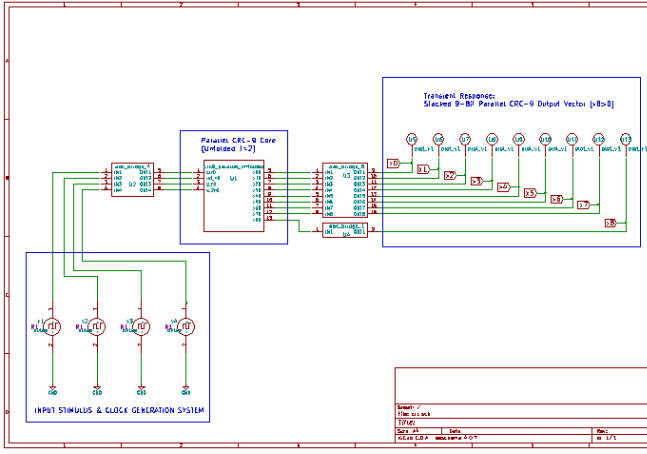


Fig. 2. schematic implemented in eSim

IV. VERILOG DESCRIPTION

```

`timescale 1ns / 1ps
module crc9_parallel_unfolded (
    input wire clk,
    input wire rst_n,
    input wire y_k,
    input wire y_2k,
    output reg x0,
    output reg x1,
    output reg x2,
    output reg x3,
    output reg x4,
    output reg x5,
    output reg x6,
    output reg x7,
    output reg x8
);

    reg next_x0;
    reg next_x1;
    reg next_x2;
    reg next_x3;
    reg next_x4;
    reg next_x5;
    reg next_x6;
    reg next_x7;
    reg next_x8;

    always @(*) begin
        next_x0 = x7 ^ y_k;
        next_x1 = x8 ^ x7 ^ y_k ^ y_2k;
        next_x2 = x0;
        next_x3 = x1;
        next_x4 = x2;
        next_x5 = x3;
        next_x6 = x4;
        next_x7 = x5;
        next_x8 = x6 ^ x7 ^ y_k;
    end

    always @(posedge clk or negedge rst_n) begin
        if (!rst_n) begin
            x0 <= 1'b0;
            x1 <= 1'b0;
            x2 <= 1'b0;
            x3 <= 1'b0;
            x4 <= 1'b0;
            x5 <= 1'b0;
            x6 <= 1'b0;
            x7 <= 1'b0;

```

```

            x8 <= 1'b0;
        end else begin
            x0 <= next_x0;
            x1 <= next_x1;
            x2 <= next_x2;
            x3 <= next_x3;
            x4 <= next_x4;
            x5 <= next_x5;
            x6 <= next_x6;
            x7 <= next_x7;
            x8 <= next_x8;
        end
    end
endmodule

```

V. TEST CASES AND VERIFICATION

To validate the parallel CRC-9 architecture, a 10-bit test frame (1100000011) is split into two synchronized bit-streams: $Y(k) = [1, 0, 0, 0, 1]$ and $Y(2k) = [1, 0, 0, 0, 1]$ at a 20 ns clock period (50 MHz).

TABLE I
CYCLE-BY-CYCLE VERIFICATION AND STATE TRANSITIONS

Cycle	Time Window	Inputs [Y_k, Y_{2k}]	Output Remainder [$x_8 \dots x_0$]	Hex Value	Active Rails (5 V)
Reset	0–20 ns	(0, 0)	9'b000000000	0x000	None
1	20–40 ns	(1, 1)	9'b000000001	0x001	x_0
2	40–60 ns	(0, 0)	9'b000000110	0x006	x_1, x_2
3	60–80 ns	(0, 0)	9'b000011000	0x018	x_3, x_4
4	80–100 ns	(0, 0)	9'b001100000	0x060	x_5, x_6
5	100–120 ns	(1, 1)	9'b010000001	0x081	x_0, x_7

- **Convergence:** At Cycle 5 (100–120 ns), only x_0 and x_7 step to 5 V, confirming exact convergence to 9'b010000001 (0x081).
- **Performance:** Reduces computation time from 9 cycles (serial) to **5 cycles**, achieving a **44.4% latency reduction** with stable mixed-signal transients.

