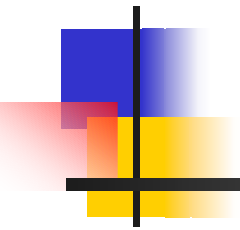


# CoreLoom

## A Semantic Model for Fine-Grained Flexible Datapaths

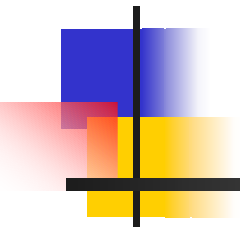
Michael Pellauer

Dependable Computing Systems

pellauer@cs.chalmers.se

# CoreLoom

## A Semantic Model for Fine-Grained Flexible Datapaths



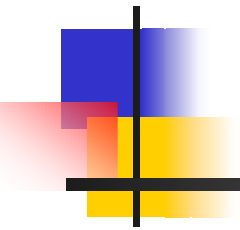
Michael Pellauer

Massachusetts Institute of Technology

(Starting Fall ’04)

# CoreLoom

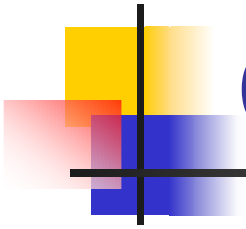
## A Semantic Model for Fine-Grained Flexible Datapaths



Michael Pellauer

Bluespec, Inc.

(Summer ’04)

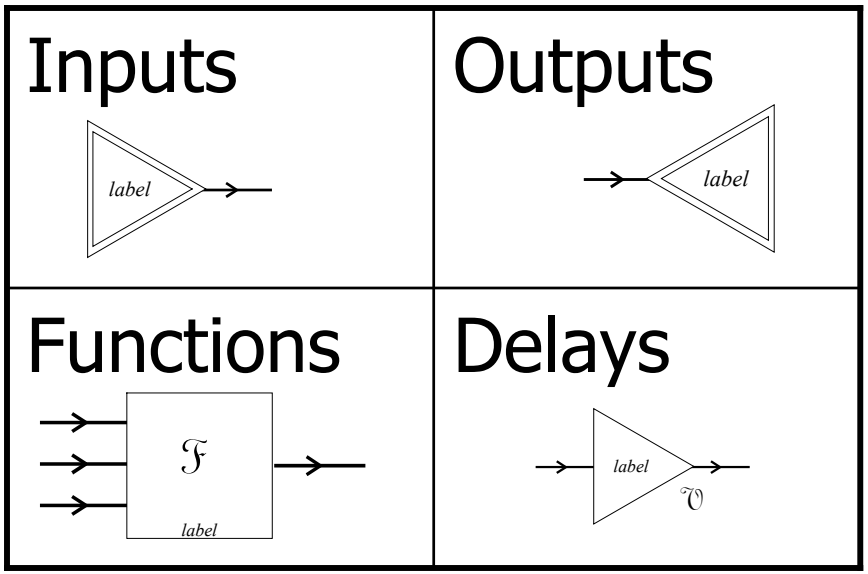


# CoreLoom Introduction

- Semantic Model for Flexible Datapaths
- Domain-Specific Embedded Language
- FlexSoC Project
- Emphasizes:
  - Design-Space exploration
  - Potential for Black-Box core representation
  - Implicit control units

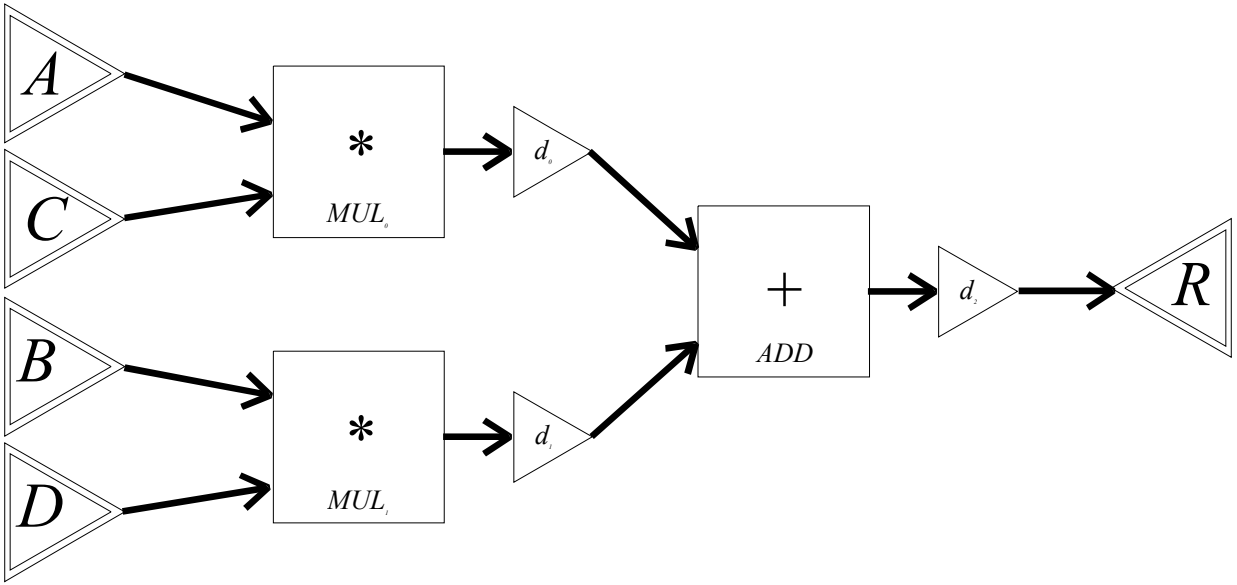
# Flexible Datapaths

- Datapath:
  - Data "walks" the path, ends CC in delay



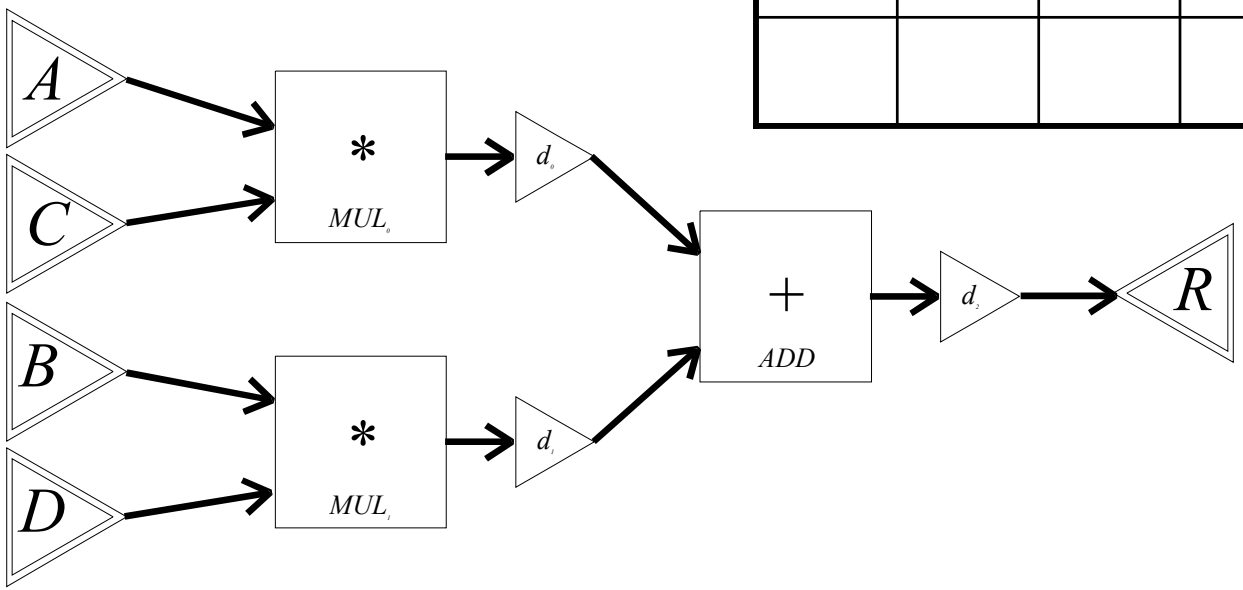
# Example

- Expresses R= [A B] • [C D] = AC + BD



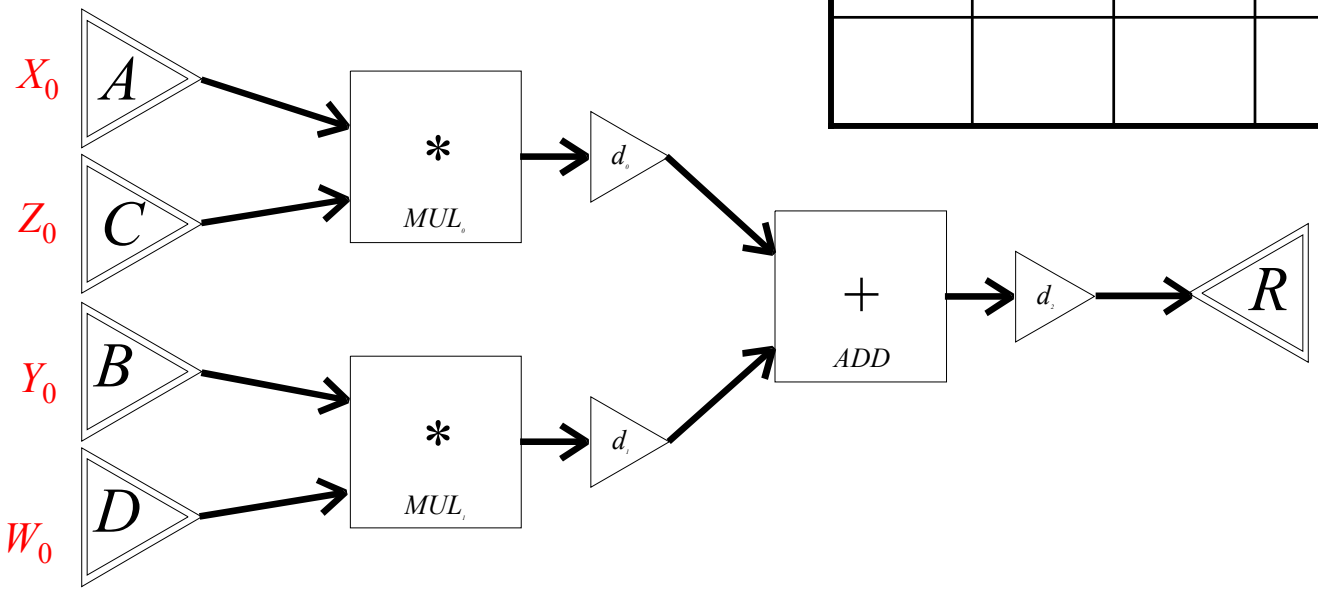
# Example

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



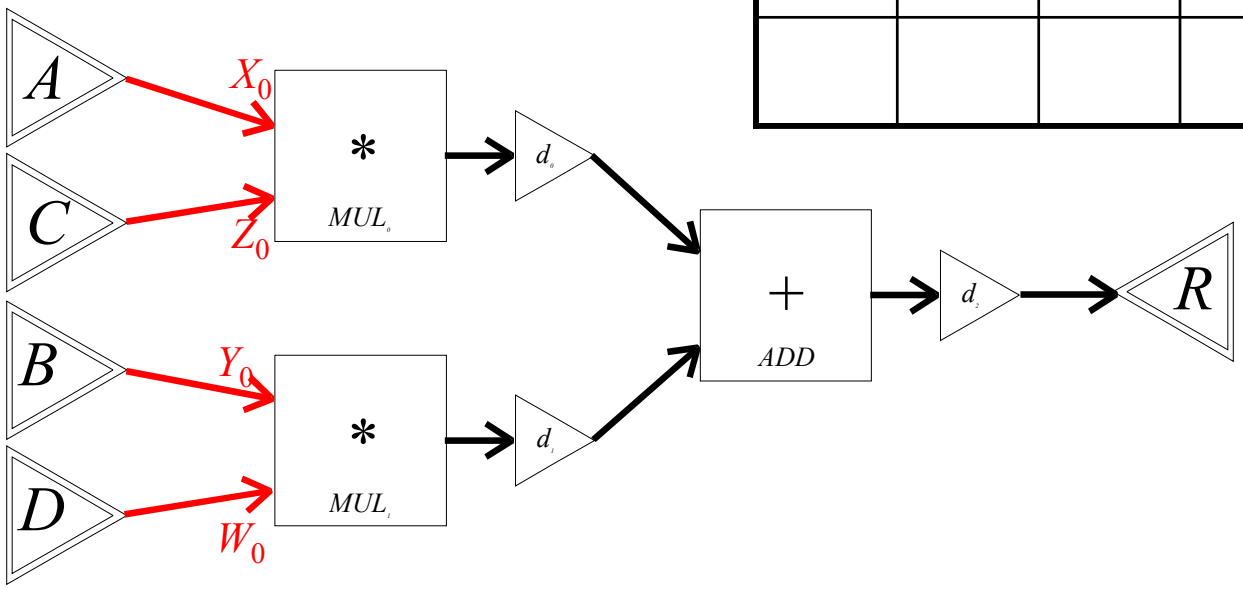
# Example

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



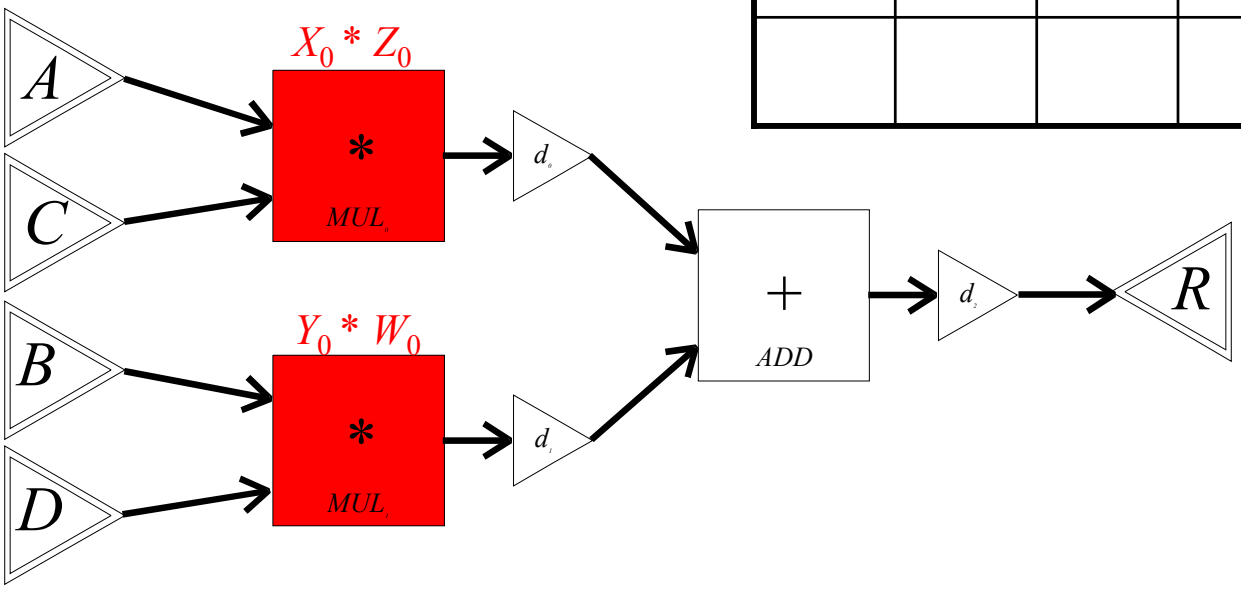
# Example

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



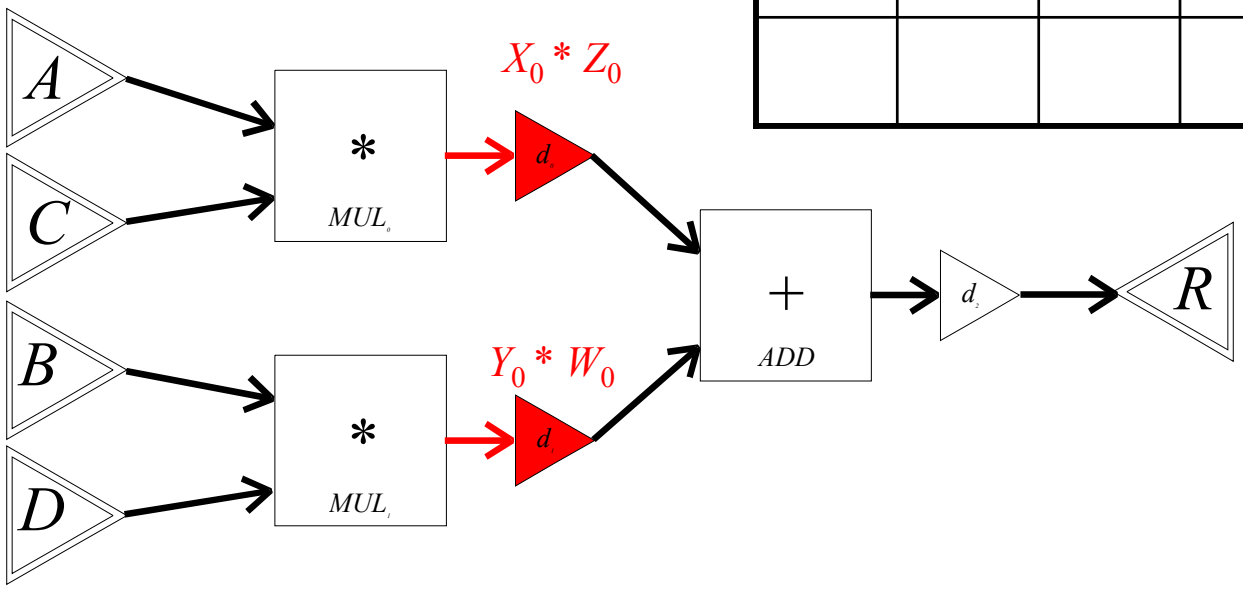
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



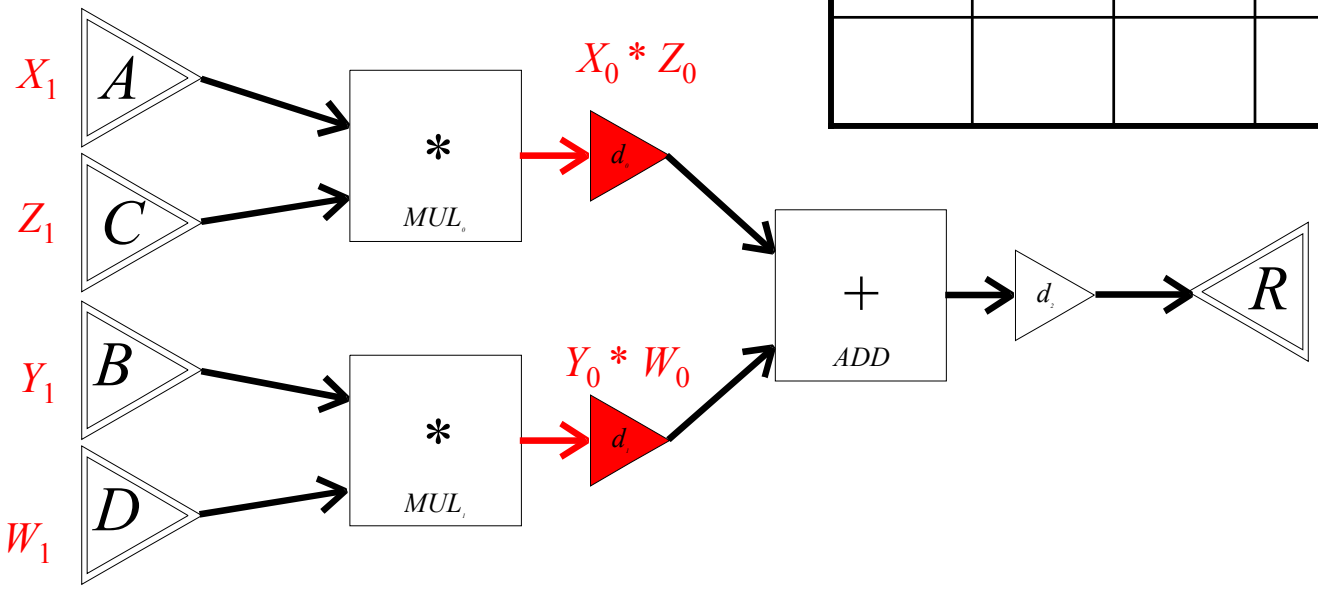
# Example

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



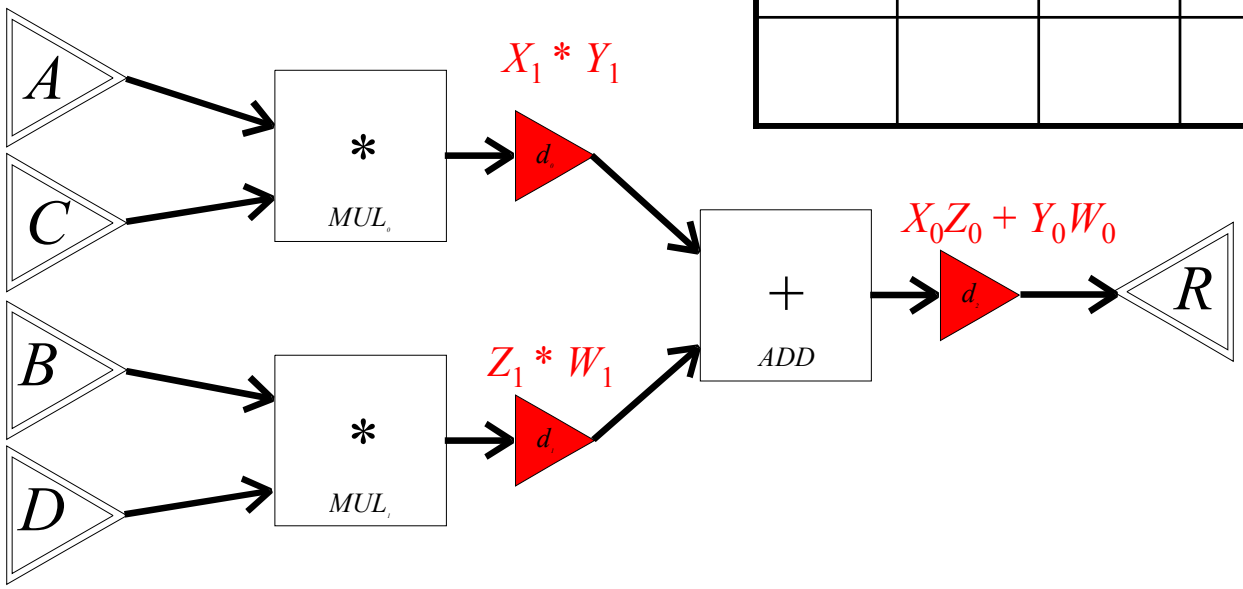
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| | | | | | |
| | | | | | |



