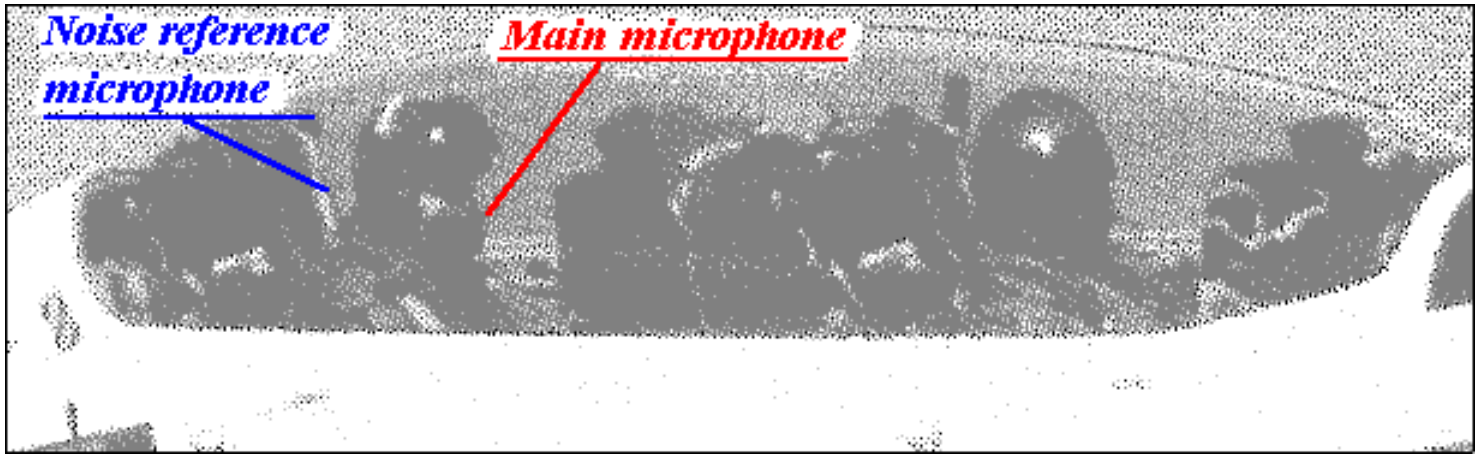
Adaptive Noise Cancellation using the LMS Algorithm

December 7, 1999

GROUP Y
Yonghui Cheng
Damian Dobric
Ping Tao
Shunxi Wang

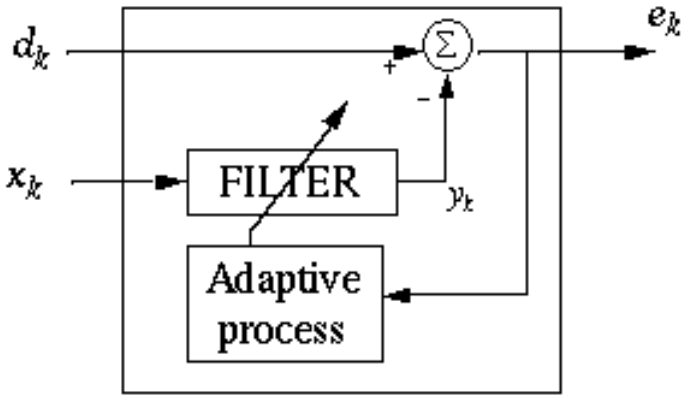
Functional Description

- Adaptive Noise Cancellation DSP Circuit
  - Removes background noise from signals
- Application: Pilot Communication
  - Two signals:
    - Noise + Pilot's Speech
    - Noise Reference Signal in Cockpit



Functional Description (cont'd)

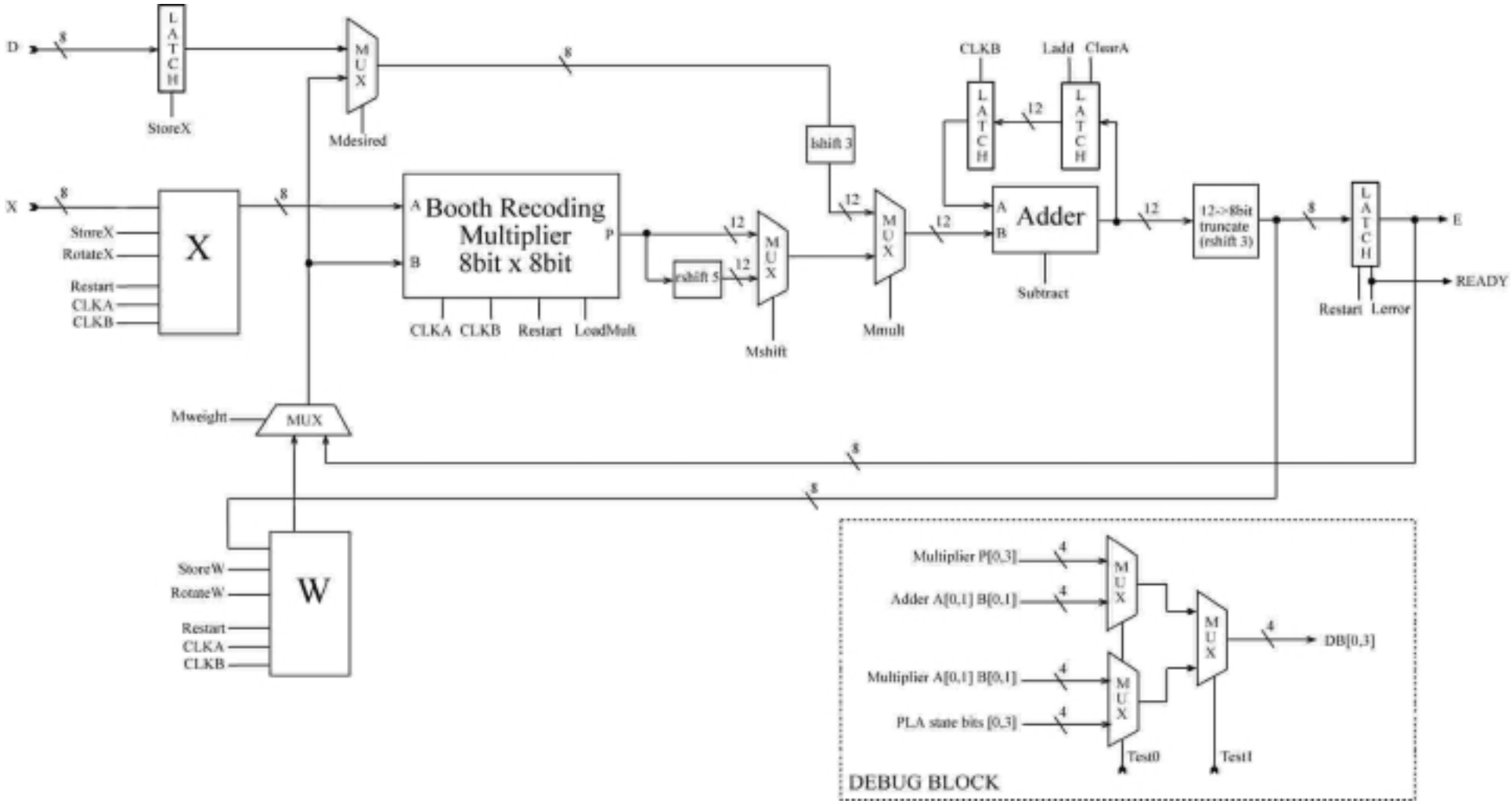
- Filter input x(k) to obtain y(k)
- Error e(k) = Desired Input d(k) – y(k)
  - Used to update filter frequency response
- 8-tap Finite Impulse Response (FIR) Filter
- LMS Algorithm
  - W(k+1) = W(k) + u*2e(k)*x(k)
  - Fixed Step-Size u



