

RELATÓRIO SOBRE PROJETO DE SISTEMA EMBARCADO COM FOCO EM ALTO DESEMPENHO

Santa Maria, 03 de setembro de 2018

Keli Tauana Prass Ruppenthal¹

Victor Dallagnol Bento²

Resumo: O relatório em questão trará dados e detalhes acerca do trabalho desenvolvido na disciplina de Projetos de Sistemas Embarcados II. A primeira etapa do projeto (demonstrada neste relatório), necessita que seja elaborado um *Data Path* e FSM a partir de um algoritmo que calcula a raiz quadrada de um número inteiro de entrada.

Palavras-chave: Desempenho, frequência, área, consumo, potência.

1. Introdução

Comumente chamados de Sistemas Embarcados, é esta a mais recente designação para programas e sistemas embutidos em microprocessadores que executam tarefas específicas em um aparelho. Eles estão presentes em diversos equipamentos do dia a dia, como semáforos, aparelhos de ar condicionado, impressoras, tablets, smartphones, MP3 players, entre outros.

No caso deste trabalho, a primeira etapa, e que será tratada aqui, é relativamente simples e consiste em uma das principais etapas para o desenvolvimento de um sistema embarcado. Serão apresentados um grafo de FSM e um *Data Path* para um circuito de cálculo de raiz quadrada e que visa um bom desempenho em sua execução.

Contudo, embora simples, é necessário que seja feita uma boa análise de todos os pontos que influenciarão no desempenho do circuito nas próximas etapas do trabalho (como caminhos críticos, por exemplo). Isso porque, uma escolha errada nesta etapa pode não poder ser desfeita futuramente, e isso prejudicará o circuito de forma permanente.

2. Objetivos

Como principais objetivos deste trabalho, destacam-se o estudo e planejamento de uma boa máquina de estados e de um *Data Path* correspondente. Como o foco do grupo é desempenho, a área e a potência não serão de grande preocupação, uma vez que estas grandezas são inversamente proporcionais, conforme a **Equação 1**.

$$P = C \times f \times V_{dd}^2$$

Equação 1: Relação entre potência e frequência.

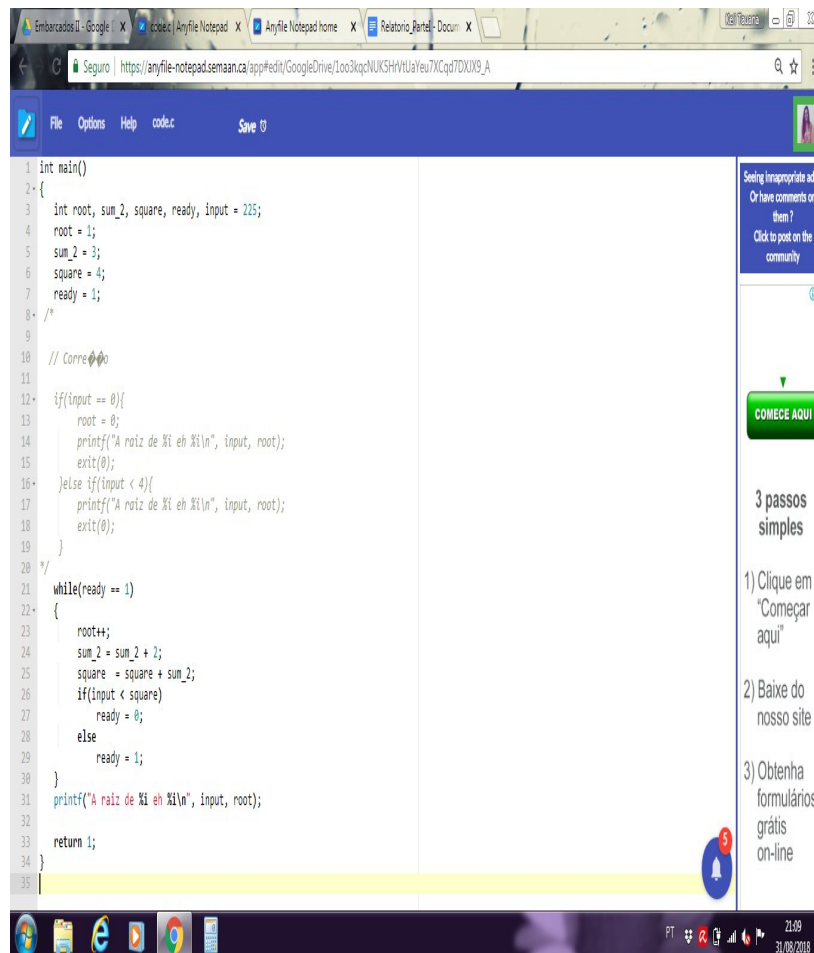
Como já foi mencionado anteriormente, esta primeira etapa do trabalho é relativamente simples, porém de grande importância, pois o modelo de FSM adotado agora não poderá mais ser alterado. Isso significa que todos os passos, ciclos, instruções e sinais devem ser bem arranjados nesta etapa inicial.