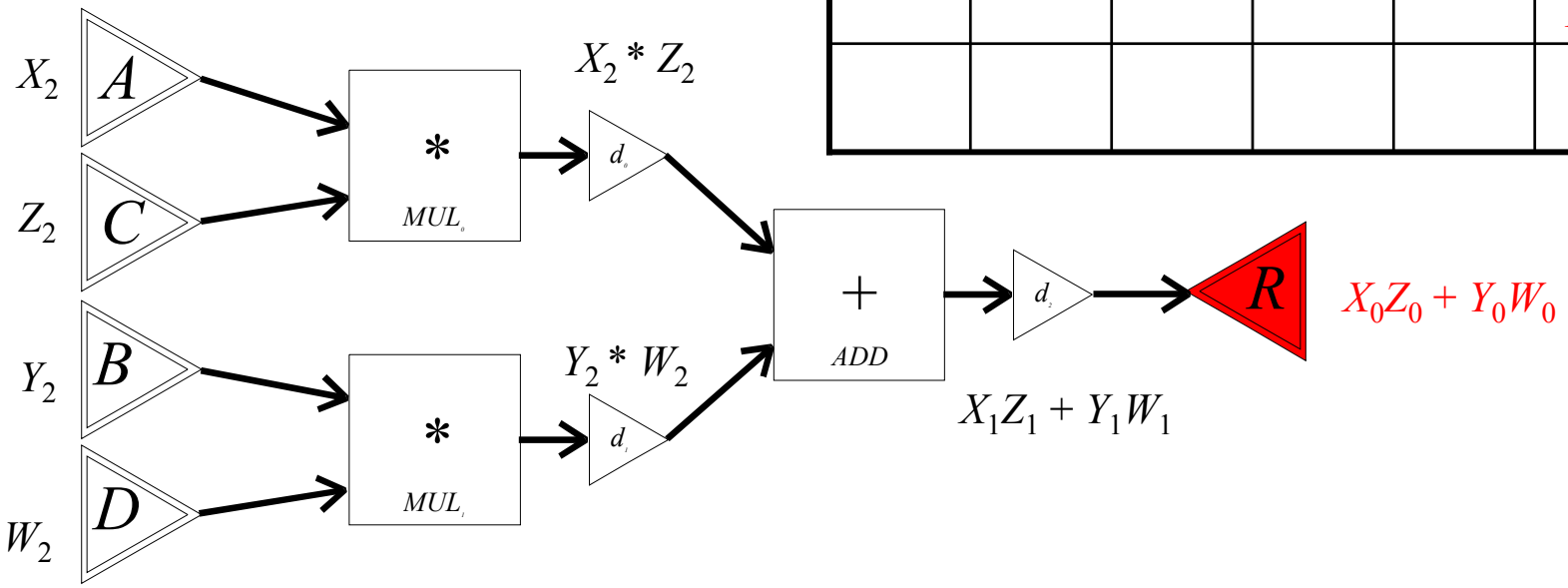
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| | | | | | |
| | | | | | |



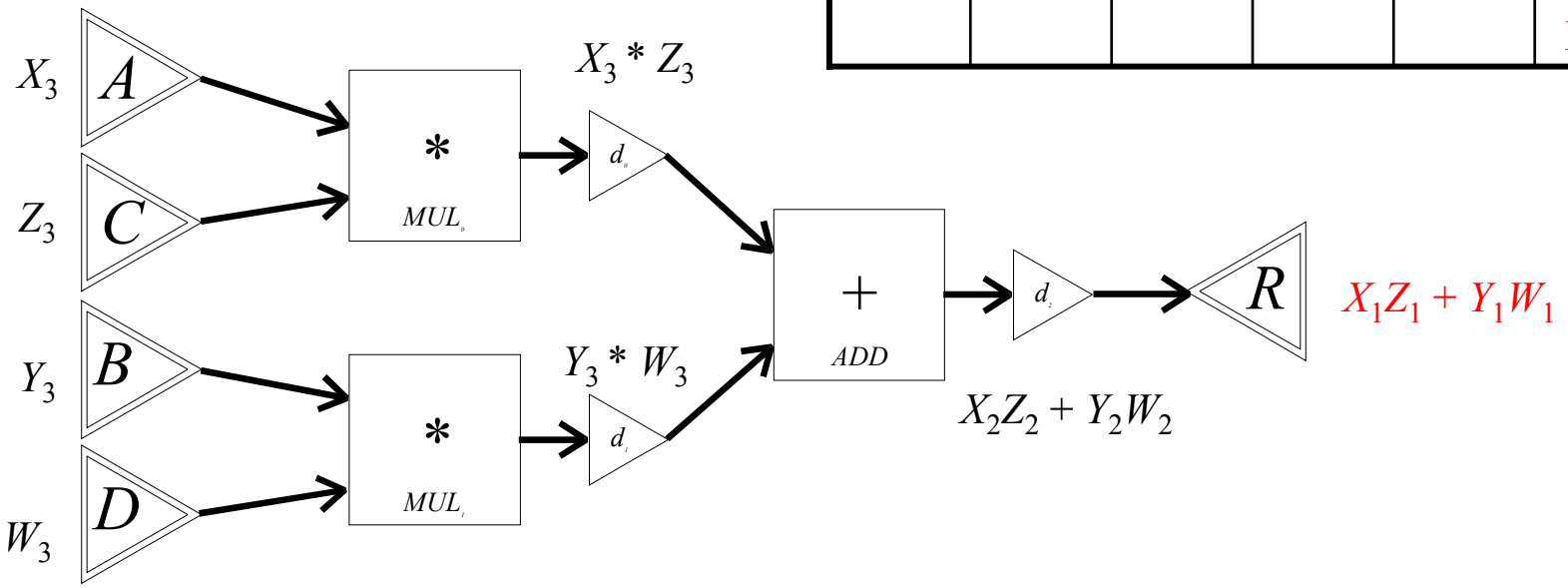
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| | | | | | |



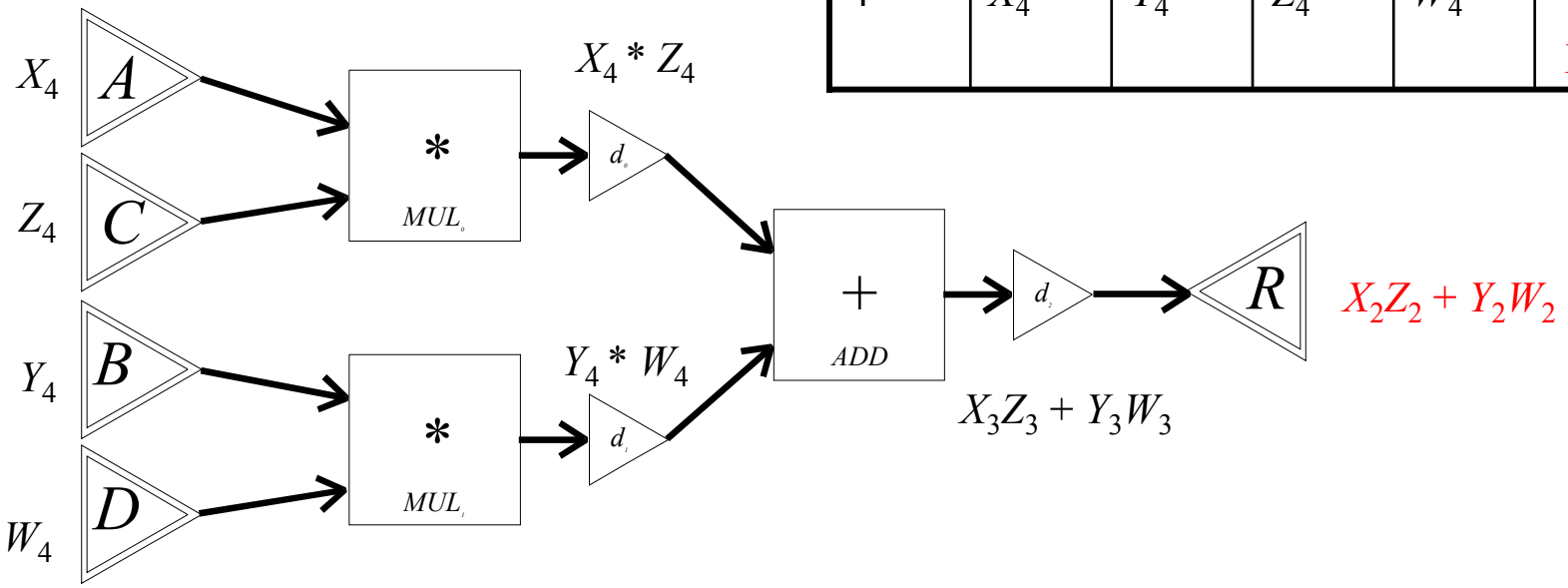
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |



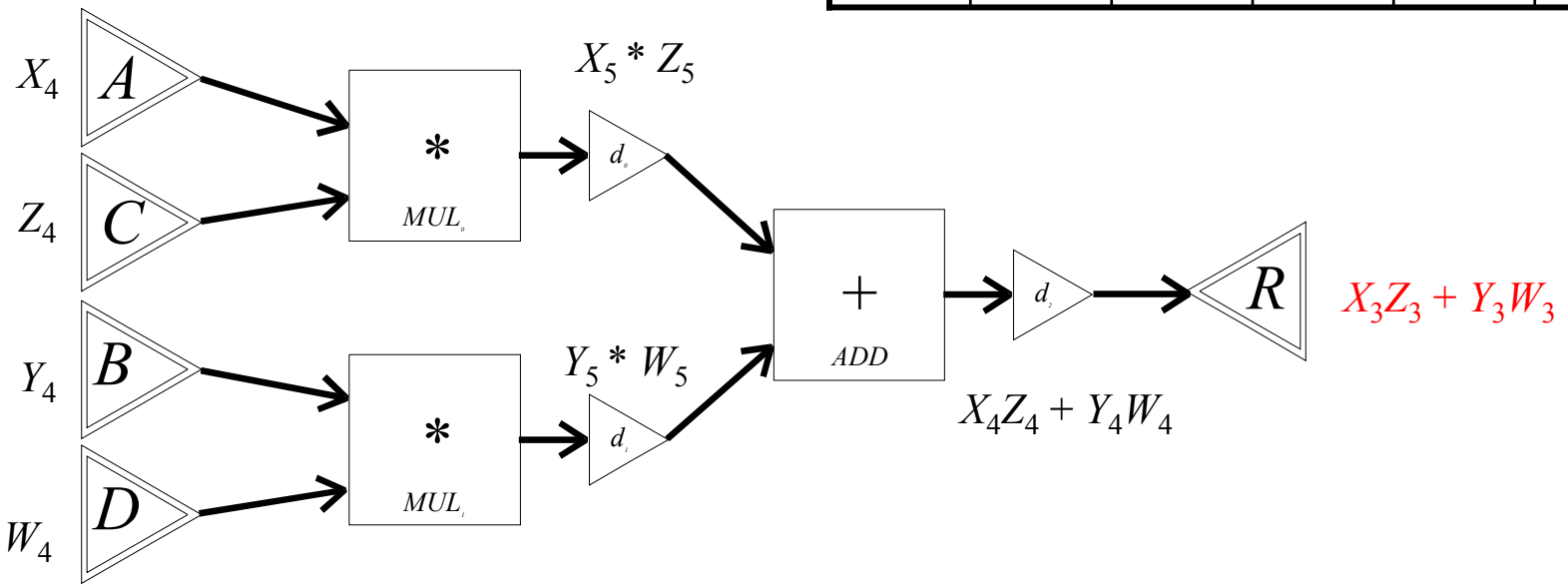
# Example

| CC | A | B | C | D | R |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |



# Example

| … | … | … | … | … | … |
|---|---|---|---|---|---|
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |
| 5 | $X_5$ | $Y_5$ | $Z_5$ | $W_5$ | $X_3Z_3 + Y_3W_3$ |



# Example

| … | … | … | … | … | … |
|---|---|---|---|---|---|
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |
| 5 | $X_5$ | $Y_5$ | $Z_5$ | $W_5$ | $X_3Z_3 + Y_3W_3$ |

# Usage Pattern

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |
| 5 | $X_5$ | $Y_5$ | $Z_5$ | $W_5$ | $X_3Z_3 + Y_3W_3$ |

# Usage Pattern

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |
| 5 | $X_5$ | $Y_5$ | $Z_5$ | $W_5$ | $X_3Z_3 + Y_3W_3$ |

Latency = 2

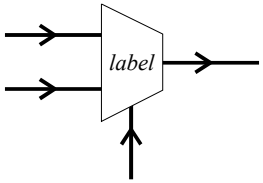
# Usage Pattern

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $X_0$ | $Y_0$ | $Z_0$ | $W_0$ | |
| 1 | $X_1$ | $Y_1$ | $Z_1$ | $W_1$ | |
| 2 | $X_2$ | $Y_2$ | $Z_2$ | $W_2$ | $X_0Z_0 + Y_0W_0$ |
| 3 | $X_3$ | $Y_3$ | $Z_3$ | $W_3$ | $X_1Z_1 + Y_1W_1$ |
| 4 | $X_4$ | $Y_4$ | $Z_4$ | $W_4$ | $X_2Z_2 + Y_2W_2$ |
| 5 | $X_5$ | $Y_5$ | $Z_5$ | $W_5$ | $X_3Z_3 + Y_3W_3$ |

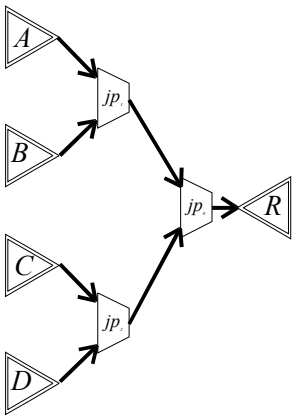
Throughput = 1

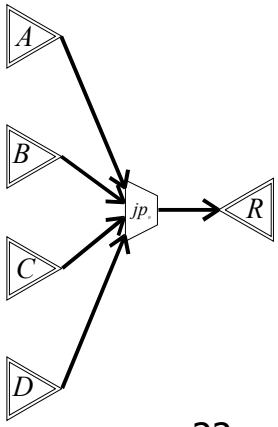
# Flexible Datapaths

- Flexible Datapath
  - Datapath where the route is determined by external environment
- Join Points

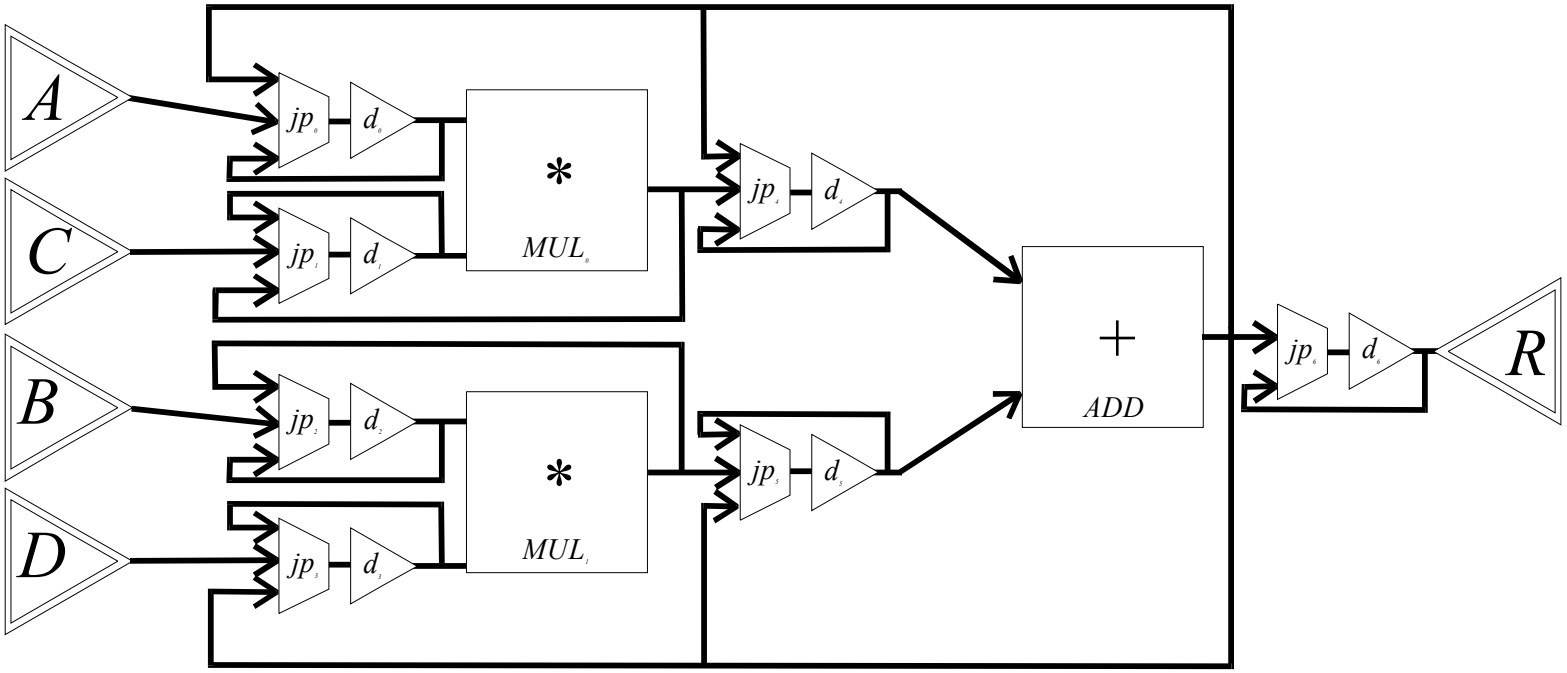



- Conventions
  - Unconnected control
  - Labels vs True/False
  - Implicit Trees

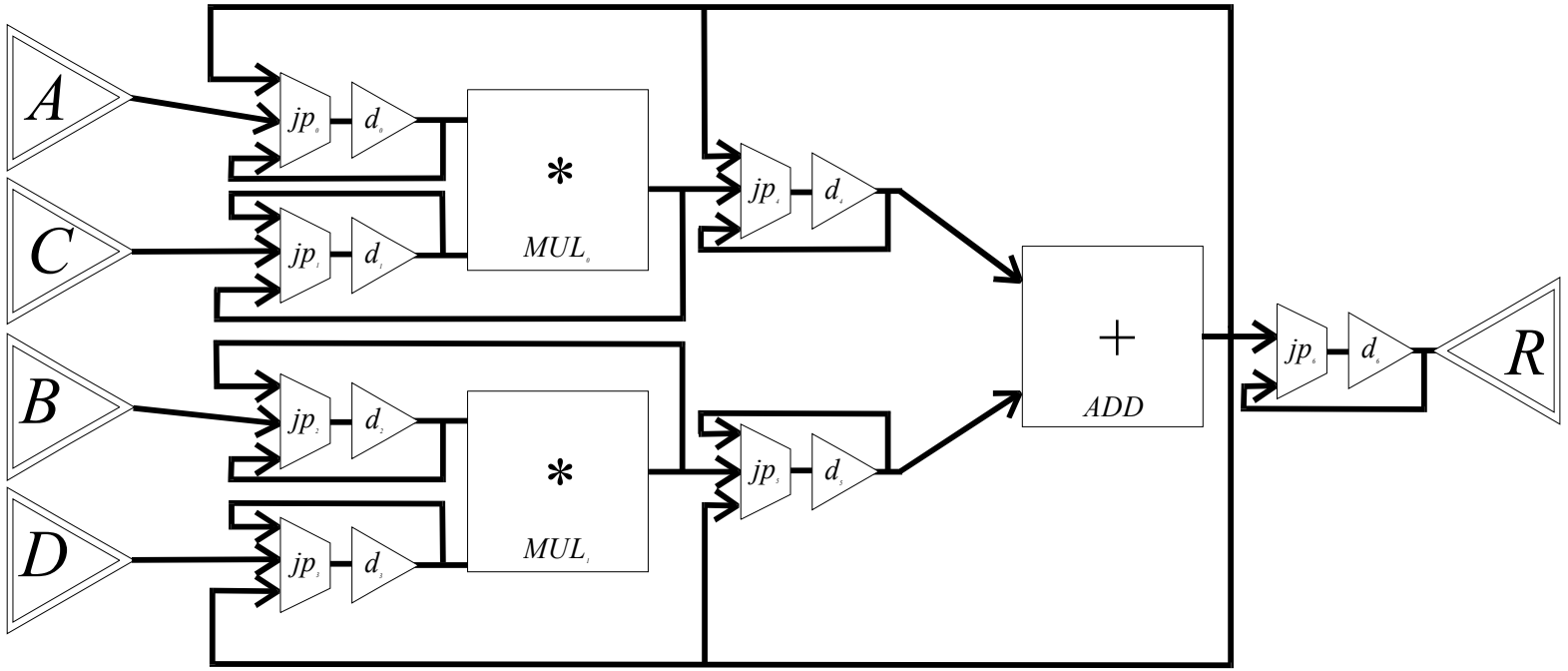


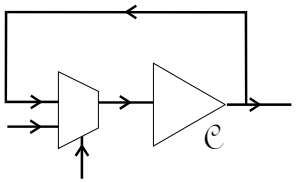
# Flexample


A
C
B
D
jp0
d0
jp1
d1
jp2
d2
jp3
d3
*
MUL0
*
MUL1
jp4
d4
jp5
d5
+
ADD
jp6
d6
R

# Flexample




- Registers



# Configurations

- Finite sequence of choices for each JP
- Sequence is repeated infinitely

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

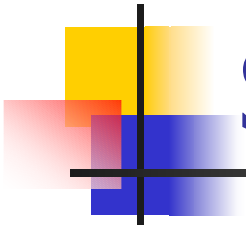
# Configurations

- Finite sequence of choices for each JP
- Sequence is repeated infinitely

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

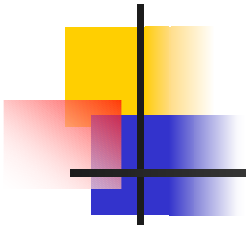
Control Word

- The **Open** Value



# Semantic Model Overview

- CoreLoom expresses the relationship between the data types:
- `run :: Datapath -> Configuration -> Usage Pattern`
- But…
- `run :: Datapath -> Configuration -> Int -> Usage Pattern`

