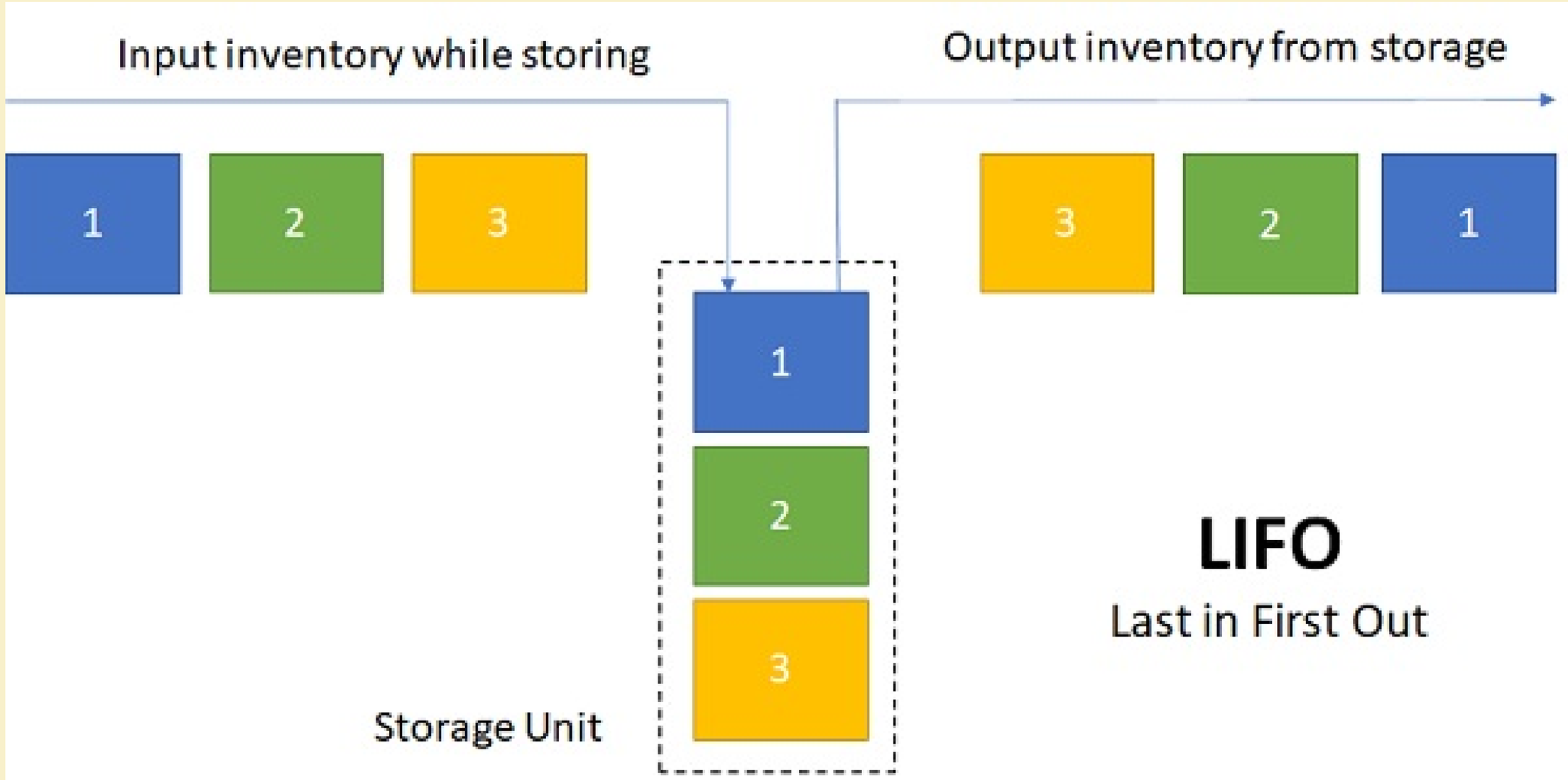
LAST IN FIRST OUT (STACK)

