Peter Wallace
Derek Barbosa
Lab 4

Modules:

- Move
  - In this module we implement the move command. The module takes R2 as an input, and uses the assign command to set the output (R1) to the input.
- Not
  - We implemented the not command by parameterizing the basic logic not gate included in vivado. We used a generate statement, assigning each bit of input and each bit of output into a single not gate.
- Add
  - We implemented a parameterized full adder, using a parameter and a generate loop. We hardwired c_in to 0, since the diagram in the lab doesn't include c_in as an input.
- Sub
  - We implemented the subtractor using a parameterized full adder, a parameterized not gate, and a hard-wired c_in of 1. Since subtracting is adding the 2's complement of the second operand, we not the b of the full adder, and include the c_in of 1 to the 1st full adder to finish the 2's complement
- Or
  - We implemented the Or function by parameterizing and generating N of the basic logic or gates included in vivado.
- And
  - We implemented the And function by parameterizing and generating N of the basic logic and gates included in vivado.
- Multiplexer
  - Our multiplexer takes as input the output of each of the command modules, as well as the ALU opcode as a select. It maps the command outputs to R1 depending on the opcode.
- ALU
  - Our ALU module combines and instantiates each command module, as well as the Mux module.
- Register
  - Our register is a parametrized D flip-flop, set to N bits.
- Top Module
  - The top module in our design puts together the ALU and the Register, taking R2, R3, and the opcode as input to the ALU, and wire R1, clk as input (D) to the

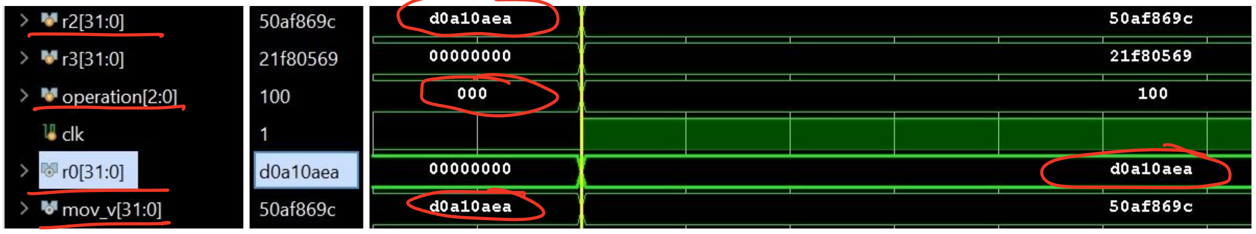
Register. It sets wire R1 as the output of the ALU, and R0 as the output (Q) of the Register.

Hierarchy

- Top Module (combine ALU and Register
  - Register
  - Alu
    - Multiplexer
    - Move module
    - Not module
    - Add module
    - Sub module
    - Or module
    - And module

