
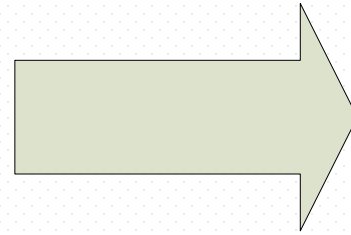


CGRA
COOKBOOK 2018



Task

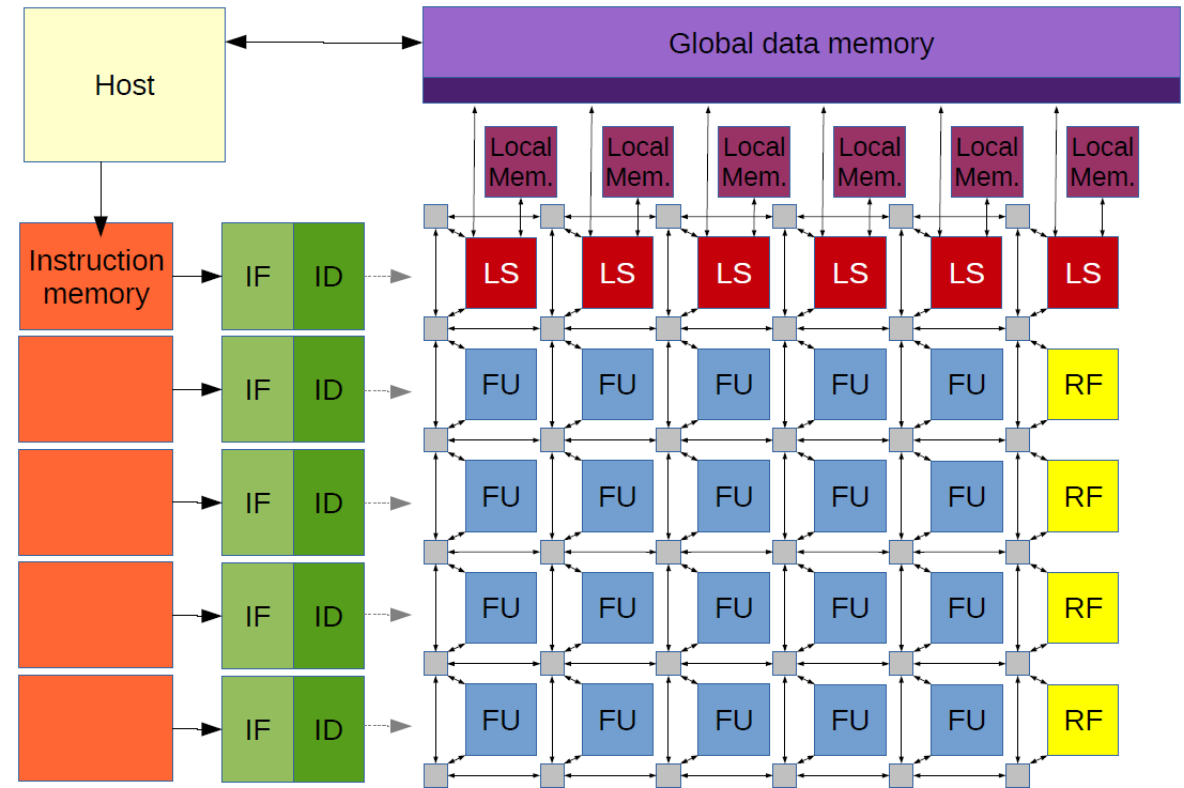
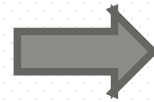
Implementing Edge Detection on the CGRA



Task

Implementing Edge Detection on the CGRA

Naive implementation of Edge Detection



Goal

Finding the best performance for minimal energy & area.

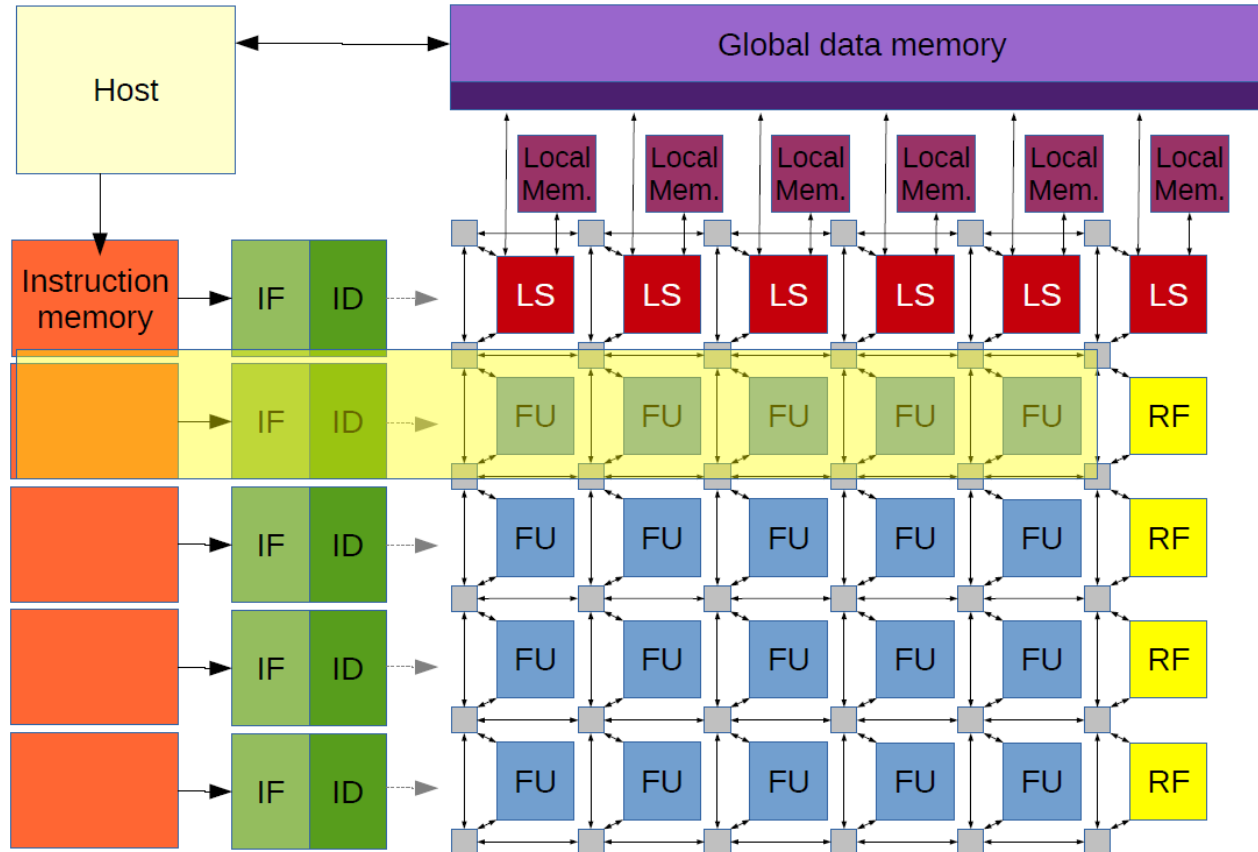
Where can you optimize?

Change the
algorithm.

Change the
architecture.

Optimized
code
mapping.

