HLS DUTh Lab

ΣΑΜΟΛΑΔΑΣ ΤΡΙΑΝΤΑΦΥΛΛΟΣ 57259

HLS DUTh Lab – Exercise 1

```
for (int i=0; i < 10000; i++)
    A[i] = B[i]*C[i] - D[i]*E[i];
```

1 operation(Add or Mul or Sub) OR 1 memory access **per clock**

For a 500MHz clock → **T = 2ns**
8 clocks per loop iteration + 1 clock for 'i' iterator incrementation("i++")

Throughput → Time needed to get all outputs
Latency → Time Needed for 'program' to finish

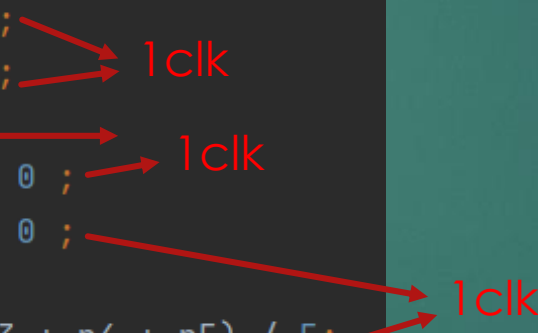- **Throughput Cycles**: 9 x 10.000 – 1 = 89.999 clks (without the last 'i' loop termination check)
- **Throughput Time**: 89.999 x 2 = 179.998 ns

- **Latency Cycles**: 9 x 10.000 = 90.000 clks
- **Latency Time**: 90.000 x 2 = 180.000 ns

HLS DUTh Lab – Exercise 2

```
// scan the image row by row
// N = #row, M = #col
short p1,p2,p3,p4,p5;
ROW:for (int i = 0; i < N; ++i) {
    // scan each row pixel by pixel from left to right
    COL:for (int j = 0; j <M; ++j) {
        // get values of the pixels in the kernel
        p1 = (j > 1) ? img[i][j-2]: 0;
        p2 = (j > 0) ? img[i][j-1]: 0;           1clk
        p3 = img[i][j];
        p4 = (j < M-1) ? img[i][j+1]: 0 ;        1clk
        p5 = (j < M-2) ? img[i][j+2]: 0 ;
        // compute the mean
        out_orig[i][j] = (p1 + p2 + p3 + p4 + p5) / 5;   1clk
    }
}
```

Ερώτημα Α

Δεδομένου ότι έχουμε dual port μνήμες(2 προσβάσεις μνήμης ανά κύκλο ρολογιού), κάθε επανάληψη της COL, εκτελείται σε 3 κύκλους ρολογιού.

Όμως κατά το pipeline της COL, οι εξαρτήσεις μεταξύ των μεταβλητών δεν επιτρέπουν την μείωση του Initiation Interval κάτω από **II=3**.

Για να υπολογιστεί το out_orig[i][j] πρέπει να έχουν προϋπολογιστεί τα p1,p2(1ος κύκλος) ,p3,p4(2ος κύκλος) και το p5(3ος κύκλος)

HLS DUTh Lab

```
void CCS_BLOCK(with_queue)(short (&img)[N][M], short (&out_Q)[N][M]){
    short sum;
    short myqueue[5];
    short front = 0;  //for queue functionality
    short rear = 0;
    ROW:for (int i = 0; i < N; ++i) {
        //init new queue
        myqueue[rear] = 0;
        rear++;
        myqueue[rear] = 0;
        rear++;
        myqueue[rear] = img[i][0];
        rear++;                      //  2 mem accesses for pre-processing(1 clk)
        //init sum variable
        sum = myqueue[rear - 1];
        myqueue[rear] = img[i][1];
        rear++;
        sum += myqueue[rear - 1];
        COL:for (int j = 2; j < M; ++j) {
            //push new element in
            myqueue[rear] = img[i][j];      //  2 mem accesses for main process(1 clk)
            rear++;
            sum += myqueue[rear - 1];
            //calculate cell in center
            out_Q[i][j-2] = sum / 5;
            //decrease sum by the oldest element and pop it out
            sum -= myqueue[front];
            //remove from queue
            myqueue[0] = myqueue[1];
            myqueue[1] = myqueue[2];
            myqueue[2] = myqueue[3];
            myqueue[3] = myqueue[4];
            myqueue[4] = 0;
            // decrement rear
            rear--;
        }
        out_Q[i][M-2] = sum / 5;         //  2 mem accesses for last two cols (1 clk)
        sum -= myqueue[front];
        out_Q[i][M-1] = sum / 5;
        // set queue ready for next row
        rear=0;
    }
}
```

Υλοποίηση ουράς (FIFO) με χρήση πίνακα 5 θέσεων.