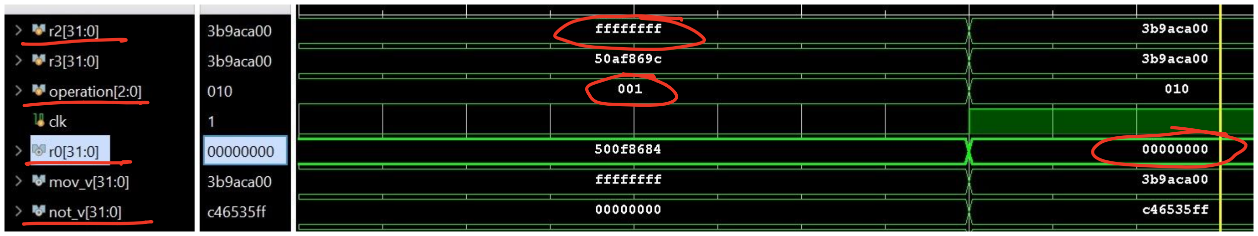
Timing Diagrams

- Move



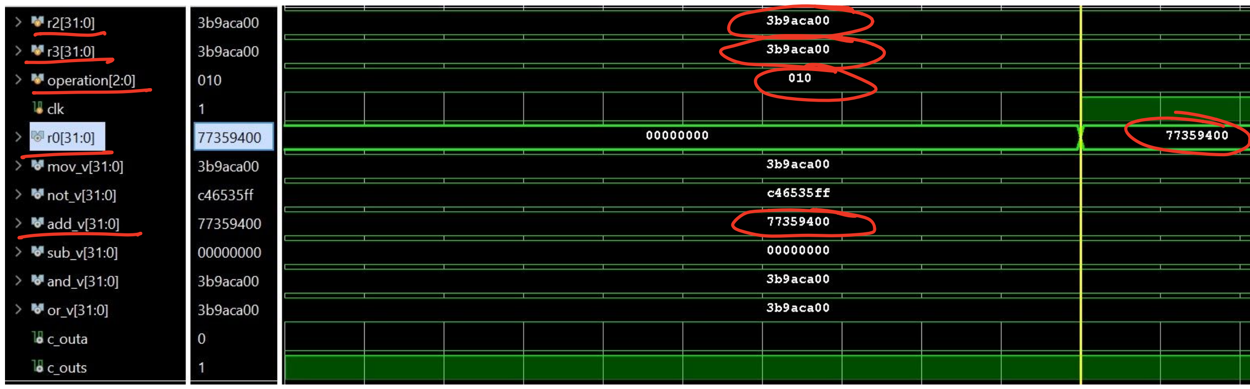
```
r2[31:0]          50af869c   d0a10aea   50af869c
r3[31:0]          21f80569   00000000   21f80569
operation[2:0]    100        000        100
clk               1
r0[31:0]          d0a10aea   00000000   d0a10aea
mov_v[31:0]       50af869c   d0a10aea   50af869c
```

- Not



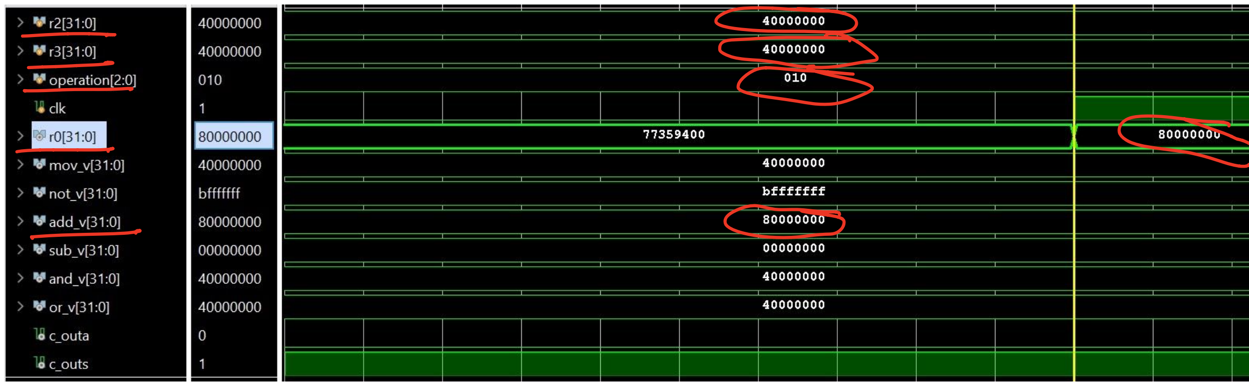
```
r2[31:0]          3b9aca00   ffffffff   3b9aca00
r3[31:0]          3b9aca00   50af869c   3b9aca00
operation[2:0]    010        001        010
clk               1
r0[31:0]          00000000   500f8684   00000000
mov_v[31:0]       3b9aca00   ffffffff   3b9aca00
not_v[31:0]       c46535ff   00000000   c46535ff
```

- Add
  - Add large w/ overflow



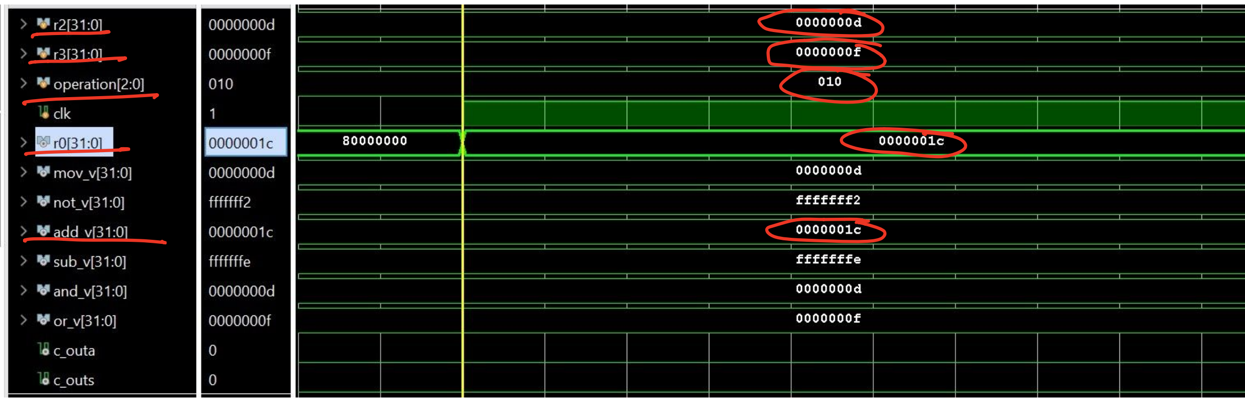
```
r2[31:0]          3b9aca00   3b9aca00
r3[31:0]          3b9aca00   3b9aca00
operation[2:0]    010        010
clk               1
r0[31:0]          77359400   00000000   77359400
mov_v[31:0]       3b9aca00   3b9aca00
not_v[31:0]       c46535ff   c46535ff
add_v[31:0]       77359400   77359400
sub_v[31:0]       00000000   00000000
and_v[31:0]       3b9aca00   3b9aca00
or_v[31:0]        3b9aca00   3b9aca00
c_outa            0
c_outs            1
```