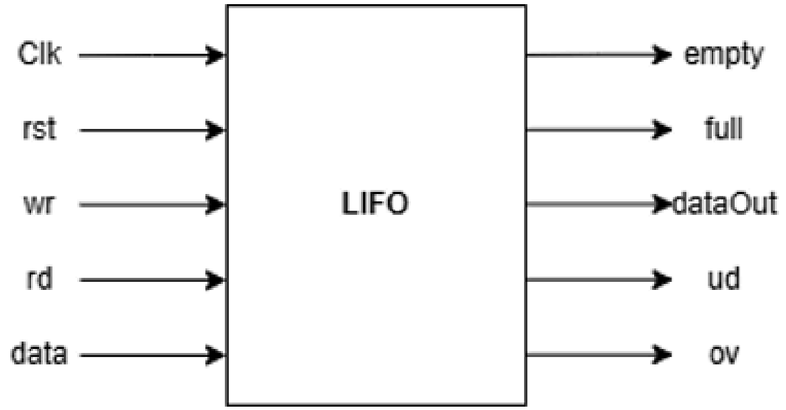
Nguyen Hong Nhat
Pham Dong Tra
Nguyen Nho Truong
Mentor: Khoa Le

ABSTRACT

What is LIFO ?

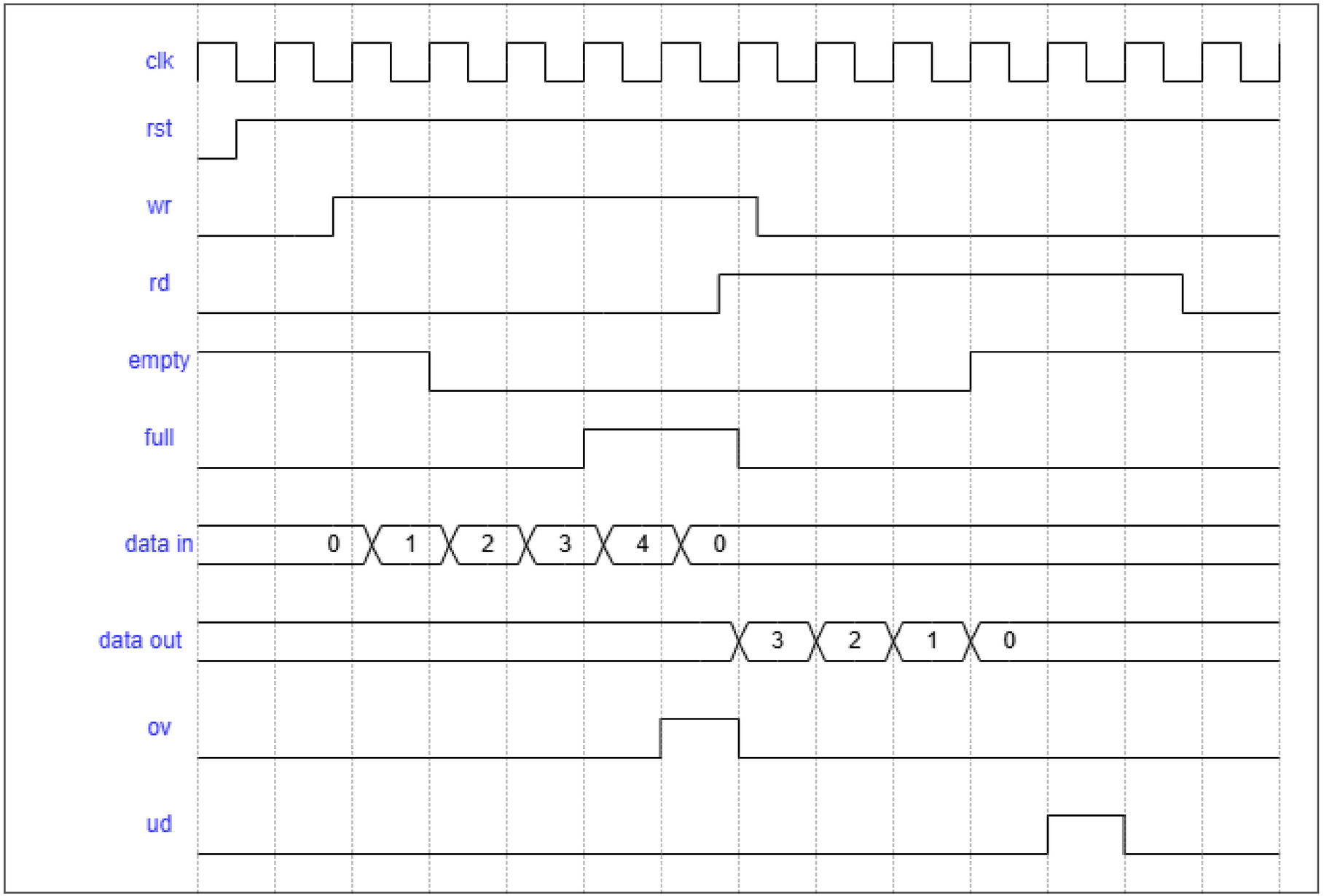

Input inventory while storing
Output inventory from storage
1
2
3
3
2
1
1
2
3
Storage Unit
LIFO
Last in First Out