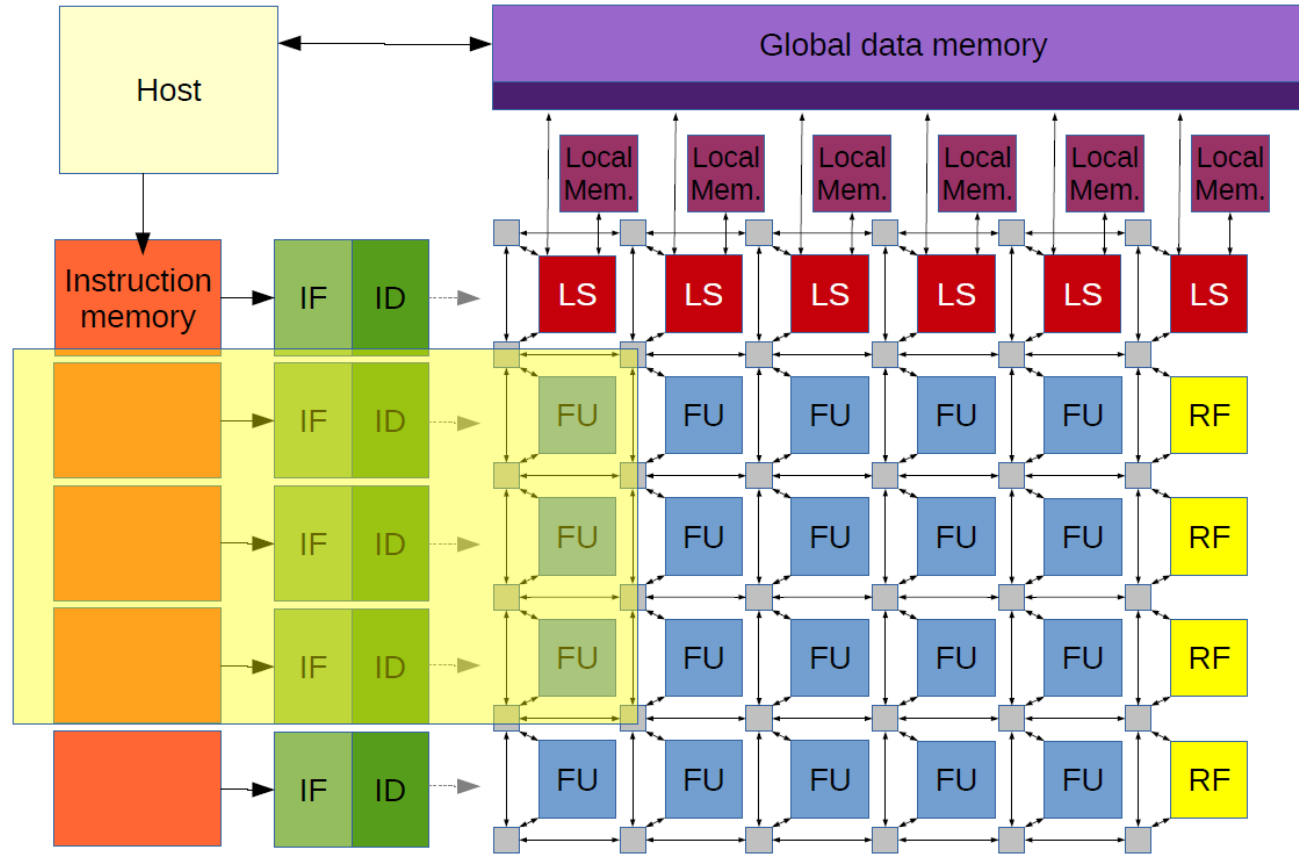
Changing the architecture.



SIMD

Exploit data level parallelism (DLP)

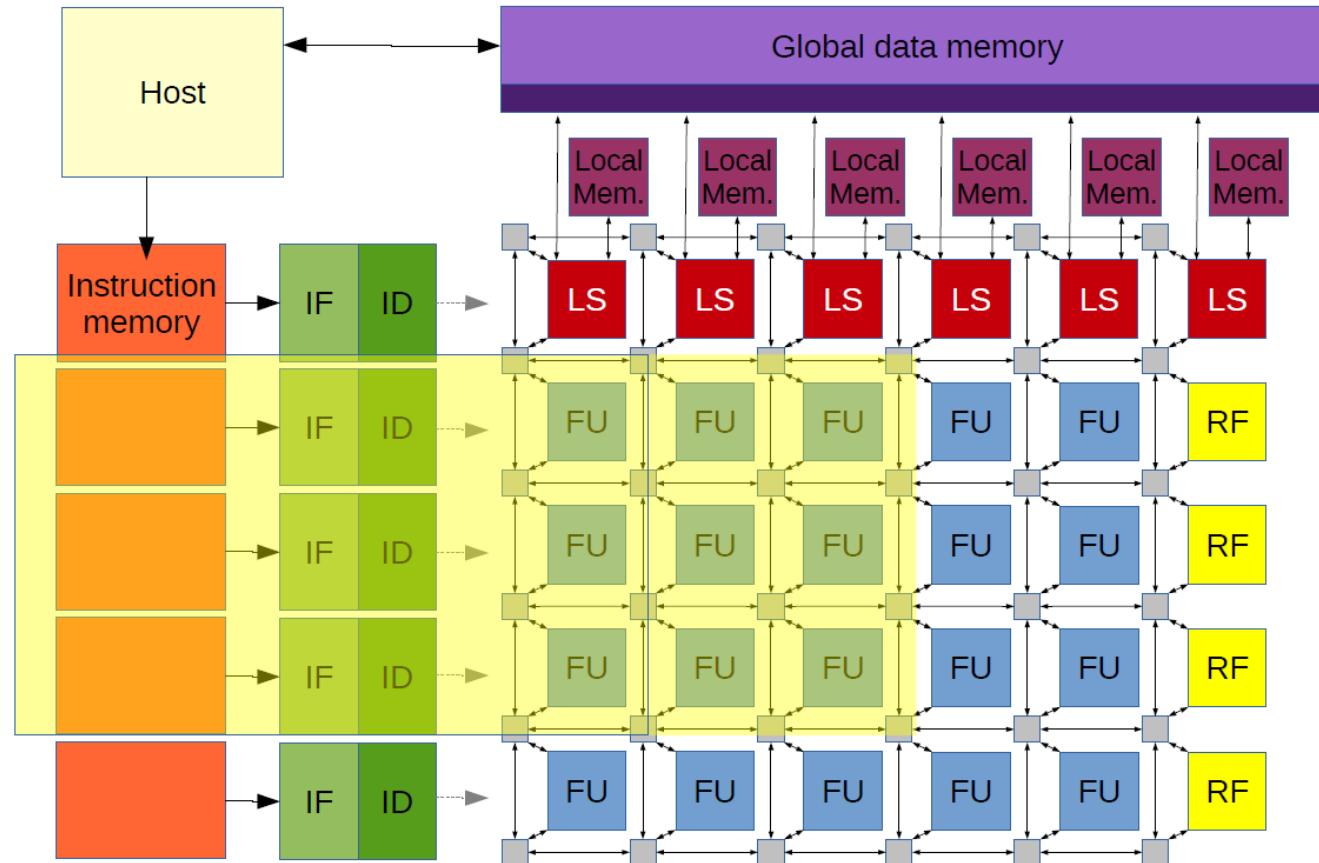
Changing the architecture.



VLIW

Exploit instruction level
parallelism (ILP)

Changing the architecture.



ILP + DLP

Changing the architecture.

```
<fu type="ID" name='id_abu'> <!-- IDs -->
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_lsu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_alu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_mul'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_rf'>
  <input index="0" source="abu.1"/>
</fu>

<fu type="LSU" name='lsu' ID="id_lsu"> <!-- FUs -->
  <input index="0" source="imm.0"/>
  <input index="1" source="rf.1"/>
  <input index="2" source="rf.0"/>
</fu>
<fu type="ALU" name='alu' ID="id_alu" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="MUL" name='mul' ID="id_mul" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="RF" name='rf' ID="id_rf">
  <input index="0" source="imm.0"/>
  <input index="1" source="lsu.1"/>
  <input index="2" source="alu.1"/>
  <input index="3" source="mul.0"/>
</fu>
<fu type="ABU" name='abu' ID="id_abu" config="1">
  <input index="0" source="alu.1"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="IU" name='imm'>
  <input index="0" source="abu.1"/>
</fu>
```