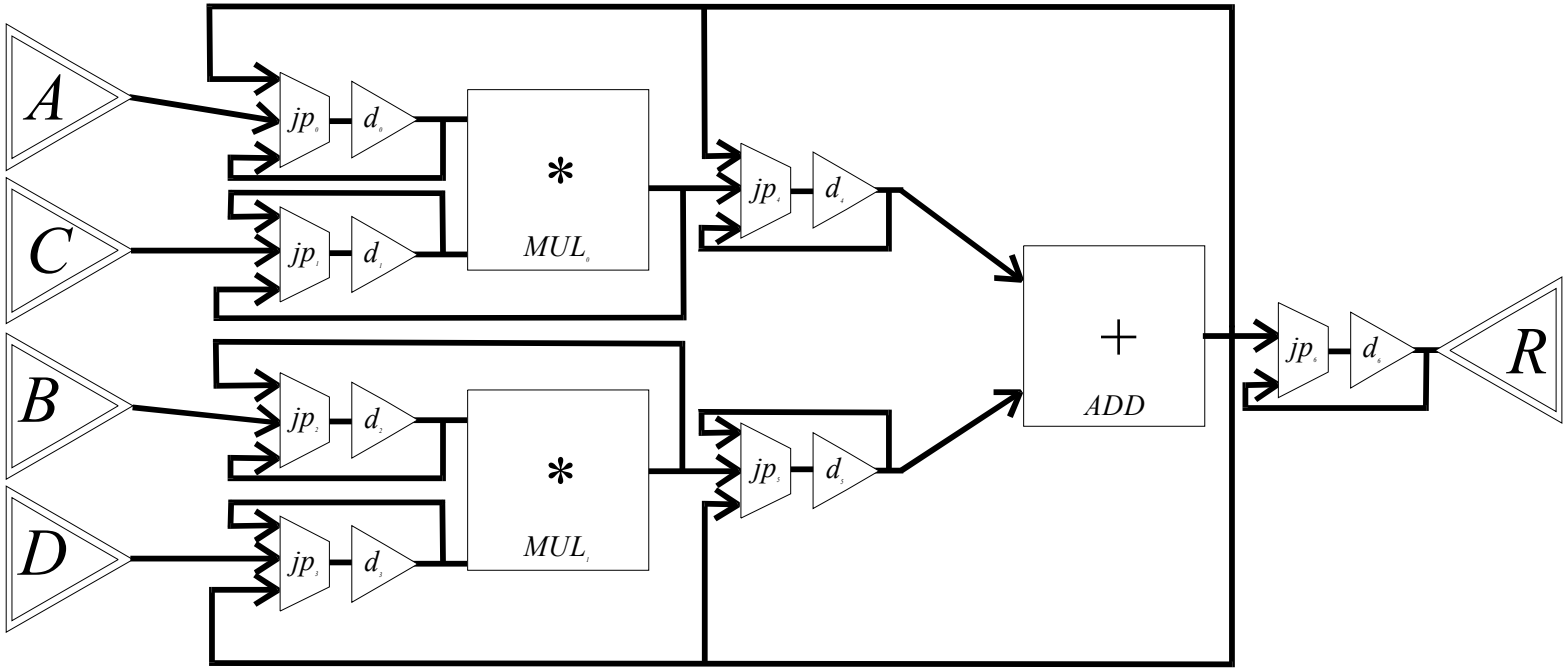


# Restrictions & Limitations

- Restrictions:
  - No combinational loops
  - No joined wires
  - At least 1 Output Node
  - Graph connected
  - No disjoint components
- Not Modeled:
  - Bit widths
  - Bit notation
  - Exceptions
  - Stalling

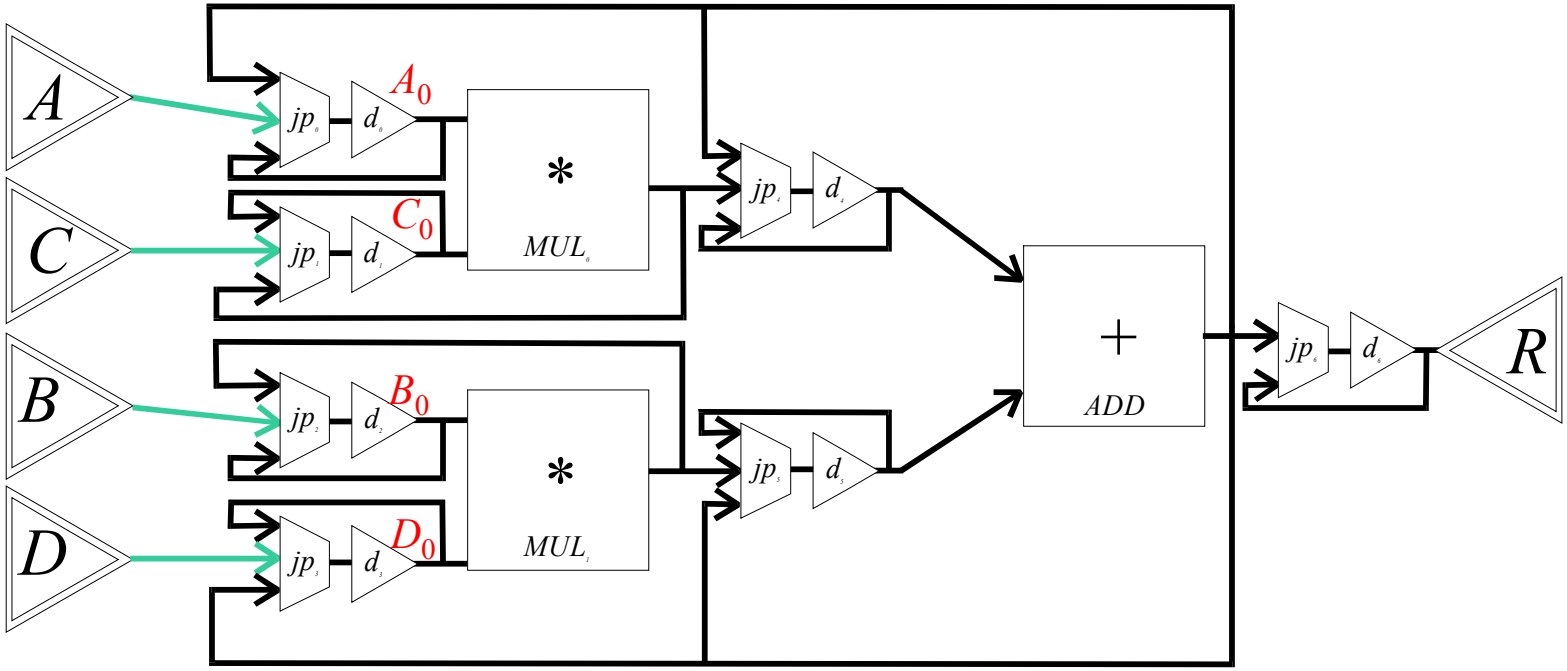
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |



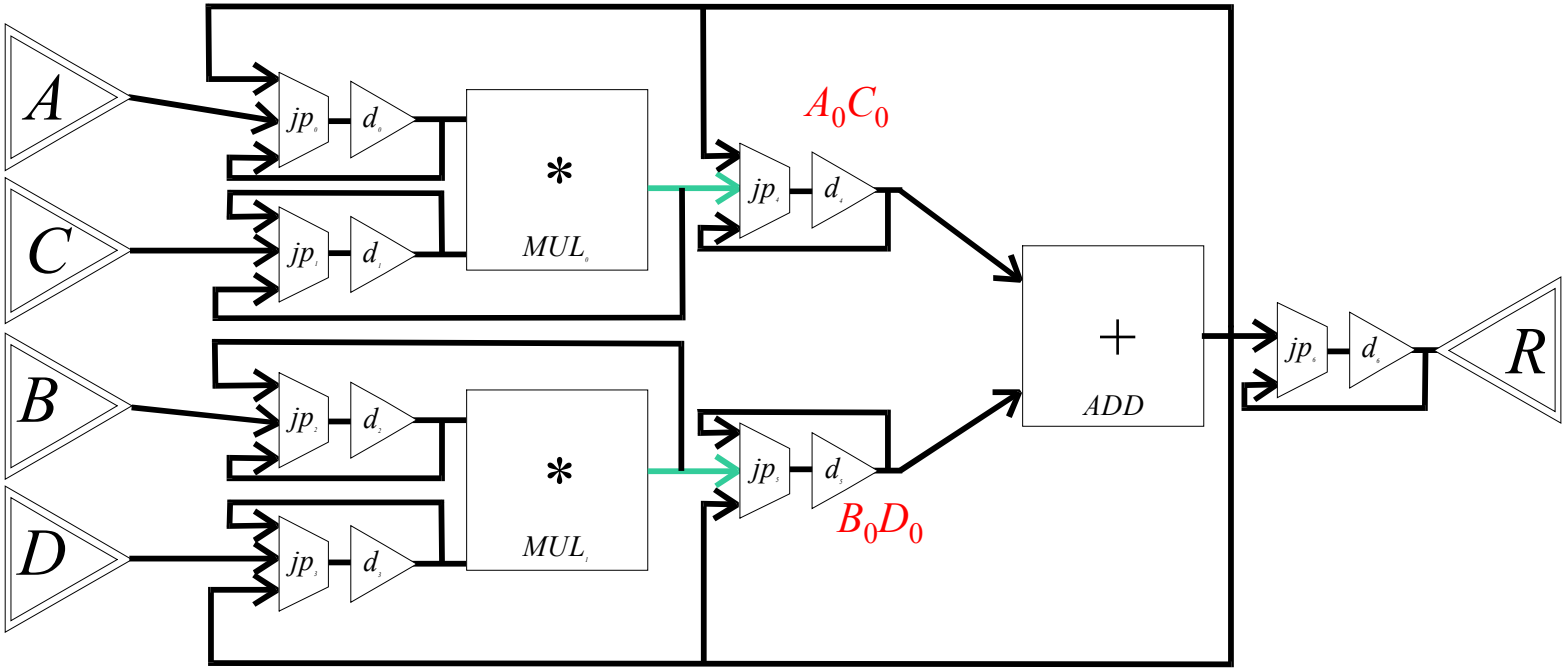
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



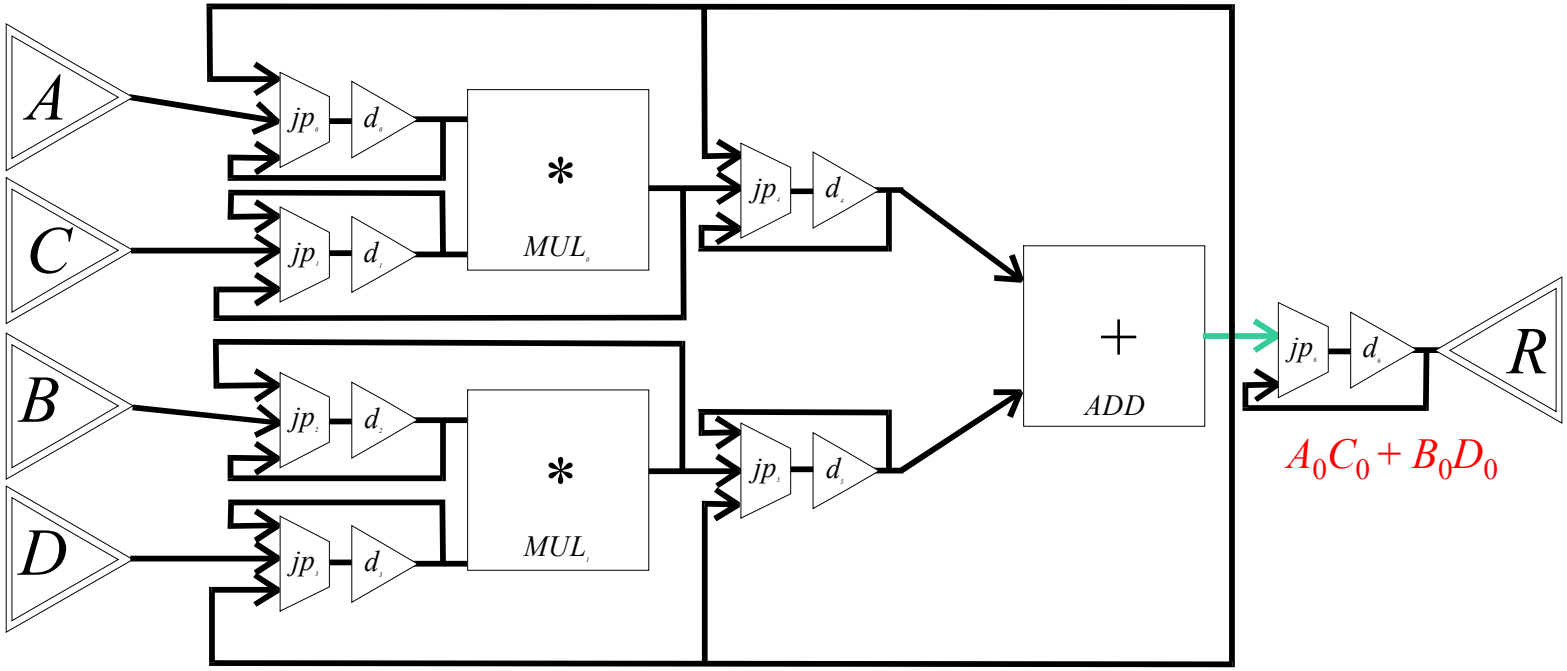
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | | | | | |
| | | | | | |
| | | | | | |



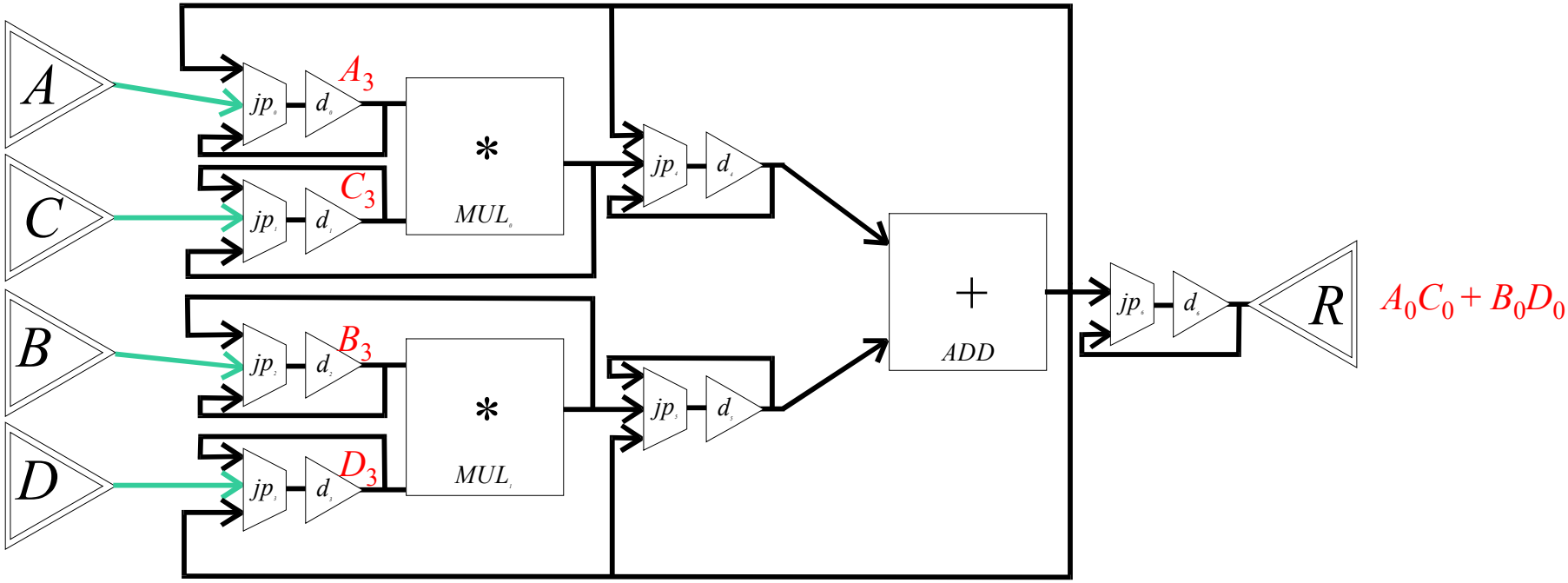
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | | | | | |
| 2 | | | | | |
| | | | | | |



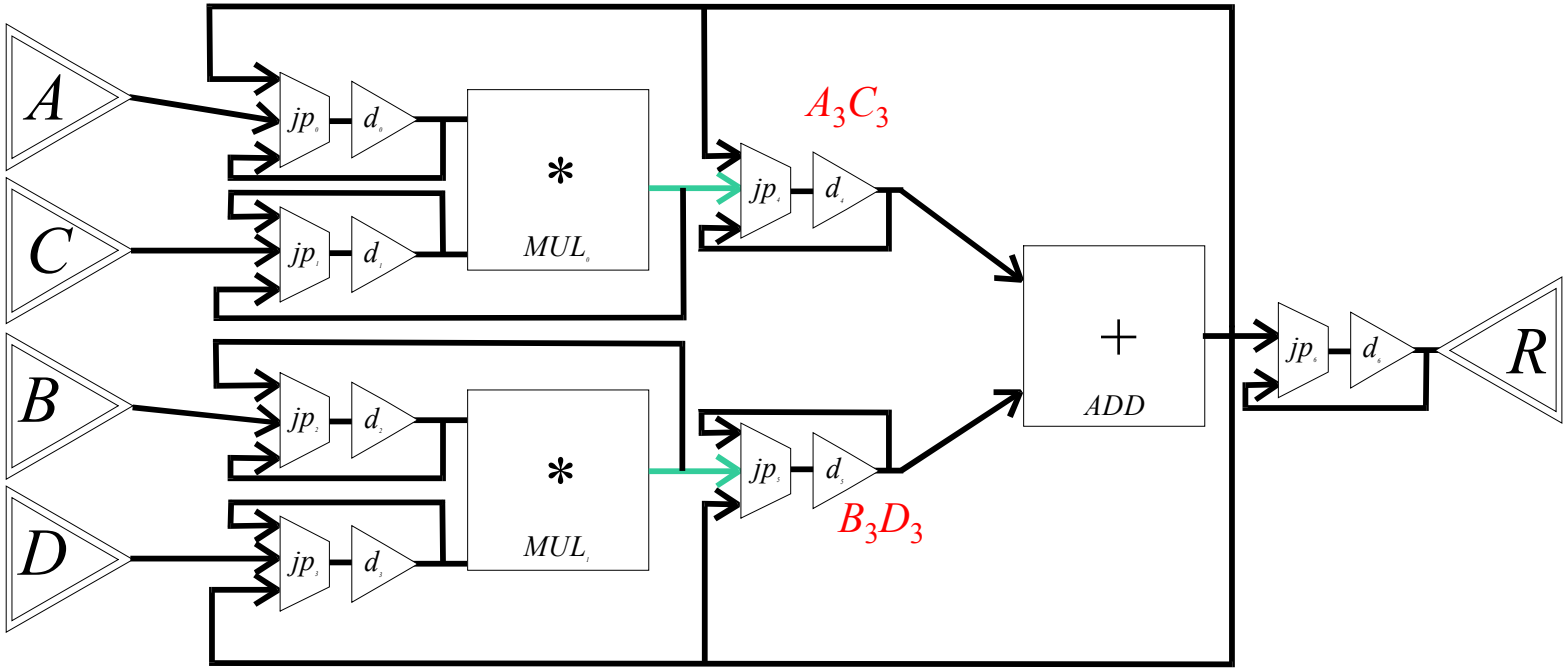
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | | | | | |
| 2 | | | | | |
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |



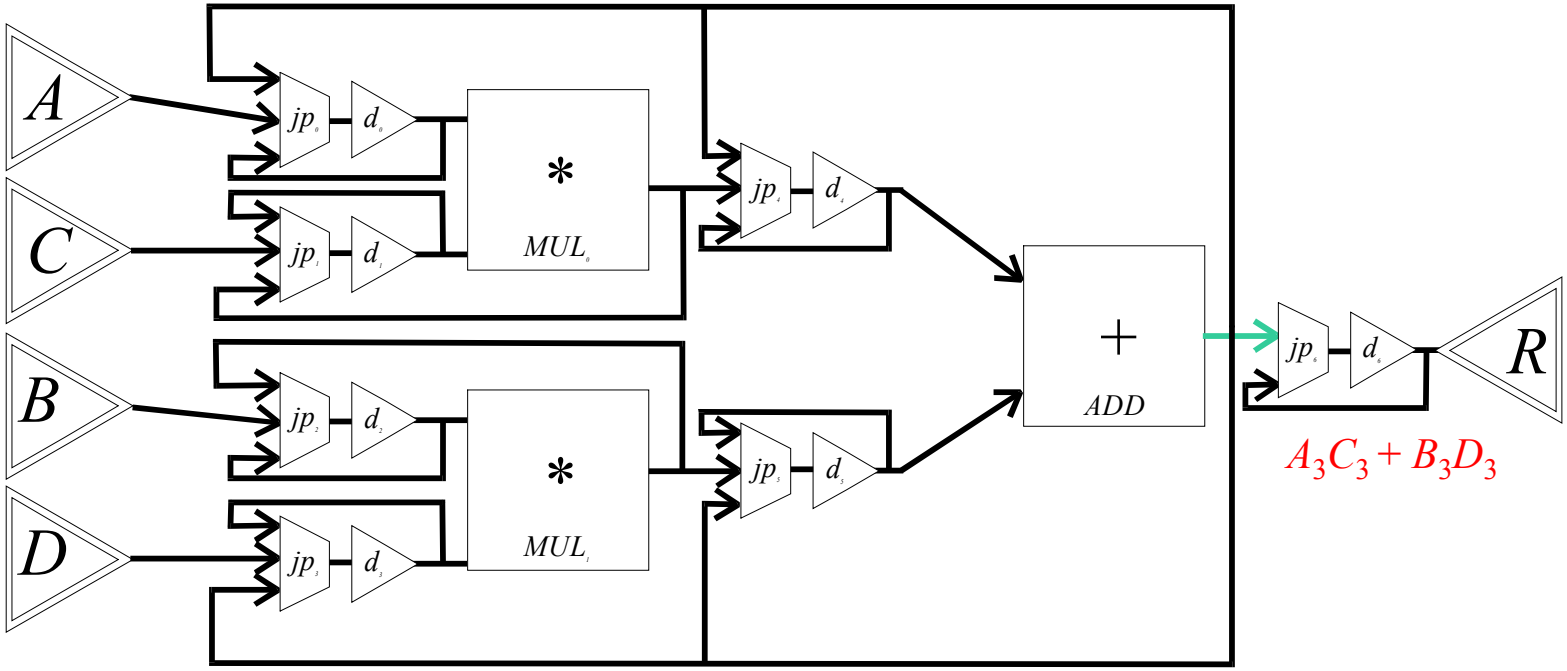
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 1 | | | | | |
| 2 | | | | | |
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |



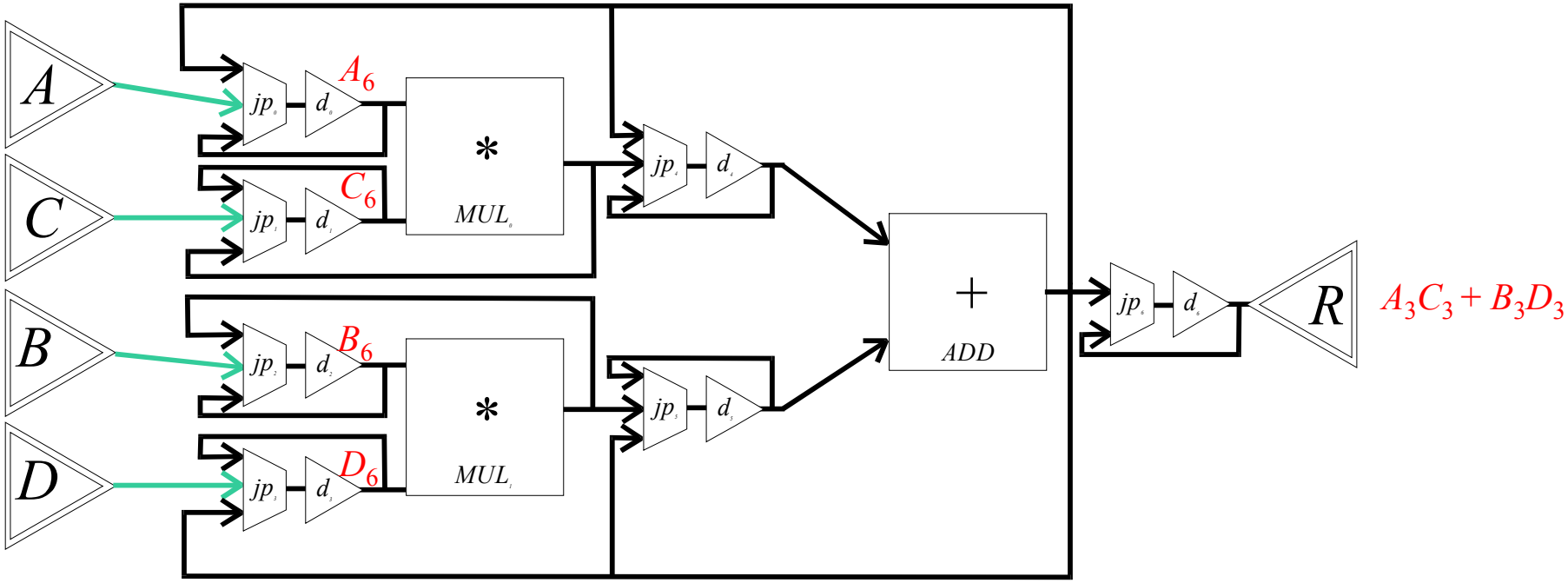
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 2 | | | | | |
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |
| 5 | | | | | |