HIGH LEVEL BLOCK

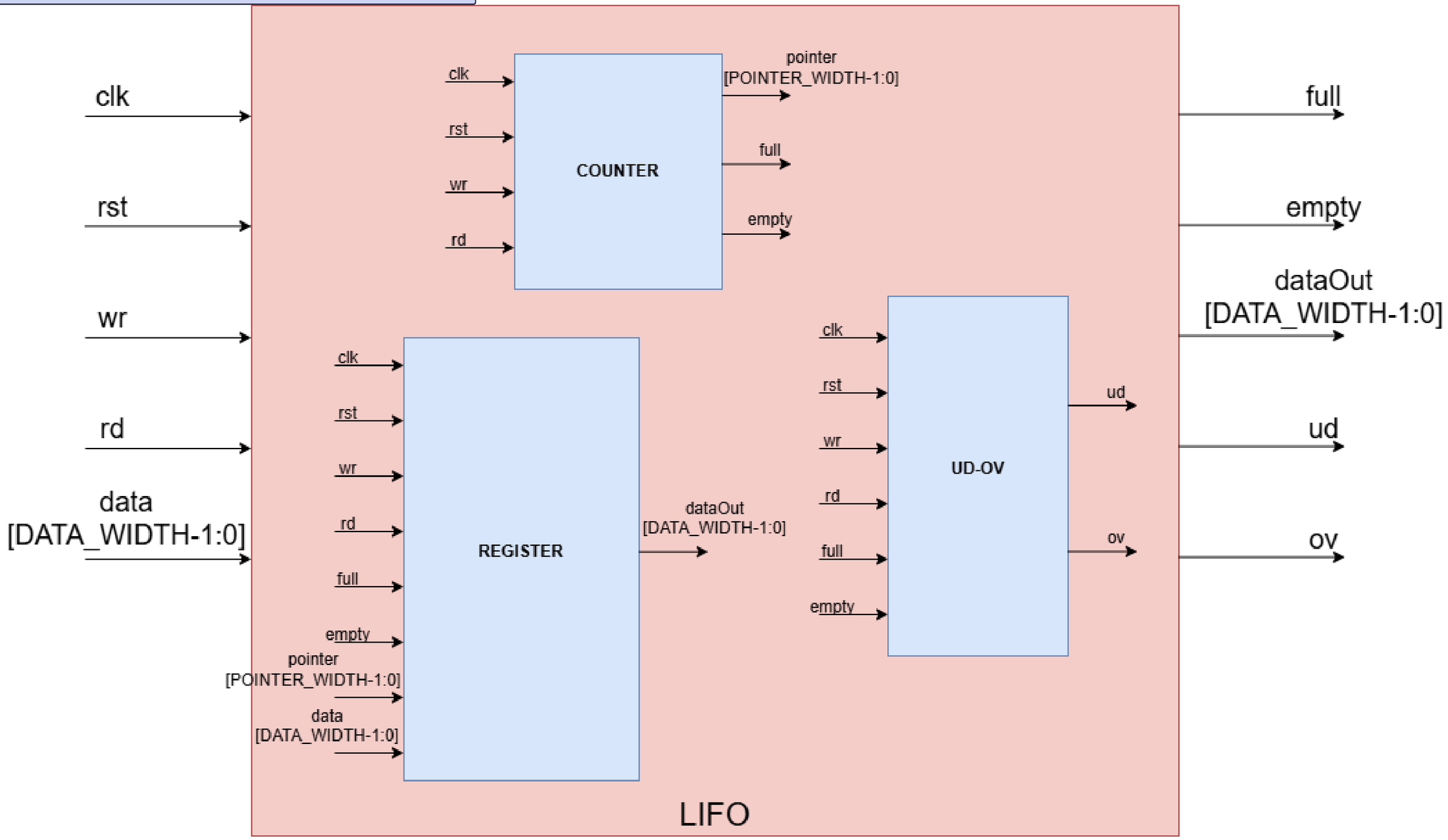


Port	Size	Direction	Description
Clk	1	Input	System clock
rst	1	Input	Asynchronous reset, active LOW
wr	1	Input	Write enable signal (Push) to store data into the stack
rd	1	Input	Read enable signal (Pop) to retrieve data from the stack
data	3	Input	Parallel input data bus (3-bit width)
dataOut	3	Output	Parallel output data bus (3-bit width)
empty	1	Output	Status flag indicating the stack is currently empty
full	1	Output	Status flag indicating the stack is full
ud	1	Output	Underflow: Error flag triggered when attempting to read from an empty stack
ov	1	Output	Overflow: Error flag triggered when attempting to write to a full stack