¹ Acadêmico(a) do curso de Engenharia de Computação da Universidade Federal de Santa Maria - UFSM, matrícula: 201520603, email: kelitauana@gmail.com

² Acadêmico(a) do curso de Engenharia de Computação da Universidade Federal de Santa Maria - UFSM, matrícula: 201520835, email: victor.bento@ecomp.ufsm.br

3. Metodologia

Inicialmente, analisou-se o algoritmo responsável pelo cálculo da raiz quadrada de um número inteiro. Verificou-se que este código não funciona corretamente para raízes menores do que 4. Embora este erro exista, o grupo optou por não consertá-lo, uma vez que não foi requerido pela tarefa em si, além de que mais hardware teria que ser adicionado, fazendo com que o desempenho caísse também. O algoritmo em questão é encontrado na **Figura 1**.



```
1 int main()
2 {
3     int root, sum_2, square, ready, input = 225;
4     root = 1;
5     sum_2 = 3;
6     square = 4;
7     ready = 1;
8     /*
9
10    // Correção
11
12    if(input == 0){
13        root = 0;
14        printf("A raiz de %i eh %i\n", input, root);
15        exit(0);
16    }else if(input < 4){
17        printf("A raiz de %i eh %i\n", input, root);
18        exit(0);
19    }
20    */
21    while(ready == 1)
22    {
23        root++;
24        sum_2 = sum_2 + 2;
25        square = square + sum_2;
26        if(input < square)
27            ready = 0;
28        else
29            ready = 1;
30    }
31    printf("A raiz de %i eh %i\n", input, root);
32
33    return 1;
34 }
35
```

Figura 1: Código base para a descrição da FSM e Data Path.

O próximo passo foi a elaboração da FSM. A ferramenta utilizada foi o *fizzim*, o qual foi bastante simples e intuitivo de usar. A FSM final está disposta na **Figura 2**.