  - Add large w/no overflow



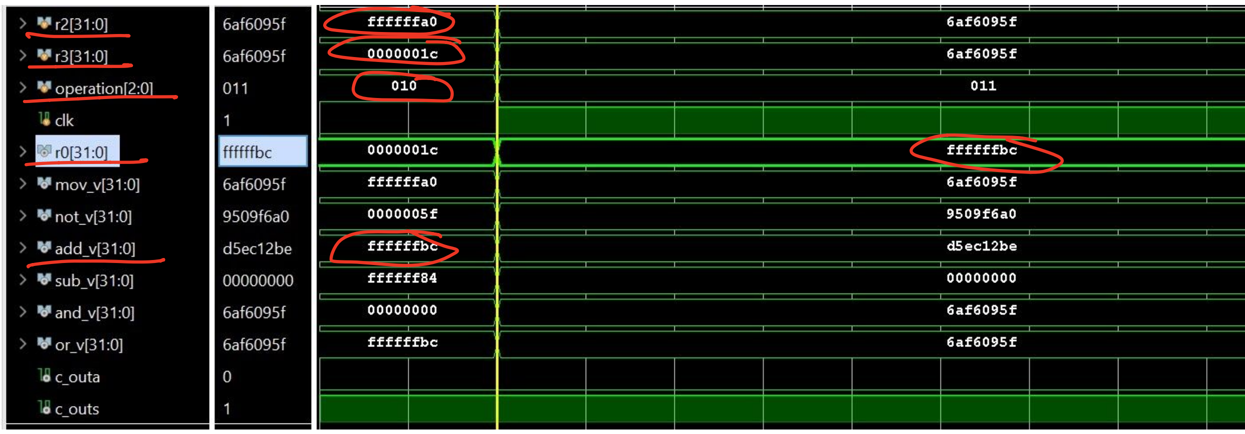
```
r2[31:0]           40000000    40000000
r3[31:0]           40000000    40000000
operation[2:0]     010         010
clk                1
r0[31:0]           80000000    77359400    80000000
mov_v[31:0]        40000000    40000000
not_v[31:0]        bfffffff    bfffffff
add_v[31:0]        80000000    80000000
sub_v[31:0]        00000000    00000000
and_v[31:0]        40000000    40000000
or_v[31:0]         40000000    40000000
c_outa             0
c_outs             1
```