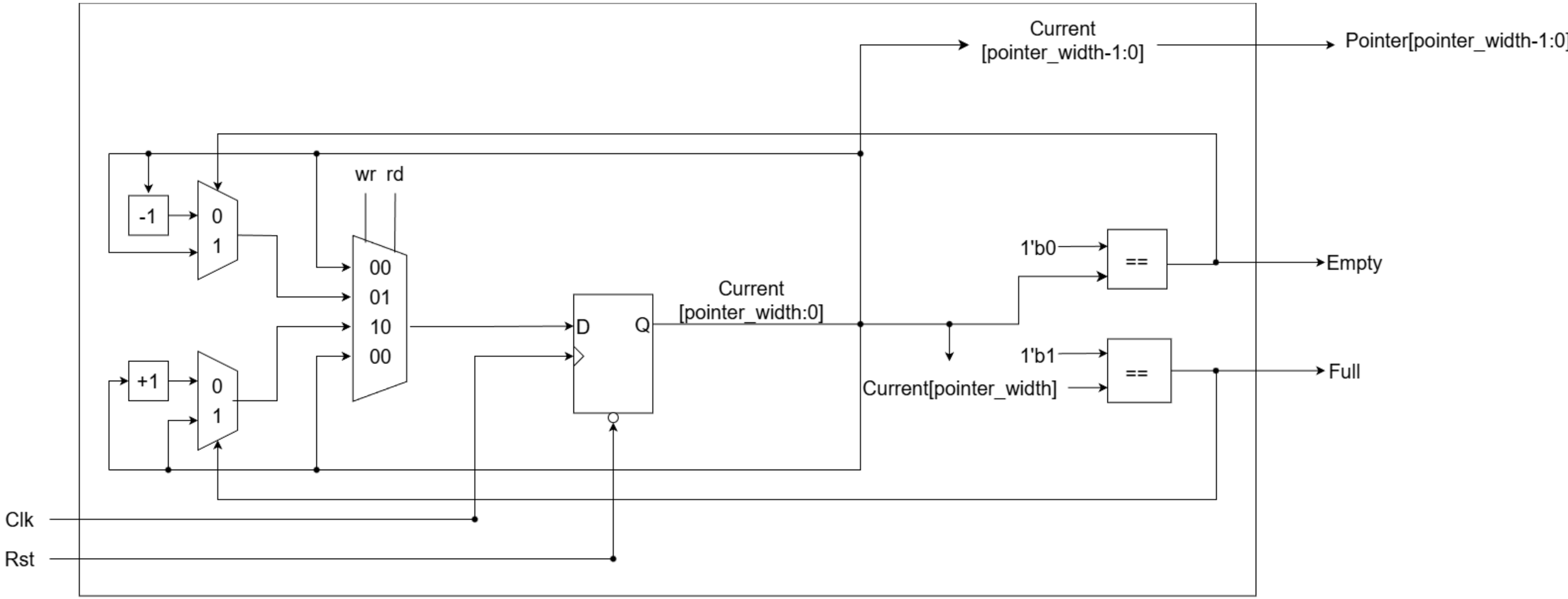
EXPECTED WAVEFORM



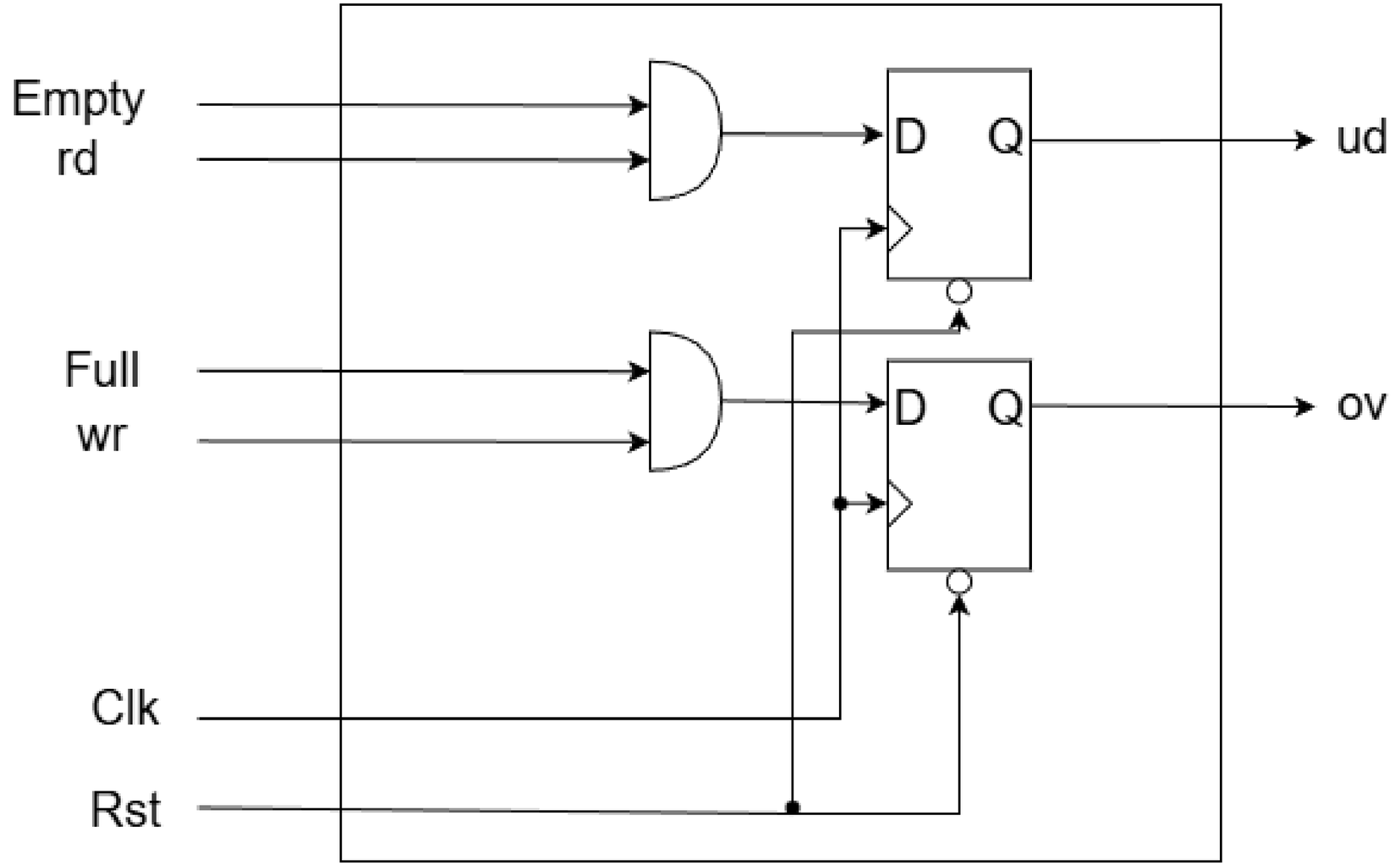
HIGH LEVEL BLOCK



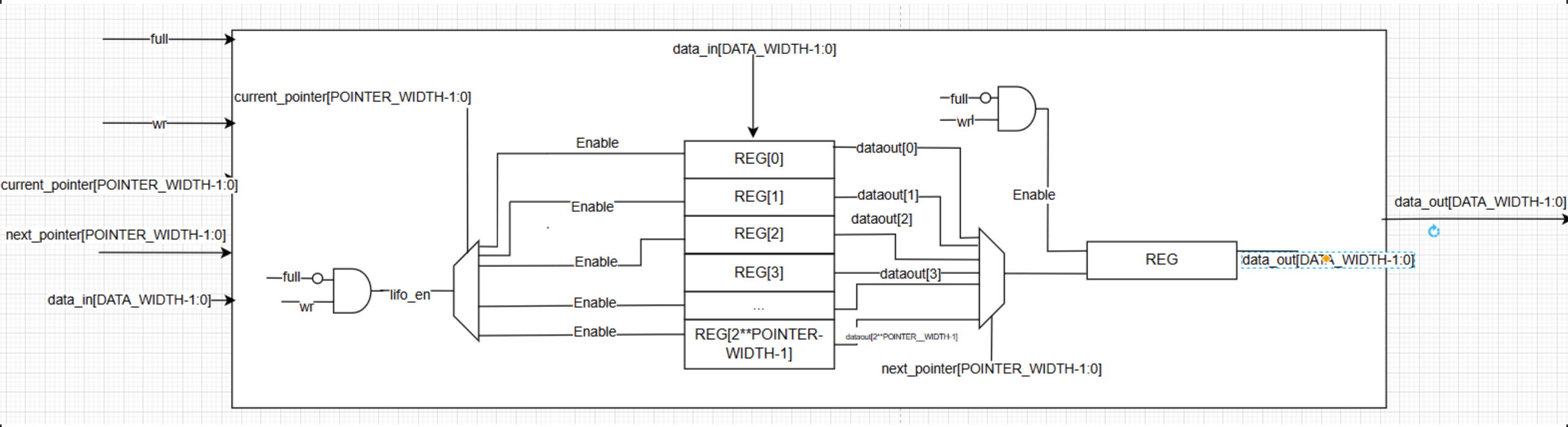
LOW LEVEL BLOCK
controller



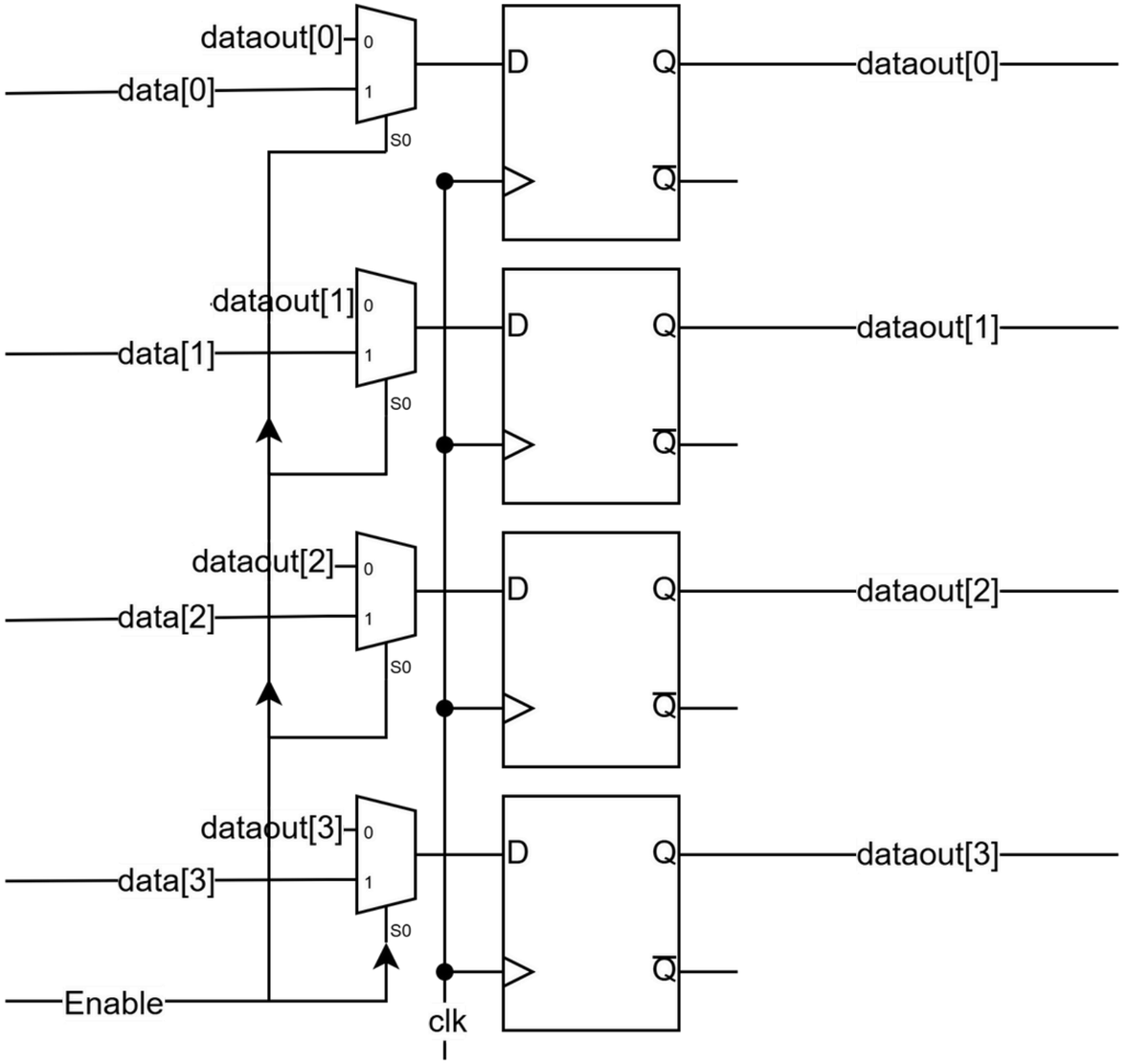
LOW LEVEL BLOCK
UD-OV



LOW LEVEL BLOCK
Register



LOW LEVEL BLOCK
REG



RTL DESCRIPTION

```
module counter#(
    parameter pointer_width = 2'd2
)(
    input clk,rst,wr,rd,
    output full, empty,
    output [pointer_width-1:0]pointer
);
    reg [pointer_width:0] current;