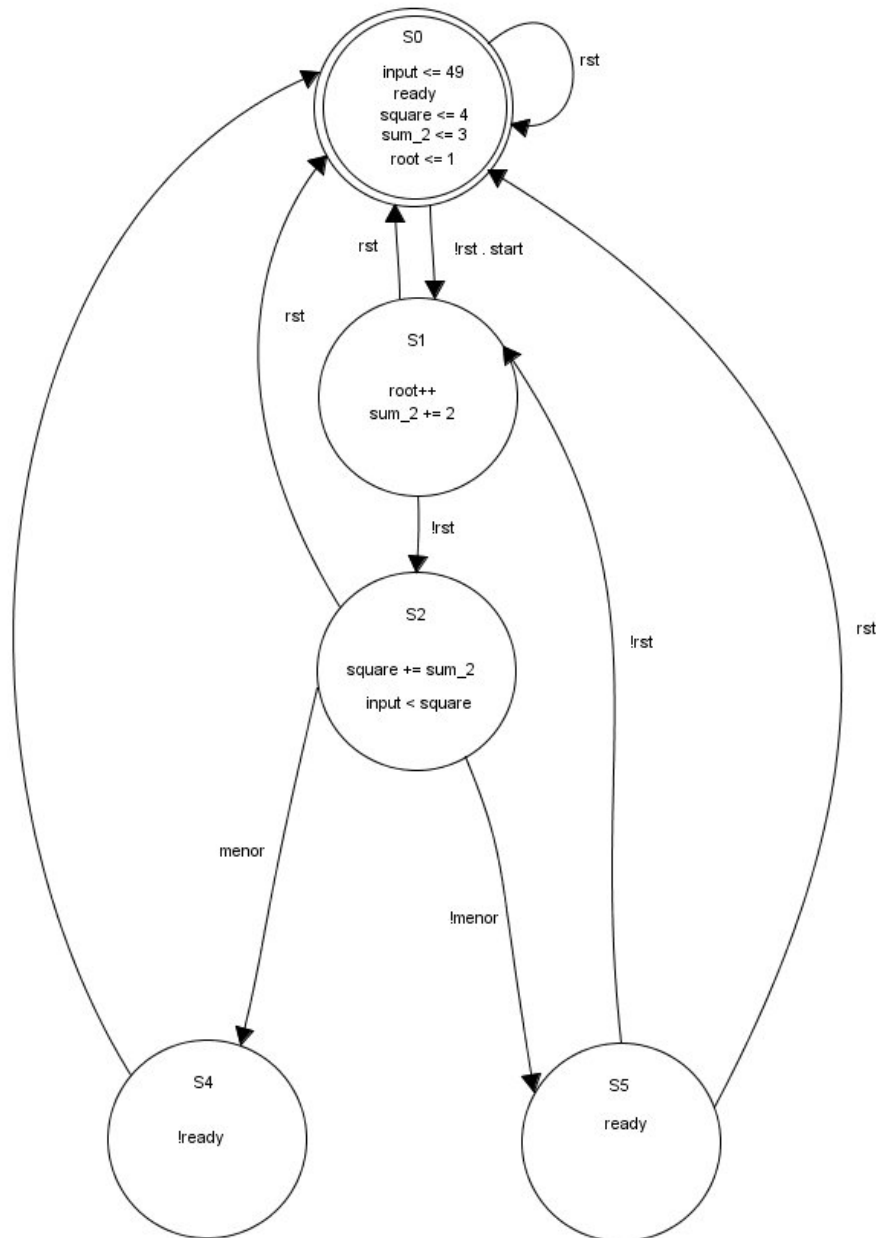


Figura 2: FSM para o código da figura anterior.

Feita a FSM, a última etapa é a confecção do *Data Path*, visto na **Figura 3**. O programa utilizado foi o *Logisim*. Nesta etapa, também seria necessária a criação dos sinais de controle para o funcionamento do circuito. Nota-se que no esquemático, o grupo não se preocupou em economizar área, pois o bom desempenho e a frequência não permitem essa preocupação. Na **Figura 4** estão as tabelas de *Next State* e *Control Path* das outras unidades do circuito.

