# Implementation Report Analysis

Vivado

August 16, 2024

# Chapter 1

# Introduction

This document is an analysis of the implementation report generated by Vivado for the `top` module. The report is generated after the implementation process is completed. The report contains information about the resources used by the design, the timing constraints, and the timing performance of the design. The report is used to analyze the design and to identify any potential issues that may need to be addressed.

# Chapter 2

# Resource Utilization

The resource utilization section of the report provides information about the resources used by the design. This includes information about the number of LUTs, FFs, BRAMs, and DSPs used by the design. The resource utilization section also provides information about the number of IOBs used by the design.

## 2.1 Common Resource Types

Here are some common resource types used in FPGA designs and Verilog codes that utilizes these resources:

### 2.1.1 LUTs

LUTs (Look-Up Tables) are fundamental building blocks in FPGA designs. They are used to implement combinational logic functions. Each LUT has a fixed number of inputs and a single output. The Verilog code snippet below shows an example of a 2-input LUT implementation:

```
module lut2 (input wire a, b, output wire y);
    assign y = a & b;
endmodule
```

### 2.1.2 FFs

FFs (Flip-Flops) are sequential elements used to store and propagate data in FPGA designs. They are commonly used to implement registers and memory elements. The Verilog code snippet below shows an example of a D flip-flop implementation:

```
module dff (input wire clk, reset, input wire d, output reg q);
    always @(posedge clk or posedge reset)
        if (reset)
```

```
            q <= 1'b0;
        else
            q <= d;
endmodule
```

### 2.1.3 BRAMs

BRAMs (Block RAMs) are memory elements in FPGA designs. They provide large storage capacity and high-speed access. They can be coded into fpga, or The Verilog code snippet below shows an example of a simple dual-port RAM implementation:

```
module dual_port_ram (input wire clk, input wire [7:0] addr_a, addr_b,
                      input wire [7:0] data_a, data_b,
                      input wire we_a, we_b,
                      output reg [7:0] q_a, q_b);
    reg [7:0] mem [0:255];

    always @(posedge clk)
        if (we_a)
            mem[addr_a] <= data_a;

    always @(posedge clk)
        if (we_b)
            mem[addr_b] <= data_b;

    always @(*)
        q_a = mem[addr_a];
        q_b = mem[addr_b];
endmodule
```

### 2.1.4 DSPs

DSPs (Digital Signal Processors) are specialized hardware blocks in FPGA designs used for high-performance signal processing operations. They are commonly used in applications such as image and audio processing. The Verilog code snippet below shows an example of a simple multiplier using DSP blocks:

```
module multiplier (input wire [7:0] a, b, output wire [15:0] result);
    assign result = a * b;
endmodule
```

### 2.1.5 IOBs

IOBs (Input/Output Blocks) are used to interface the FPGA design with external devices. They provide the necessary logic to drive and receive signals from the external world. The Verilog code snippet below shows an example of an IOB implementation for a simple GPIO (General Purpose Input/Output) interface:

```
module gpio (input wire clk, input wire reset, input wire [7:0] data_in,
             output wire [7:0] data_out, inout wire [7:0] io_pins);
    reg [7:0] internal_reg;

    always @(posedge clk or posedge reset)
        if (reset)
            internal_reg <= 8'b0;
        else
            internal_reg <= data_in;

    assign data_out = internal_reg;
    assign io_pins = internal_reg;
endmodule
```

## 2.2 Resource Utilization Report

The resource utilization report provides information about how many of these components are available in the FPGA and how many are used by your design. It helps you understand how efficiently your design is utilizing the available resources and whether any adjustments or optimizations are needed.

**There are 8 parts in the resource utilization report.**

1. Slice Logic: How many LUTs and FFs are used in the design. How many of LUTs are used as Logic or Memory and how many of FFs are used as Registers or Latches.
2. Slice Logic Distribution: How the LUTs and FFs are distributed across the slices.
3. Memory: Usage report of build in memory blocks. There can be different types for different fpga families.
4. DSPs: Usage report of DSP blocks.
5. IOs
6. BUFG/BUFGCTRL (Clocking): Usage report of clocking resources.
7. Specific Feature: Usage report of specific featured blocks of the fpga card.
8. Primitive: How many of each primitive type are utilized in your design. A more detailed report of the primitives used in the design.

Exampla scenarios to chose which part to look at:

- If you wanted to know exactly how many flip-flops or LUTs of a particular configuration (like FDRE for a flip-flop with enable) were used, you would consult the Primitive Types Report.
- If the design is using a lot of memory resources as LUT, you may want to look at the Memory section to see if there are any optimizations that can be made.
- If the design is using a lot math operations, you may want to look at the DSP section to see if the design is close to the limits.
- *****The synthesis tool might not be able to utilize the desing as desired. A different block of fpga can be used to implement your design. For example a buffer can be utilized in BRAMs, LUTRAMs, or FFs. Simple changes in your design can make implementation tool to chose a different resource.**

An example table from resource utilization report for the `top` module is shown below:

```
8. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  178 |        Flop & Latch |
| LUT6     |   79 |                 LUT |
| LUT4     |   48 |                 LUT |
| LUT5     |   22 |                 LUT |
| LUT2     |   21 |                 LUT |
| LUT3     |   18 |                 LUT |
| CARRY4   |   11 |          CarryLogic |
| FDSE     |   10 |        Flop & Latch |
| OBUF     |    7 |                  IO |
| RAMB18E1 |    4 |        Block Memory |
| IBUF     |    3 |                  IO |
| LUT1     |    2 |                 LUT |
| BUFG     |    2 |               Clock |
+----------+------+---------------------+
```

General file names for this report are `Module_Name_utilization_routed.rpt` or `Module_Name_utilization_placed.rpt`.