Επαναχρησιμοποίηση δεδομένων:
- 2 προσβάσεις μνήμης(1 κύκλος ρολογιού) ανά επανάληψη της COL
- 4 προσβάσεις μνήμης(2 κύκλοι ρολογιού) ανά επανάληψη της ROW

HLS DUTh Lab – Exercise 2

Solution △	Latency Cycles	Latency Time	Throughput Cycles	Throughput Time	Total Area
original.v1 (allocate)	150096	300192.00	150102	300204.00	1913.52
original.v2 (allocate)	45497	90994.00	45502	91004.00	2338.44
with_queue.v1 (allocate)	74797	149594.00	74802	149604.00	2821.98
with_queue.v2 (allocate)	15798	31596.00	15802	31604.00	3442.28
with_queue.v3 (allocate)	14803	29606.00	14800	29600.00	4756.37

- **original.v1** → Original Proposed Version
- **original.v2** → Original Proposed Version with COL Loop Pipeline(II = 3)
- **with_queue.v1** → Queue Implementation
- **with_queue.v2** → Queue Implementation with COL Loop Pipeline(II = 1)
- **with_queue.v3** → Queue Implementation with main Loop Pipeline(II = 2) and COL Loop Partial Unroll(U = 2)

HLS DUTh Lab – Exercise 2
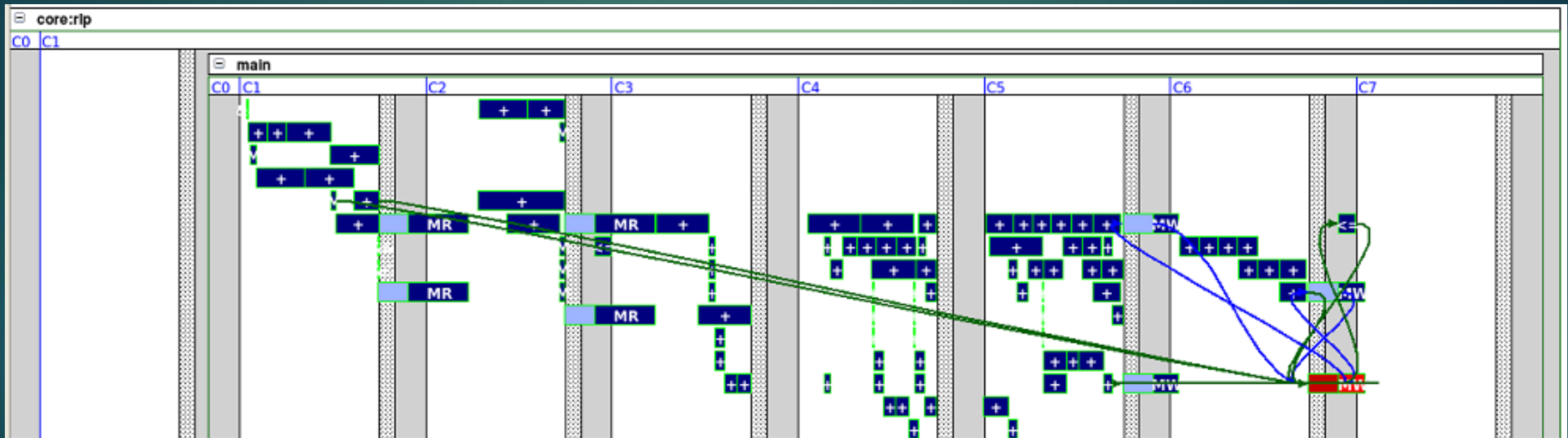
HLS DUTh Lab – Exercise 2

```
VSIM 2> run -all
# Creating Example Image...
# SCVerify intercepting C++ function 'with_queue' for RTL block 'with_queue'
# Info: HW reset: TLS_rst active @ 0 s
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Creating Example Image...
# Checking Complete...
# Info: Execution of user-supplied C++ testbench 'main()' has completed with exit code = 0
#
# Info: Collecting data completed
#    captured 10 values of img
#    captured 10 values of out_Q
# Info: scverify_top/user_tb: Simulation completed
#
# Checking results
# 'out_Q'
#    capture count        = 10
#    comparison count     = 10
#    ignore count         = 0
#    error count          = 0
#    stuck in dut fifo    = 0
#    stuck in golden fifo = 0
#
# Info: scverify_top/user_tb: Simulation PASSED @ 296016 ns
# ** Note: (vsim-6574) SystemC simulation stopped by user.
# 1
```

Testbench consists of 10 Example images randomly created and compared with original proposed code.

Simulation PASSED : 296016ns