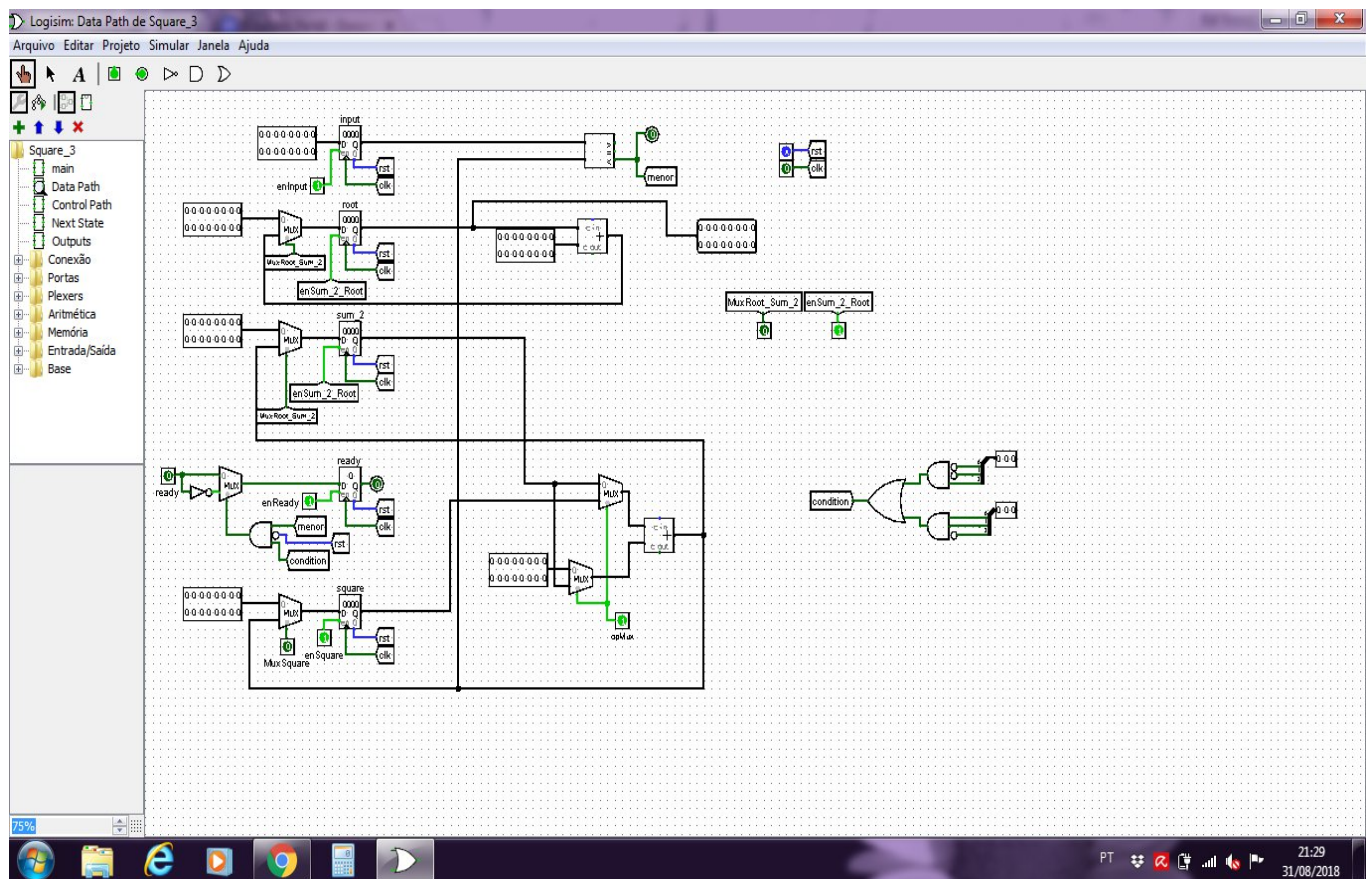


Figura 3: Data Path para o código da figura anterior.

Next State									
Q2	Q1	Q0	rst	menor	start	*s2	*s1	*s0	
0	0	0	1	x	x	0	0	0	
0	0	0	0	x	0	0	0	0	
0	0	0	0	x	1	0	0	1	
0	0	1	1	x	x	0	0	0	
0	0	1	0	x	x	0	1	0	
0	1	0	1	x	x	0	0	0	
0	1	0	0	1	x	0	1	1	
0	1	0	0	0	x	1	0	0	
0	1	1	x	x	x	0	0	0	
1	0	0	0	x	x	0	0	1	
1	0	0	1	x	x	0	0	0	

Tabela Control Path									
Q2	Q1	Q0	enInput	enReady	enSquare	enSum_2_Root	MuxRoot_Sum_2	MuxSquare	OpMux
0	0	0	1	1	1	1	0	0	x
0	0	1	0	0	0	1	1	0	0
0	1	0	0	0	1	0	0	1	1
0	1	1	0	1	0	0	x	x	x
1	0	0	0	1	0	0	x	x	x

Figura 4: Tabelas de Next State e Control Path.