- Add small



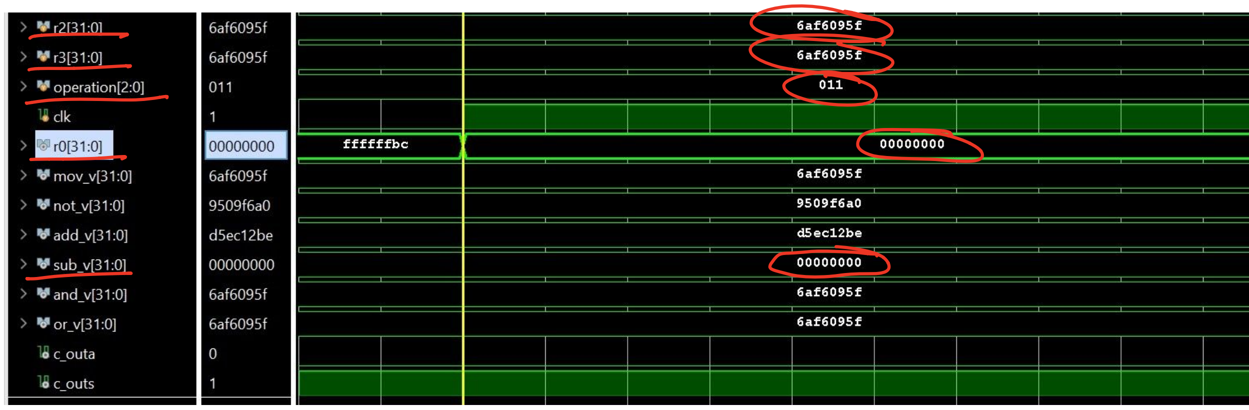
```
r2[31:0]           0000000d    0000000d
r3[31:0]           0000000f    0000000f
operation[2:0]     010         010
clk                1
r0[31:0]           0000001c    80000000    0000001c
mov_v[31:0]        0000000d    0000000d
not_v[31:0]        fffffff2    fffffff2
add_v[31:0]        0000001c    0000001c
sub_v[31:0]        fffffffe    fffffffe
and_v[31:0]        0000000d    0000000d
or_v[31:0]         0000000f    0000000f
c_outa             0
c_outs             0
```

- Add a negative



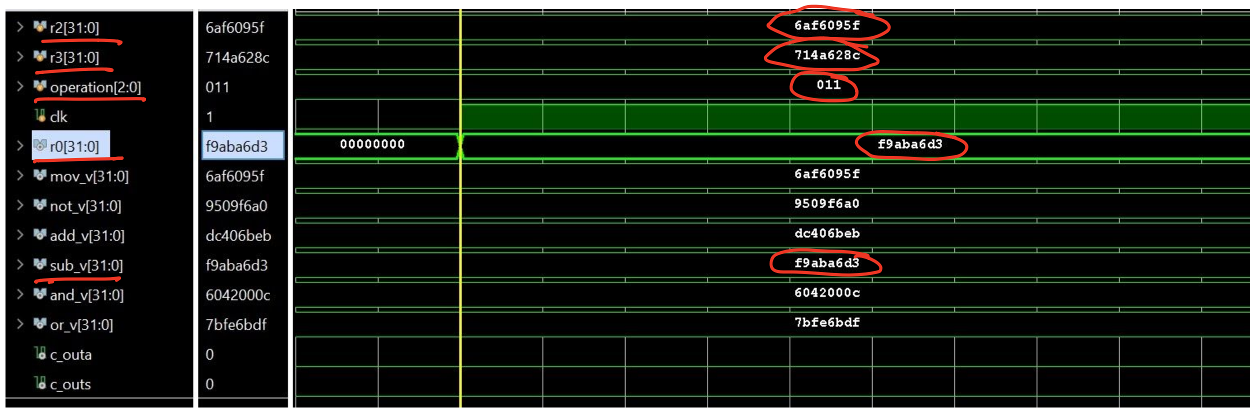
```
r2[31:0]           6af6095f    ffffffa0    6af6095f
r3[31:0]           6af6095f    0000001c    6af6095f
operation[2:0]     011         010         011
clk                1
r0[31:0]           ffffffbc    0000001c    ffffffbc
mov_v[31:0]        6af6095f    ffffffa0    6af6095f
not_v[31:0]        9509f6a0    0000005f    9509f6a0
add_v[31:0]        d5ec12be    ffffffbc    d5ec12be
sub_v[31:0]        00000000    ffffff84    00000000
and_v[31:0]        6af6095f    00000000    6af6095f
or_v[31:0]         6af6095f    ffffffbc    6af6095f
c_outa             0
c_outs             1
```