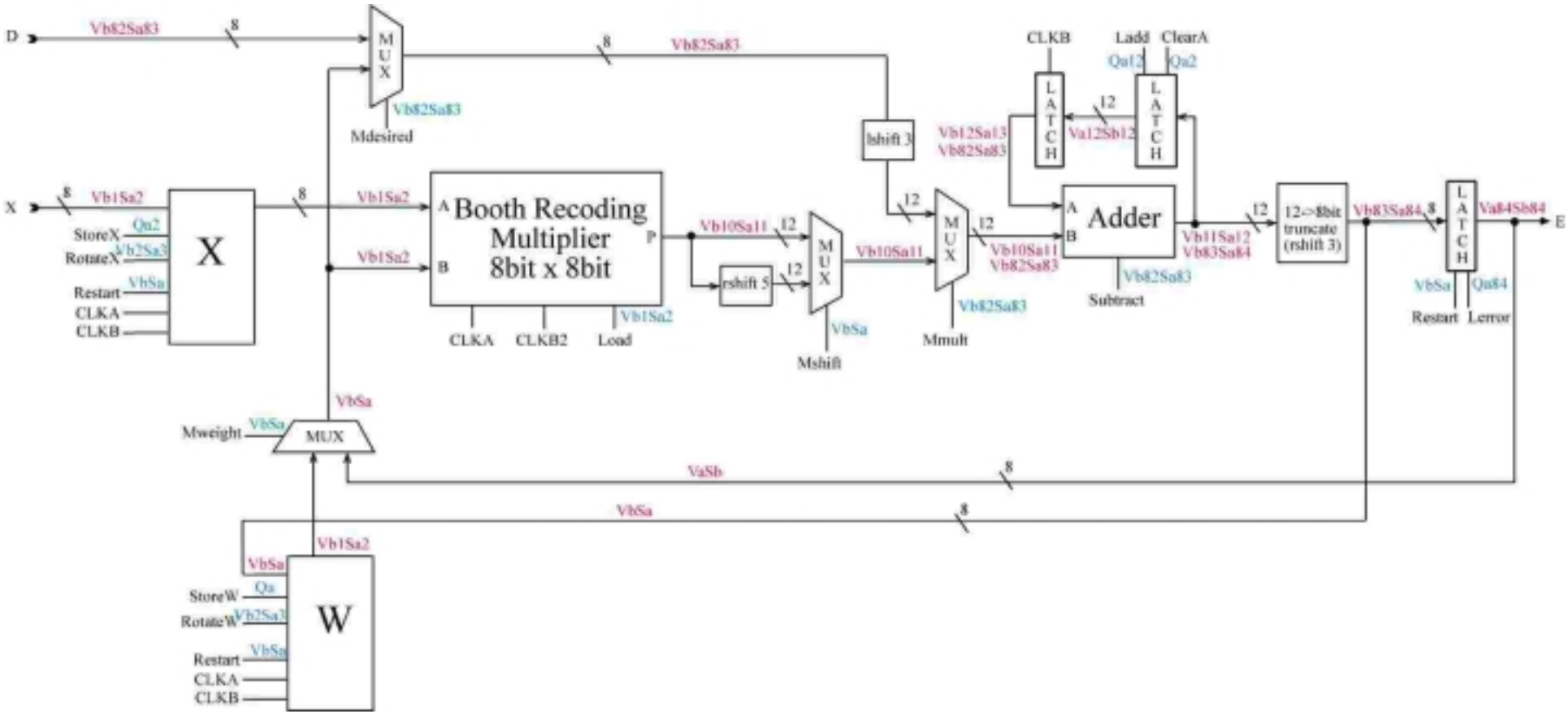
Overview

- Circuit Design
- PLA Description
- Layout
- System Performance Analysis

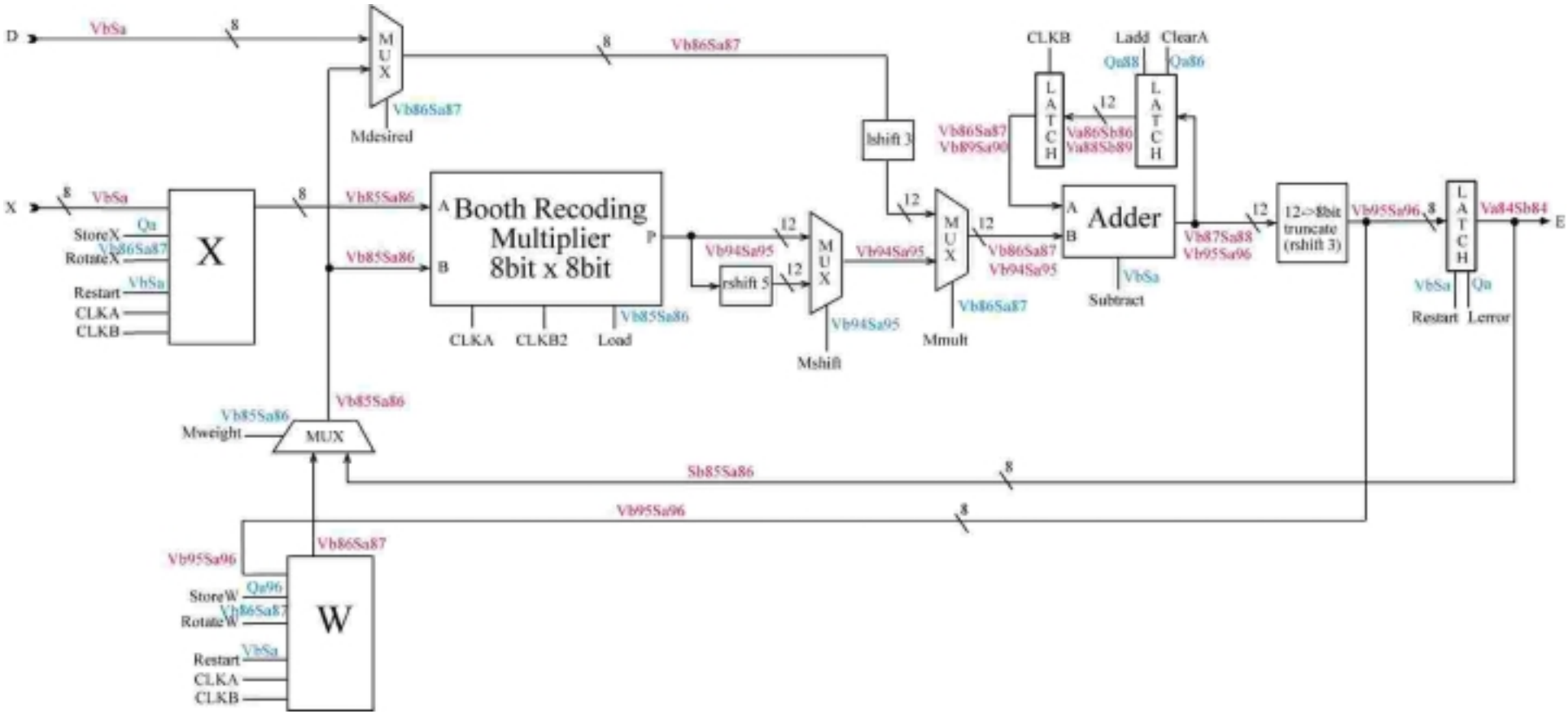
System Block Diagram



System Timing: Error Calculation



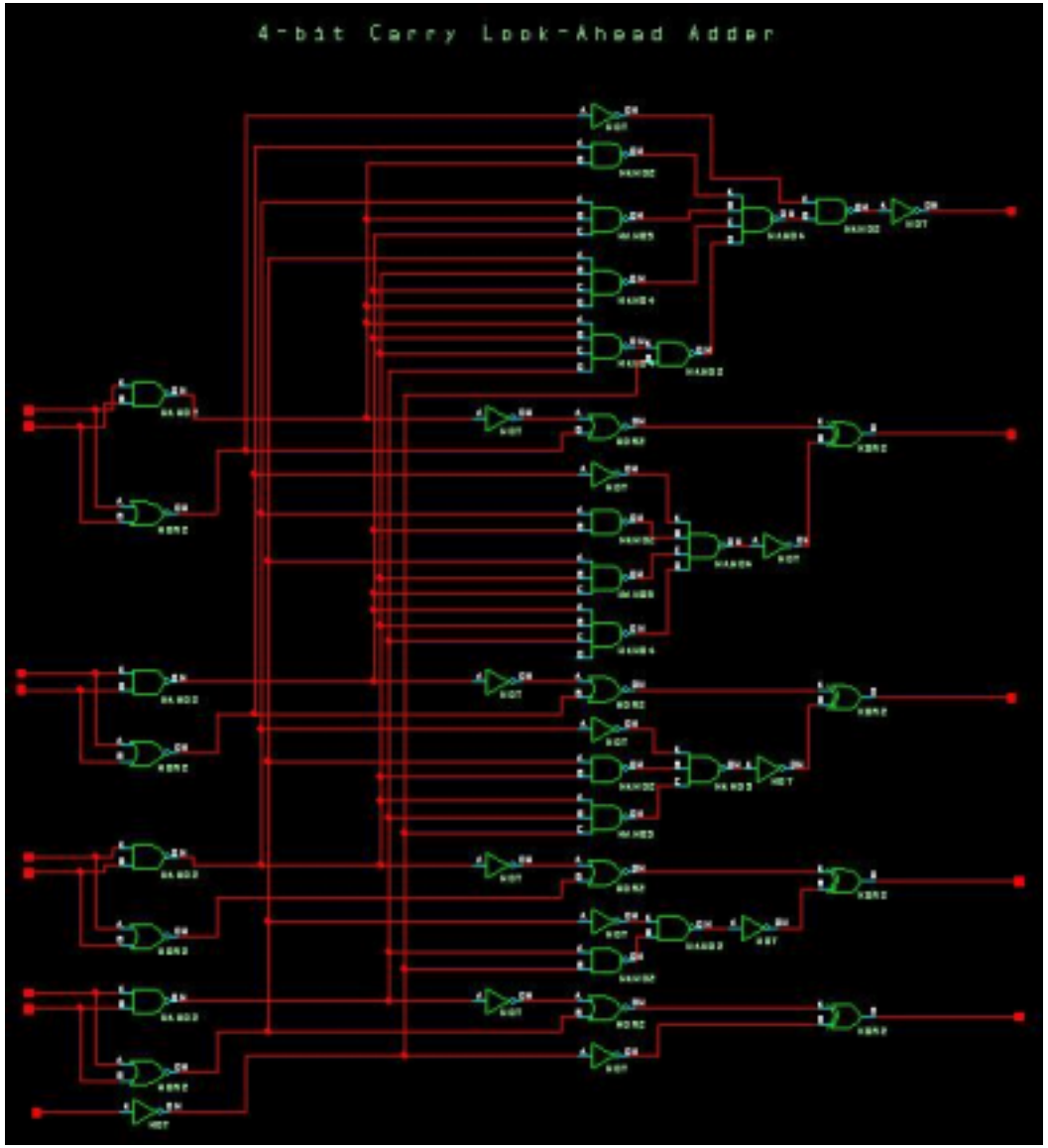
System Timing: Weight Update



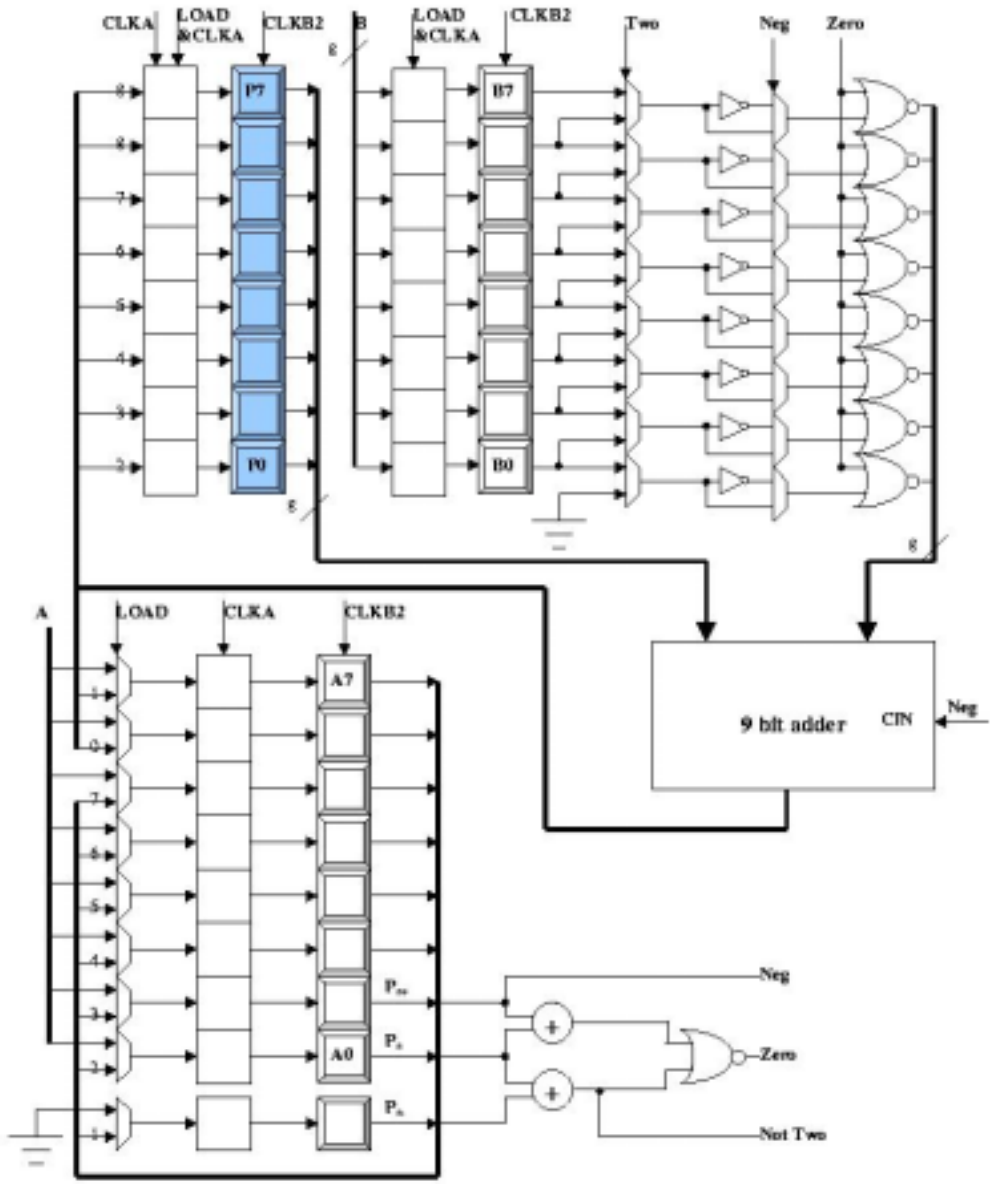
Circuit Design

- 12-bit Carry-Look Ahead Adder
- 8-bit Radix-4 Booth Recoding Multiplier