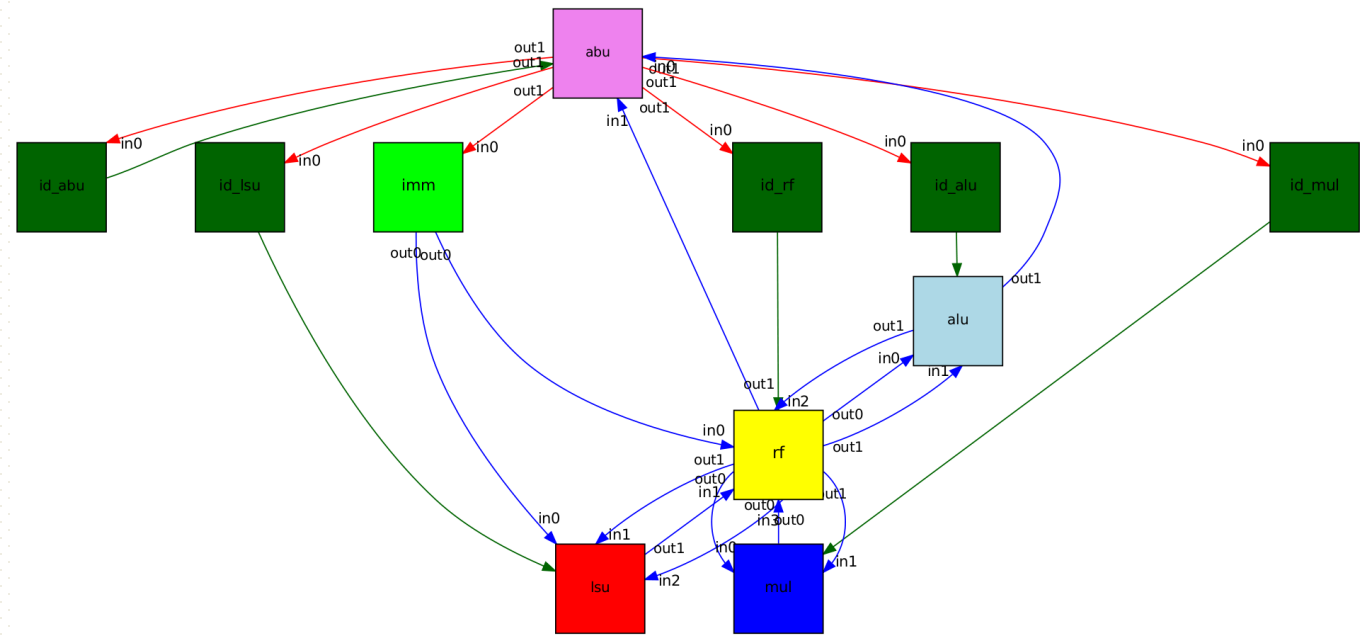

Changing the architecture.

```
<fu type="ID" name='id_abu'> <!-- IDs -->
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_lsu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_alu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_mul'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_rf'>
  <input index="0" source="abu.1"/>
</fu>

<fu type="LSU" name='lsu' ID="id_lsu"> <!-- FUs -->
  <input index="0" source="imm.0"/>
  <input index="1" source="rf.1"/>
  <input index="2" source="rf.0"/>
</fu>
<fu type="ALU" name='alu' ID="id_alu" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="MUL" name='mul' ID="id_mul" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="RF" name='rf' ID="id_rf">
  <input index="0" source="imm.0"/>
  <input index="1" source="lsu.1"/>
  <input index="2" source="alu.1"/>
  <input index="3" source="mul.0"/>
</fu>
<fu type="ABU" name='abu' ID="id_abu" config="1">
  <input index="0" source="alu.1"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="IU" name='imm'>
  <input index="0" source="abu.1"/>
</fu>
```

Graphical view.

\$ xdot ~/CGRA/build/convolution/<name>/report/architecture.dot



Changing the architecture.

```
<fu type="ID" name='id_abu'> <!-- IDs -->
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_lsu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_alu'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_mul'>
  <input index="0" source="abu.1"/>
</fu>
<fu type="ID" name='id_rf'>
  <input index="0" source="abu.1"/>
</fu>

<fu type="LSU" name='lsu' ID="id_lsu"> <!-- FUs -->
  <input index="0" source="imm.0"/>
  <input index="1" source="rf.1"/>
  <input index="2" source="rf.0"/>
</fu>
<fu type="ALU" name='alu' ID="id_alu" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="MUL" name='mul' ID="id_mul" config="1">
  <input index="0" source="rf.0"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="RF" name='rf' ID="id_rf">
  <input index="0" source="imm.0"/>
  <input index="1" source="lsu.1"/>
  <input index="2" source="alu.1"/>
  <input index="3" source="mul.0"/>
</fu>
<fu type="ABU" name='abu' ID="id_abu" config="1">
  <input index="0" source="alu.1"/>
  <input index="1" source="rf.1"/>
</fu>
<fu type="IU" name='imm'>
  <input index="0" source="abu.1"/>
</fu>
```

Important points.

1. Each ID must have a unique name.
2. All IDs must be connected to the ABU, for the program to advance.
3. FUs are connected to their corresponding IDs.

Optimizing code mapping.

| id_abu | id_rf | id_lsu | imm | id_alu | id_mul |
|--------------|---------------------|--------------------|------------|--------------------|--------|
| .text | .text | .text | .text | .text | .text |
| nop | nop | nop | nopi | nop | nop |
| | srm r0, in0 | | imm 0 | | |
| | srm r1, in0 | | imm 4 | | |
| | srm r2, in0 | | imm 65536 | | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm -64572 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm 0 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 63818 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm 0 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm -25323 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | | | nopi | | |
| | | | nopi | | |
| | | | nopi | | |
| bcr in1, in0 | lrm_srm r7, r2, in2 | | nopi | | |
| | | | nopi | | |
| jai 0 | nop | nop | nopi | nop | nop |

Parallel Assembly (PASM)

Optimizing code mapping.

| id_abu | id_rf | id_lsu | imm | id_alu | id_mul |
|--------------|---------------------|--------------------|---------------|--------------------|--------|
| .text | .text | .text | .text | .text | .text |
| nop | nop | nop | nopi imm 0 | nop | nop |
| | srm r0, in0 | | imm 4 | | |
| | srm r1, in0 | | imm 65536 | | |
| | srm r2, in0 | | nopi | | |
| | lrm r2 | sla WORD, in2, in1 | imm 0 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | | nopi | | |
| | srm r0, in2 | sla WORD, in2, in1 | imm -64572 | add out1, in0, in1 | |
| | lrm r2 | | nopi | | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | | nopi | | |
| | lrm r2 | sla WORD, in2, in1 | imm 0 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | | nopi | | |
| | srm r0, in2 | sla WORD, in2, in1 | imm 63818 | add out1, in0, in1 | |
| | lrm r2 | | nopi | | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | | nopi | | |
| | lrm r2 | sla WORD, in2, in1 | imm -25323 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | | nopi | | |
| | srm r0, in2 | sla WORD, in2, in1 | imm 0 | add out1, in0, in1 | |
| | lrm r2 | | nopi | | |
| | lrm_srm r1, r2, in0 | | nopi | | |
| bcr in1, in0 | lrm_srm r7, r2, in2 | | nopi | | |
| | | | nopi | | |
| jai 0 | nop | nop | nopi | nop | nop |

Important points.

1. All IDs are specified in different columns.

Parallel Assembly (PASM)

Optimizing code mapping.

| id_abu | id_rf | id_lsu | imm | id_alu | id_mul |
|--------------|---------------------|--------------------|------------|--------------------|--------|
| .text | .text | .text | .text | .text | .text |
| nop | nop | nop | nopi | nop | nop |
| | srm r0, in0 | | imm 0 | | |
| | srm r1, in0 | | imm 4 | | |
| | srm r2, in0 | | imm 65536 | | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm -64572 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm 0 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 63818 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm 0 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm -25323 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | | | nopi | | |
| bcr in1, in0 | lrm_srm r7, r2, in2 | | nopi | | |
| | | | nopi | | |
| jai 0 | nop | nop | nopi | nop | nop |

Important points.

1. All IDs are specified in different columns
2. Instruction execution starts at count 1 , (not count 0)

Parallel Assembly (PASM)

Optimizing code mapping.

| id_abu | id_rf | id_lsu | imm | id_alu | id_mul |
|--------------|---------------------|--------------------|------------|--------------------|--------|
| .text | .text | .text | .text | .text | .text |
| nop | nop | nop | nopi | nop | nop |
| | srm r0, in0 | | imm 0 | | |
| | srm r1, in0 | | imm 4 | | |
| | srm r2, in0 | | imm 65536 | | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm -64572 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm 0 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 63818 | add out1, in0, in1 | |
| | srm r0, in2 | sla WORD, in2, in1 | nopi | | |
| | lrm r2 | | imm 0 | add out1, in0, in1 | |
| | lrm_srm r1, r2, in0 | sla WORD, in2, in1 | nopi | | |
| | srm r0, in2 | | imm -25323 | add out1, in0, in1 | |
| | lrm r2 | sla WORD, in2, in1 | nopi | | |
| | lrm_srm r1, r2, in0 | | imm 0 | add out1, in0, in1 | |
| | srm r0, in2 | | nopi | | |
| | lrm r2 | | nopi | | |
| | lrm_srm r1, r2, in0 | | nopi | | |
| bcr in1, in0 | lrm_srm r7, r2, in2 | | nopi | | |
| | | | nopi | | |
| | | | nopi | | |
| jai 0 | nop | nop | nopi | nop | nop |

Important points.

1. All IDs are specified in different columns
2. Instruction execution starts at count 1 , (not count 0)
3. "jai 0" ends program execution.