    always @(posedge clk, negedge rst) begin
        if(!rst) current<=0;
        else begin
          case({wr,rd})
            2'b00: current<=current;
            2'b01: current<=(empty==0)? (current-1):current;
            2'b10: current<=(full==0)?(current+1): current;
            2'b11: current<=current;
            default current<=current;
          endcase
        end
    end
  assign empty=(current==0)? 1'b1:0;
  assign full= (current[pointer_width]==1'b1) ? 1:0;
  assign pointer= current[pointer_width-1:0];
endmodule
```

RTL DESCRIPTION

```
26 module OVUD(
27     input empty,full,clk,rst,wr,rd,
28     output reg ud,ov
29 );
30   always @(posedge clk,negedge rst) begin
31     if(!rst) begin
32       ud<=0;
33       ov<=0;
34     end
35     else begin
36       ud<=empty&rd;
37       ov<=full&wr;
38     end
39   end
40 endmodule
```

RTL DESCRIPTION

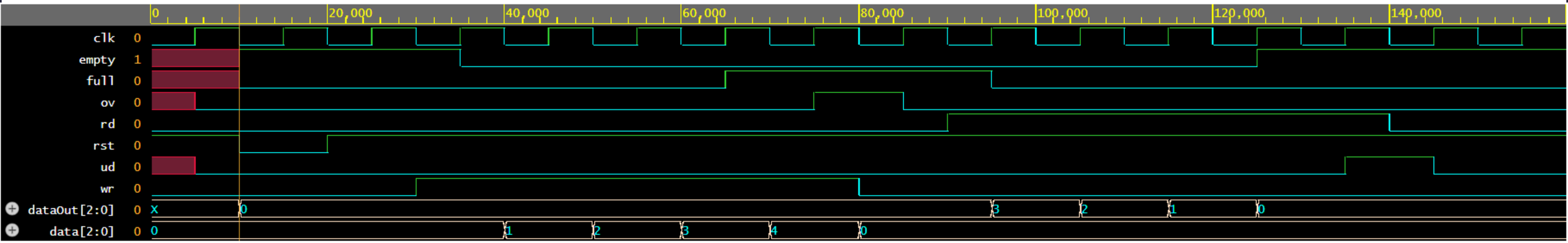
```
42 module register#(
43     parameter pointer_width=2'd2,
44     parameter data_width=2'd3
45 )(
46     input [pointer_width-1:0] pointer,
47     input [data_width-1:0] data,
48     input clk,wr,rd,rst,full,empty,
49   output reg [data_width-1:0] dataOut
50 );
51   reg [data_width-1:0] ghi [(1<<pointer_width)-1:0];
52   always @(posedge clk) begin
53     if(wr&&!full) ghi[pointer]<=data;
54   end
55   always @(posedge clk, negedge rst) begin
56     if( !rst) dataOut<=0;
57     else if (rd&&!empty) dataOut<=ghi[pointer-1'b1];
58     else dataOut<=dataOut;
59   end
60 endmodule
```