- 8-bit Rotate Stack

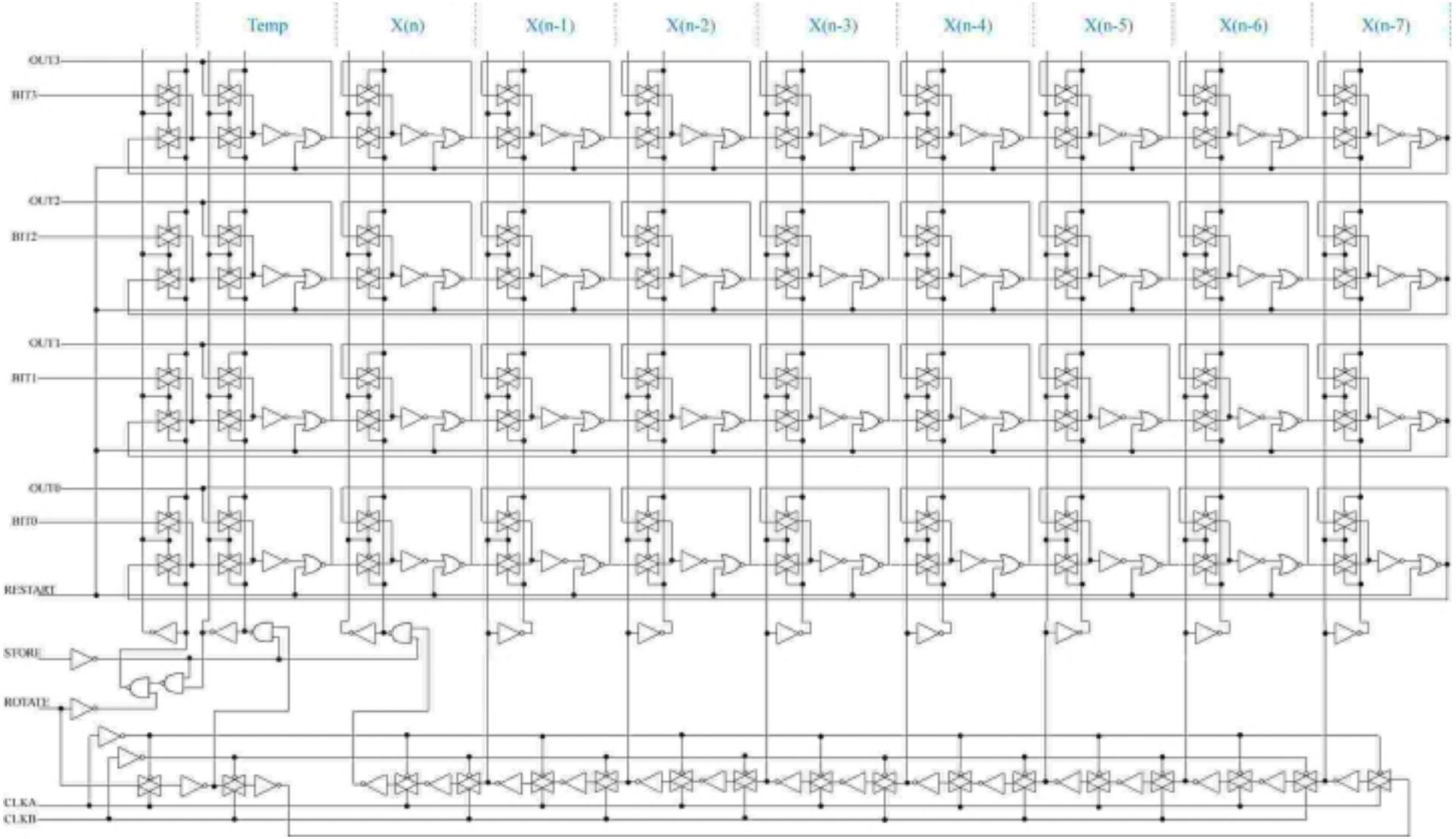
Carry-Look Ahead Adder



Radix-4 Booth Recoding Multiplier



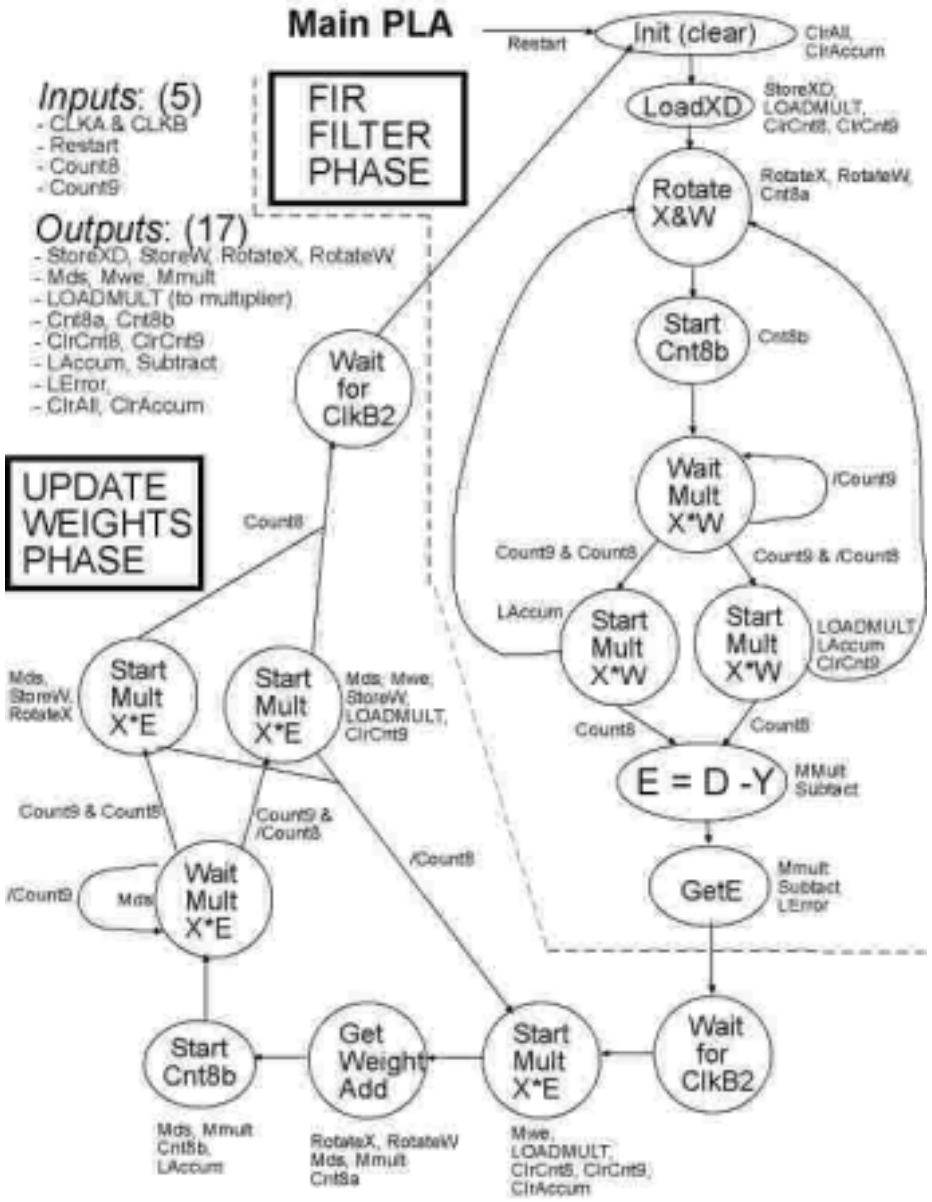
Rotate Stack



PLA Description

- 15 states, 5 inputs, 17 outputs
- 2 Phases:
  - Error Computation
  - Filter Tap Weight Update
- 2 Counters (count to 8):
  - Counter to count multiplier clock cycles
  - Counter to count # of multiply operations
- One Iteration ~200 clock cycles