Detailed ISA can be found
on
<https://oncourse.tue.nl>

Parallel Assembly (PASM)

Debugging

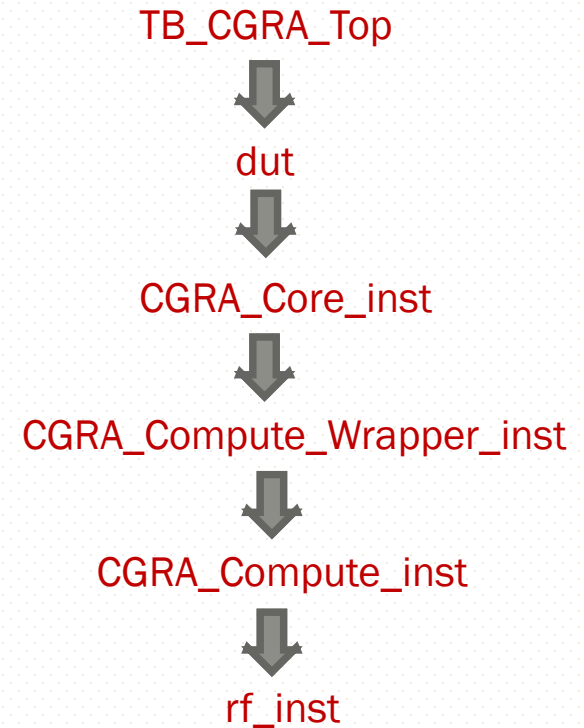
The screenshot displays the ModelSim simulation environment. On the left, the 'Instance' tree shows the hierarchy of components, including 'TB_CGRA_Top', 'dut', and various sub-modules like 'CLogB2', 'CGRA_Core_inst', and 'CGRA_Compute_Wrapper_inst'. The 'Wave - Default' window on the right shows a list of signals and their values over time. The signals are indexed from [15] down to [0]. The values for these signals are: [15]: 19375, [14]: x, [13]: x, [12]: 188, [11]: 44, [10]: 0, [9]: x, [8]: x, [7]: -72, [6]: 8000, [5]: 0, [4]: 20940, [3]: -740, [2]: 0, [1]: 4, [0]: -1. The waveform shows a sequence of values over time, with a cursor at 9032210 ns.

| Signal | Value |
|--------|-------|
| [15] | 19375 |
| [14] | x |
| [13] | x |
| [12] | 188 |
| [11] | 44 |
| [10] | 0 |
| [9] | x |
| [8] | x |
| [7] | -72 |
| [6] | 8000 |
| [5] | 0 |
| [4] | 20940 |
| [3] | -740 |
| [2] | 0 |
| [1] | 4 |
| [0] | -1 |

Use Modelsim.

```
$ cd ~/CGRA/benchmarks/convolution/<name>  
$ make sim
```

To add register file :



Makefile

make sim

Runs modelsim with GUI.

make run

Runs modelsim in the command line.

make compare

Runs simulation and compares the output with actual one (given in compare directory).

make performance

Runs simulation and gives quick performance estimates (**less accurate**).

make report

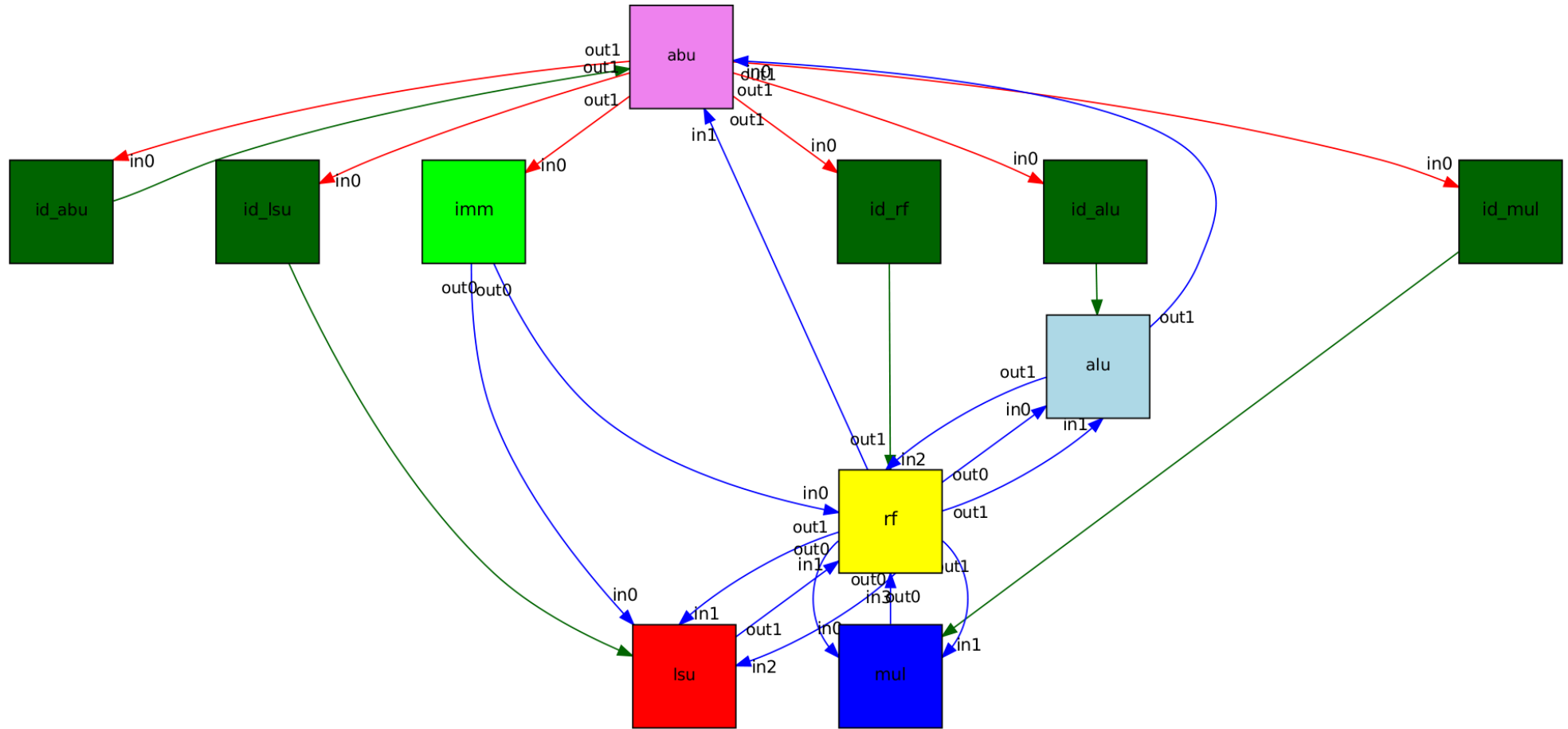
Runs simulation and generates a report in build/<name>/report (**slower but accurate**).

make clean

Cleans the build directory for the benchmark.