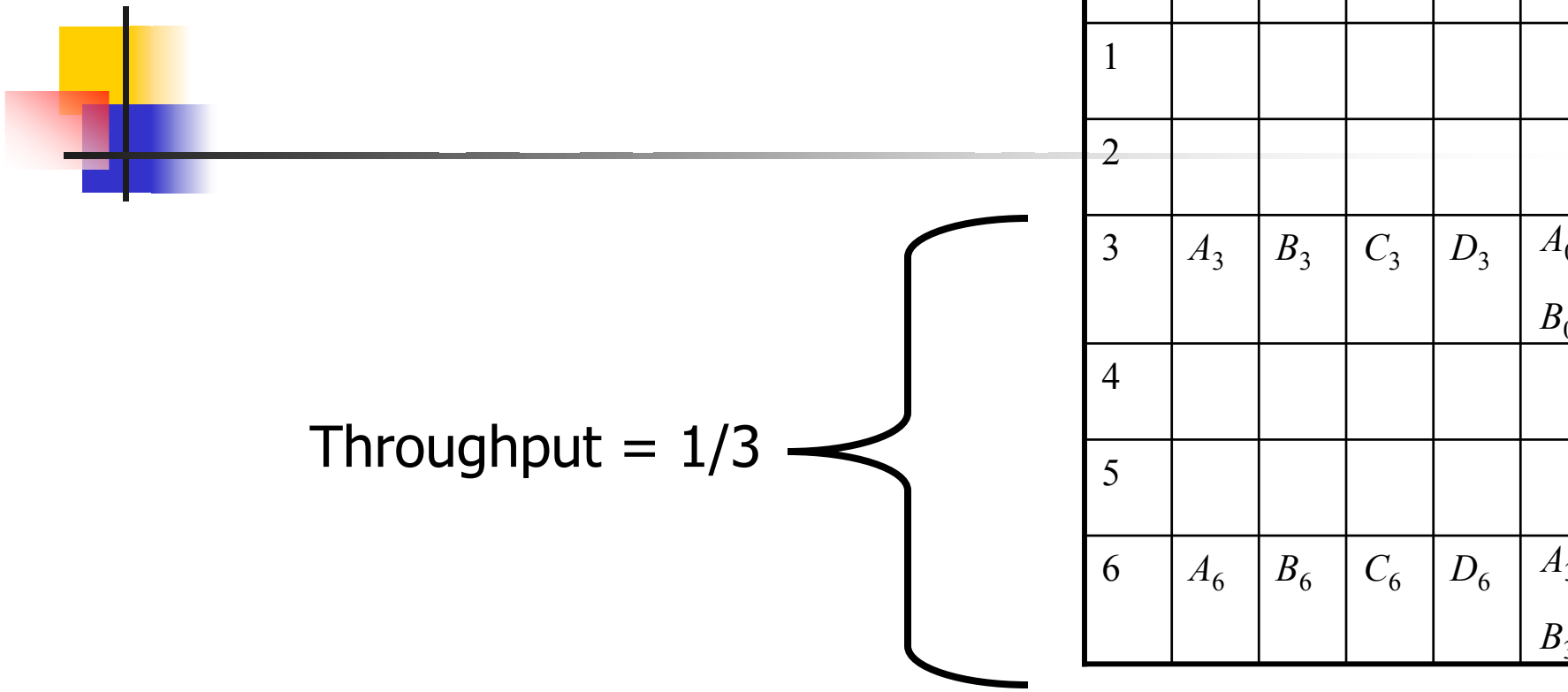
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |
| 5 | | | | | |
| 6 | $A_6$ | $B_6$ | $C_6$ | $D_6$ | $A_3C_3 + B_3D_3$ |



| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |
| 5 | | | | | |
| 6 | $A_6$ | $B_6$ | $C_6$ | $D_6$ | $A_3C_3 + B_3D_3$ |

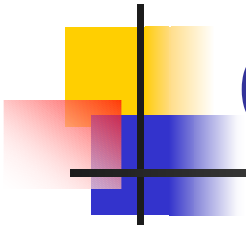
| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | | | | | |
| 2 | | | | | |
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |
| 5 | | | | | |
| 6 | $A_6$ | $B_6$ | $C_6$ | $D_6$ | $A_3C_3 + B_3D_3$ |



| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | | | | | |
| 2 | | | | | |
| 3 | $A_3$ | $B_3$ | $C_3$ | $D_3$ | $A_0C_0 + B_0D_0$ |
| 4 | | | | | |
| 5 | | | | | |
| 6 | $A_6$ | $B_6$ | $C_6$ | $D_6$ | $A_3C_3 + B_3D_3$ |

Throughput = 1/3

What's the problem?



# Configuration Scheduling

- Problem: Simple repetition does not maximize throughput
- Solution: Configuration scheduling
- CoreLoom does two types:
  - Concatenation Scheduling
  - Overlapping Scheduling

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

Original Length

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

# Configuration Scheduling

- Concatenation Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |

# Configuration Scheduling

- Configuration Compression:

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |

# Configuration Scheduling

- Configuration Compression:

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |

# Configuration Scheduling

- Final Result:

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |

- Optimum throughput

# Configuration Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

# Configuration Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| $ADD$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |

# Configuration Scheduling

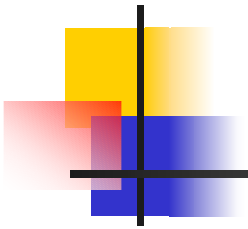
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | *ADD* |

# Overlapping Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | | | |
| | | | | $MUL_0$ | $MUL_1$ | |
| | | | | | | $ADD$ |
| | | | | | | |

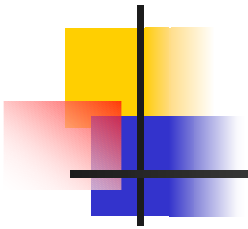
# Overlapping Scheduling

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | | | |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| | | | | $MUL_0$ | $MUL_1$ | $ADD$ |
| | | | | | | $ADD$ |

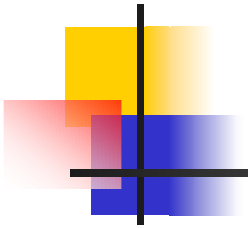


| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | | | |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| | | | | $MUL_0$ | $MUL_1$ | *ADD* |
| | | | | | | *ADD* |

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | | | |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| | | | | $MUL_0$ | $MUL_1$ | *ADD* |
| | | | | | | *ADD* |



| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |



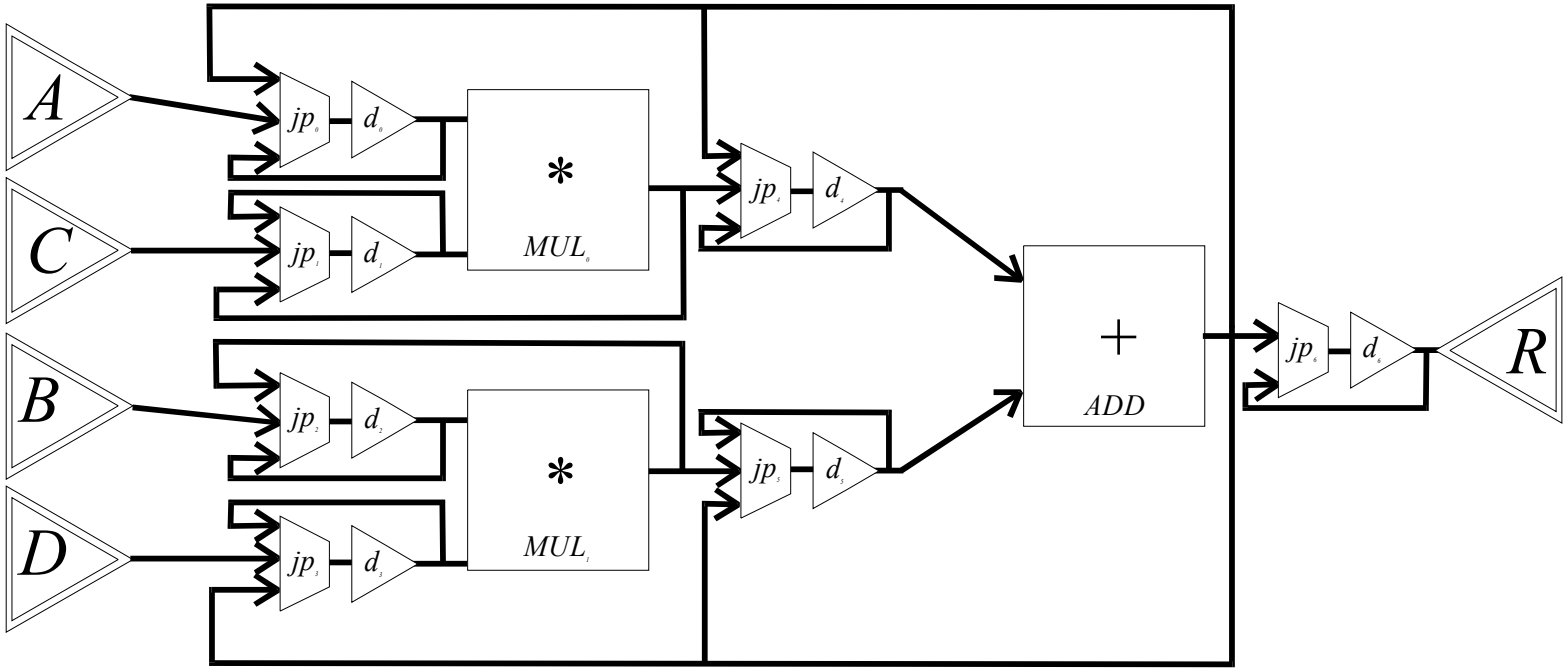
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |

## Final Configuration:

| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |

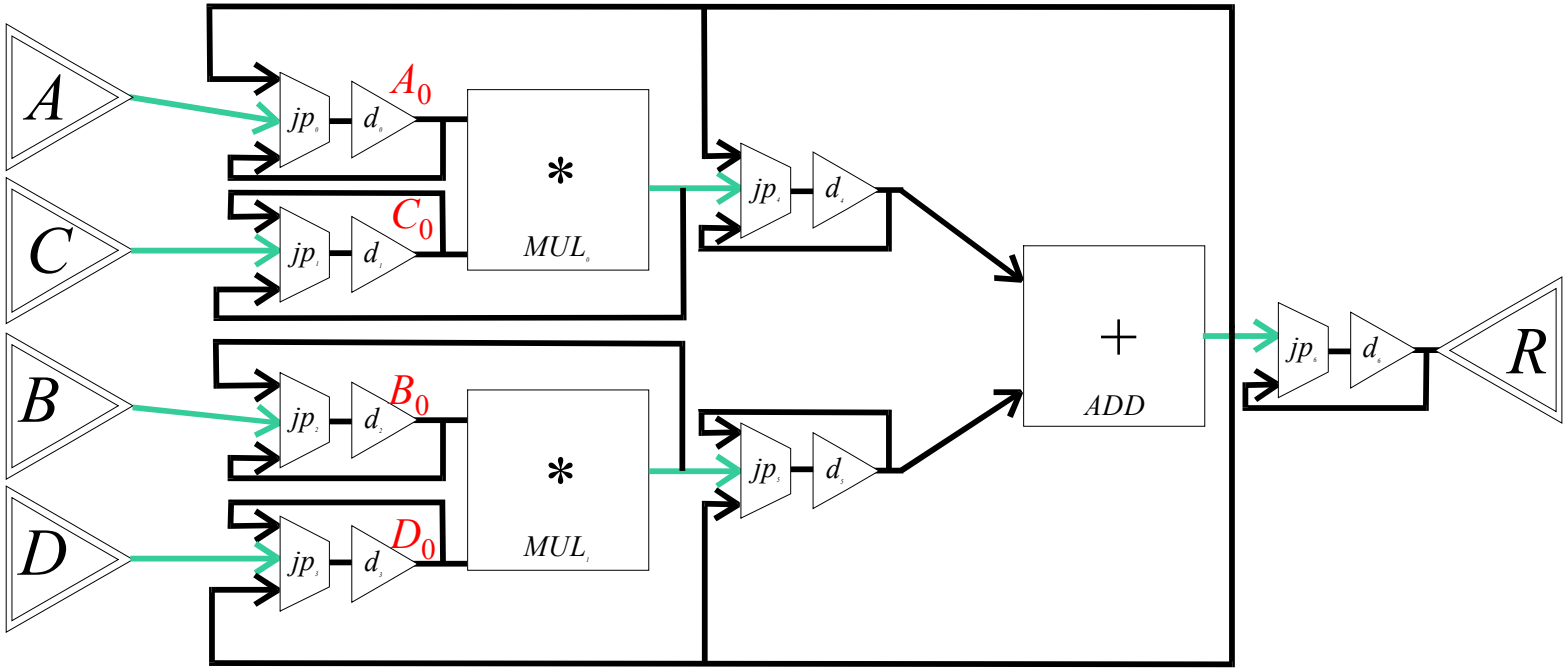
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |

| *CC* | *A* | *B* | *C* | *D* | *R* |
|---|---|---|---|---|---|
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |



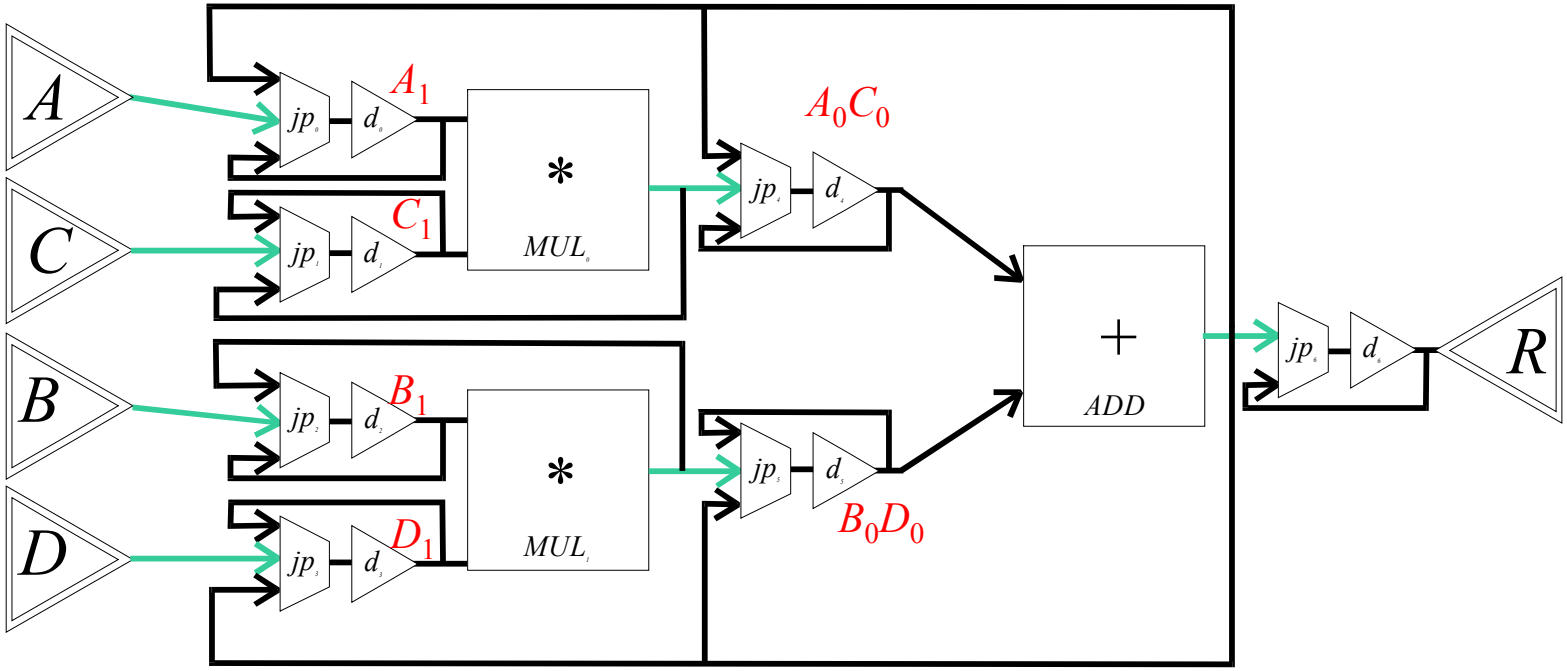
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |

| *CC* | *A* | *B* | *C* | *D* | *R* |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| | | | | | |
| | | | | | |
| | | | | | |



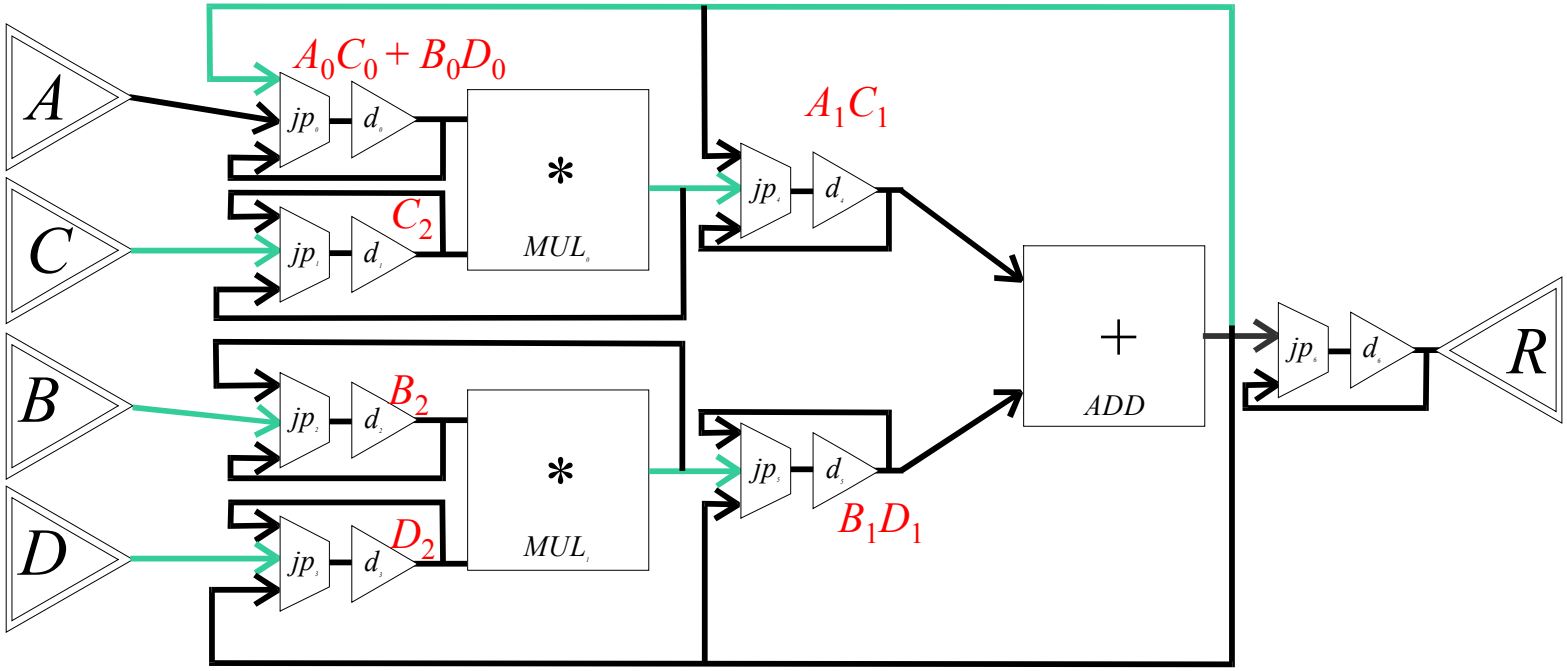
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |

| *CC* | *A* | *B* | *C* | *D* | *R* |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| | | | | | |
| | | | | | |



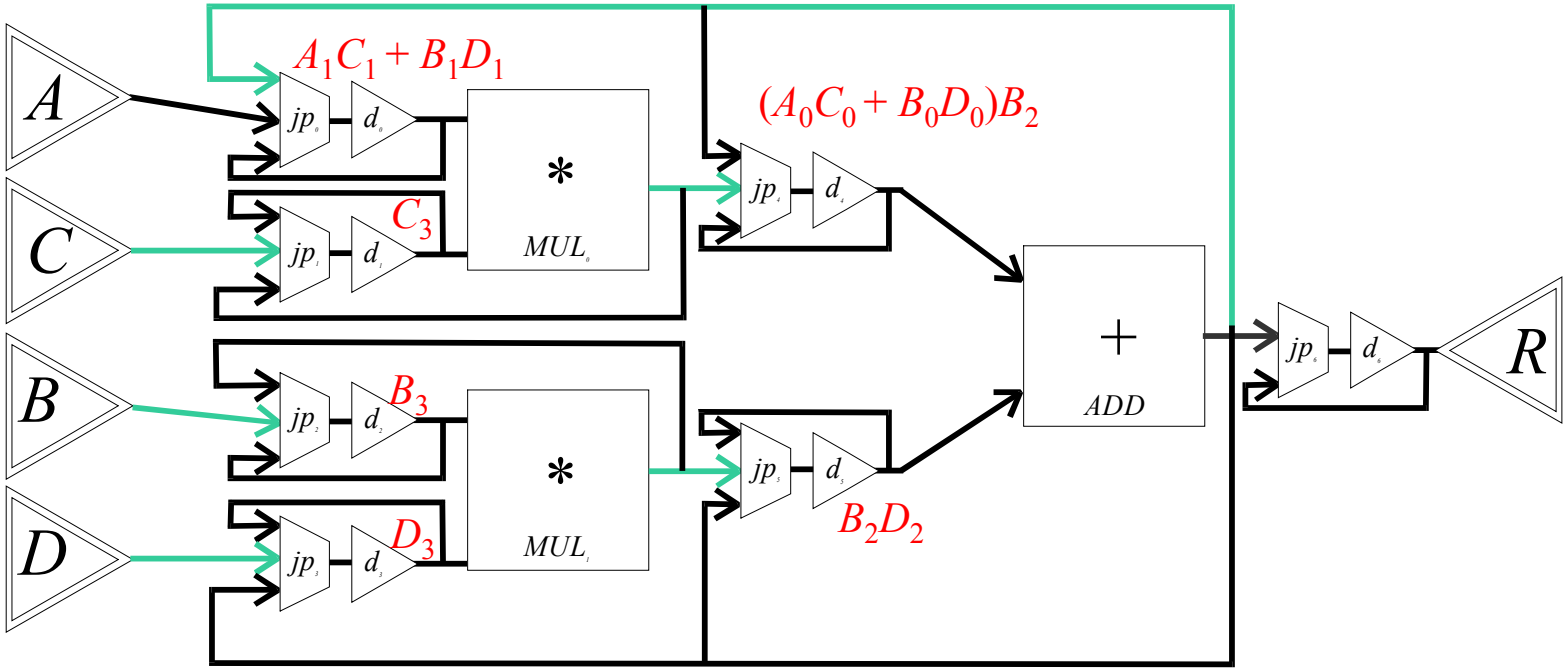
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |

| *CC* | *A* | *B* | *C* | *D* | *R* |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| | | | | | |