State Diagram



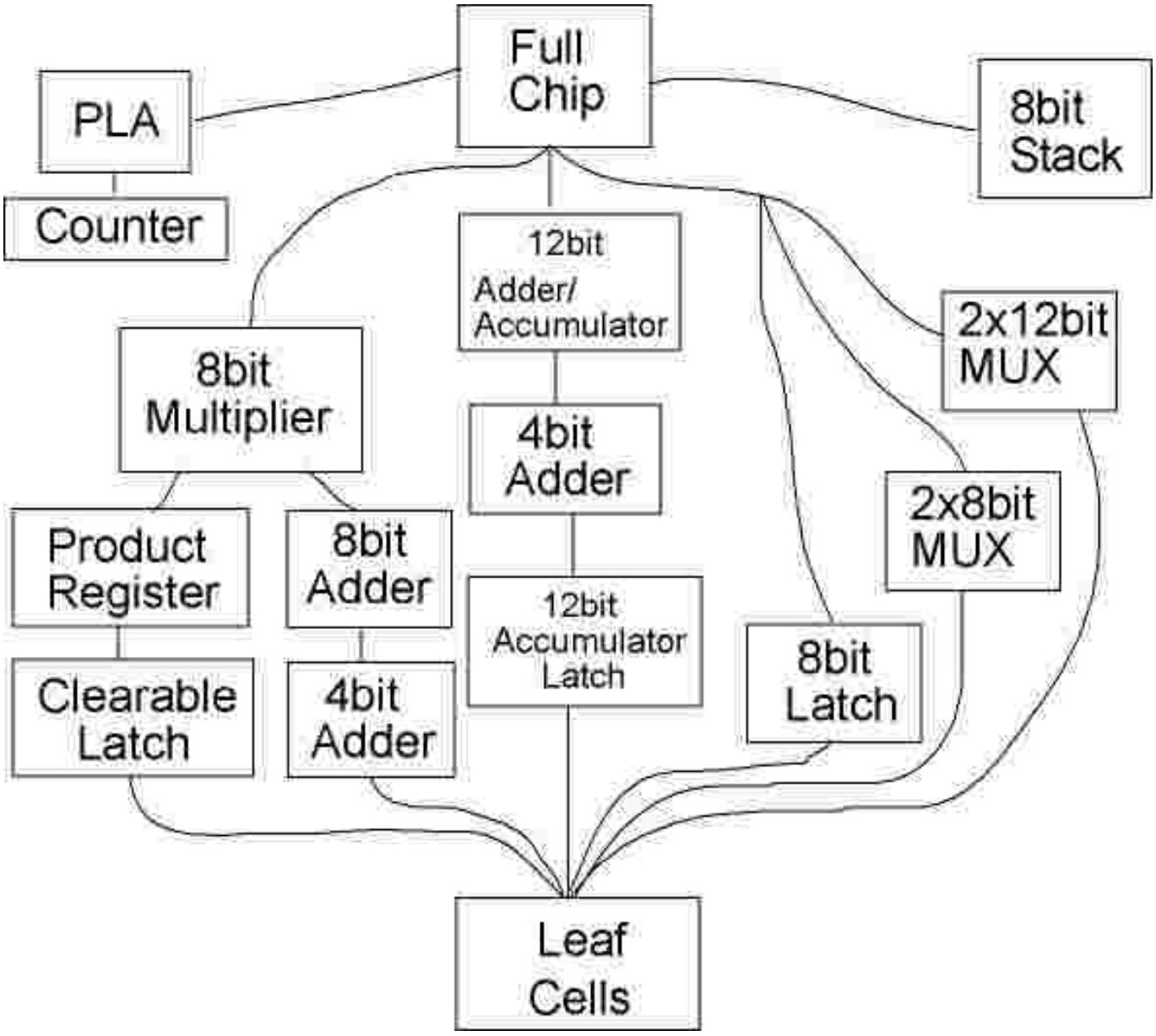
Input to Meg


```
-- Elec 422 Project MAIN CONTROLLER PLA

INPUTS:   RESTART COUNT8 COUNT9;

OUTPUTS:  STOREXD STOREW ROTATEX ROTATEW MDS MWE
        MMULT LOADMULT CLRCNT8 CLRCNT9 CNT8A CNT8B
        SUBTRACT LACCUM LERROR CLRALL CLRACCUM;

-- reset logic
reset on RESTART to init(CLRALL CLRACCUM);

init:   -- initial state --> clear all latches
        goto LoadXD(STOREXD LOADMULT CLRCNT8 CLRCNT9);

LoadXD:  -- Load X & D inputs
        goto RotateXW(ROTATEX ROTATEW CNT8A);

RotateXW:  -- Rotate X & W stacks
        goto StartCNT8b(CNT8B);

StartCNT8b: -- signal 8 counter w/ second control signal
        goto WaitMultXWDone;

WaitMultXWDone: -- wait for the multiply of X & W to finish
        case (COUNT9 COUNT8)
        10 => StartNextMultXW(LOADMULT LACCUM CLRCNT9);
        11 => StartNextMultXW(LACCUM);
        endcase => WaitMultXWDone;

StartNextMultXW:
        if COUNT8 then CalculateE(MMULT SUBTRACT)
        else RotateXW(ROTATEX ROTATEW CNT8A);

CalculateE:
        goto GetE(MMULT SUBTRACT LERROR);

GetE:
        goto WaitBeforeMult;  -- CLKB2 happens on EVEN cycles

WaitBeforeMult:
        goto StartMultXE(MWE LOADMULT CLRCNT8 CLRCNT9
        CLRACCUM);

StartMultXE:
        goto GetWAdd(ROTATEX ROTATEW MDS MMULT CNT8A);

GetWAdd:
        goto StartCNT8b2(MDS MMULT CNT8B LACCUM);

StartCNT8b2:
        goto WaitMultXEDone(MDS);

WaitMultXEDone:
        case (COUNT9 COUNT8)
        10 => StartNextMultXE(MDS MWE STOREW LOADMULT
        CLRCNT9);
        11 => StartNextMultXE(MDS STOREW ROTATEX);
        endcase => WaitMultXEDone(MDS);

StartNextMultXE:
        if COUNT8 then WaitBeforeMult2  -- CLKB2 happens on EVEN
        cycles
        else StartMultXE(CLRACCUM);

WaitBeforeMult2:
        goto init(CLRALL CLRACCUM);
```