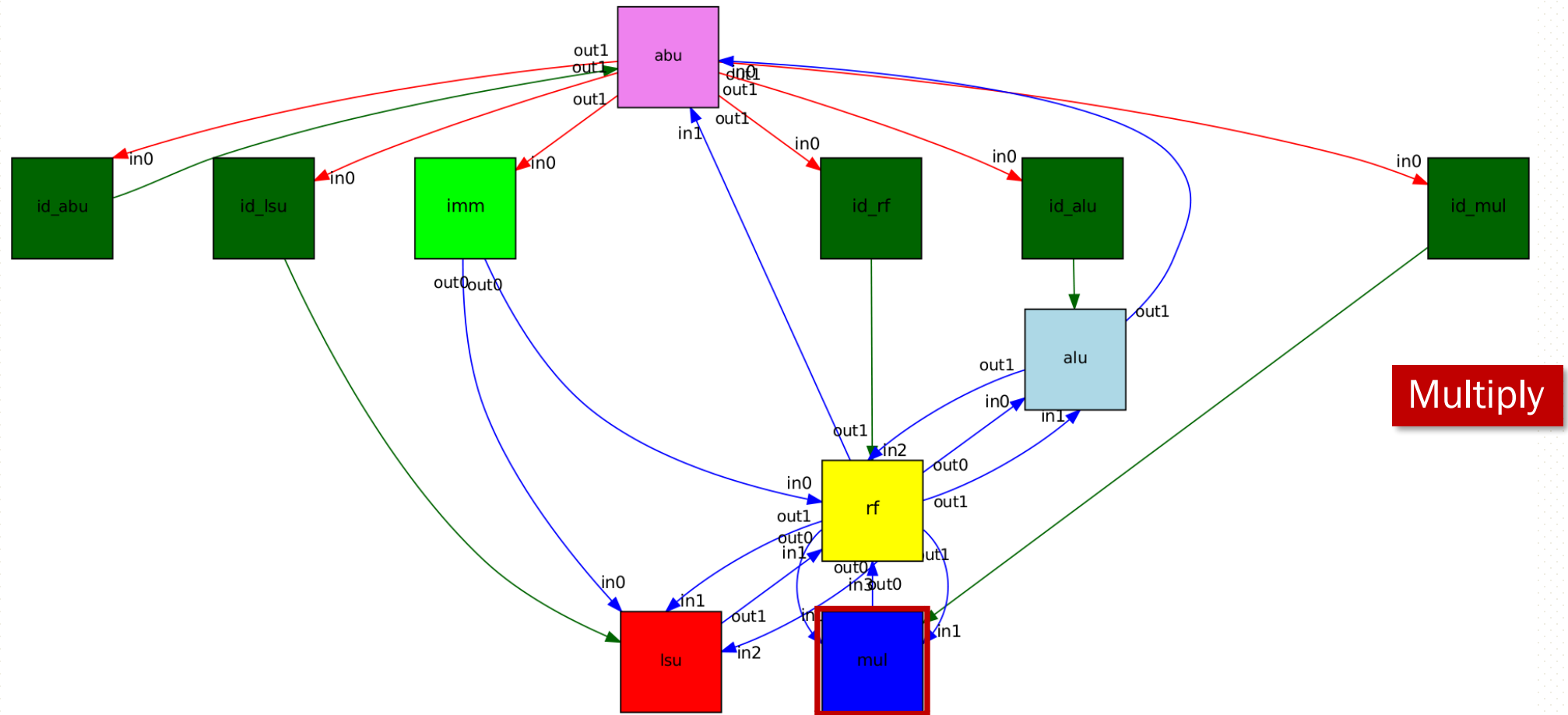
Optimization Hints

Bypassing



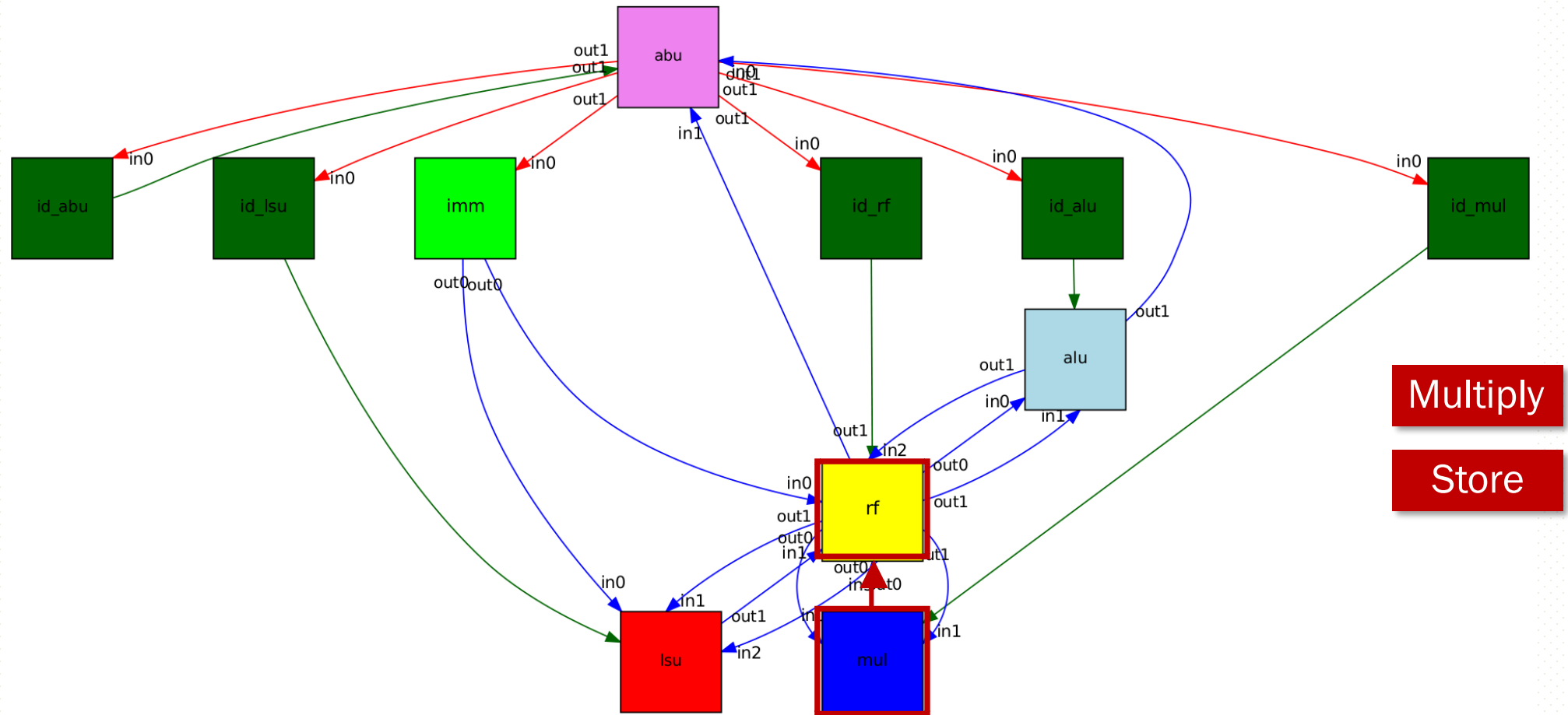
Optimization Hints

Bypassing



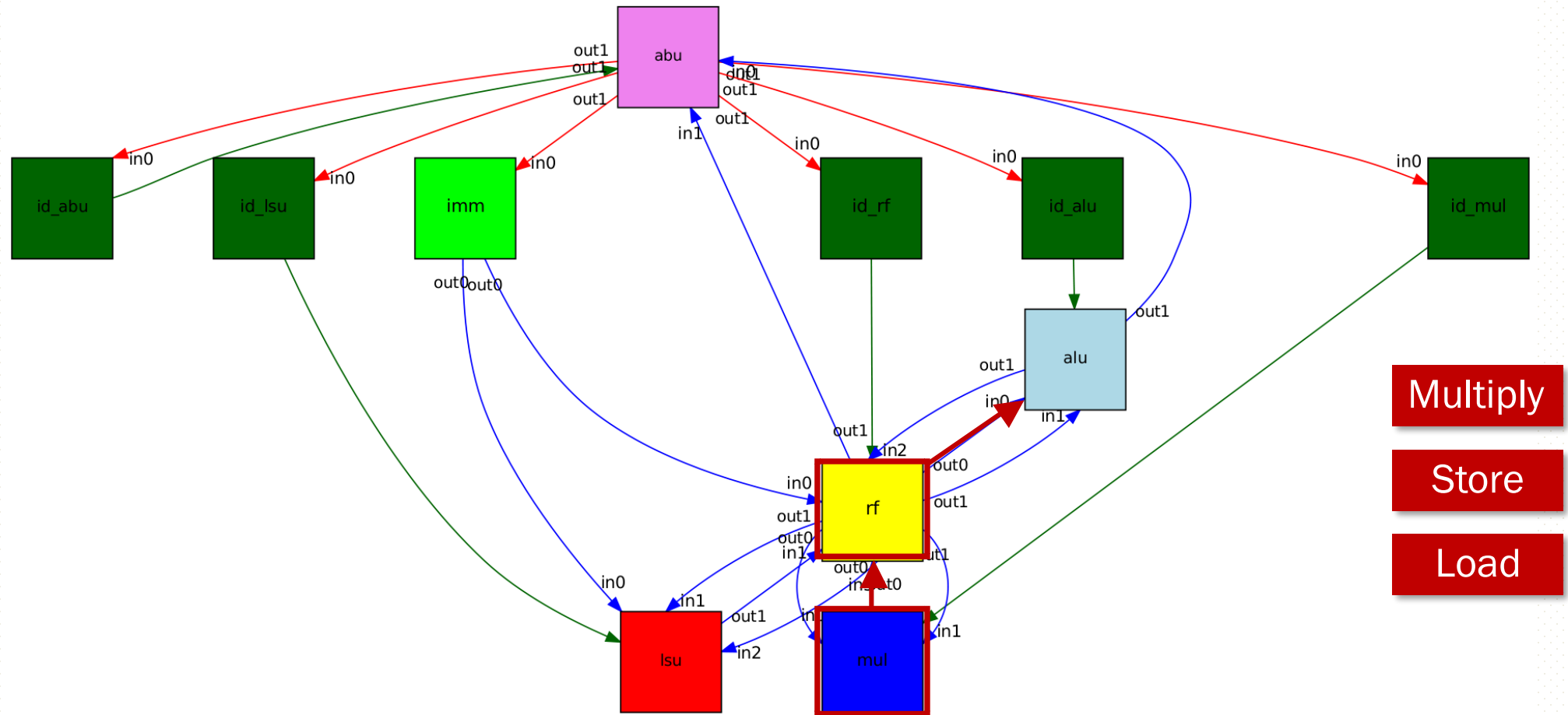
Optimization Hints

Bypassing



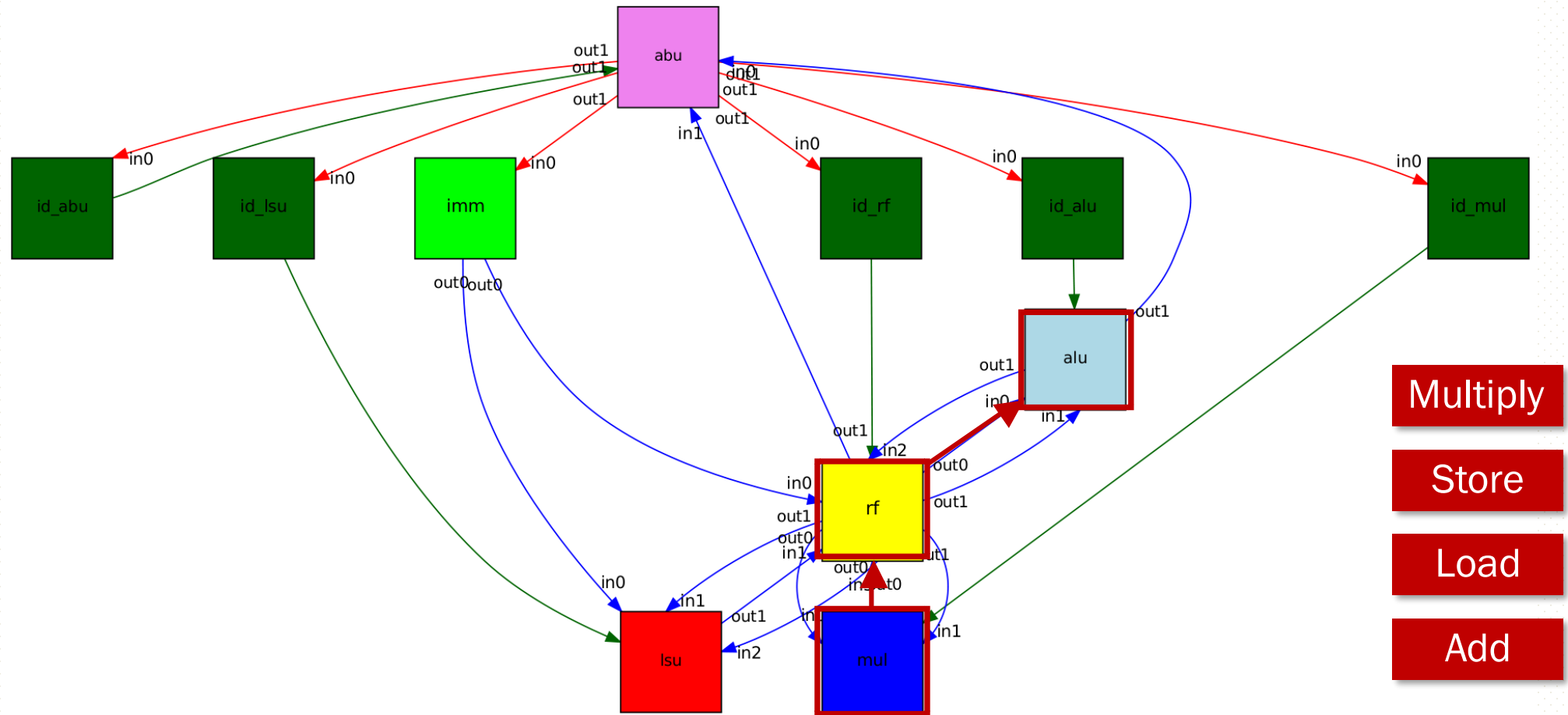
Optimization Hints

Bypassing



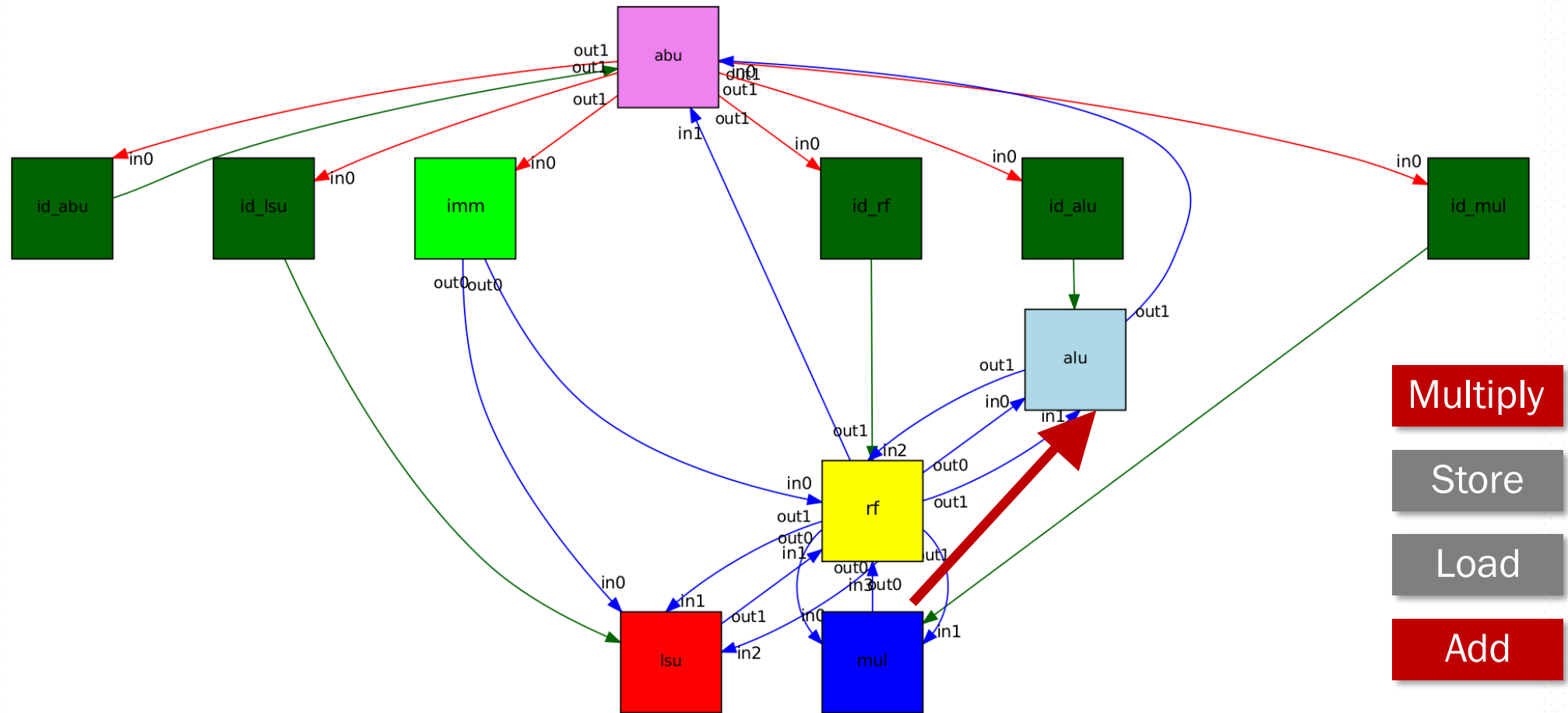
Optimization Hints

Bypassing



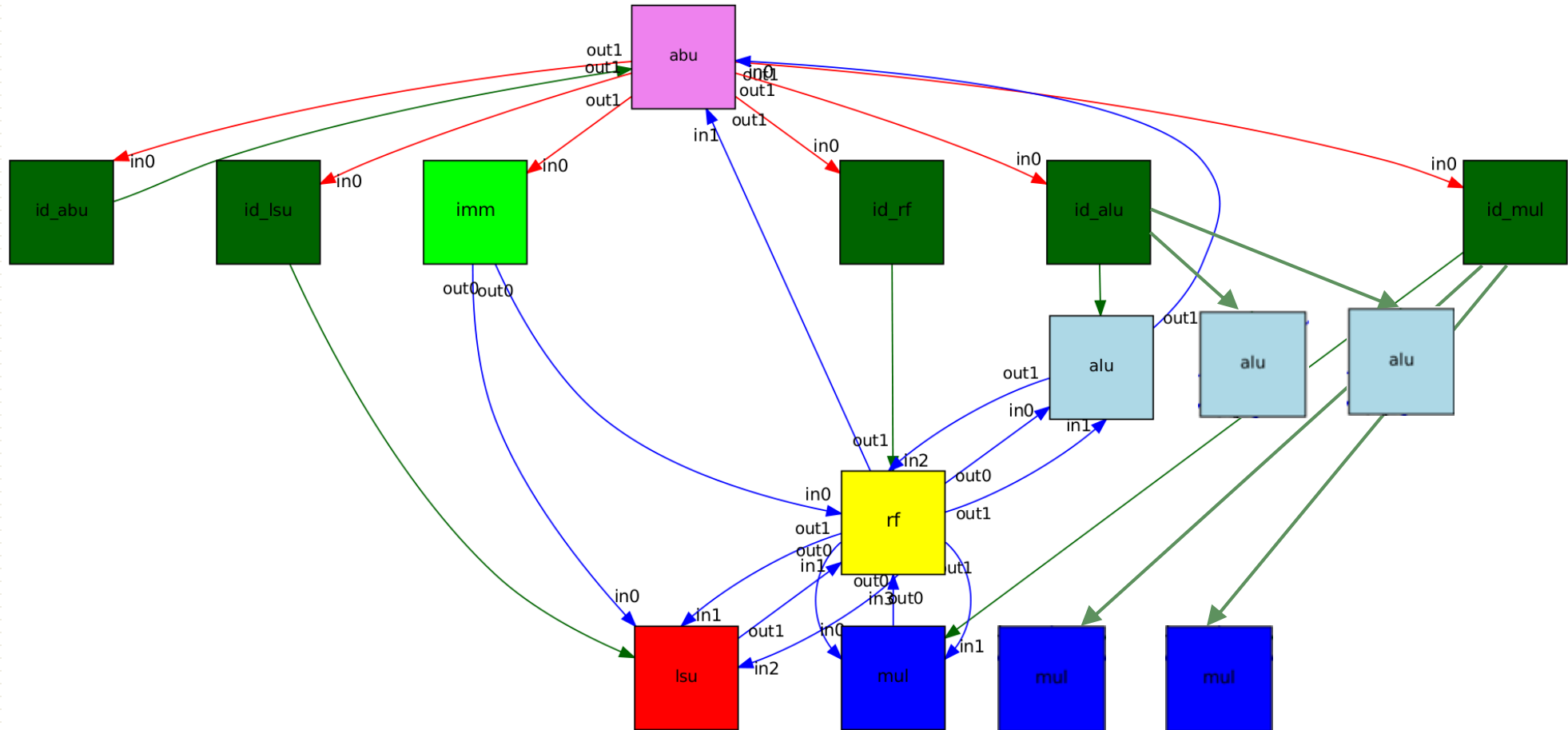
