UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
DISCIPLINA DE INTEGRAÇÃO SOFTWARE/HARDWARE
ALUNOS: DANIEL PEREIRA VOLPATO
LEONARDO LUIZ ECCO

Multilateration IP

Grupo β1

SUMÁRIO

OBJETIVOS 3

PLANEJAMENTO 4

MULTILATERAÇÃO 5

ORIGEM: A TRILATERAÇÃO 5

O ALGORITMO 6

API MATEMÁTICA DEFINIDA 8

PROJETO DO IP 9

PROBLEMAS INICIAIS 9

LARGURA DE PALAVRA INTEIRA MUITO GRANDE 9

A ÁREA LIMITADA DISPONÍVEL NA PLATAFORMA 10

DESENVOLVIMENTO DE EXPRESSÃO ÚNICA E SIMPLIFICAÇÃO DA MESMA 10

MODELO DE REFERÊNCIA 11

PARTICIONAMENTO 11

COMPONENTE MULTILATERATION 12

COMPONENTE EXPS_CALC 13

COMPONENTE DIV_FACTOR_CALC 14

COMPONENTE NUMERATORS_CALC 14

COMPONENTE XY_CALC 15

VERIFICAÇÃO DO IP 16

INTEGRAÇÃO 17

ON-CHIP PERIPHERAL BUS 17

INTEGRAÇÃO DA PLATAFORMA 17

PROBLEMAS COM A INTEGRAÇÃO 18

Objetivos

O grupo β1 tinha como seguintes objetivos:

- Desenvolver um IP que implementa um algoritmo de multilateração (definido pelo grupo de localização β2) utilizando uma FPGA.
- Integrá-lo numa plataforma composta também por: um PowerPC (hard-core), uma FPU, três UARTs e um controlador de LED.

Planejamento

O grupo desenvolveu no início do semestre um cronograma. Este foi sendo refinado, através de auxílio do professor e monitores, até atingir o seguinte formato:

Atividade	Semana Prevista
1) Definição da API matemática	1-2
2) Estudo da viabilidade do uso de ponto flutuante	1-2
3) Familiarização com tecnologias	3-9
4) Projeto do IP	3-7
5) Verificação do IP	4-9
6) Apresentação	8
7) Integração	9-11

Segue abaixo uma breve descrição das atividades:

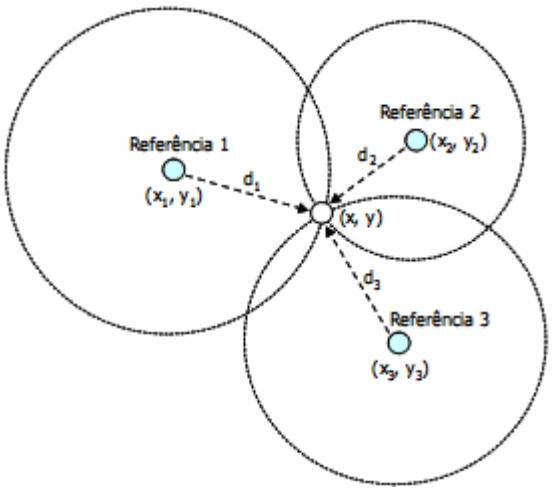
1) Auxiliar o grupo β1 na definição do algoritmo de localização mais apropriado para o grupo; e definir a API matemática que o IP fornecerá.
2) Estudo da viabilidade do uso de valores em ponto flutuante para o cálculo da multilateração.
3) Familiarização com tecnologias como os softwares da Xilinx ISE e EDK, com o kit ML403, entre outros.
4) O projeto do IP em si.
5) Verificação funcional do IP.
6) Apresentação de andamento do grupo.
7) Integração do IP na plataforma sendo projetado por todo o grupo β.

Multilateração

A multilateração é um algoritmo para cálculo de posição. Não se pode falar dela sem falar da onde foi derivada, ou seja, da trilateração.

Origem: a trilateração

O método mais básico e intuitivo, é usado quando são conhecidas as posições de três referências e as distâncias delas até o ponto que se deseja estimar, como mostra a figura abaixo:



Trilateração

Dada a figura, pode-se afirmar que a trilateração se dá pela intersecção de três círculos centrados na posição das referências, tendo o raio igual a cada medição de alcance.