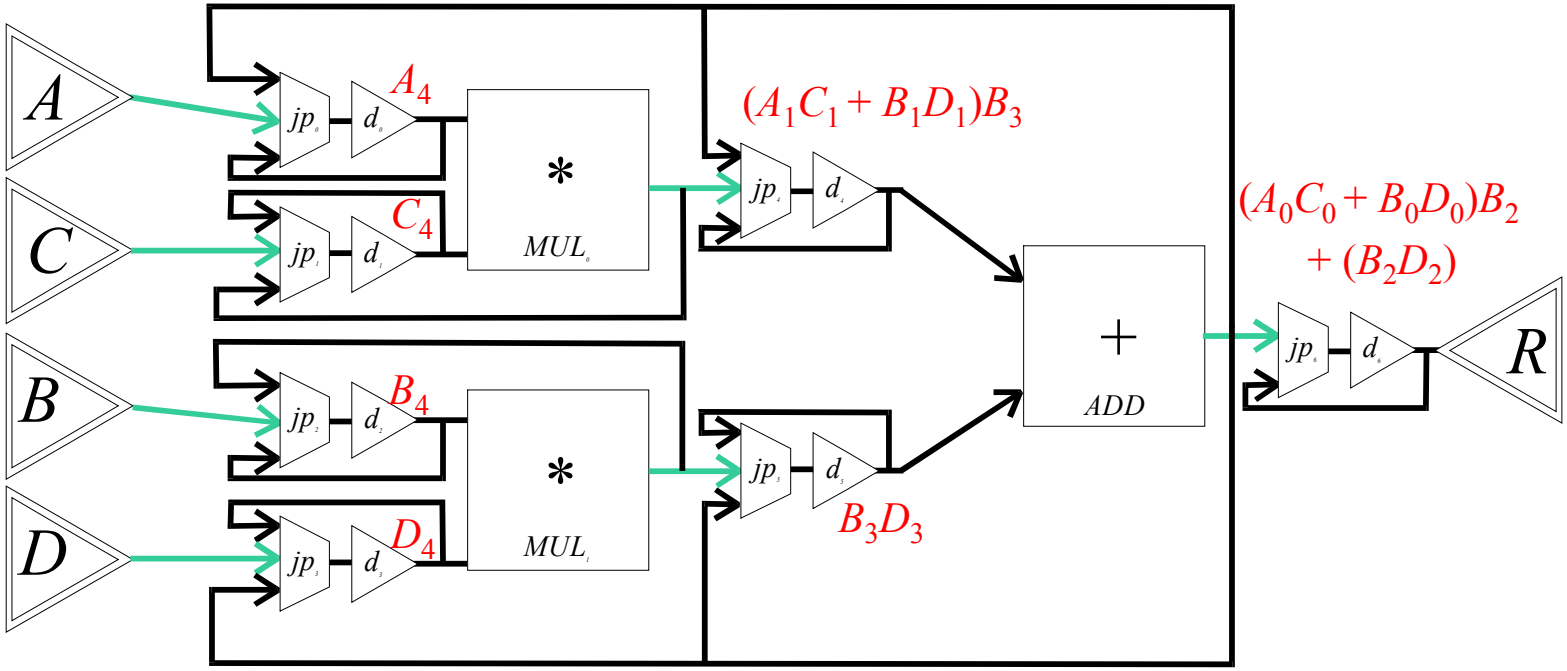
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |



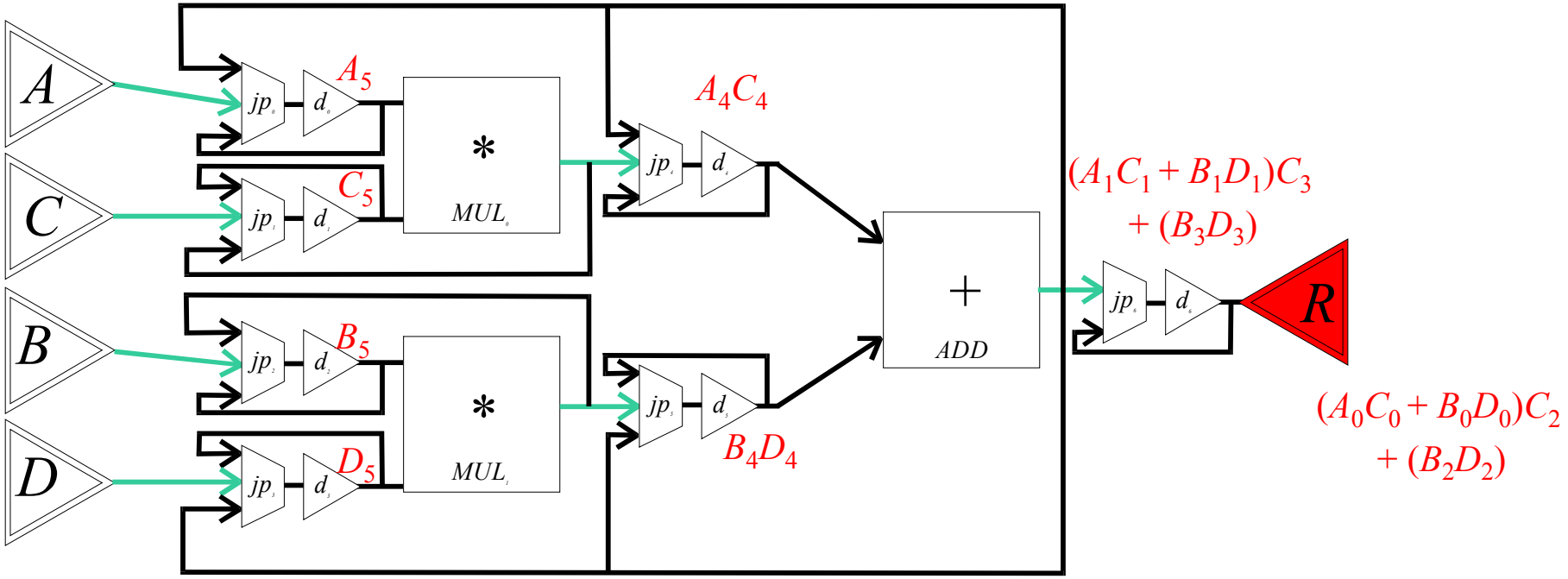
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *A* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | *ADD* |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |
| *ADD* | *C* | *B* | *D* | $MUL_0$ | $MUL_1$ | |

| *CC* | *A* | *B* | *C* | *D* | *R* |
|---|---|---|---|---|---|
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |
| 4 | $A_4$ | $B_4$ | $C_4$ | $D_4$ | |



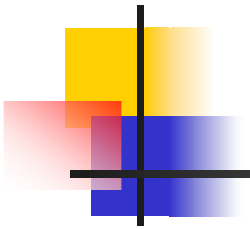
| $jp_0$ | $jp_1$ | $jp_2$ | $jp_3$ | $jp_4$ | $jp_5$ | $jp_6$ |
|---|---|---|---|---|---|---|
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $A$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | $ADD$ |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |
| $ADD$ | $C$ | $B$ | $D$ | $MUL_0$ | $MUL_1$ | |

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |
| 4 | $A_4$ | $B_4$ | $C_4$ | $D_4$ | |
| 5 | $A_5$ | $B_5$ | $C_5$ | $D_5$ | $(A_0C_0 + B_0D_0)C_2 + (B_2D_2)$ |



Latency = ??

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |
| 4 | $A_4$ | $B_4$ | $C_4$ | $D_4$ | |
| 5 | $A_5$ | $B_5$ | $C_5$ | $D_5$ | $(A_0C_0 + B_0D_0)C_2 + (B_2D_2)$ |
| 6 | | $B_6$ | $C_6$ | $D_6$ | $(A_1C_1 + B_1D_1)C_3 + (B_3D_3)$ |
| 7 | | $B_7$ | $C_7$ | $D_7$ | |
| 8 | $A_8$ | $B_8$ | $C_8$ | $D_8$ | |
| 9 | $A_9$ | $B_9$ | $C_9$ | $D_9$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |
| 10 | $A_{10}$ | $B_{10}$ | $C_{10}$ | $D_{10}$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |



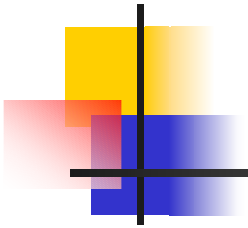
Latency = ??

5 from first input

3 from last input

CoreLoom arbitrarily uses last

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |
| 4 | $A_4$ | $B_4$ | $C_4$ | $D_4$ | |
| 5 | $A_5$ | $B_5$ | $C_5$ | $D_5$ | $(A_0C_0 + B_0D_0)C_2 + (B_2D_2)$ |
| 6 | | $B_6$ | $C_6$ | $D_6$ | $(A_1C_1 + B_1D_1)C_3 + (B_3D_3)$ |
| 7 | | $B_7$ | $C_7$ | $D_7$ | |
| 8 | $A_8$ | $B_8$ | $C_8$ | $D_8$ | |
| 9 | $A_9$ | $B_9$ | $C_9$ | $D_9$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |
| 10 | $A_{10}$ | $B_{10}$ | $C_{10}$ | $D_{10}$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |



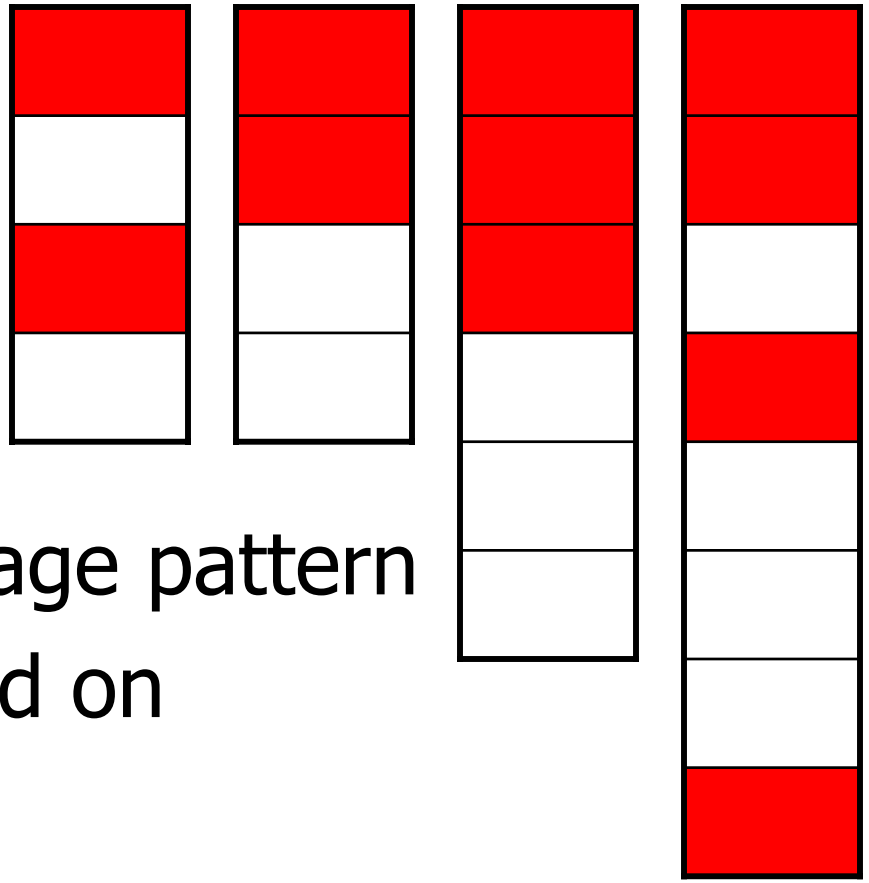
Throughput = ½?

| $CC$ | $A$ | $B$ | $C$ | $D$ | $R$ |
|---|---|---|---|---|---|
| 0 | $A_0$ | $B_0$ | $C_0$ | $D_0$ | |
| 1 | $A_1$ | $B_1$ | $C_1$ | $D_1$ | |
| 2 | | $B_2$ | $C_2$ | $D_2$ | |
| 3 | | $B_3$ | $C_3$ | $D_3$ | |
| 4 | $A_4$ | $B_4$ | $C_4$ | $D_4$ | |
| 5 | $A_5$ | $B_5$ | $C_5$ | $D_5$ | $(A_0C_0 + B_0D_0)C_2 + (B_2D_2)$ |
| 6 | | $B_6$ | $C_6$ | $D_6$ | $(A_1C_1 + B_1D_1)C_3 + (B_3D_3)$ |
| 7 | | $B_7$ | $C_7$ | $D_7$ | |
| 8 | $A_8$ | $B_8$ | $C_8$ | $D_8$ | |
| 9 | $A_9$ | $B_9$ | $C_9$ | $D_9$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |
| 10 | $A_{10}$ | $B_{10}$ | $C_{10}$ | $D_{10}$ | $(A_0C_0 + B_0D_0)B_2 + (B_2D_2)$ |

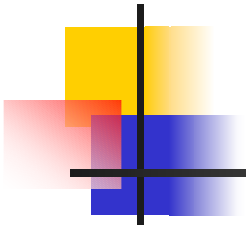
# Overlapping Example

- Throughput becomes ambiguous

Throughput = ½?



- Only the usage pattern can be relied on

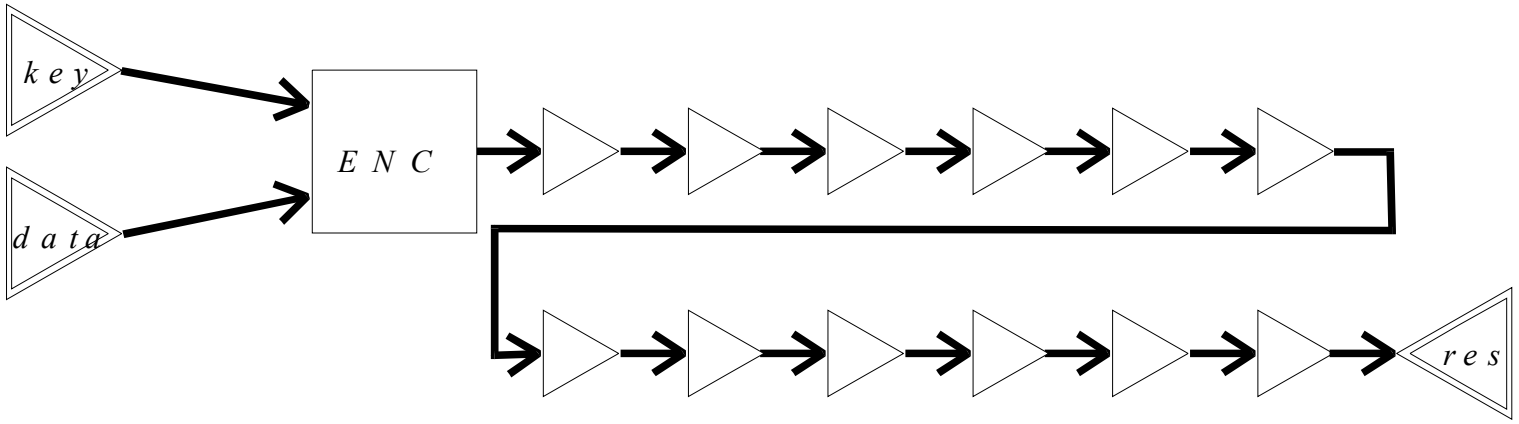


# Black-Box Approach

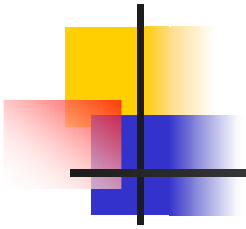
- Can model cores as “black boxes”
- Method: Extrapolate Datapath from Usage Pattern
- Example: Encryption Core
  - Encrypts Key, Data
  - 12 CC Latency
  - TP 1

# Black Box Example

- Pipelined Encryptor



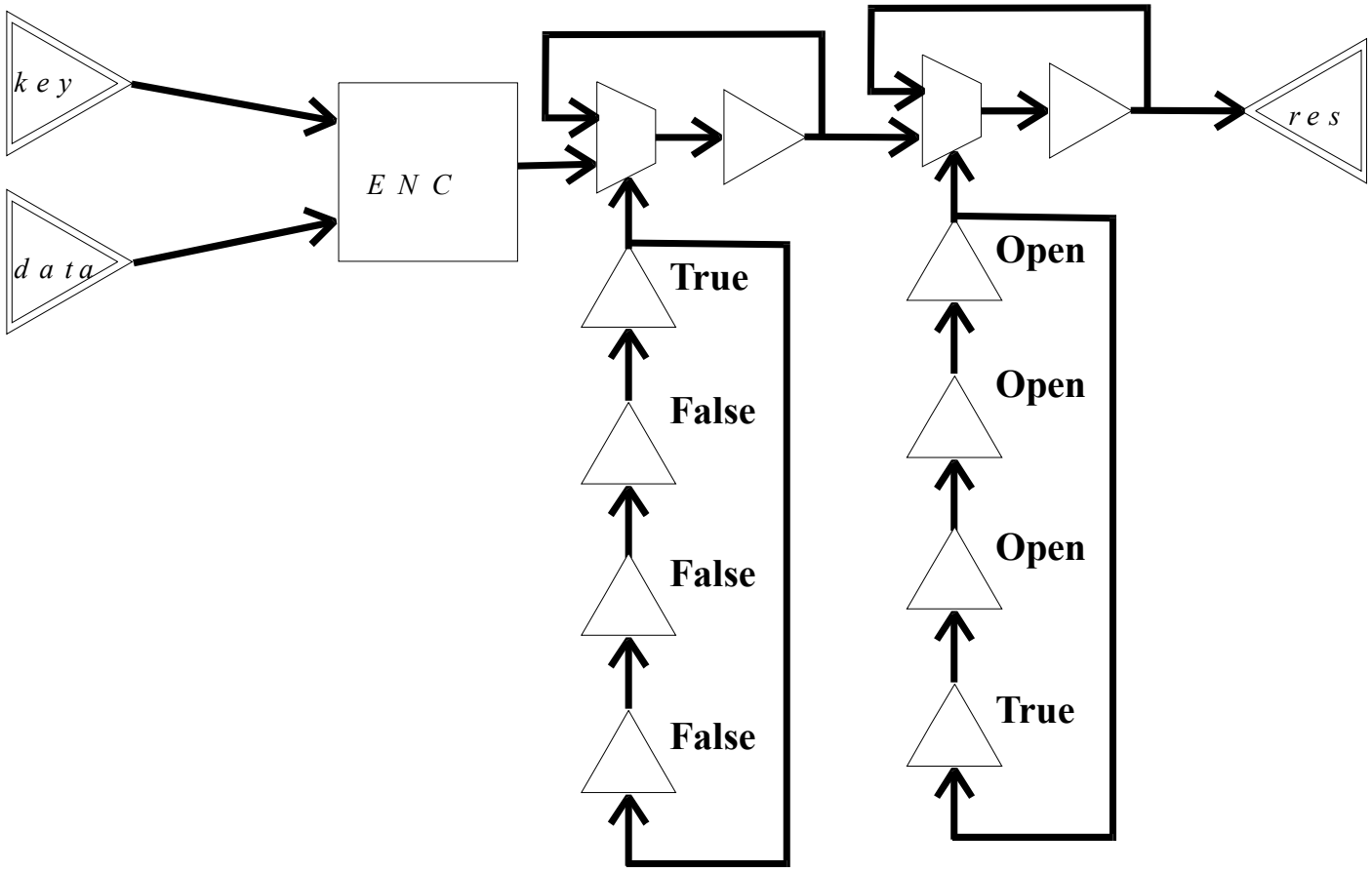

- Note: *Enc* Function is unimportant



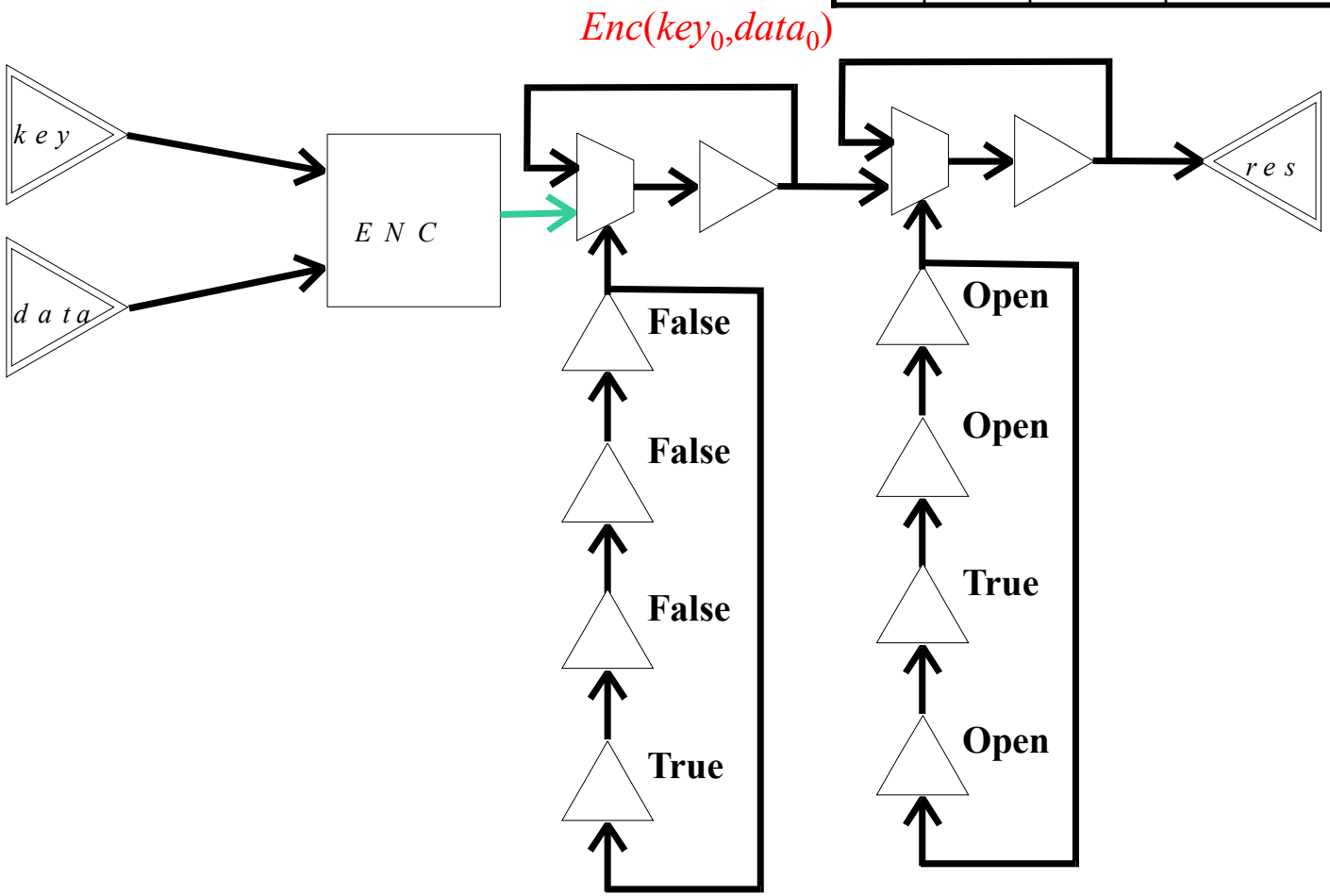
# Black Box Example 2

- Replacement Core:
  - Latency 4
  - Unpipelined (Throughput ¼)