Optimization Hints

Bypassing



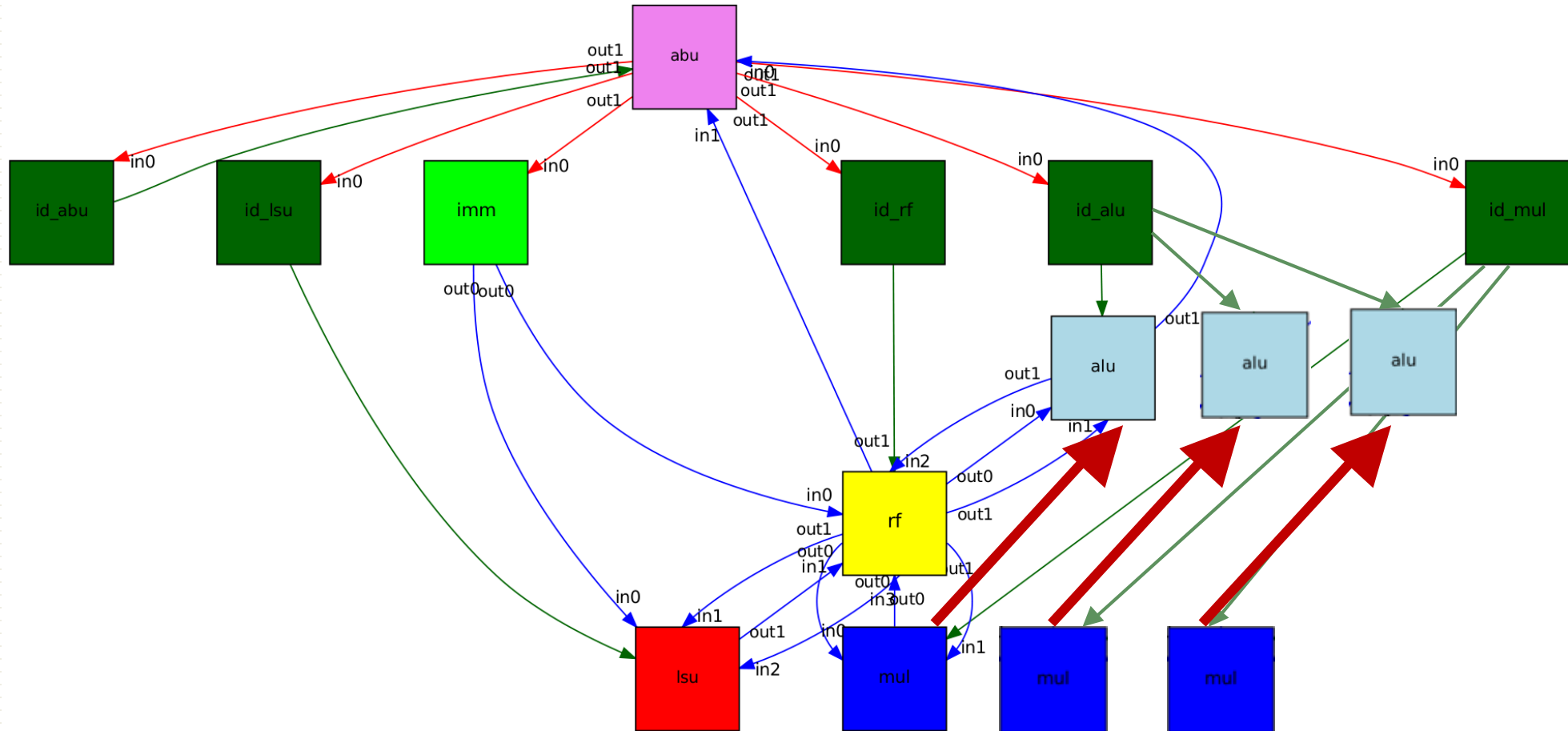
Optimization Hints

SIMD



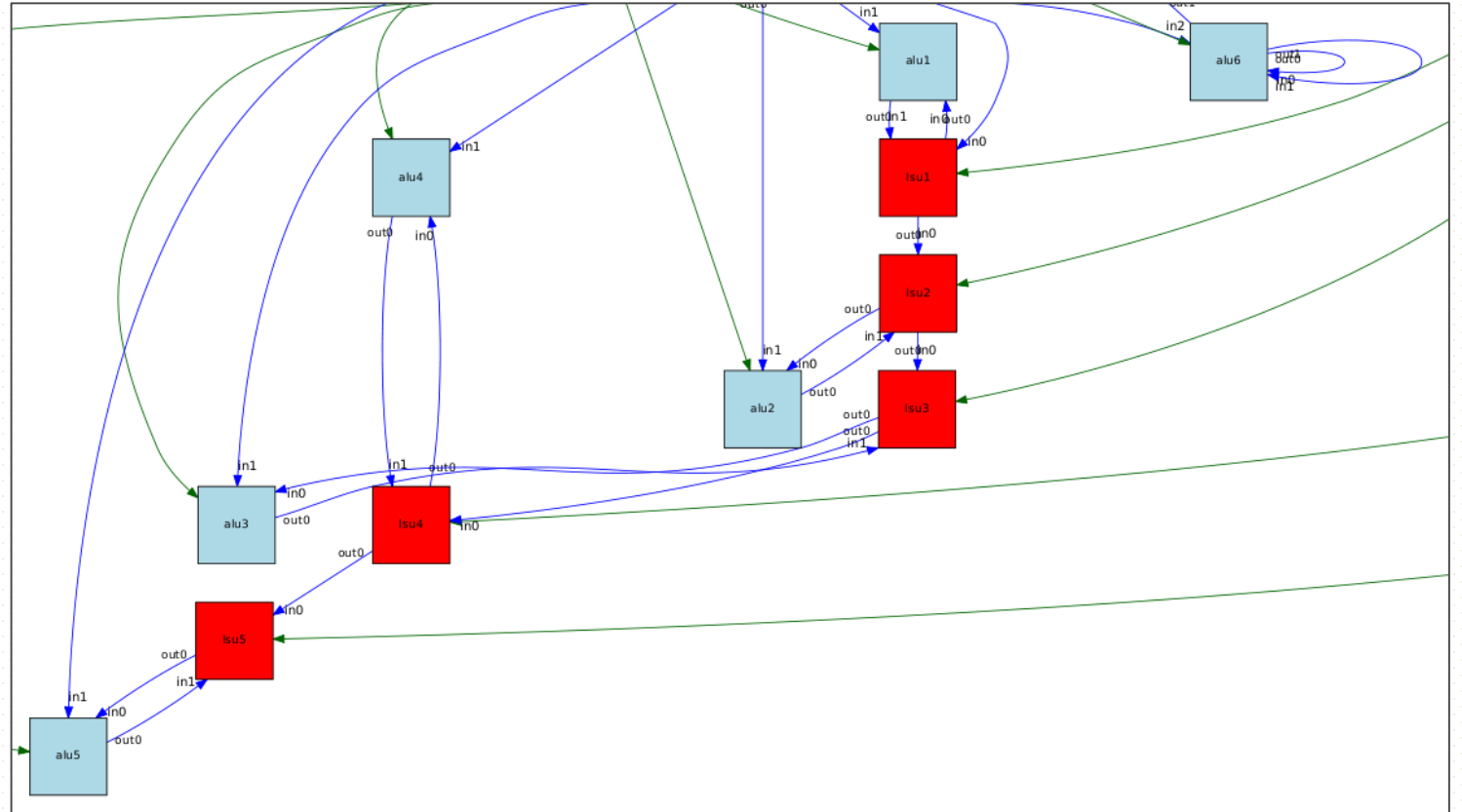
Optimization Hints

Bypassing + SIMD



Optimization Hints

Single Cycle Loop



| id_abu | id_lsu | imm | id_alu1 |
|-----------------------------|-------------------------|---------|--------------------|
| nop | lgi BYTE, out0 | imm 128 | nop |
| nop | lgi BYTE, out0 | imm 128 | ltu out0, in0, in1 |
| bcai 20, in0 | lgi_sgi BYTE, out0, in1 | imm 128 | ltu out0, in0, in1 |
| nop ; delay slot 1 | lgi_sgi BYTE, out0, in1 | imm 128 | ltu out0, in0, in1 |
| nop ; delay slot 2 | lgi_sgi BYTE, out0, in1 | imm 128 | ltu out0, in0, in1 |
| nop | sgi BYTE, in1 | nopi | ltu out0, in0, in1 |
| nop | sgi BYTE, in1 | nopi | ltu out0, in0, in1 |
| jai 0 ; terminate execution | nop | nopi | nop |
| | | | nop |

