My LaTeX Template

pastglory*

目录

摘 要

这是一段摘要, 这个仓库主要保存我的 LaTeX 模版，用于各种文档的书写，目前实现的功能较少，有待日后在使用中不断优化。

第一章 简介

你好，LaTeX! 这个仓库主要保存我的 LaTeX 模版，用于各种文档的书写。为了实现自由扩展的需求，一切格式上的改动都放在 cls 文件中，并且所有实质性内容都放在 src 文件夹下，main.tex 只用于整理，作为顶层。

为了测试参考文献格式是否正确，使用一篇稀疏运算加速的论文[1] 以及一篇老化预测的论文[2] 作为参考文献样例。

第二章 测试

2.1 结构

图1为 FPGA 基本单元结构图。

*sunyata000@hotmail.com

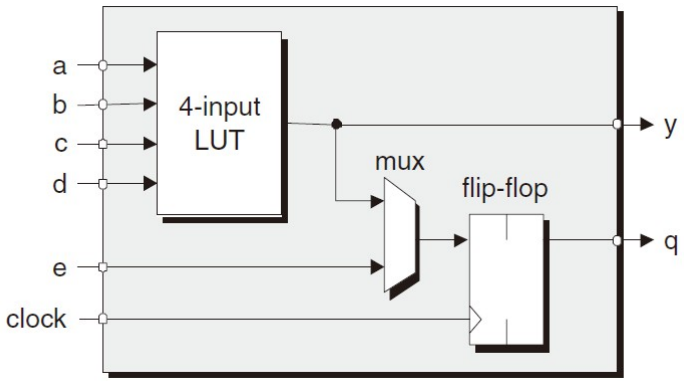

图 1: Slice of FPGA

接下来是代码风格的描述。

2.2 数学

这里有一个数学公式

$$\int_1^2 x dx = \frac{3}{2} \tag{2-1}$$

2.3 代码

推荐的 verilog 代码风格如下所示。推荐的 verilog 代码风格如下所示。推荐的 verilog 代码风格如下所示。

```
module keyboard(
    input             clk,
    input             rst_n,
    input  [3 : 0]    col, // column input, 1 enable
    output [3 : 0]    key_pulse // if key pushed, output 1
);

reg  [11 : 0] treg0; // tmp reg
reg  [11 : 0] treg1; // tmp reg
reg  [11 : 0] treg2; // tmp reg
reg  [11 : 0] treg3; // tmp reg
wire [11 : 0] treg0_nxt = treg0 + 1'b1; // tmp reg next val
wire [11 : 0] treg1_nxt = treg1 + 1'b1; // tmp reg next val
wire [11 : 0] treg2_nxt = treg2 + 1'b1; // tmp reg next val
```

```
wire [11 : 0] treg3_nxt = treg3 + 1'b1; // tmp reg next val

// when tregx = 12'hffe and tregx_nxt = 12'hfff
// key_pulse will be 1
// then, tregx will be 12'hfff
// when tregx == 12'hfff, it will keep until keyboard not pushed
always @ (posedge clk or negedge rst_n) begin
    if (~rst_n) begin
        treg0 <= 12'b0;
        treg1 <= 12'b0;
        treg2 <= 12'b0;
        treg3 <= 12'b0;
    end
    else begin
        if (col[0]) begin
            if (treg0 != 12'hfff)
                treg0 <= treg0_nxt;
        end
        else begin
            treg0 <= 12'b0;
        end

        if (col[1]) begin
            if (treg1 != 12'hfff)
                treg1 <= treg1_nxt;
        end
        else begin
            treg1 <= 12'b0;
        end

        if (col[2]) begin
            if (treg2 != 12'hfff)
                treg2 <= treg2_nxt;
```

```
            end
            else begin
                treg2 <= 12'b0;
            end

            if (col[3]) begin
                if (treg3 != 12'hfff)
                    treg3 <= treg3_nxt;
            end
            else begin
                treg3 <= 12'b0;
            end
        end
    end

    assign key_pulse[3] = (treg3 != 12'hfff) & (treg3_nxt == 12'hfff);
    assign key_pulse[2] = (treg2 != 12'hfff) & (treg2_nxt == 12'hfff);
    assign key_pulse[1] = (treg1 != 12'hfff) & (treg1_nxt == 12'hfff);
    assign key_pulse[0] = (treg0 != 12'hfff) & (treg0_nxt == 12'hfff);

endmodule
```

参考文献

[1] Z. Zhang, H. Wang, S. Han, and W. J. Dally, "Sparch: Efficient architecture for sparse matrix multiplication," in *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2020, pp. 261–274.

[2] M. Sadi, G. K. Contreras, J. Chen, L. Winemberg, and M. Tehranipoor, "Design of reliable socs with bist hardware and machine learning," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 11, pp. 3237–3250, 2017.