INSTITUTO FEDERAL
São Paulo

Linguagem e Descrição de Hardware

Greatest Common Divisor - (GCD)

Giovanna Fantacini[1], Higor Grigorio[2], Leonardo Reneres[3], Luis[4], Raul Prado Dantas[5]

RESUMO: Este artigo apresenta a implementação do algoritmo para cálculo do Greatest Common Divisor (GCD) em hardware, utilizando uma abordagem de Control e Datapath. Serão abordadas as principais etapas de desenvolvimento, incluindo a descrição do algoritmo em linguagem C, a modelagem dos módulos de controle e datapath em Verilog, e a simulação e validação do sistema.

PALAVRAS-CHAVE: GCD; Greatest Common Divisor; Hardware Description Language; Control; Datapath.

TITLE IN ENGLISH

ABSTRACT: This paper presents the implementation of the Greatest Common Divisor (GCD) algorithm in hardware, using a Control and Datapath approach. The main development steps will be addressed, including the description of the algorithm in C language, the modeling of control and datapath modules in Verilog, and the system simulation and validation.

KEYWORDS: GCD; Greatest Common Divisor; Hardware Description Language; Control; Datapath.

INTRODUÇÃO

O Greatest Common Divisor (GCD) é um problema clássico da teoria dos números, com aplicações em diversas áreas da computação. Este trabalho baseia-se nas notas de aula da disciplina CSE 141L: Introduction to Computer Architecture Lab, ministrada na Universidade da Califórnia. O objetivo é desenvolver uma implementação eficiente do GCD utilizando linguagem de descrição de hardware.

O cálculo do GCD é frequentemente utilizado em algoritmos de criptografia, compressão de dados e outros campos onde operações matemáticas eficientes são essenciais. Este projeto visa criar um design em hardware que maximize a eficiência dessas operações.

As referências devem estar citadas no trabalho conforme a sua forma de citação, como por exemplo (ALVES; RODRIGUES, 2004), (GALVANI, 2008) e (INSTRUMENTS, 2019) ou em Pandorfi, et al, (2007). Na seção Referências devem ser listadas em ordem alfabética (PANDORFI et al.,).

MATERIAIS E MÉTODOS

Os materiais e métodos utilizados no desenvolvimento da pesquisa incluem a descrição do algoritmo em C, a modelagem dos módulos de controle e datapath em Verilog, e a simulação utilizando o ambiente Intel QuestaSim.

Descrição do Algoritmo em C

O algoritmo para cálculo do GCD pode ser implementado de diversas maneiras. Abaixo, apresentamos uma implementação em linguagem C:

```
int GCD(int inA, int inB) {
    int swap;
    int done = 0;
    int A = inA;
    int B = inB;
    while (!done) {
        if (A < B) {
            swap = A;
            A = B;
            B = swap;
        } else if (B != 0) {
            A = A - B;
        } else {
            done = 1;
        }
    }
    return A;
}
```

Modelagem em Verilog

A implementação do GCD em hardware envolve a criação de módulos para o controle e o datapath. O datapath lida com a movimentação e transformação dos dados, enquanto o módulo de controle gerencia as operações de controle.

Módulo Datapath

```
module GCDdatapath#( parameter W=16 )
(
    input clk,
    input [W-1:0] operand_A,
    input [W-1:0] operand_B,
    output [W-1:0] result_data,
    input A_ld,
    input B_ld,
    input [1:0] A_sel,
```

```
    input B_sel,
    output B_zero,
    output A_lt_B,
    output [W-1:0] A_chk, B_chk, sub_chk, A_mux_chk, B_mux_chk
);

    wire [W-1:0] A;
    wire [W-1:0] B;
    wire [W-1:0] sub_out;
    wire [W-1:0] A_mux_out;
    wire [W-1:0] B_mux_out;

    Mux3#(W) A_mux
    (.in0 (operand_A),
    .in1 (sub_out),
    .in2 (B),
    .sel (A_sel),
    .out (A_mux_out) );

    register#(W) A_reg
    (.clk (clk),
    .d (A_mux_out),
    .en (A_ld),
    .q (A) );

    Mux2#(W) B_mux
    (.in0 (A),
    .in1 (operand_B),
    .sel (B_sel),
    .out (B_mux_out) );

    register#(W) B_reg
    (.clk (clk),
    .d (B_mux_out),
    .en (B_ld),
    .q (B) );

    assign B_zero = (B==0);
    assign A_lt_B = (A < B);
    assign sub_out = A - B;
    assign result_data = A;
    // send checking signals only for debugging purposes
    assign A_chk = A;
    assign B_chk = B;
```

```
    assign sub_chk = sub_out;
    assign A_mux_chk = A_mux_out;
    assign B_mux_chk = B_mux_out;
endmodule
```