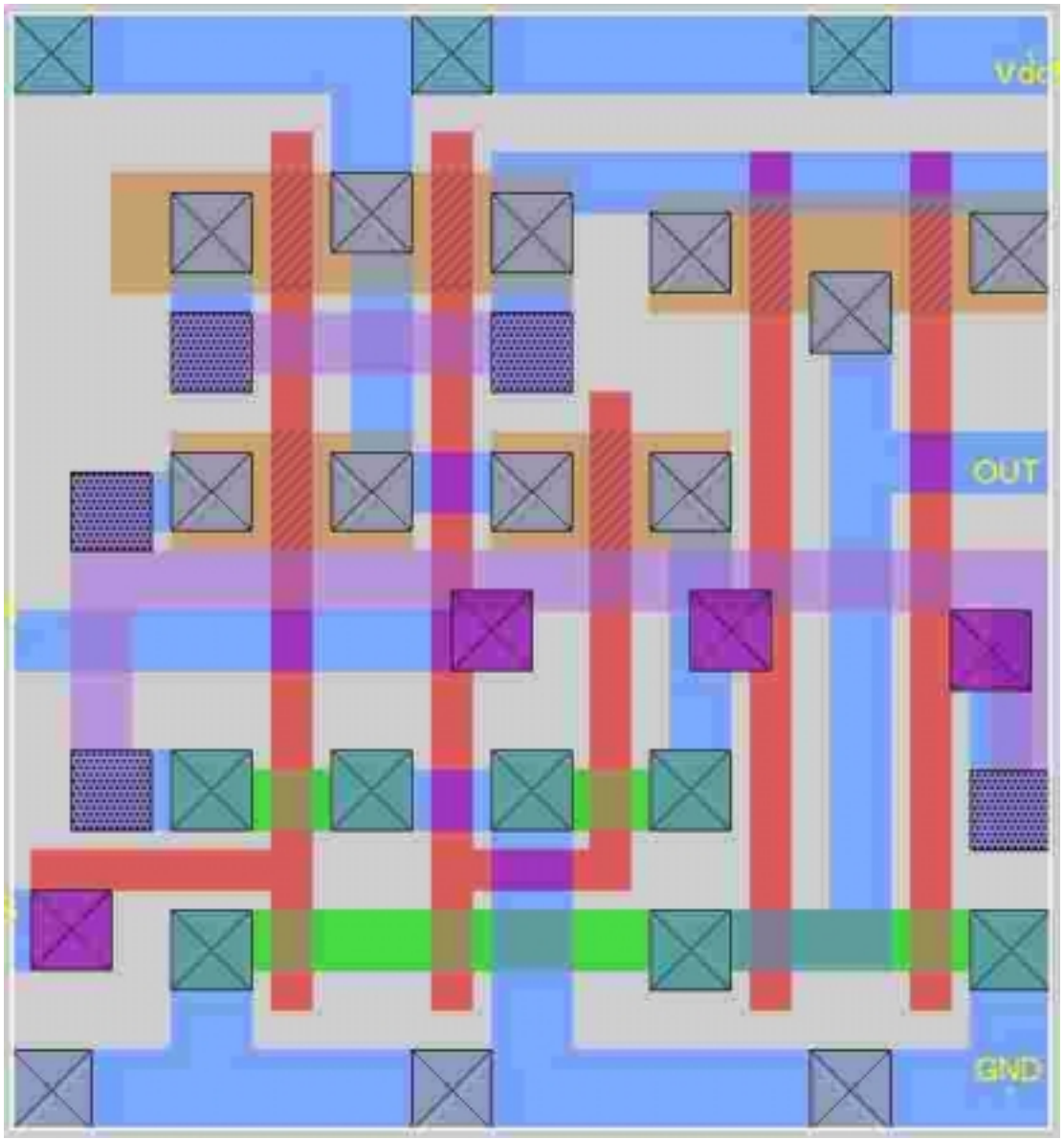
Layout

- Cell Hierarchy
- Plots of Low-Level Cells
- Floorplan
- Full plot of Chip

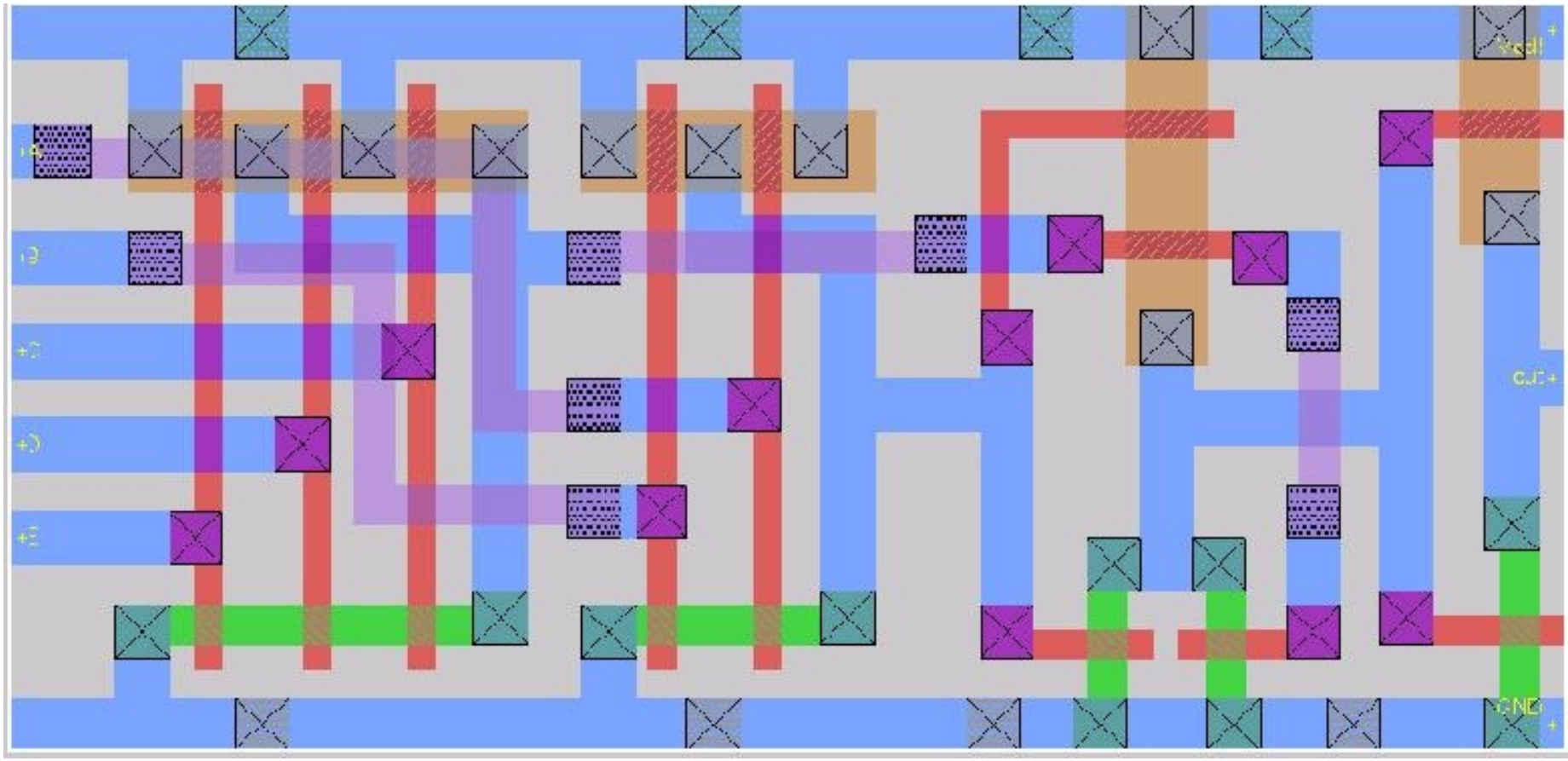
Cell Hierarchy



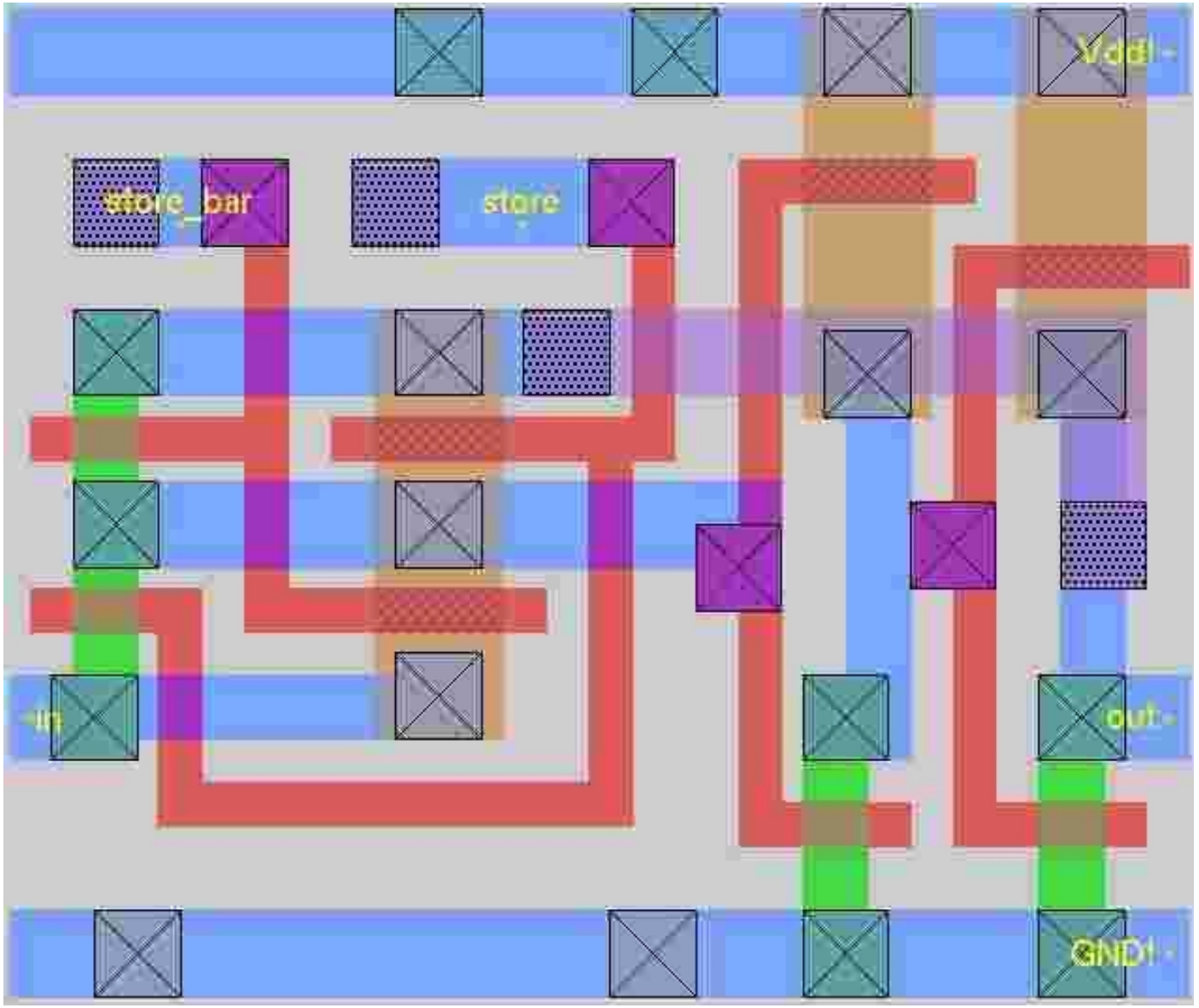