RTL DESCRIPTION

```
module LIFO#(
    parameter pointer_width=2'd2,
    parameter data_width=2'd3
)(
    input clk,rst,wr,rd,
    input [data_width-1:0] data,
    output full, empty,
  output [data_width-1:0] dataOut,
  output ud,ov
);
    wire [pointer_width-1:0] pointer;
    counter #(.pointer_width(pointer_width))
khoi1(.clk(clk),.rst(rst),.wr(wr),
                .rd(rd),.full(full),.empty(empty),.pointer(pointer));
  OVUD
khoi2(.clk(clk),.rst(rst),.wr(wr),.rd(rd),.full(full),.empty(empty),
               .ud(ud),.ov(ov));
  register #(.pointer_width(pointer_width),.data_width(data_width))
  khoi3(.pointer(pointer),.data(data),.clk(clk),

.dataOut(dataOut),.wr(wr),.rd(rd),.rst(rst),.empty(empty),.full(full));
endmodule
```

ACTUAL WAVEFORM



SYNTHESIS + STA

QOR

Timing

Clock	Period
des_clk	500.0

Cost Group	Critical Path Slack	TNS	Violating Paths
cg_enable_group_des_clk	67.2	0.0	0
default	No paths	0.0	
des_clk	No paths	0.0	
in2out	No paths	0.0	
in2reg	59.3	0.0	0
reg2out	60.4	0.0	0
reg2reg	213.0	0.0	0
Total		0.0	0

Instance Count

Leaf Instance Count	338
Physical Instance count	0
Sequential Instance Count	161
Combinational Instance Count	177
Hierarchical Instance Count	18

Area

Cell Area	1204.182
Physical Cell Area	0.000
Total Cell Area (Cell+Physical)	1204.182
Net Area	0.000
Total Area (Cell+Physical+Net)	1204.182

Power

Leakage Power	128.445 nW
Dynamic Power	651442.199 nW
Total Power	651570.644 nW

Number of Clock Gating Logic	18
Max Fanout	20 (clk)
Min Fanout	0 (rst)
Average Fanout	2.5
Terms to net ratio	3.5014
Terms to instance ratio	3.6361
Runtime	11.052196 seconds
Elapsed Runtime	32 seconds
Genus peak memory usage	1609.55
Innovus peak memory usage	no_value
Hostname	dtvt05.taile60e1a.ts.net

Netlists

```
module RC_CG_MOD(enable, ck_in, ck_out, test);
  input enable, ck_in, test;
  output ck_out;
  wire enable, ck_in, test;
  wire ck_out;
  TLATNTSCAX2 RC_CGIC_INST(.E (enable), .CK (ck_in), .SE (test), .ECK (ck_out));
endmodule

module RC_CG_MOD_1(enable, ck_in, ck_out, test);
  input enable, ck_in, test;
  output ck_out;
  wire enable, ck_in, test;
  wire ck_out;
  TLATNTSCAX2 RC_CGIC_INST(.E (enable), .CK (ck_in), .SE (test), .ECK (ck_out));
endmodule

module RC_CG_MOD_2(enable, ck_in, ck_out, test);
  input enable, ck_in, test;
  output ck_out;
  wire enable, ck_in, test;
  wire ck_out;
  TLATNTSCAX2 RC_CGIC_INST(.E (enable), .CK (ck_in), .SE (test), .ECK (ck_out));
endmodule

module RC_CG_MOD_3(enable, ck_in, ck_out, test);
  input enable, ck_in, test;
  output ck_out;
  wire enable, ck_in, test;
  wire ck_out;
  TLATNTSCAX2 RC_CGIC_INST(.E (enable), .CK (ck_in), .SE (test), .ECK (ck_out));
endmodule

module RC_CG_MOD_4(enable, ck_in, ck_out, test);
  input enable, ck_in, test;
  output ck_out;
  wire enable, ck_in, test;
  wire ck_out;
  TLATNTSCAX2 RC_CGIC_INST(.E (enable), .CK (ck_in), .SE (test), .ECK (ck_out));
endmodule
```