Módulo Controle

```
module GCDcontrol(
    input input_available,
    output reg idle,
    input clk, reset,
    output reg A_ld, B_sel, B_ld,
    output reg [1:0] A_sel,
    input B_zero, A_lt_B,
    output reg result_rdy,
    input result_taken,
    output [1:0] State
);

    // States naming
    localparam WAIT = 2'd0;
    localparam CALC = 2'd1;
    localparam DONE = 2'd2;

    // Constants naming for A_mux selector
    localparam A_SEL_IN = 2'b00;
    localparam A_SEL_SUB = 2'b01;
    localparam A_SEL_B = 2'b10;
    localparam A_SEL_X = 2'b11;
    // Constants naming for B_mux selector
    localparam B_SEL_A = 1'b0;
    localparam B_SEL_IN = 1'b1;
    localparam B_SEL_X = 1'bx;

    reg [1:0] CurrentState, NextState;

    always @(posedge clk or posedge reset)
    begin
        if (reset)
            CurrentState <= WAIT;
        else
            CurrentState <= NextState;
    end

    always @(CurrentState)
```

```
begin
    // default is to stay in the same state
    NextState <= CurrentState;
    case ( CurrentState )
        WAIT :
            if ( input_available )
                NextState <= CALC;
        CALC :
            if ( B_zero )
                NextState <= DONE;
        DONE :
            if ( result_taken )
                NextState <= WAIT;
    endcase
end

always @( * )
begin
    // Default control signals
    A_sel <= A_SEL_X;
    A_ld <= 1'b0;
    B_sel <= B_SEL_X;
    B_ld <= 1'b0;
    idle <= 1'b0;
    result_rdy = 1'b0;

    case ( CurrentState )
        WAIT :
            begin
                idle <= 1'b1;
                if(input_available)begin
                    A_sel <= A_SEL_IN;
                    B_sel <= B_SEL_IN;
                    A_ld <= 1'b1;
                    B_ld <= 1'b1;
                end
            end
        CALC :
            if ( A_lt_B )begin
                A_sel <= A_SEL_B;
                B_sel <= B_SEL_A;
                A_ld <= 1'b1;
                B_ld <= 1'b1;
            end
```

```
                else if ( !B_zero )begin
                    A_sel <= A_SEL_SUB;
                    A_ld <= 1'b1;
                end
            DONE :
                result_rdy <= 1'b1;
        endcase
    end

    assign State = CurrentState;
endmodule
```

RESULTADOS E DISCUSSÃO

Para validar a implementação, foi realizada a simulação dos módulos utilizando o ambiente Intel QuestaSim. A Figura 1 mostra a simulação do módulo GCD.

Figura 1: Simulação do módulo GCD

Os resultados da simulação confirmam que o design do GCD em hardware funciona conforme esperado. O módulo datapath executa corretamente as operações de subtração e troca, enquanto o módulo de controle gerencia os estados do sistema de maneira eficiente.

CONCLUSÕES

Neste artigo, apresentamos a implementação do algoritmo GCD em hardware, detalhando a modelagem dos módulos de controle e datapath em Verilog. A simulação mostrou que a abordagem utilizada é eficiente e atende aos requisitos do projeto. Futuras melhorias podem incluir a otimização do datapath e a integração com outros módulos de um sistema maior.

CONTRIBUIÇÕES DOS AUTORES

Raul Prado Dantas, Higor Grigorio dos Santos, Leonardo Reneres, Giovanna Fantacini e Luis contribuíram com a concepção e escopo do estudo. Higor e Raul procederam com a metodologia e experimentos. Leonardo, Giovanna e Luis escreveram o trabalho. Todos os autores contribuíram com a revisão do trabalho e aprovaram a versão submetida.

AGRADECIMENTOS

Gostaríamos de agradecer ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo pelo suporte e infraestrutura fornecidos para a realização deste projeto. Agradecemos também aos professores Michael B. Taylor e Matt Devuyst por suas valiosas contribuições através das notas de aula da disciplina CSE 141L.

REFERÊNCIAS

ALVES, S. P.; RODRIGUES, E. Sombreamento arbóreo e orientação de instalações avícolas. *Engenharia Agrícola*, v. 24, n. 2, p. 241 – 245, 2004.

GALVANI, E. Estudo comparativo dos elementos do balanço hídrico climatológico para duas cidades do estado de são paulo e para paris. Confins [Online], v. 4, n. 4, p. 1–106, 2008. Disponível em: <<http://confins.revues.org/4733>.doi:10.400/confins.4733>.

INSTRUMENTS, N. *Data Acquisition*. [S.l.], 2019. Disponível em: <http://www.ni.com/pt-br/shop/select/compactdaq-controller>. Acesso em: 19 Abril. 2019.

PANDORFI, H. et al. Estudo da lógica fuzzy na caracterização do ambiente produtivo para matrizes gestantes. *Engenharia Agrícola*, v. 27, n. 1, p. 83–92. Disponível em: <http://www.scielo.br/pdf/eagri/v27n1/01.pdf>. Acesso em: 24 Setembro. 2007.