XOR Gate Cell



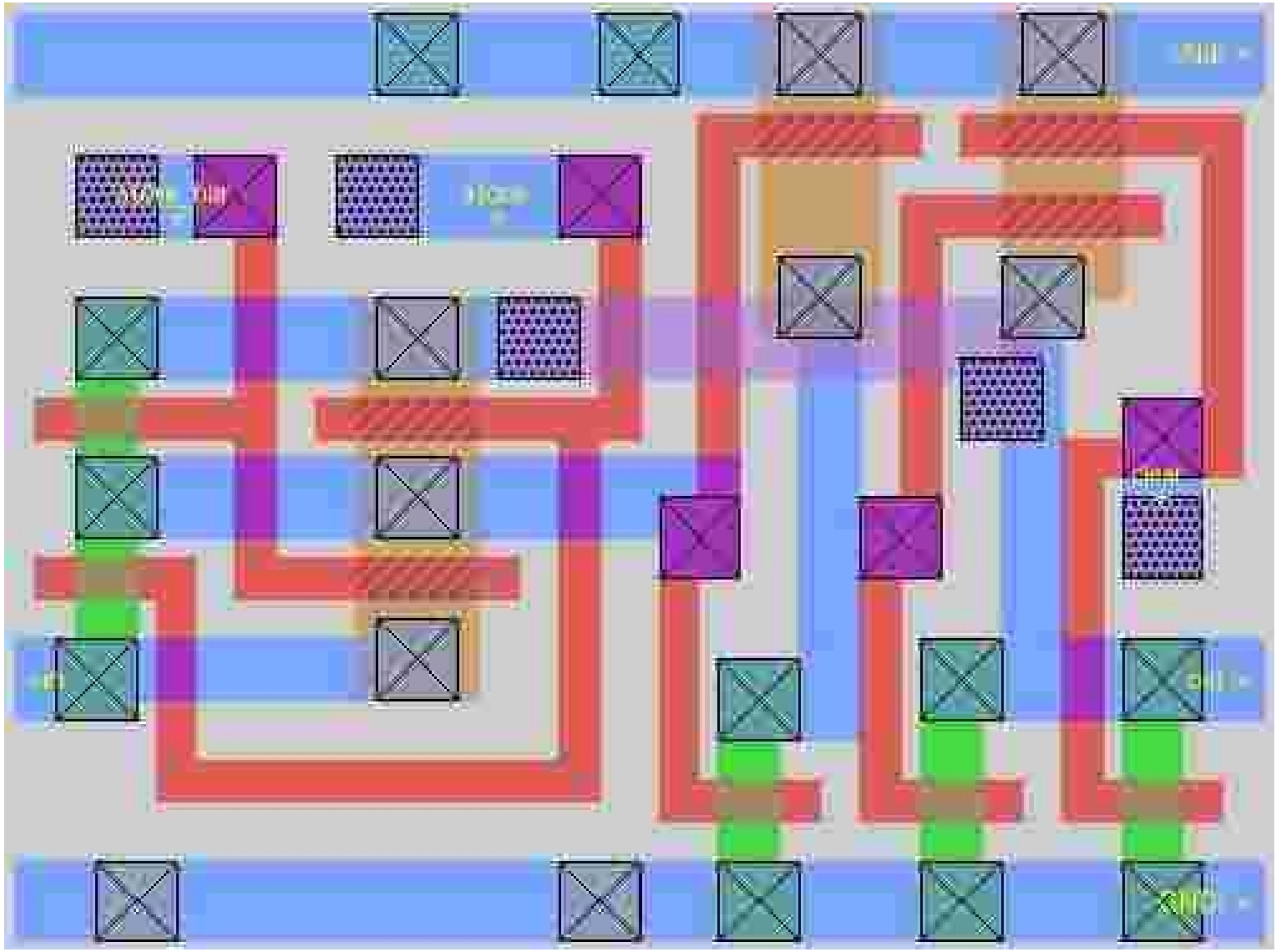
5-Input NAND Gate Cell



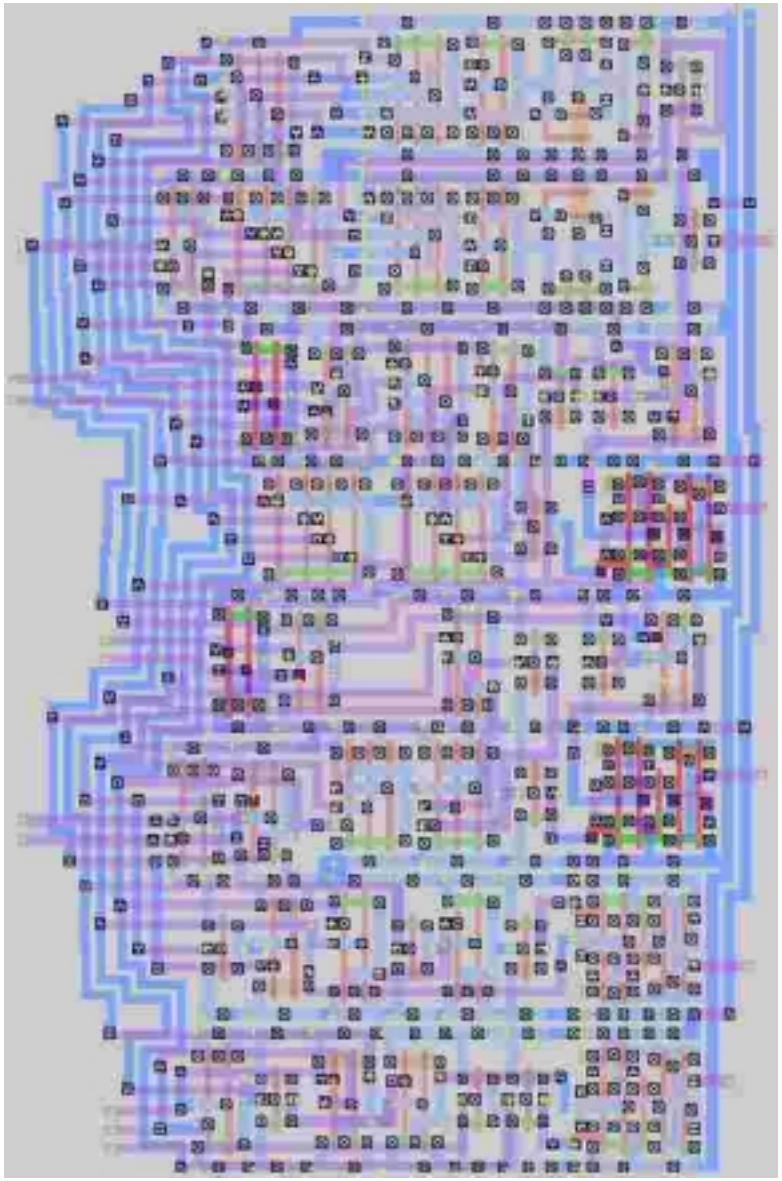
Static Latch



Clearable Static Latch