VI. RESULTS AND DISCUSSION

The 120 ns Ngspice transient simulation in eSim confirms correct functionality of the 2-level unfolded parallel CRC-9 generator at 50 MHz (20 ns period):

- **Waveform Verification:** Stacked voltage traces ($v(x_0)$ to $v(x_8)$) with 5 V vertical offsets show clean transitions across all 5 clock cycles following reset release at 20 ns.
- **Exact Remainder Convergence:** In the final cycle (100–120 ns), only $v(x_0)$ and $v(x_7)$ assert high (5 V), producing the exact remainder vector 9'b010000001 (0x081).
- **Performance Gain:** Processing two bits per cycle reduces total computation from 9 cycles to **5 cycles**, delivering a **44.4% latency reduction** over the serial baseline.
- **Electrical Stability:** The ADC/DAC interfaces demonstrate sharp 0–5 V switching with zero metastable states or glitches.

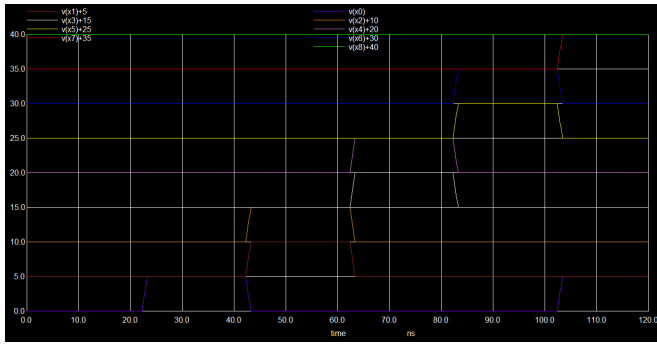


Fig. 3. Transient Output Waveform of Stacked 9-Bit Parallel CRC-9 Remainder Vector

```
ngspice -p C:\esim-workspace\crc\crc.cir.out
x6=0
x7=1
x8=0

Inside foo after eval....
c[k]=1
rst_n=1
y_k=1
y_2k=1
x0=1
x1=0
x2=0
x3=0
x4=0
x5=0
x6=0
x7=1
x8=0

=====crc9_parallel_unfolded : New Iteration=====
Instance : 0

Inside foo before eval....
c[k]=1
rst_n=1
```

Fig. 4. Ngspice / NgVeri Co-Simulation Terminal Log Confirming State Evaluation and Final Remainder Convergence ($x_0=1$, $x_7=1$)

VII. CONCLUSION

A high-throughput 2-level unfolded parallel CRC-9 architecture with pipelining and retiming was successfully designed and verified in eSim. The implementation reduced critical path delay to $1T_{\text{XOR}}$ and processed a 10-bit input frame in **5 clock cycles** instead of 9, achieving a **44.4% latency reduction** over serial architectures. NgVeri–Ngspice co-simulation verified exact remainder convergence to $9'b010000001$ ($0x081$), confirming the suitability of parallel CRC structures for high-speed digital communications using open-source EDA tools.

REFERENCES

- [1] S. Singh, S. Sujana, I. Babu, and K. Latha, "VLSI Implementation of Parallel CRC Using Pipelining, Unfolding and Retiming," *IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)*, vol. 2, no. 5, pp. 66–72, May–Jun. 2013, doi: 10.9790/4200-0256672. Available: https://www.researchgate.net/publication/271250418_VLSI_Implementation_of_Parallel_CRC_Using_Pipelining_Unfolding_and_Retiming
- [2] FOSSEE, IIT Bombay, "eSim: An Open Source EDA Tool for Circuit Design, Simulation, Analysis and PCB Design," [Online]. Available: <https://esim.fossee.in/>
- [3] S. Icarus, "Icarus Verilog Simulation and Synthesis Engine," [Online]. Available: <https://steveicarus.github.io/iverilog/>