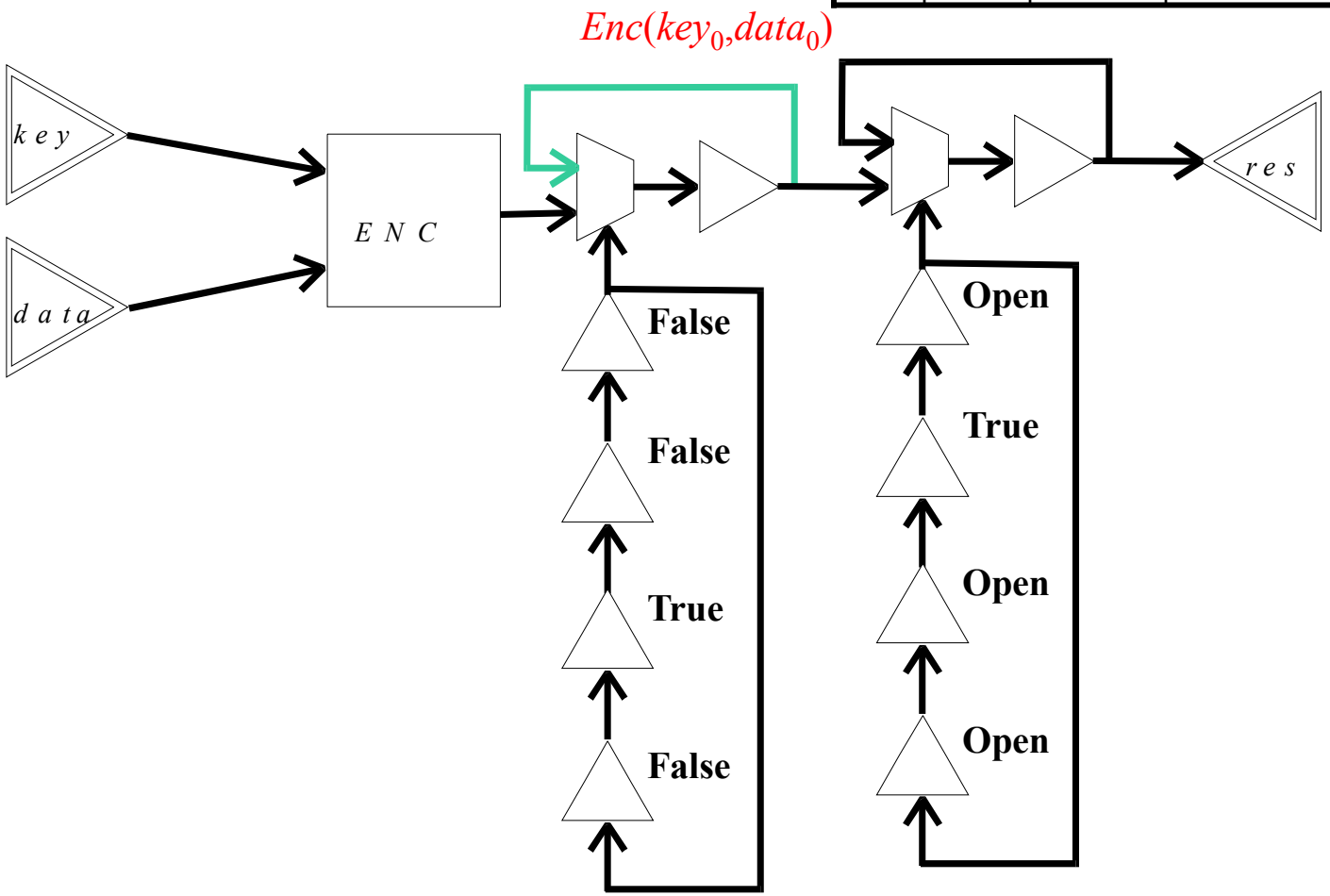
# Black Box Example 2



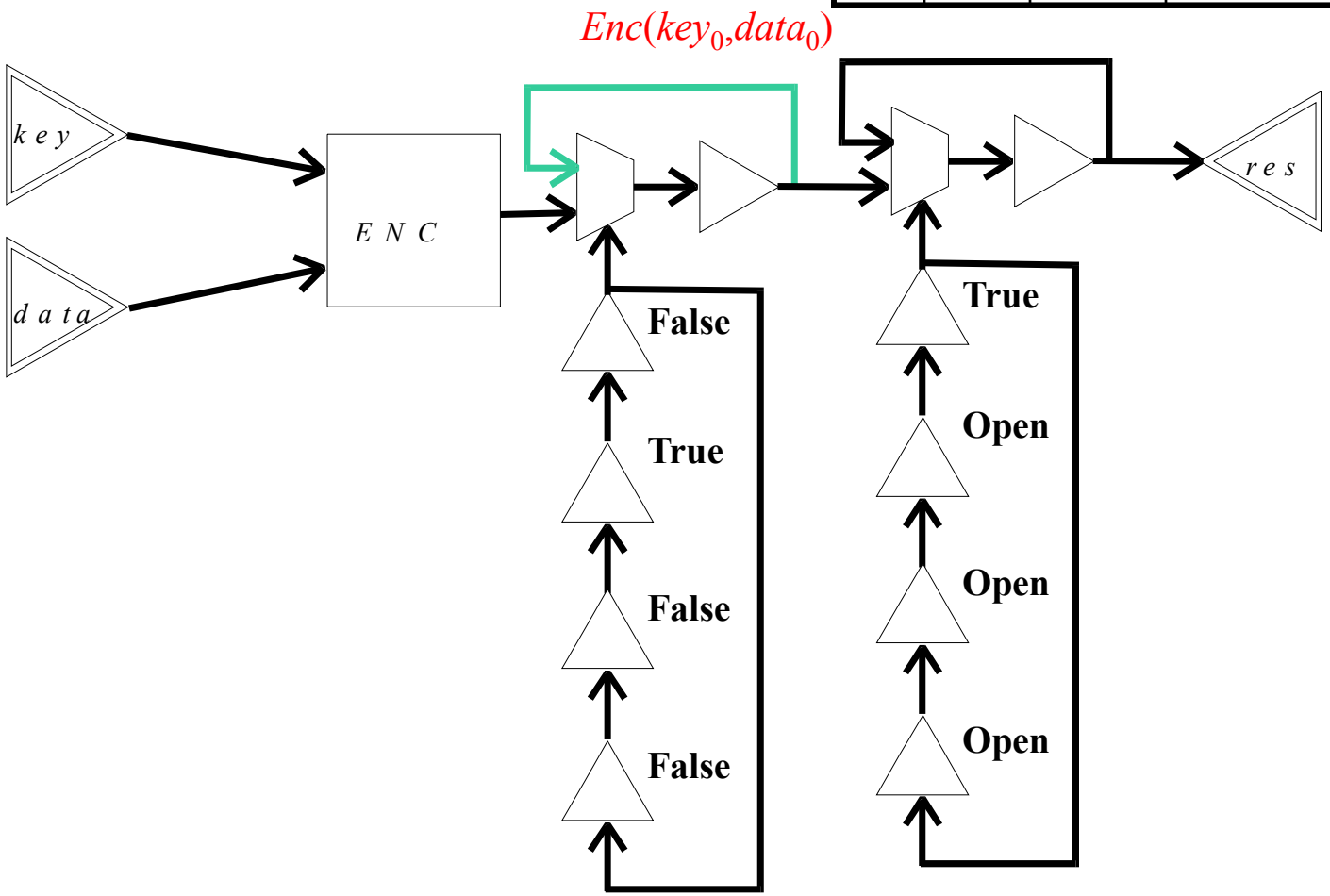
| CC | key | data | res |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| | | | |
| | | | |
| | | | |
| | | | |



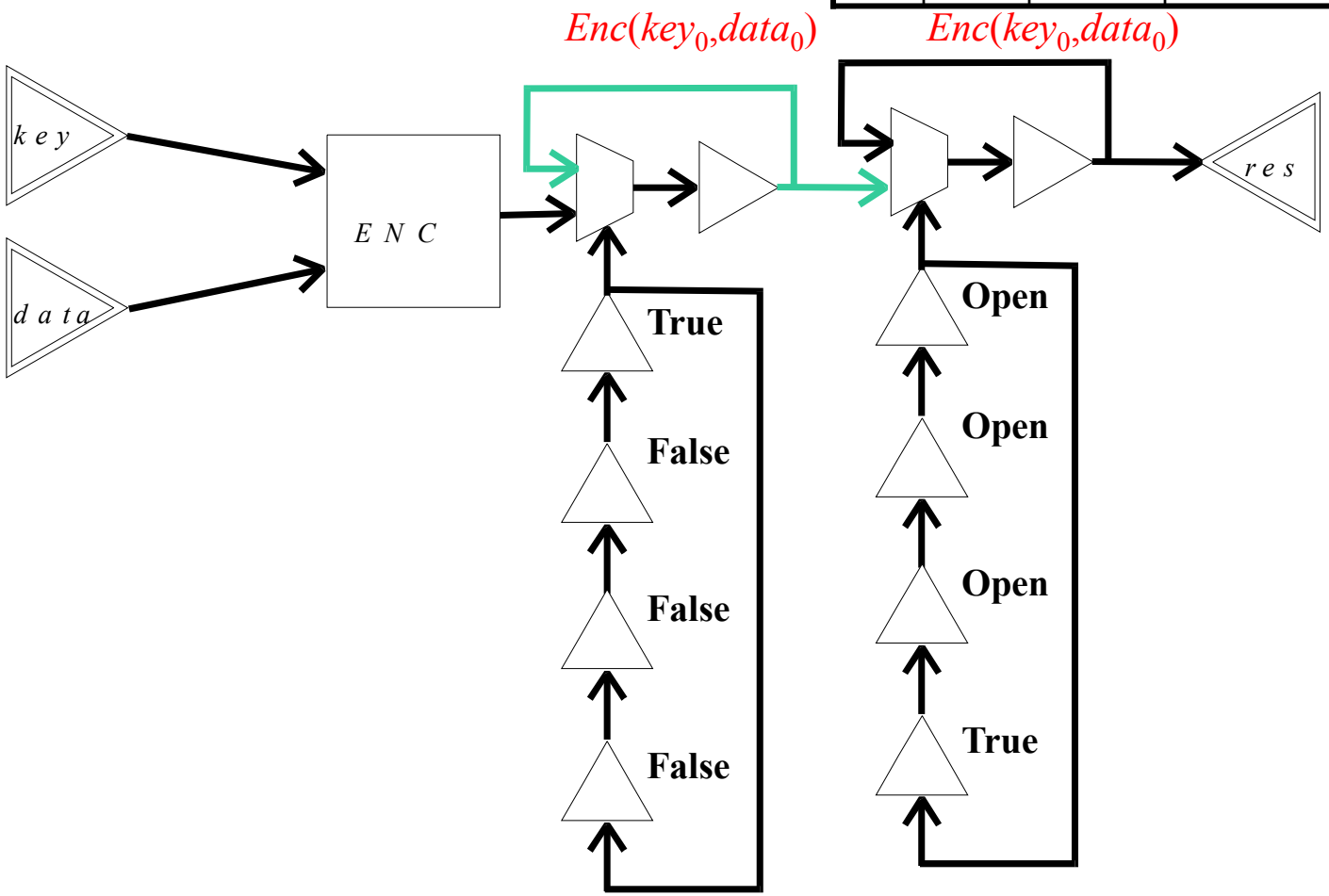
| CC | key | data | res |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| 1 | | | |
| | | | |
| | | | |
| | | | |



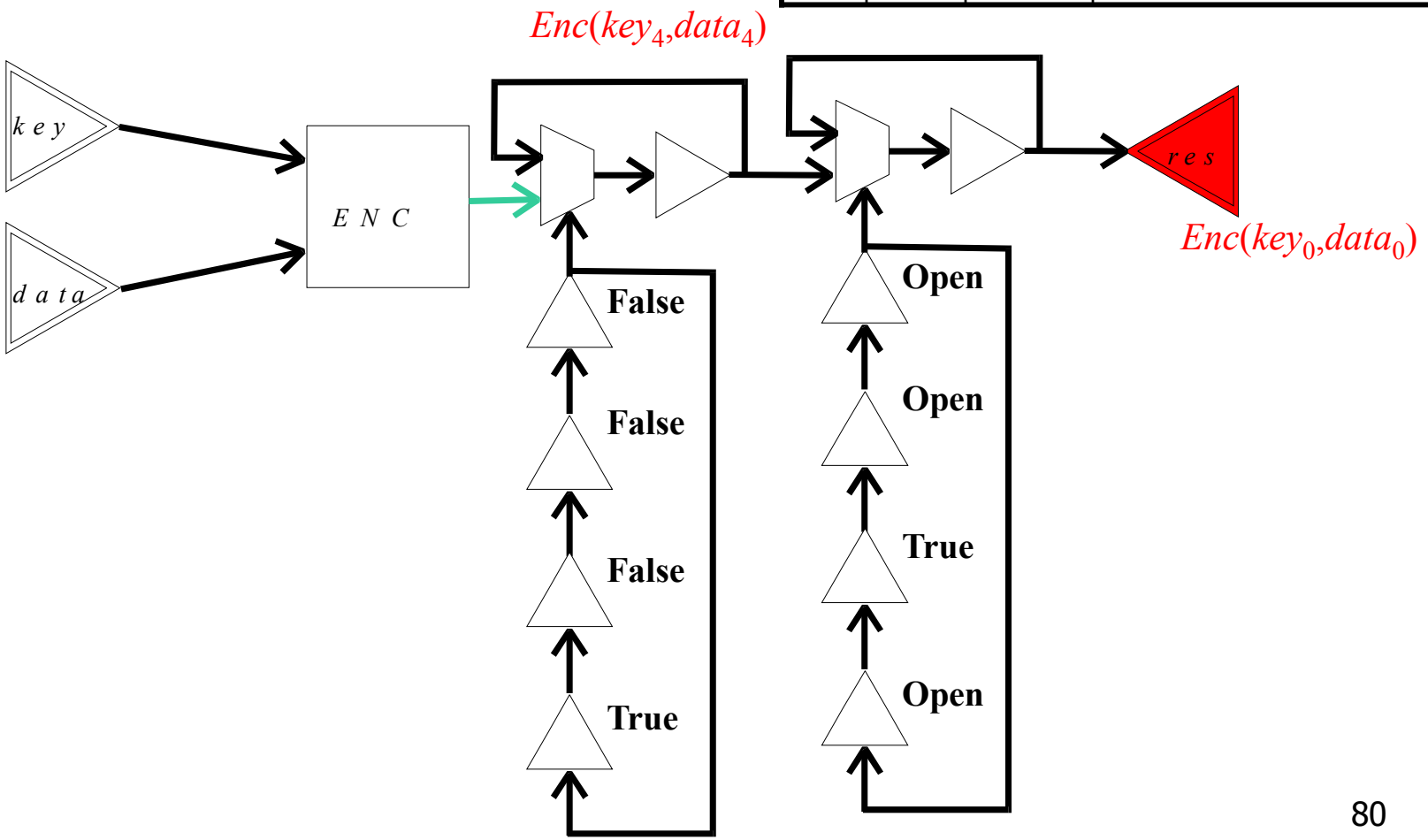
| CC | key | data | res |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| 1 | | | |
| 2 | | | |
| | | | |
| | | | |

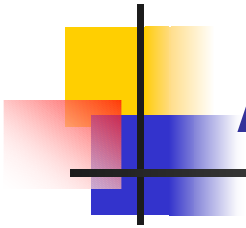


| CC | key | data | res |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| 1 | | | |
| 2 | | | |
| 3 | | | |
| | | | |



| CC | key | data | res |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | $key_4$ | $data_4$ | $Enc(key_0,data_0)$ |



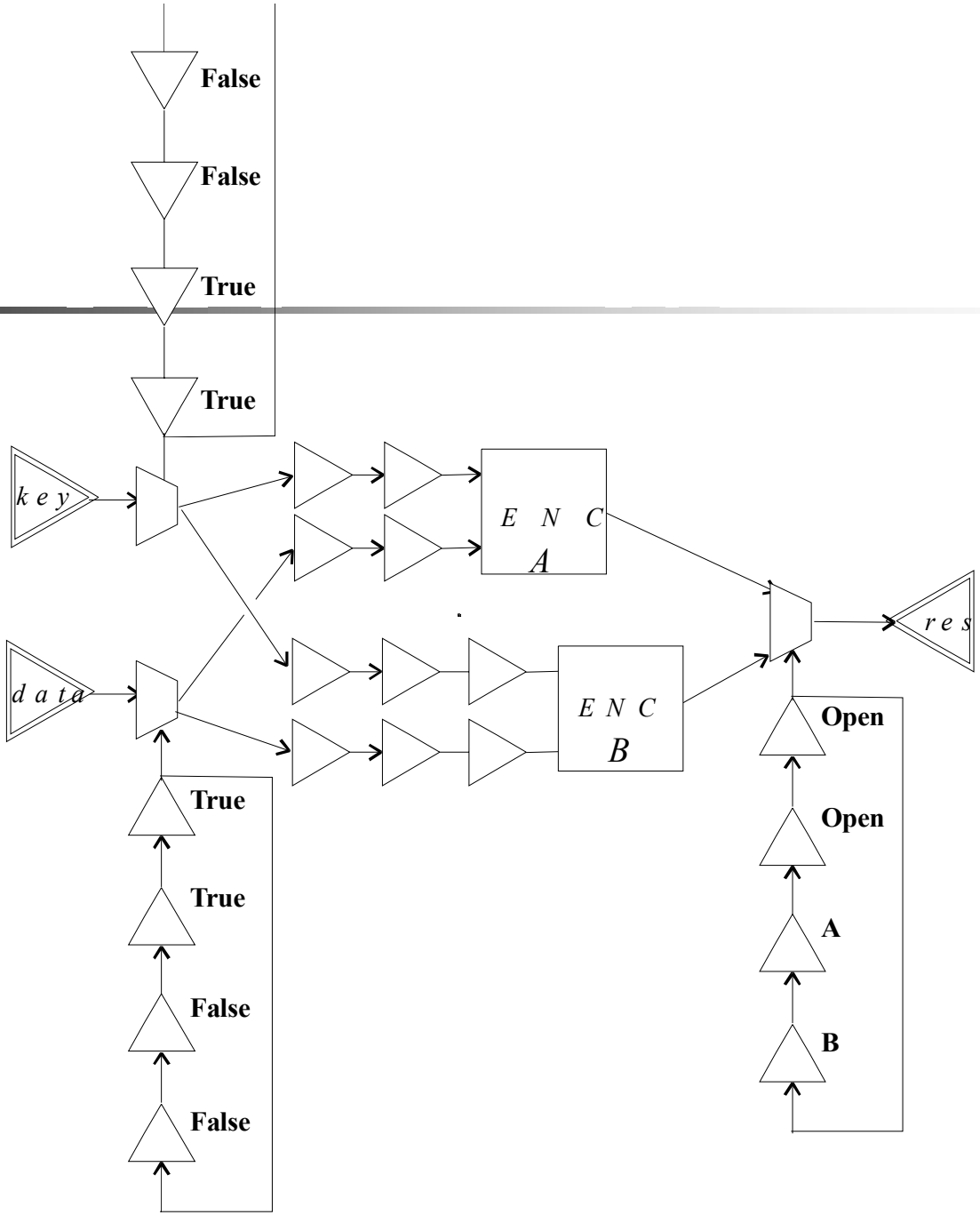


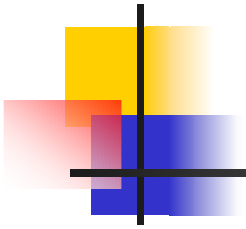
# Arbitrary Usage Patterns

- But what about extracting DPs from arbitrary usage patterns?
- Hard problem!
- CoreLoom can do it via an extremely ugly method
  - enumerates each result as a path
  - Places “guard” JPs on inputs/outputs

# Arbitrary UP Example

| *CC* | *key* | *data* | *res* |
|---|---|---|---|
| 0 | $key_0$ | $data_0$ | |
| 1 | $key_0$ | $data_0$ | |
| 2 | | | $Enc(key_0, data_0)$ |
| 3 | | | $Enc(key_1, data_1)$ |
| 4 | $key_4$ | $data_4$ | |
| 5 | $key_5$ | $data_5$ | |
| 6 | | | $Enc(key_4, data_4)$ |
| 7 | | | $Enc(key_5, data_5)$ |


False
False
True
True
key
data
E N C
A
E N C
B
res
Open
Open
A
B
True
True
False
False

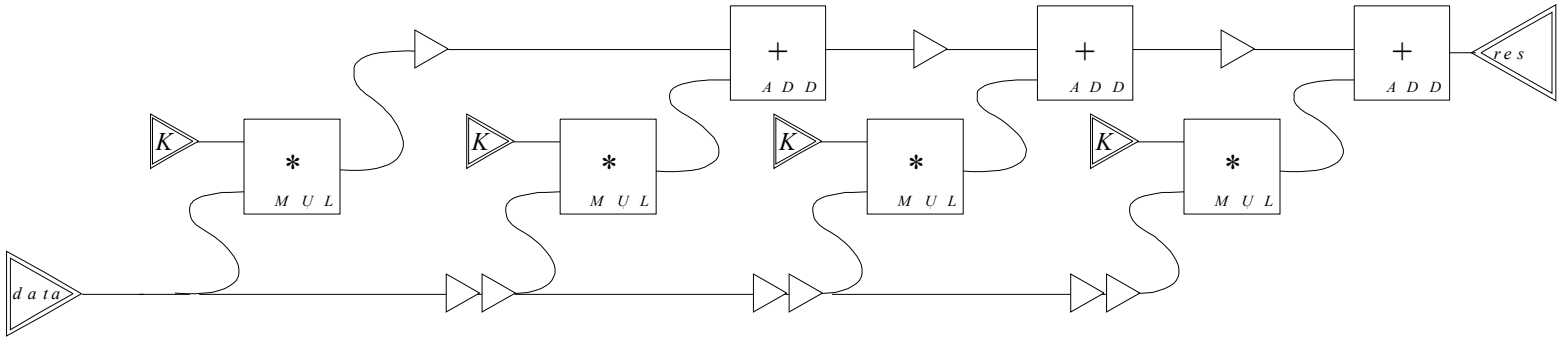


# Real-World Example

- AES Encryption Core
  - By Rudolf Usselmann
  - Donated to OpenCores.org
- Specification:
  - Separate Encryption/Decryption cores
  - Much overlapping hardware
- Used CoreLoom to combine the cores
- See Thesis Report