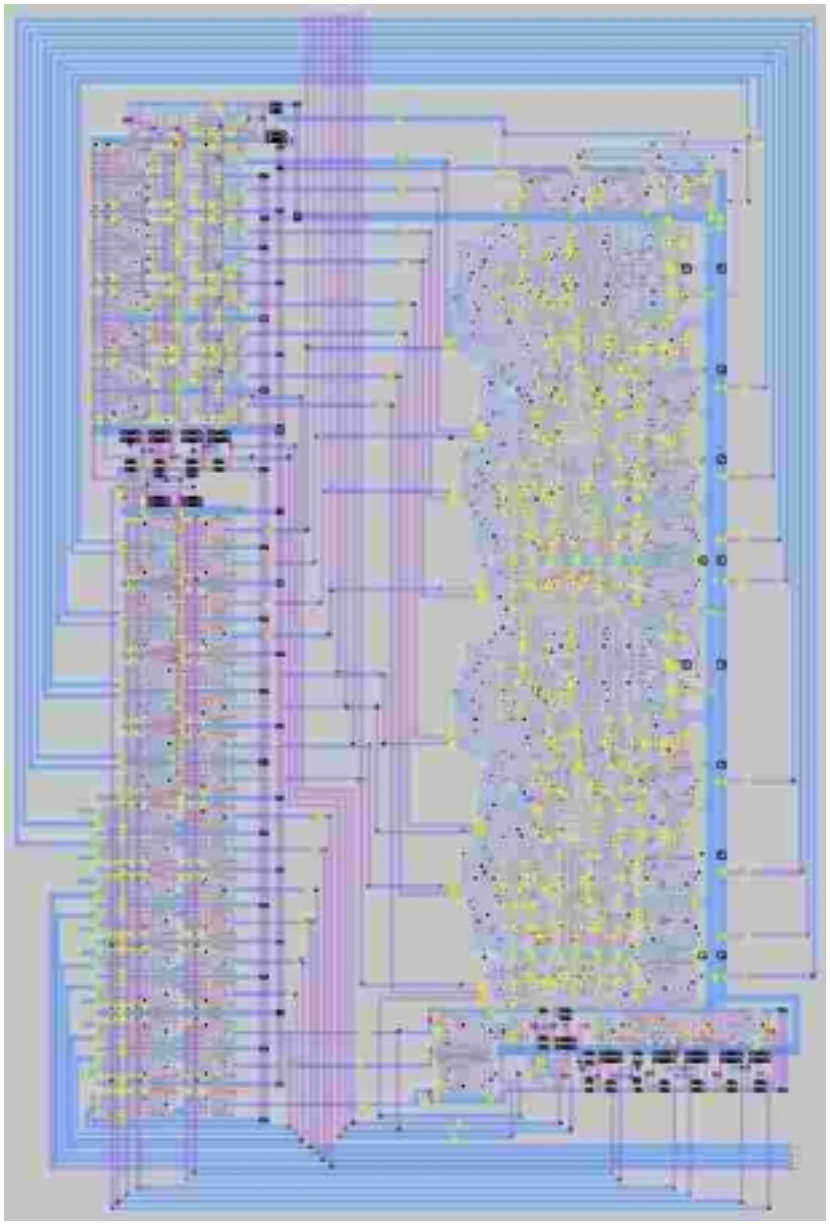


4-bit Carry-Look Ahead Adder

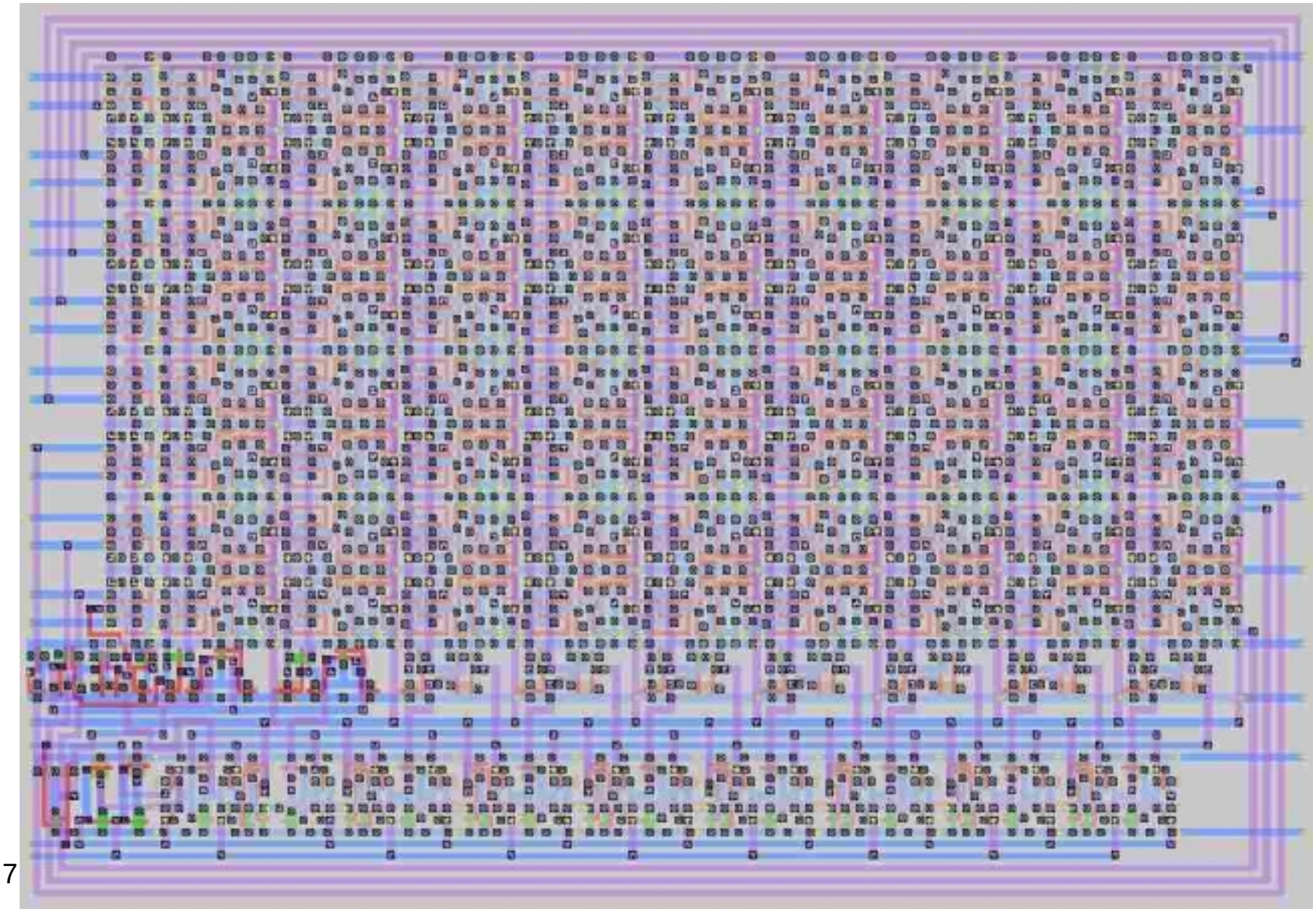


12-bit Carry-Look Ahead Adder

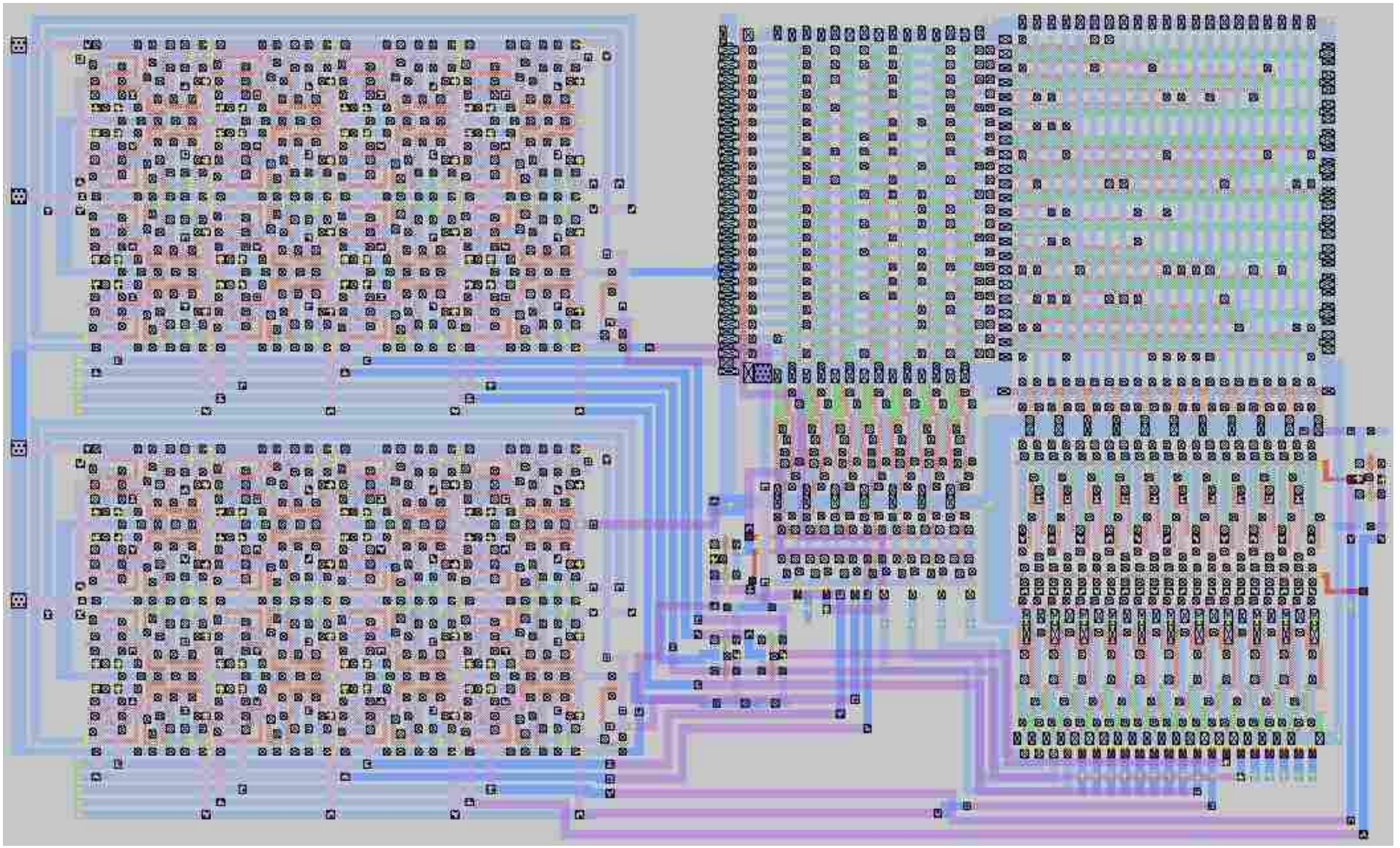
8-bit Radix-4 Booth Recoding Mult.