Ainda, criou-se a **Tabela 1** com todos os componentes utilizados no Data Path. Nota-se a utilização de dois somadores, os quais são os caminhos críticos na etapa atual, e que poderão ser melhorados nas próximas etapas do projeto.

Componente	Quantidade
Somador	2
Multiplexador	6
Registradores	5
Comparadores	1
Subtratores	0
Porta AND	3
Porta OR	1
Porta NOT	5
Entradas (bits)	6 (16), 1 (1)

Tabela 1: Quantidade de componentes para o *Data Path*.

4. Resultados e Discussão

Tendo feitas todas as partes solicitadas nesta etapa do projeto, algumas análises podem ser feitas. Primeiramente, como já foi dito, a raiz para números menores do que 4 não funciona. Esta escolha do grupo reflete em uma economia de hardware e um bônus para o desempenho, entretanto o funcionamento não é 100%.

Ademais, a FSM foi feita basicamente de duas formas: uma delas continha 6 estados, sendo que um deles era utilizado apenas para fazer uma comparação (pois o valor do registrador estaria atualizado só na transição para este estado). Esse estado “a mais” se repete conforme o número aumente de valor. Por exemplo, para a raiz de 4, ele se repete apenas uma vez, mas para a raiz de 121, ele se repete dez vezes. Assim, surgiu a segunda opção, na qual a comparação é feita imediatamente após a soma, antes do valor ser gravado no registrador. Dessa forma, o grupo optou pela segunda opção de implementação, uma vez que maiores possibilidades de mudança (troca de somadores, comparadores...) podem ser feitas neste modelo, quando forem sendo especificadas outras partes para o projeto.

Decidido o modelo de FSM a ser seguido, construiu-se o *Data Path* (e também todos os outros blocos que não serão especificados completamente aqui). Os valores lidos no estado **S0** ficam armazenados em registradores, os quais possuem multiplexadores em suas entradas para a opção de *reset* e mudança do valor atual. O registrador *Ready* possui uma lógica de escrita um pouco diferente. Como seu valor é dependente do término ou continuidade do cálculo, os estados em que ele muda são **S3** e **S4**. Dessa forma, a porta lógica OR faz essa atribuição. Contudo, a flag *Menor* deve ser levada em consideração também. Ela compara os valores de *input* e *square* e, caso *input* seja menor, o cálculo acaba e o valor de *Ready* é alterado. Ainda, se o sinal de *reset* estiver ativado, o valor a ser escrito em *Ready* deve ser o de default, ou seja, 1. O sinal *rst* negado na entrada da porta

AND, responsável por toda esta lógica, garante a passagem da entrada zero do multiplexador. Dessa maneira, a porta AND recebe a flag *Menor*, o sinal de *rst* negado e o túnel *Condition*, que traz o resultado da porta OR citada anteriormente.

Tratando-se dos somadores, os dois que foram utilizados neste circuito estão dispostos de modo a serem aproveitados ao máximo. O somador superior faz apenas o incremento do registrador *Root*, enquanto que o segundo faz dois cálculos em ciclos diferentes. Este somador possui dois multiplexadores em suas entradas, tornando possível o reaproveitamento desta unidade, uma vez que ela é o caminho crítico do circuito até o momento, principalmente pelo acréscimo do comparador em sua saída.

Levando-se em consideração o tempo para a elaboração desta etapa do projeto, o grupo preocupou-se em não deixar a análise e elaboração para o final do prazo estipulado. Isso contribuiu para uma boa organização de ideias e opções de mudança e melhorias do circuito, pois o sistema pode ser testado muitas vezes. Além disso, dúvidas foram sendo sanadas com o auxílio do professor, o que rendeu bons resultados no desenvolvimento do trabalho em questão.

Referências

- [1] Conhecimento adquirido em disciplinas anteriores.