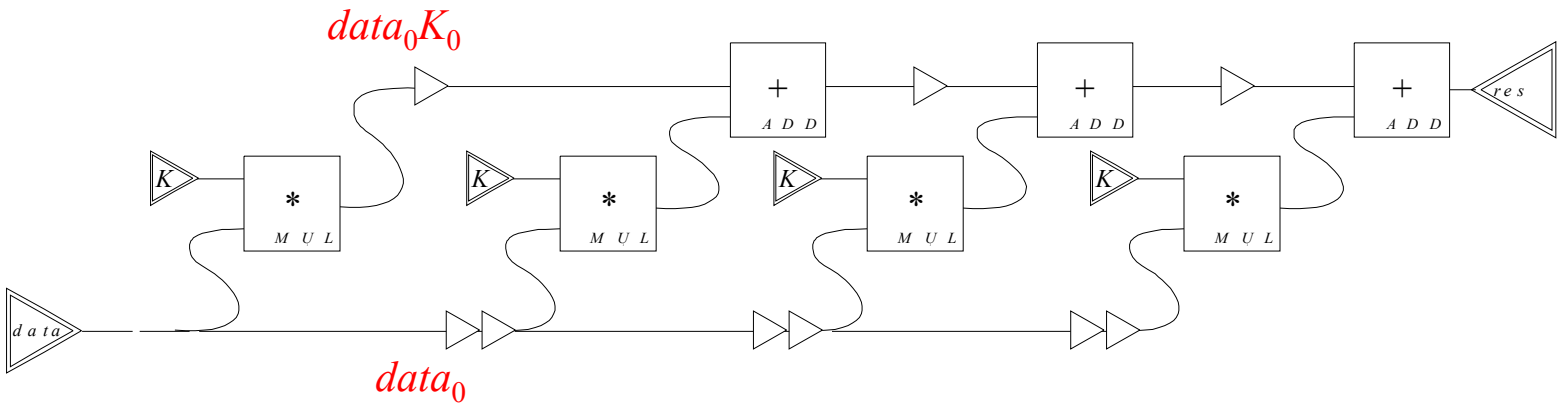
# FIR Filter Example

- FIR Filter in CoreLoom



# FIR Filter Example

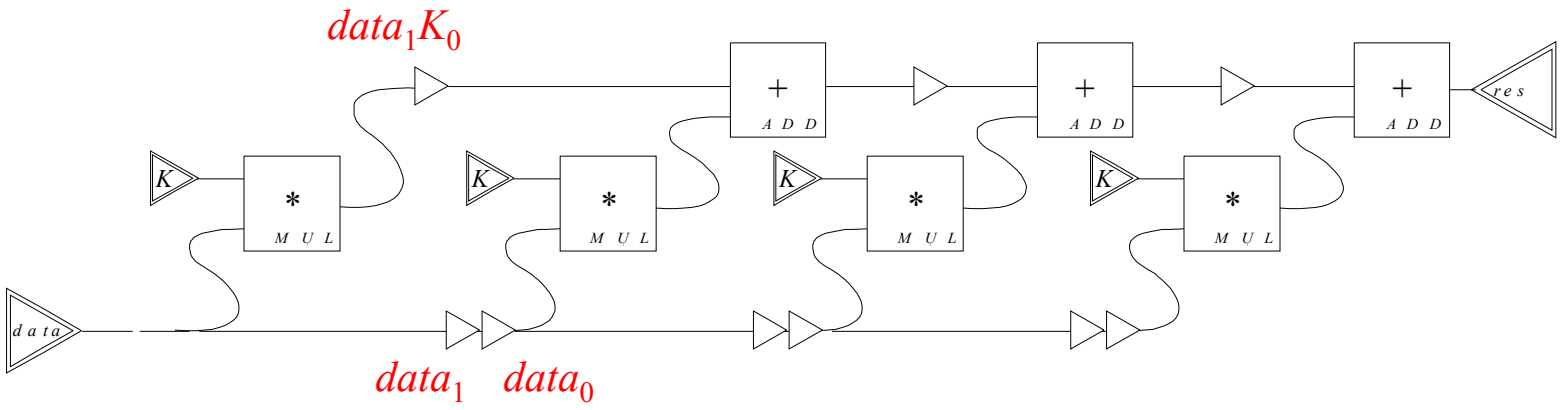
- FIR Filter in CoreLoom




- $res_0$ =

# FIR Filter Example

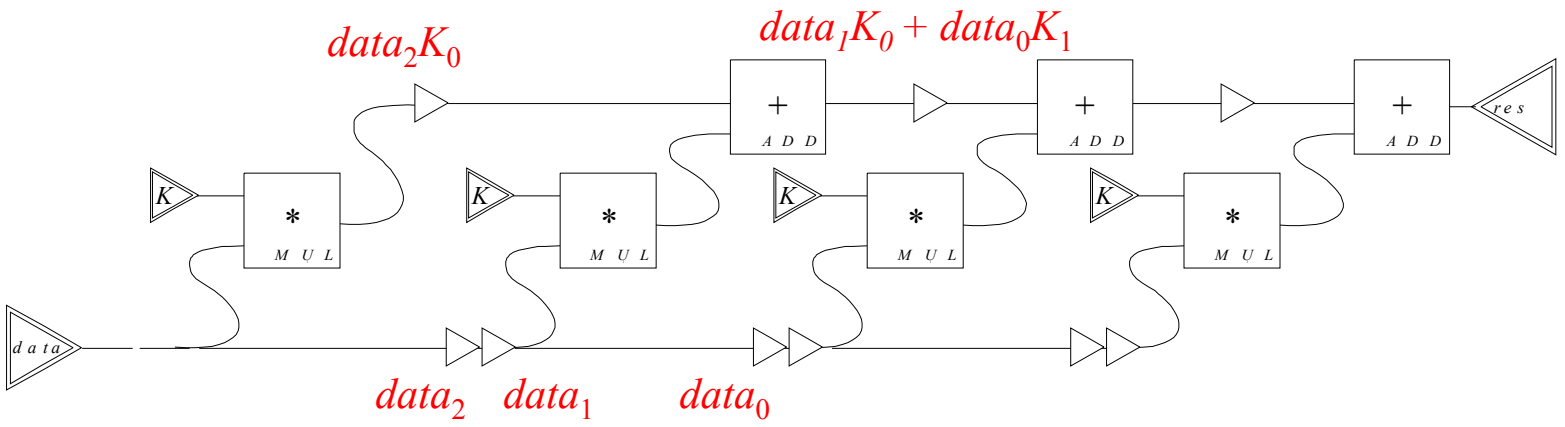
- FIR Filter in CoreLoom




- $res_0$ =
- $res_1$ =

# FIR Filter Example

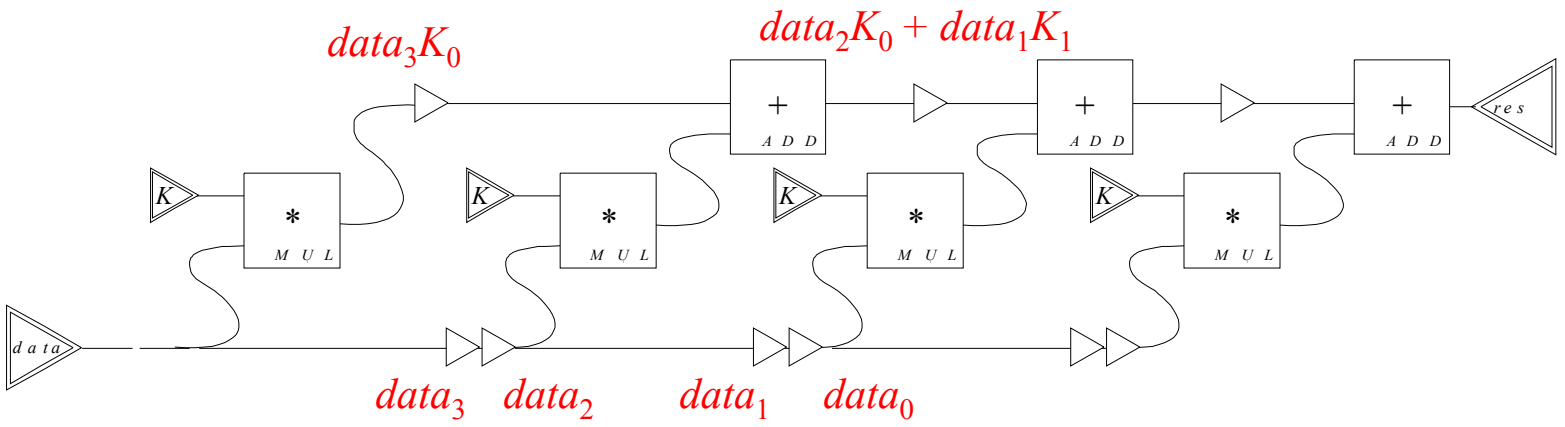
- FIR Filter in CoreLoom




- $res_0$ =
- $res_1$ =
- $res_2$ =

# FIR Filter Example

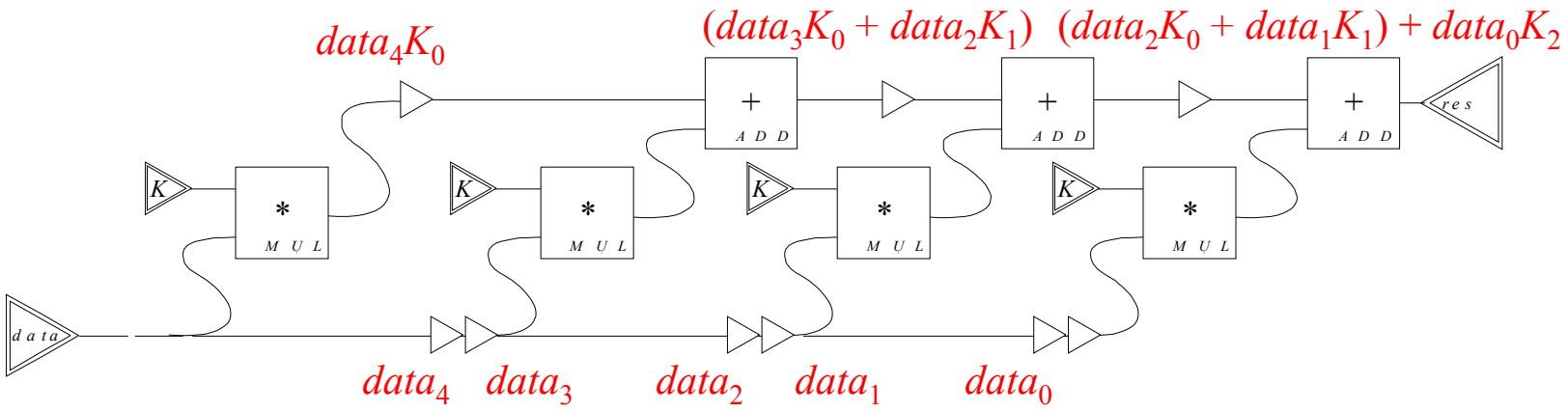
- FIR Filter in CoreLoom




- $res_1$ =
- $res_2$ =
- $res_3$ =

# FIR Filter Example

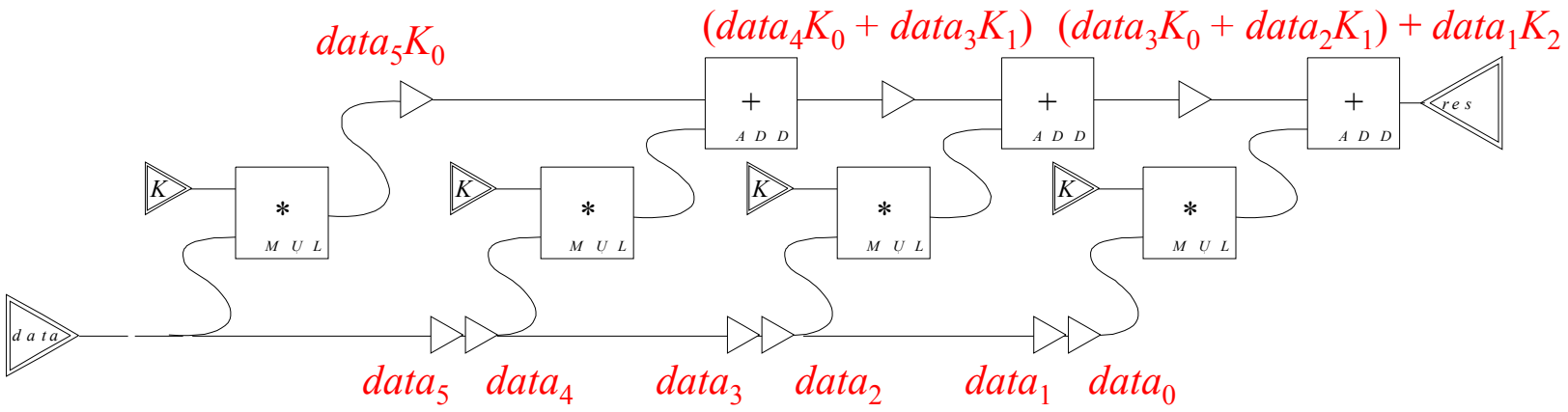
- FIR Filter in CoreLoom




- $res_2$ =
- $res_3$ =
- $res_4$ =

# FIR Filter Example

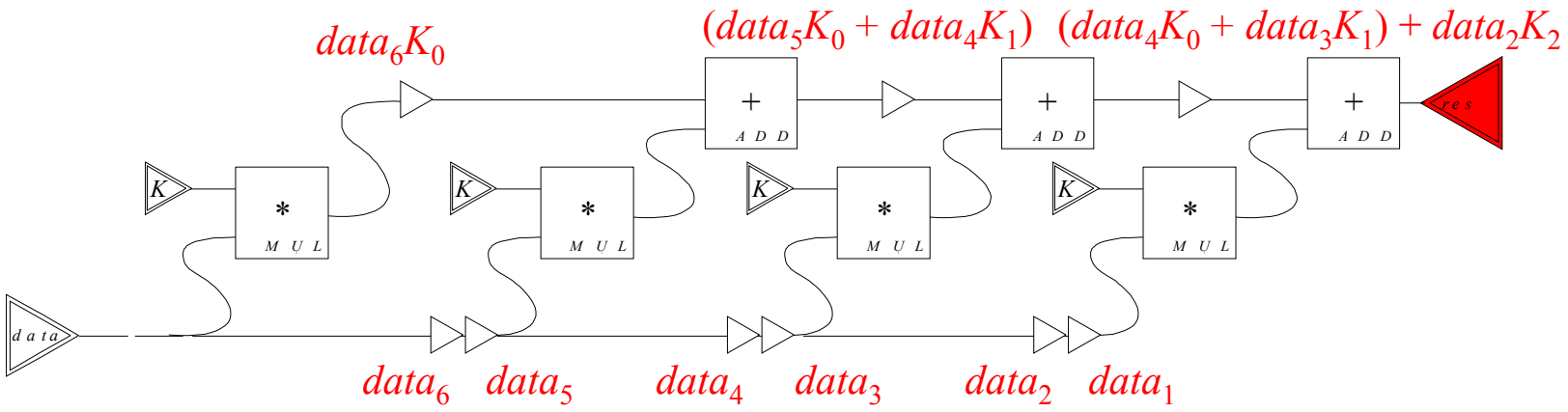
- FIR Filter in CoreLoom




- $res_3$ =
- $res_4$ =
- $res_5$ =

# FIR Filter Example

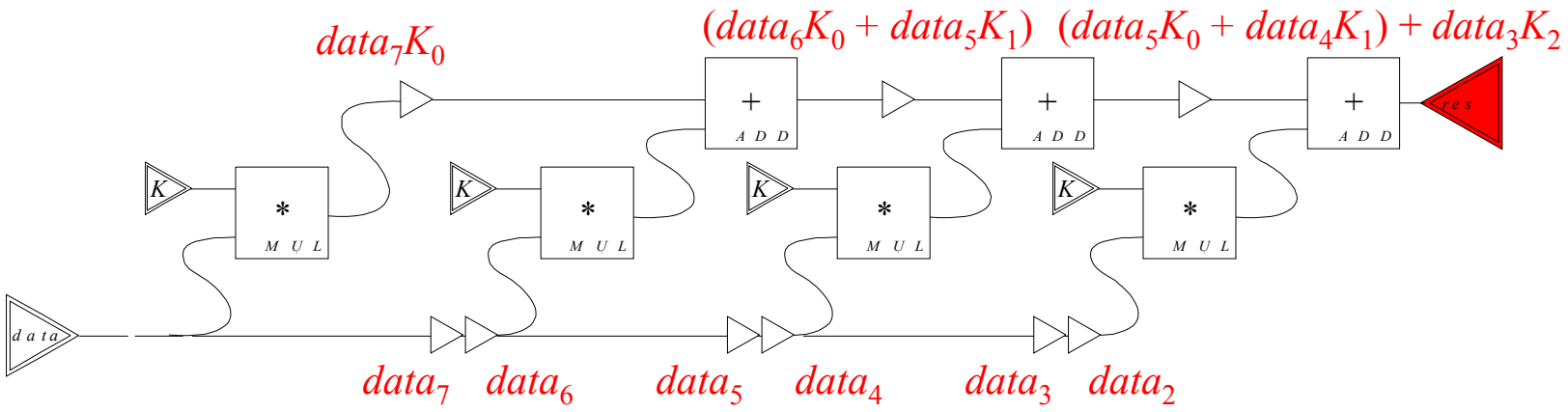
- FIR Filter in CoreLoom




- $res_4$ =
- $res_5$ =
- $res_6$ = $((data_3K_0 + data_2K_1) + data_1K_2) + data_0K_3$

# FIR Filter Example

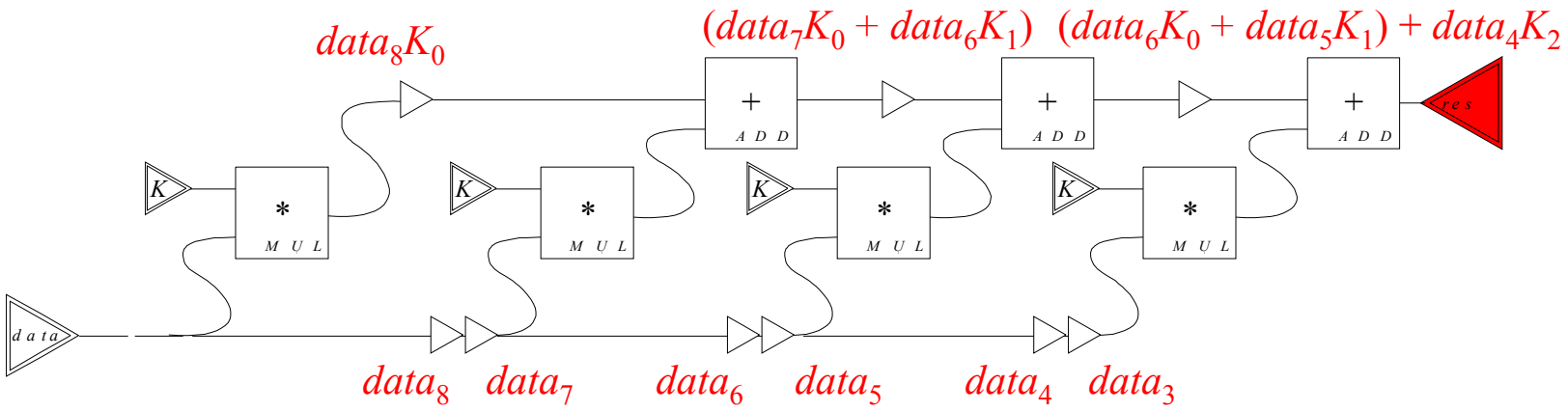
- FIR Filter in CoreLoom



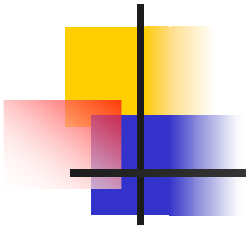

- $res_5$ =
- $res_6$ = $((data_3K_0 + data_2K_1) + data_1K_2) + data_0K_3$
- $res_7$ = $((data_4K_0 + data_3K_1) + data_2K_2) + data_1K_3$

# FIR Filter Example

- FIR Filter in CoreLoom




- $res_6 = ((data_3K_0 + data_2K_1) + data_1K_2) + data_0K_3$
- $res_7 = ((data_4K_0 + data_3K_1) + data_2K_2) + data_1K_3$
- $res_8 = ((data_5K_0 + data_4K_1) + data_3K_2) + data_2K_3$

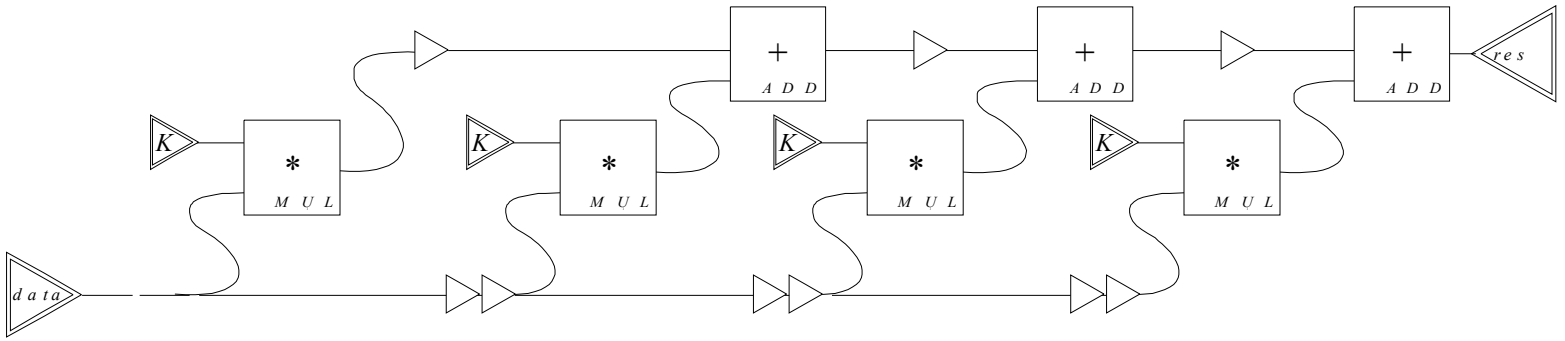


# FIR Filter Example

- Situation:
  - SoC must also do basic vector operations
    - [A B C D] n = [An Bn Cn Dn]
    - [A B C D] + [X Y Z W] = [A+X B+Y C+Z D+W]
    - [A B C] • [X Y Z] = AX + BY + CZ
    - Investigate whether FIR Filter core can be extended to do this

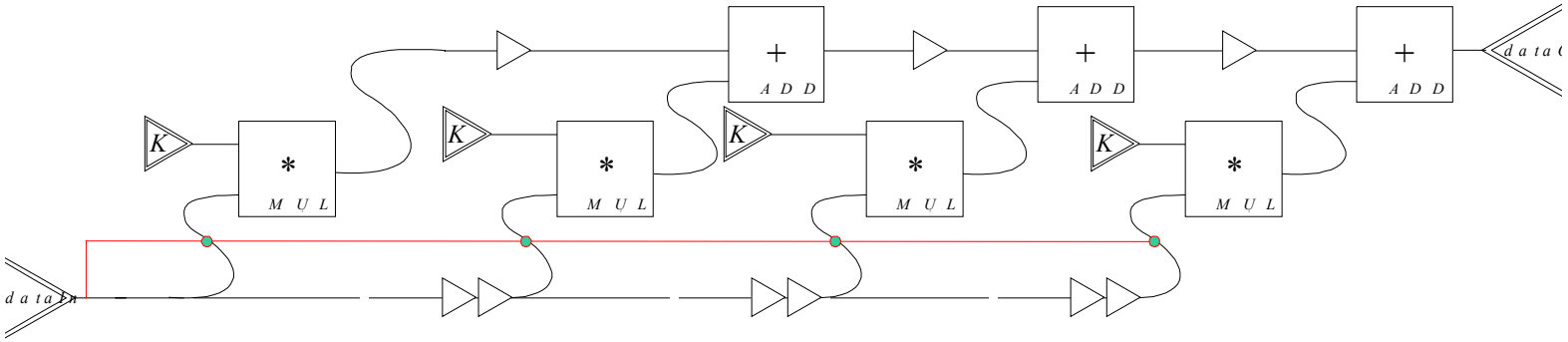
# Adding Vector-Scalar Multiply

- [A B C D] n = [An Bn Cn Dn]