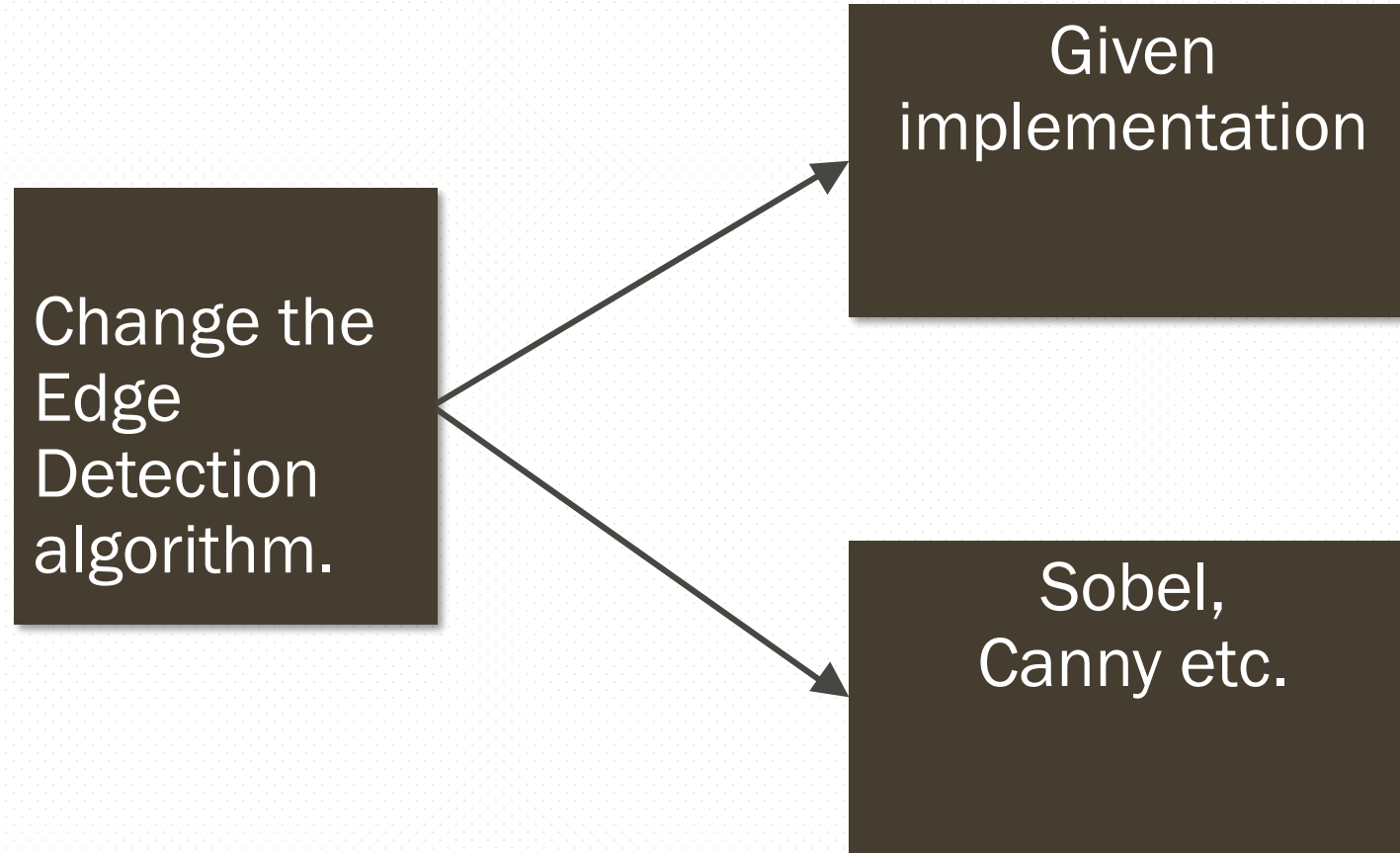
Optimization Hints

Data reuse

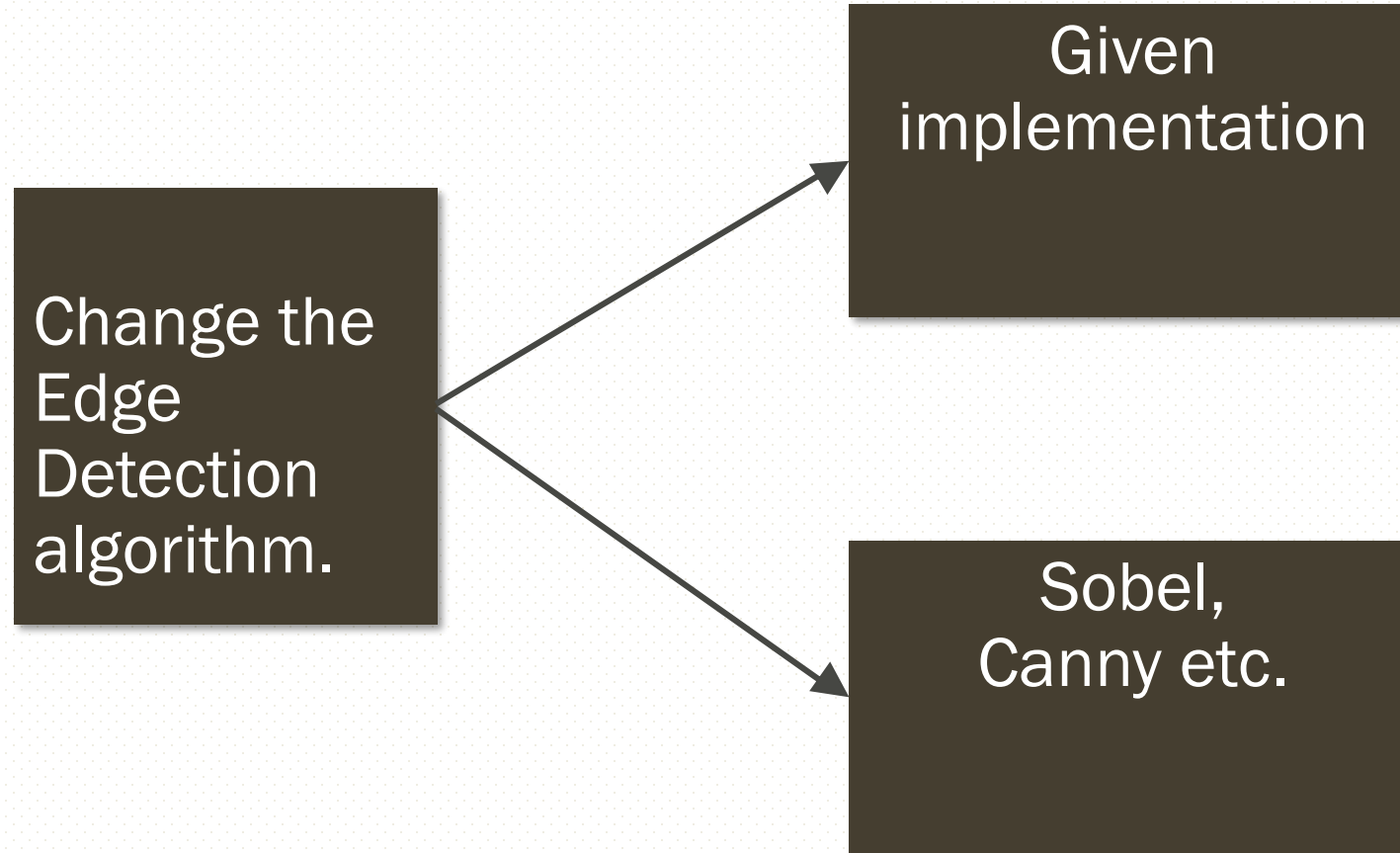
Automatic
address
generation

Adding ABU
(for
accumulation)

Changing the algorithm



Changing the algorithm



In case, you use a different algorithm, provide us with the C implementation as well.

Server List

co2.ics.ele.tue.nl

co3.ics.ele.tue.nl

co4.ics.ele.tue.nl

co9.ics.ele.tue.nl

co10.ics.ele.tue.nl

co13.ics.ele.tue.nl

co17.ics.ele.tue.nl

Resources

- Assignment details and guideline

<https://oncourse.tue.nl>