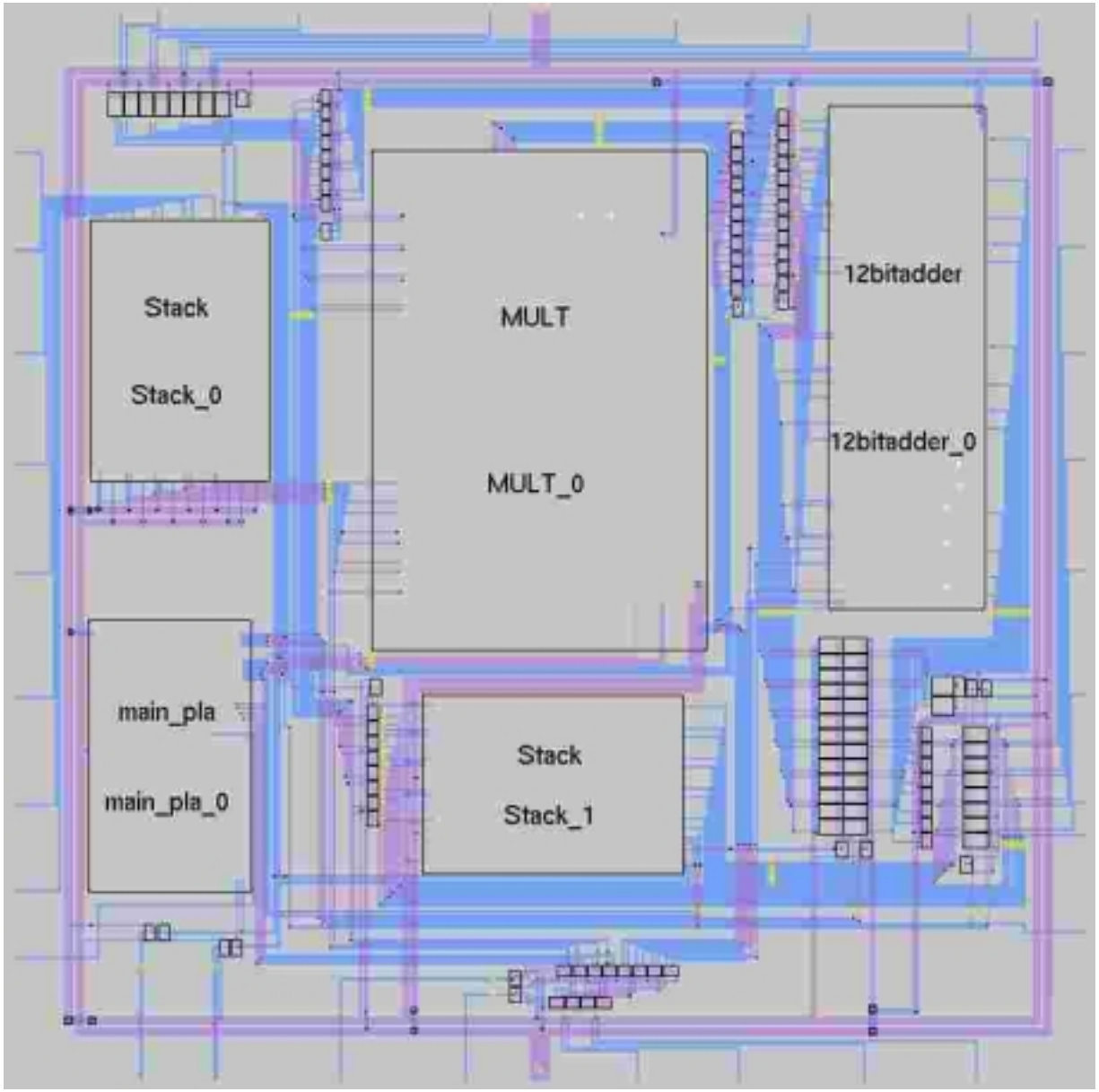
8x8-bit Rotate Stack



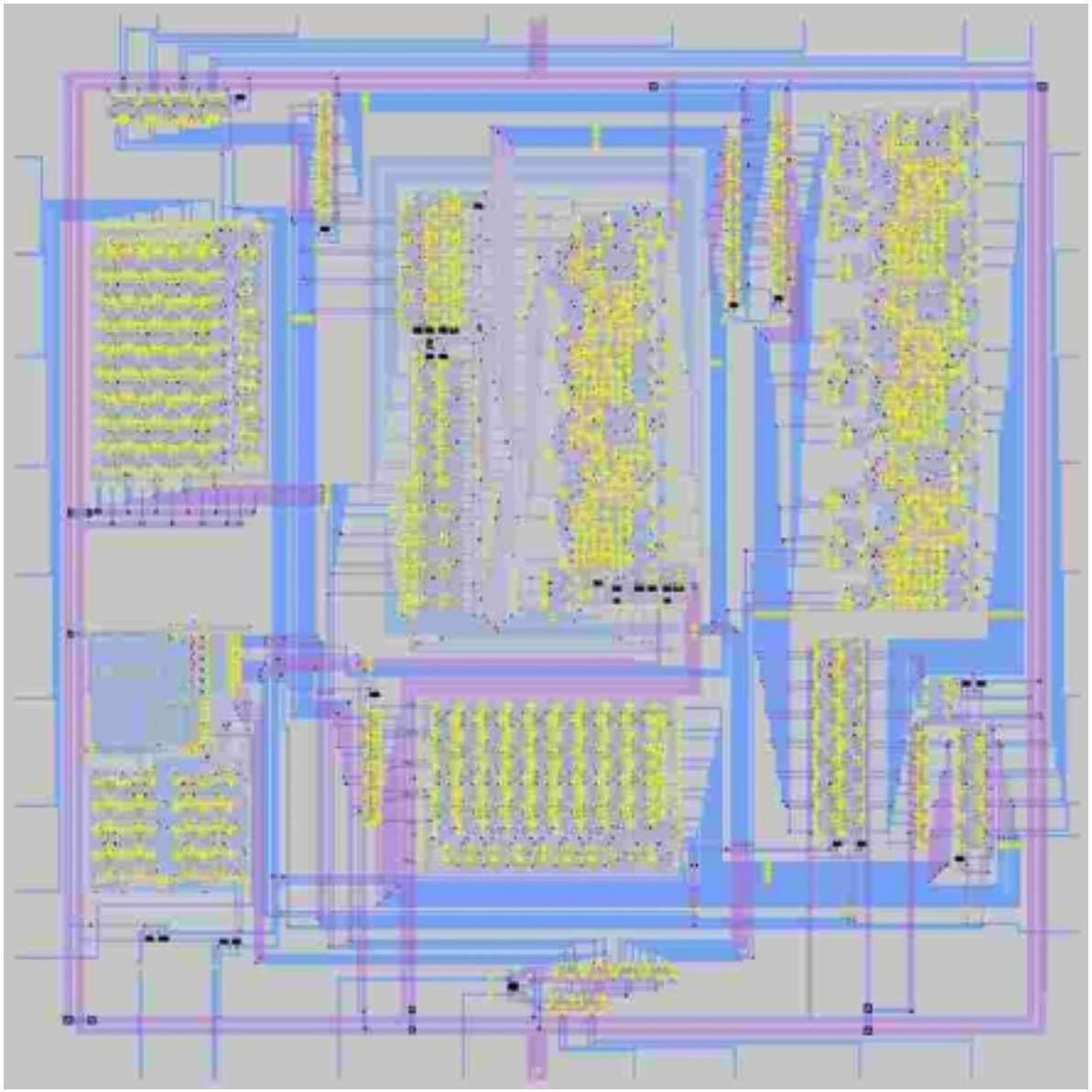
Control PLA & Counters



Floorplan


Stack
Stack_0
MULT
MULT_0
12bitadder
12bitadder_0
main_pla
main_pla_0
Stack
Stack_1

Full Plot of Chip

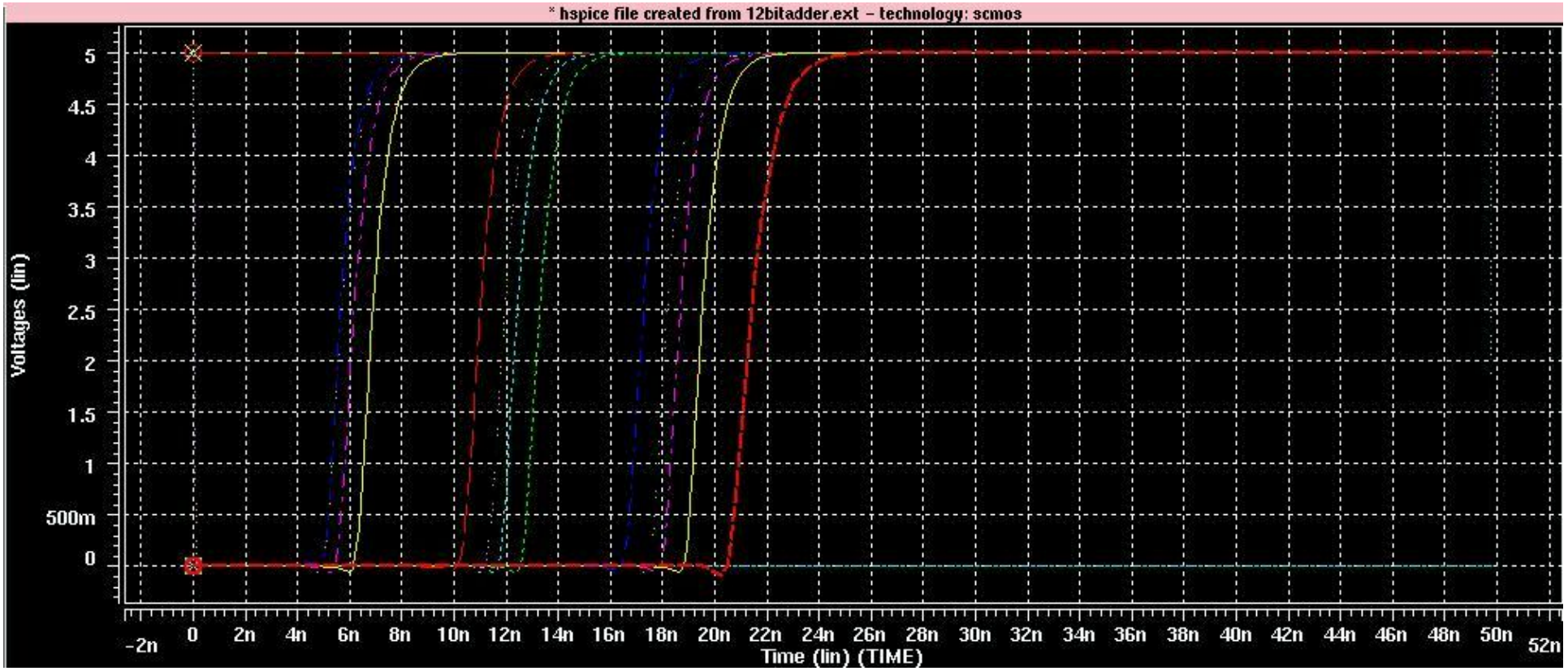


- 5,600T Total

System Performance Analysis

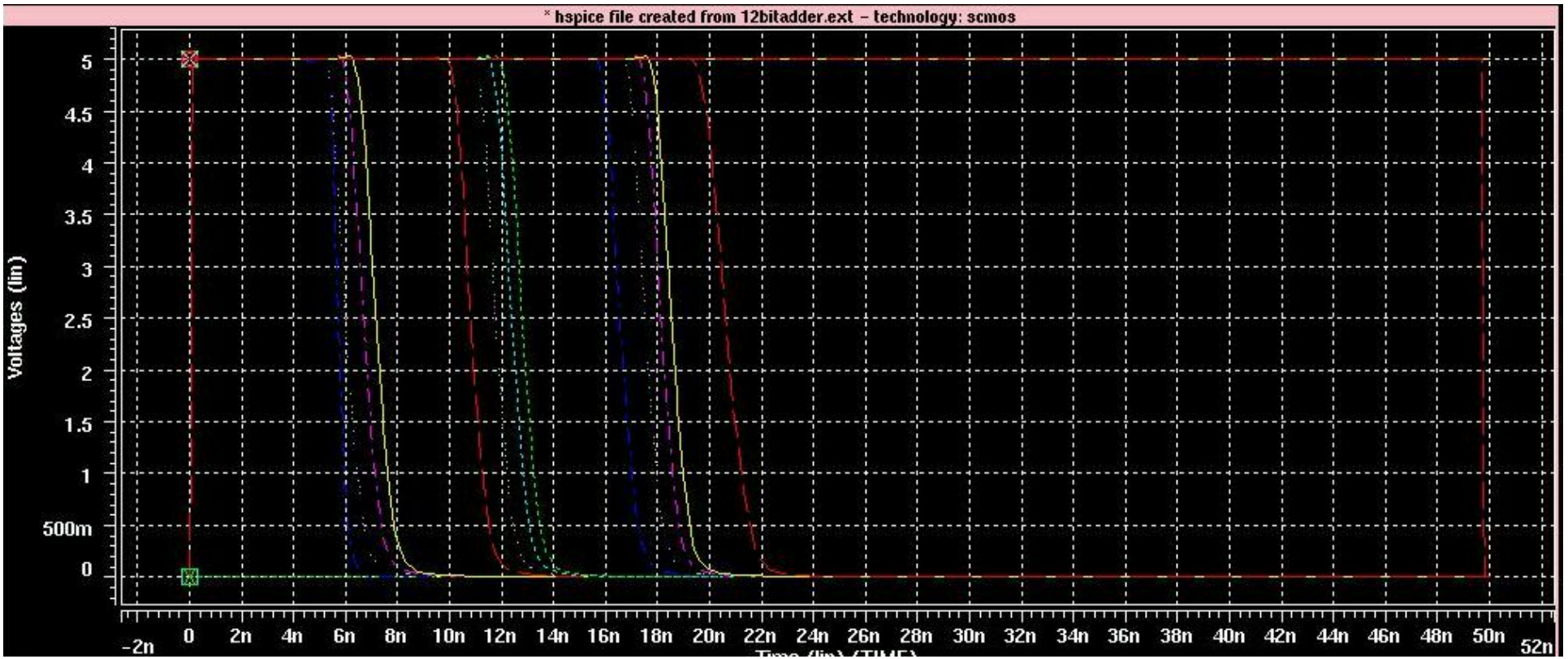
- Longest Path: Multiplier Output to Error Output Latch
  - 12-bit Adder in critical path
- Spice Analysis of 12-bit Adder

Spice Analysis (Rise Time)



- Rise Time ~23.5 ns

Spice Analysis (Fall Time)



- Fall Time ~22 ns

Summary of System Performance

- 12-bit Adder Worst Case Delay: ~25 ns
- Maximum Clock Speed: ~42.6 MHz