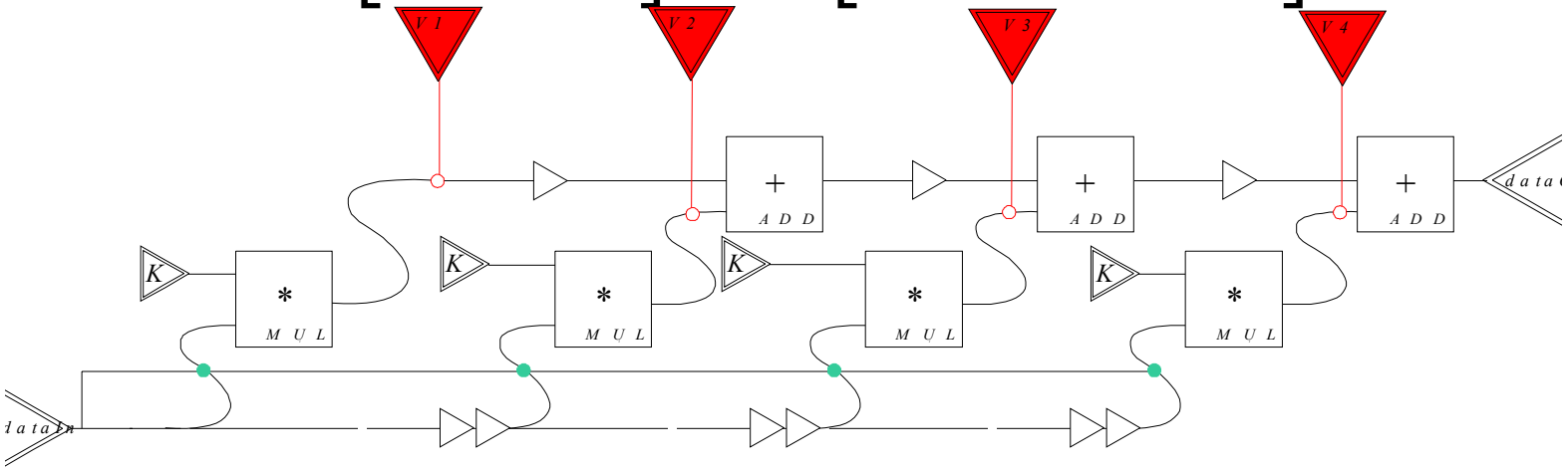
# Adding Vector-Scalar Multiply

- [A B C D] n = [An Bn Cn Dn]



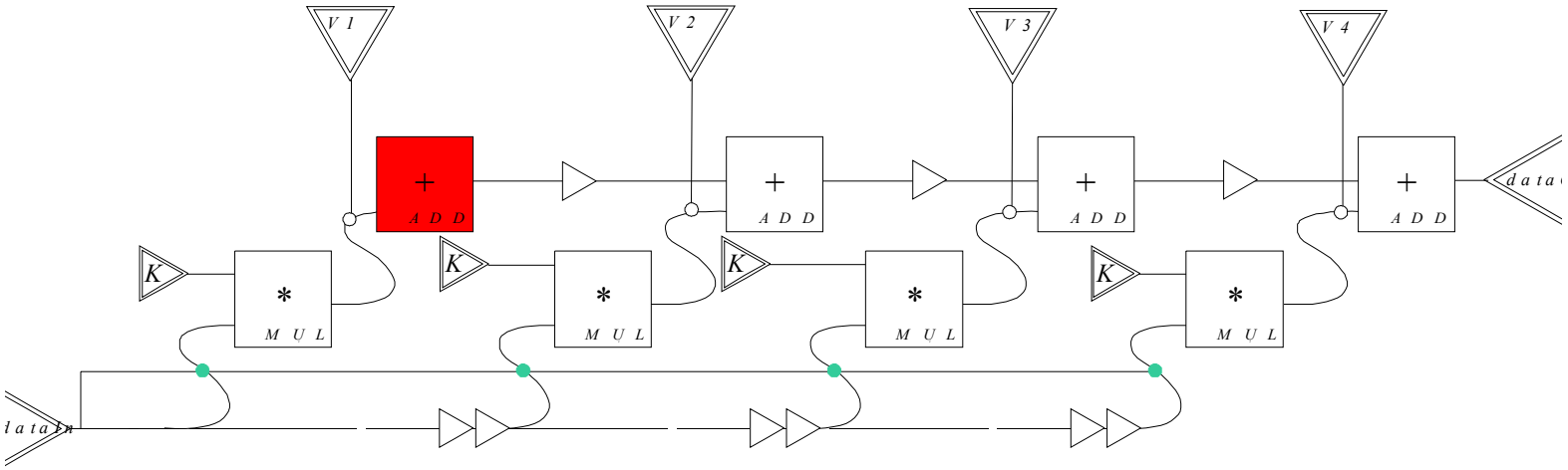
# Adding Vector-Scalar Multiply

- [A B C D] n = [An Bn Cn Dn]



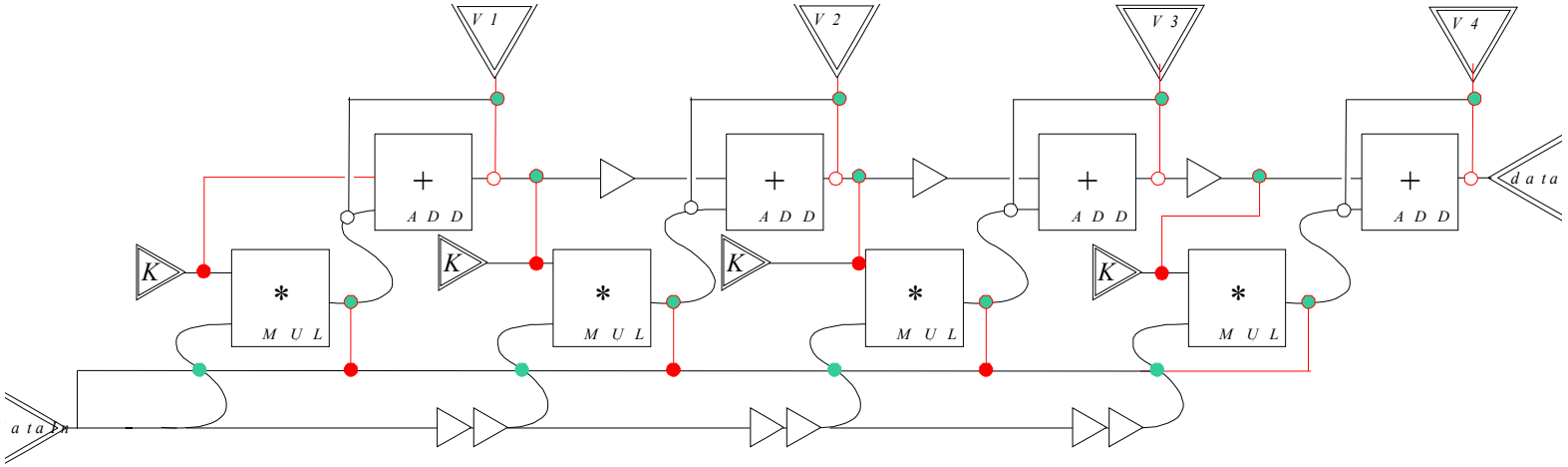
# Adding Vector-Scalar Addition

- [A B C D]+n = [A+n B+n C+n D+n]



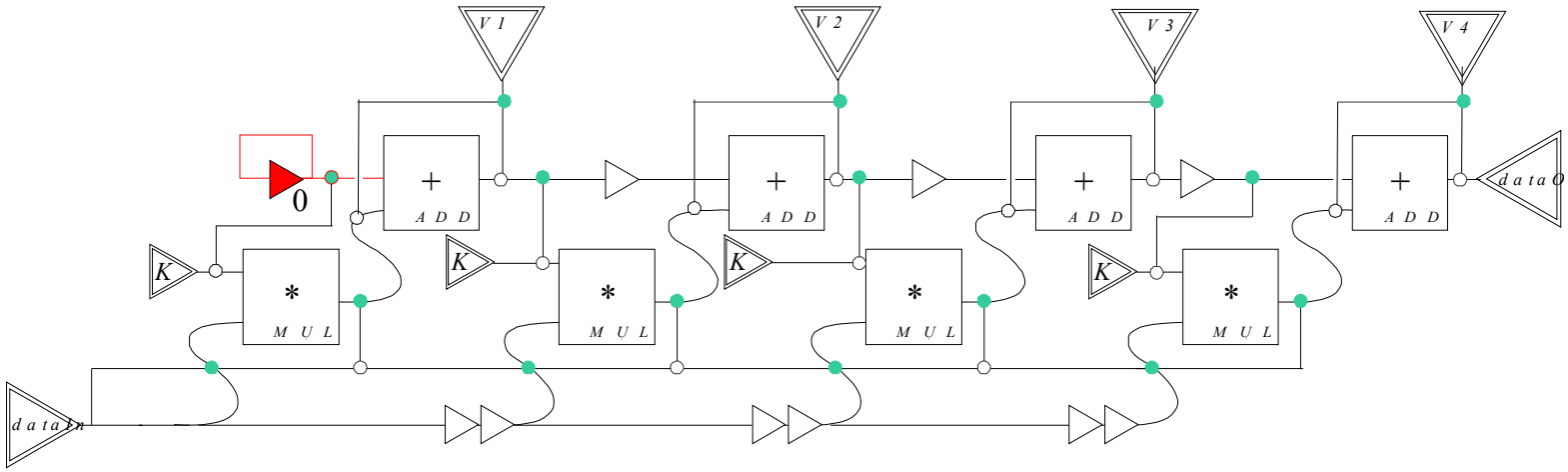
# Adding Vector-Scalar Addition

- [A B C D]+n = [A+n B+n C+n D+n]



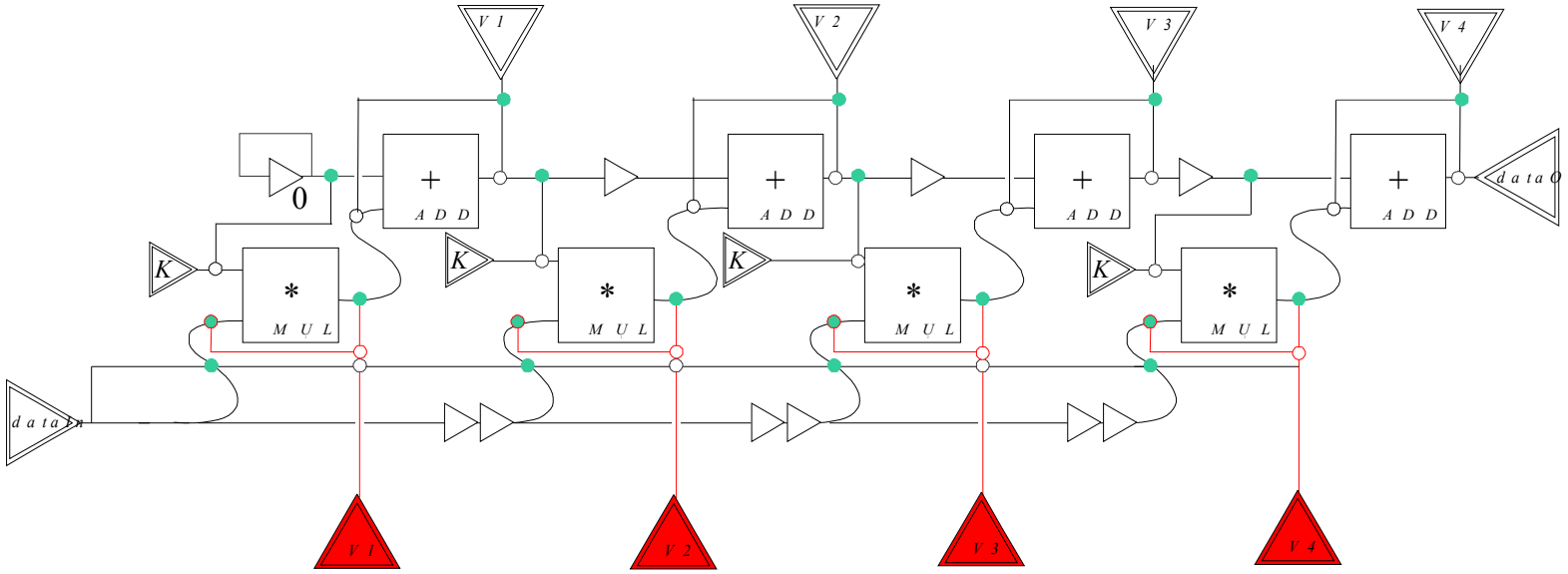
# Adding Vector-Scalar Addition

- [A B C D]+n = [A+n B+n C+n D+n]



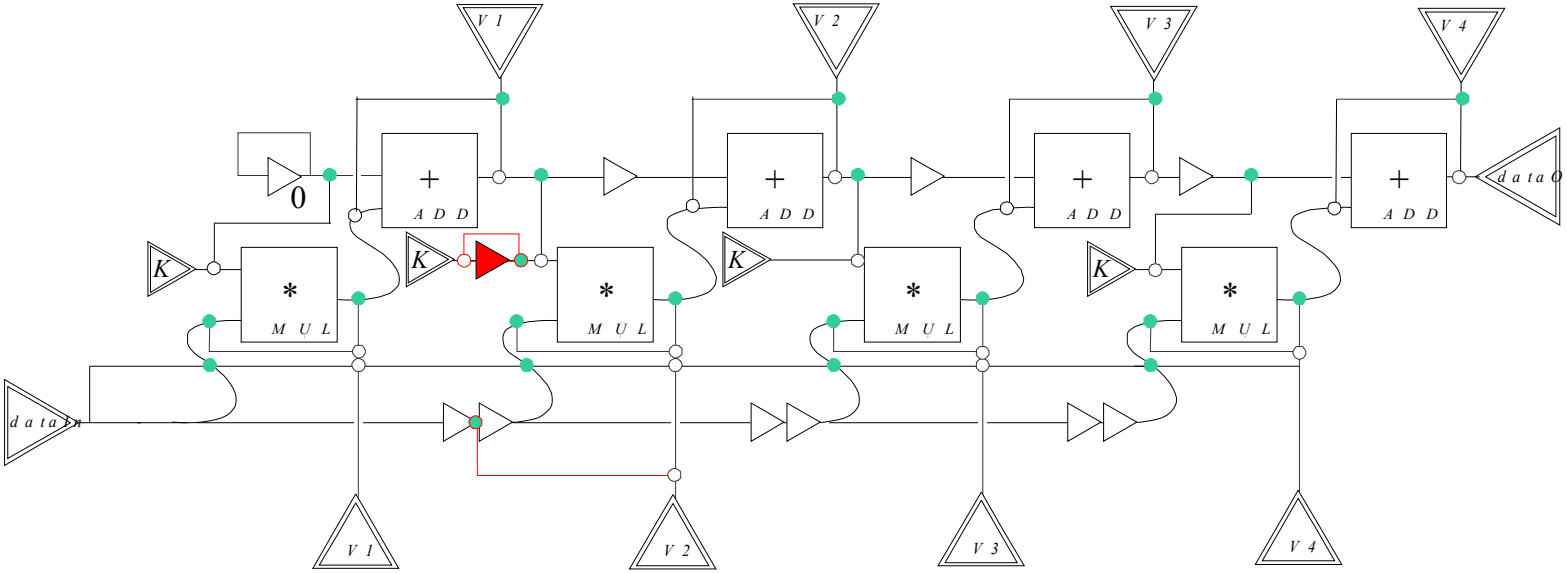
# Vector-Vector Operations

- [A B C D] + [X Y Z W] = [A+X B+Y C+Z D+W]
- [A B C D] * [X Y Z W] = [AX BY CZ DW]



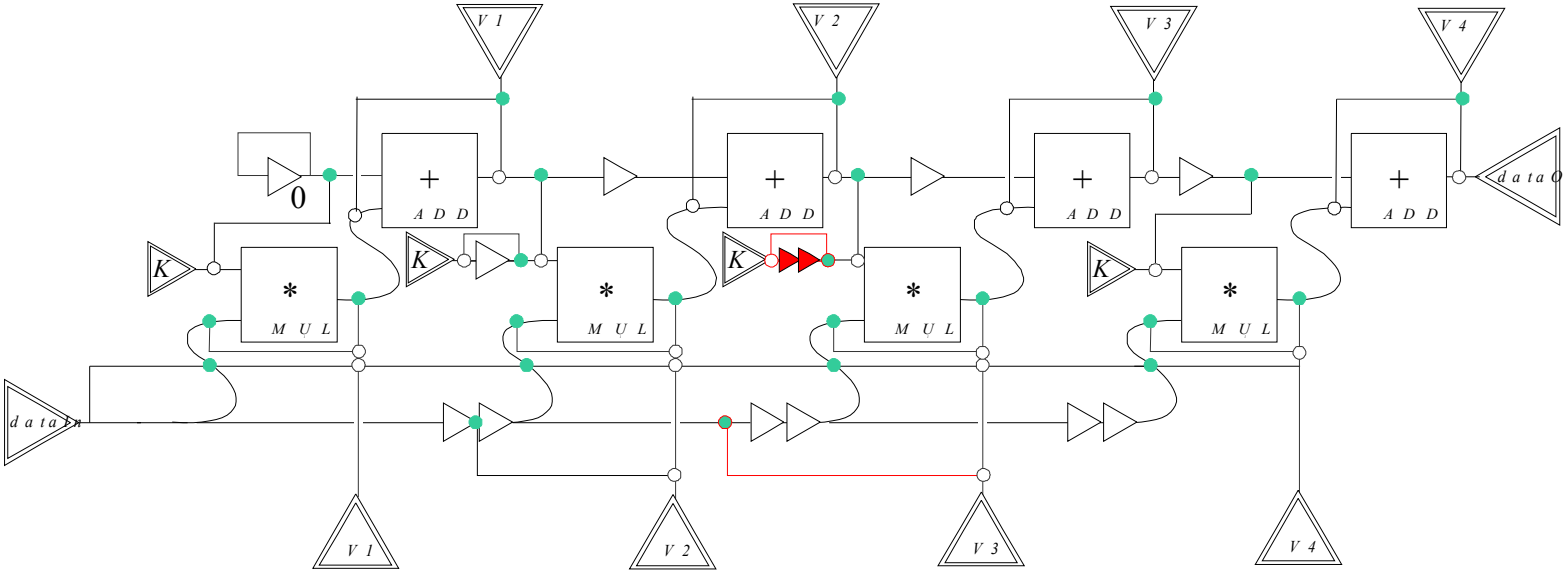
# Adding Dot Product

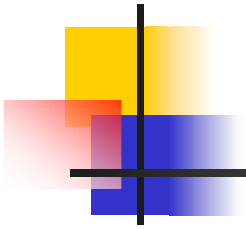
- [A B C] • [X Y Z] = AX + BY + CZ



# Adding Dot Product

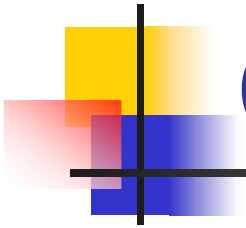
- [A B C] • [X Y Z] = AX + BY + CZ





# FlexFIR Summary

- Final Version:
  - 22 Join Points
  - $2^{22}$ possible control words (!)
  - Capable of doing FIR with original L/TP
  - Certainly has higher circuit area, critical path, etc.
- Architect must decide:
  - Is the combined core more efficient *system-wide* than using separate cores



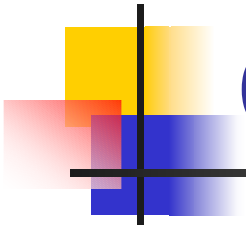
# Configuration Characteristics

- Motivation:
  - FlexSoC will perform configuration compression
  - Compression technique depends on typical config
- Explored characteristics
  - Width
  - Length
  - Variance
    - # of unique control words per configuration

# AES + FIR Characteristics

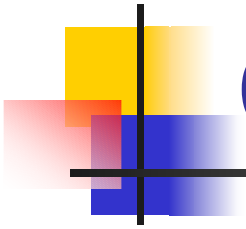
| Config | Width | Length | Unique CWs |
|---|---|---|---|
| aes_enc | 3 | 12 | 3 |
| aes_dec | 3 | 24 | 5 |
| FIR | 16 | 1 | 1 |
| VSMUL | 16 | 1 | 1 |
| VSADD | 16 | 1 | 1 |
| VVMUL | 16 | 1 | 1 |
| VVADD | 16 | 1 | 1 |
| FIR_3 | 16 | 1 | 1 |
| DPROD | 20 | 1 | 1 |

| Config | Width | Unscheduled | | Scheduled | |
|---|---|---|---|---|---|
| | | Length | Unique CWs | Length | Unique CWs |
| dp0_flat | 2 | 2 | 2 | 1 | 1 |
| dp0_loop | 2 | 4 | 3 | 3 | 2 |
| dp1_flat | 3 | 4 | 3 | 1 | 1 |
| dp1_sched | 3 | 7 | 4 | 6 | 3 |
| dotP_dotP | 4 | 1 | 1 | 1 | 1 |
| dotP_back | 4 | 3 | 3 | 4 | 2 |
| dotP_stuck | 4 | 1 | 1 | 1 | 1 |
| rg_dotP_flat | 7 | 3 | 4 | 1 | 1 |
| rg_dotP_rep | 7 | 5 | 5 | 3 | 3 |
| rg_dotP_over | 7 | 5 | 4 | 4 | 2 |
| rg_dotP_nosched | 7 | 5 | 5 | 5 | 5 |
| Average | 4.5 | 3.6 | 3 | 2.7 | 2 |



# Configuration Characteristics

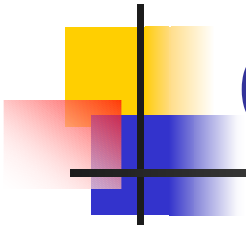
- Conclusion: 2 general types
  - Long/Narrow vs Short/Wide
    - ie
  - Unpipelined vs Heavily pipelined
    - ie
  - Reuse FUs vs add FUs
  - Perhaps one approach should be encouraged
- Third type: Medium/Medium
  - Scheduling affects characteristics
  - represents dps not being used for original purpose



# Conclusions

- CoreLoom model exposed interesting research questions:
  - Expressing functional vs non-functional capabilities
  - Automatic translation of configs across datapaths
  - A metric of flexibility which handles states/interconnects
  - Automatic inference of control/communication units
  - Creating optimum datapaths from given UPs
  - Expanding the model to handle limited non-determinism, or multiple clock domains

However…



# Conclusions

- Should have been a graphical language
  - Currently too easy to misconnect
- Model limitations are extreme
  - Currently impractical for synthesis
  - Needs bit widths, formats, exceptions, etc
- A balance must be found between:
  - level of detail
  - elegance of model