- Subtract
    - Subtract same number



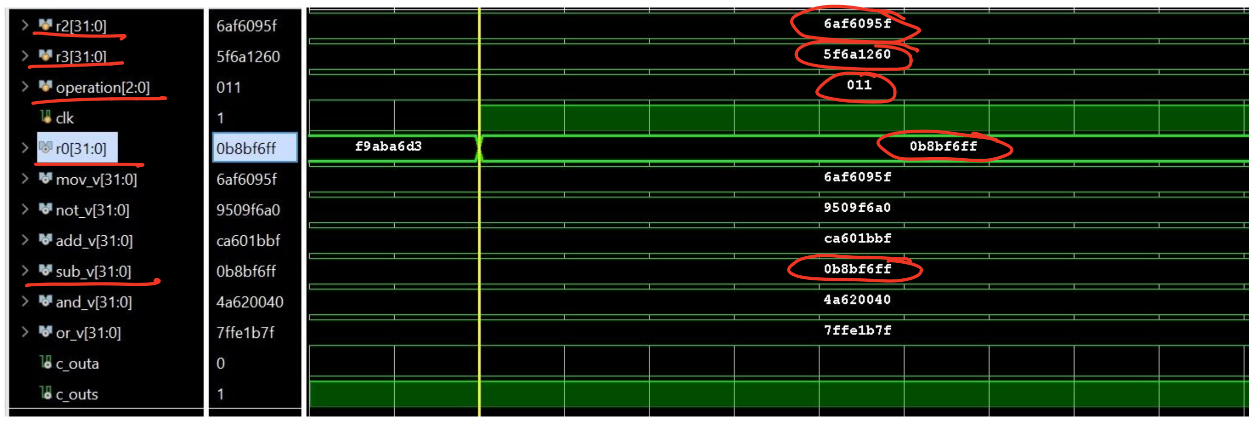
r2[31:0]	6af6095f	6af6095f
r3[31:0]	6af6095f	6af6095f
operation[2:0]	011	011
clk	1	
r0[31:0]	00000000	ffffffbc 00000000
mov_v[31:0]	6af6095f	6af6095f
not_v[31:0]	9509f6a0	9509f6a0
add_v[31:0]	d5ec12be	d5ec12be
sub_v[31:0]	00000000	00000000
and_v[31:0]	6af6095f	6af6095f
or_v[31:0]	6af6095f	6af6095f
c_outa	0	
c_outs	1	

- ○ Subtract large w/negative result



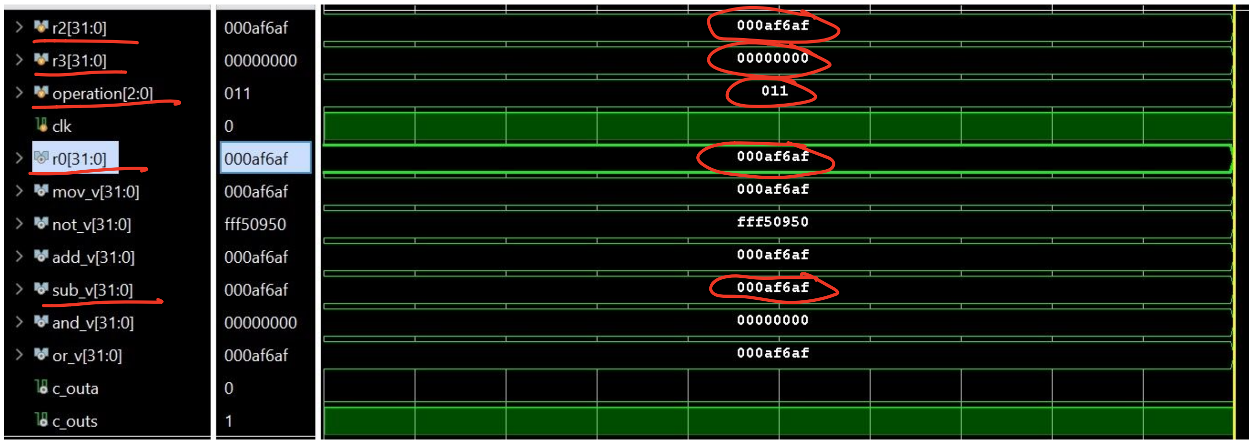
r2[31:0]	6af6095f	6af6095f
r3[31:0]	714a628c	714a628c
operation[2:0]	011	011
clk	1	
r0[31:0]	f9aba6d3	00000000 f9aba6d3
mov_v[31:0]	6af6095f	6af6095f
not_v[31:0]	9509f6a0	9509f6a0
add_v[31:0]	dc406beb	dc406beb
sub_v[31:0]	f9aba6d3	f9aba6d3
and_v[31:0]	6042000c	6042000c
or_v[31:0]	7bfe6bdf	7bfe6bdf
c_outa	0	
c_outs	0	

- ○ Subtract large w/positive result



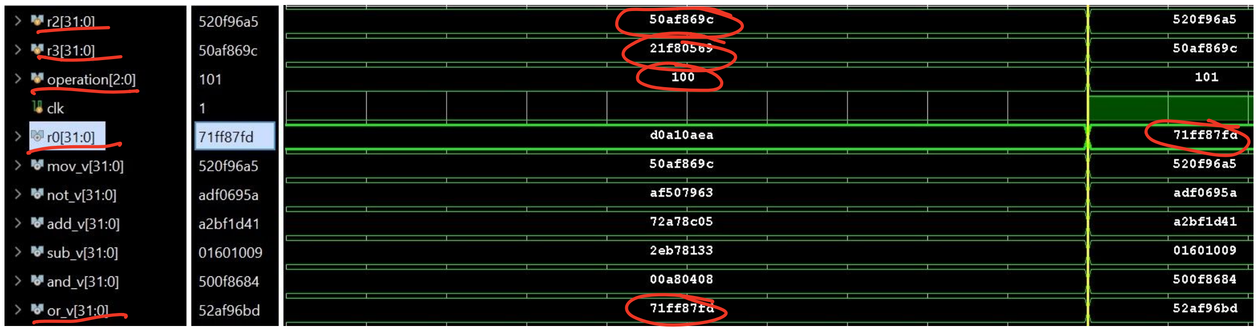
r2[31:0]	6af6095f	6af6095f
r3[31:0]	5f6a1260	5f6a1260
operation[2:0]	011	011
clk	1	
r0[31:0]	0b8bf6ff	f9aba6d3 0b8bf6ff
mov_v[31:0]	6af6095f	6af6095f
not_v[31:0]	9509f6a0	9509f6a0
add_v[31:0]	ca601bbf	ca601bbf
sub_v[31:0]	0b8bf6ff	0b8bf6ff
and_v[31:0]	4a620040	4a620040
or_v[31:0]	7ffe1b7f	7ffe1b7f
c_outa	0	
c_outs	1	

- ○ Subtract zero



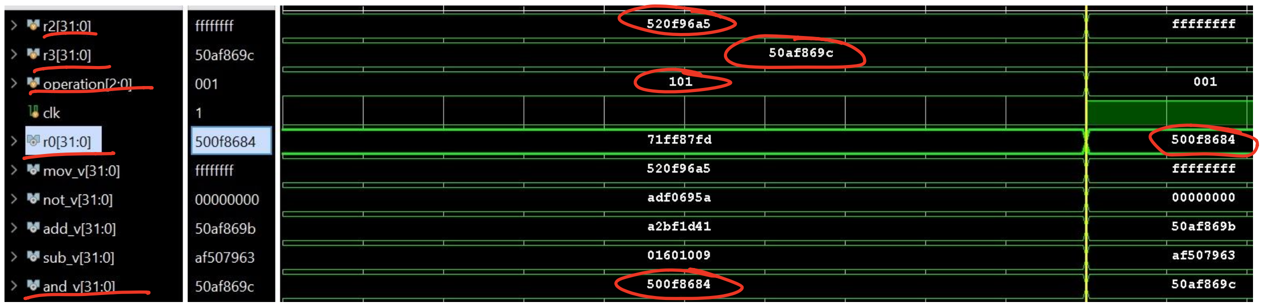
```
r2[31:0]          000af6af    000af6af
r3[31:0]          00000000    00000000
operation[2:0]    011         011
clk               0
r0[31:0]          000af6af    000af6af
mov_v[31:0]       000af6af    000af6af
not_v[31:0]       fff50950    fff50950
add_v[31:0]       000af6af    000af6af
sub_v[31:0]       000af6af    000af6af
and_v[31:0]       00000000    00000000
or_v[31:0]        000af6af    000af6af
c_outa            0
c_outs            1
```

- Or



```
r2[31:0]          520f96a5    50af869c    520f96a5
r3[31:0]          50af869c    21f80569    50af869c
operation[2:0]    101         100         101
clk               1
r0[31:0]          71ff87fd    d0a10aea    71ff87fd
mov_v[31:0]       520f96a5    50af869c    520f96a5
not_v[31:0]       adf0695a    af507963    adf0695a
add_v[31:0]       a2bf1d41    72a78c05    a2bf1d41
sub_v[31:0]       01601009    2eb78133    01601009
and_v[31:0]       500f8684    00a80408    500f8684
or_v[31:0]        52af96bd    71ff87fd    52af96bd
```

- And



```
r2[31:0]          ffffffff    520f96a5    ffffffff
r3[31:0]          50af869c    50af869c
operation[2:0]    001         101         001
clk               1
r0[31:0]          500f8684    71ff87fd    500f8684
mov_v[31:0]       ffffffff    520f96a5    ffffffff
not_v[31:0]       00000000    adf0695a    00000000
add_v[31:0]       50af869b    a2bf1d41    50af869b
sub_v[31:0]       af507963    01601009    af507963
and_v[31:0]       50af869c    500f8684    50af869c
```