Para cada referência, teremos uma equação do tipo:

$$(x-x_i)^2 + (y-y_i)^2 = d_i^2,$$

onde:

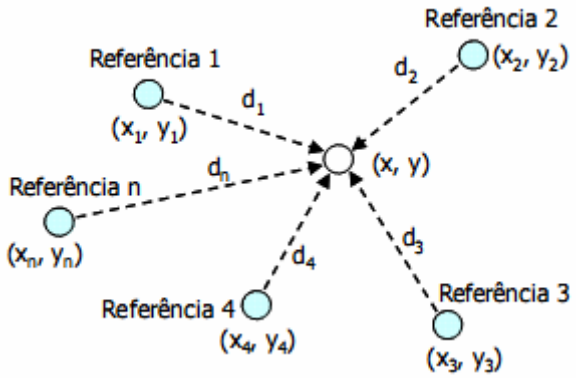
(x,y) é a posição que se deseja calcular;

(x_i, y_i) é a posição da referência i;

d_i é a distância do sensor a referência i.

O algoritmo

A multilateração nada mais é que a trilateração, porém considerando mais de três referências para o cálculo da posição, como demonstra a figura abaixo:



Multilateração

Dadas as referências, obtém-se um sistema de equações como este:

$$\begin{cases} (x_1 - x)^2 + (y_1 - y)^2 = d_1^2 \\ \vdots \\ (x_n - x)^2 + (y_n - y)^2 = d_n^2 \end{cases}$$

Cada tupla (x_i, y_i, d_i) da equação pode ser interpretada da seguinte forma: um beacon, localizado em (x_i, y_i) está à uma distância d_i do alvo. Considerando que nas extremidades do campo de jogo haverão quatro âncoras posicionados, tem-se ***n* = *4***, ou seja, quatro equações.

A resolução do sistema acontece substituindo uma das equações nas outras n-1 equações, e resolvendo o sistema resultante através da fórmula:

$$\hat{x} = (A^T A)^{-1} A^T b \text{ ,}$$

onde:

$$A = \begin{bmatrix} 2(x_1 - x_n) & 2(y_1 - y_n) \\ \vdots & \vdots \\ 2(x_{n-1} - x_n) & 2(y_{n-1} - y_n) \end{bmatrix}$$

$$b = \begin{bmatrix} x_1^2 - x_n^2 + y_1^2 - y_n^2 + d_n^2 - d_1^2 \\ \vdots \\ x_{n-1}^2 - x_n^2 + y_{n-1}^2 - y_n^2 + d_n^2 - d_{n-1}^2 \end{bmatrix}$$

A equação acima resultará num vetor contendo os respectivos valores de *x* e *y*.

API matemática definida

Para o cálculo da multilateração através do IP, foi definida a seguinte API, declarada no arquivo *coord_estimate.h*:

```
/*
 * Estima as coordenadas do alvo
 * @param d1 Distância estimada do alvo até o beacon 1
 * @param d2 Distância estimada do alvo até o beacon 2
 * @param d3 Distância estimada do alvo até o beacon 3
 * @param d4 Distância estimada do alvo até o beacon 4
 */
void estimateCoordinates(float d1, float d2, float d3,
  float d4);
```

API definida

Uma vez que as coordenadas dos *beacons* seriam fixas, não há necessidade de passá-las na chamada da função.

Projeto do IP

Após determinar o algoritmo e especificar a API deu-se início ao projeto do IP, como um todo, que envolveu os seguintes passos:

1) Solução de alguns problemas iniciais encontrados com o estudo de viabilidade efetuado.
2) Desenvolvimento de uma expressão única para cálculo do x e y resultantes;
3) Simplificação da expressão;
4) Desenvolvimento do modelo de referência do IP em software (*golden model)*;
5) Projeto do IP;
6) Desenvolvimento do Wrapper para plugar IP no barramento CoreConnect;
7) Desenvolvimento de um driver no EPOS para acessar o IP.

Problemas iniciais

Largura de palavra inteira muito grande

A idéia inicial era usar valores inteiros, já que quando formados por um número não tão grande de bits, costumam ocupar menos espaço do que *floats*. Porém isso só seria possível diminuindo a escala para obter uma precisão maior, já que sem isso a divisão inteira resultaria em uma perda muito grande, devido ao truncamento.

Observou-se, no entanto, que a largura de palavra necessária quando da diminuição da escala seria muito grande, em torno de 64 bits. Isso inviabilizou esta proposta, já que um multiplicador inteiro de ponto-flutuante de duas entradas variáveis gerado com o *Core Generator*, da *Xilinx*, ocupava, sozinho, 4250 LUTs, das aproximadamente 10000 disponíveis.

A solução foi o uso de cores de ponto flutuante (de 32 bits), que gerados também pelo *Core Generator*, ocupavam uma área menor: em torno de 300 LUTs e alguns DSP48 (dos 32 disponíveis).

A área limitada disponível na plataforma

Apesar de a FPGA possuir uma área considerável, a plataforma, com todas suas funcionalidades, ocupava muito do espaço disponível. Essa limitação forçou o abandono da idéia de ter como parâmetros de entrada do IP as quatro tuplas (x_i, y_i).

A solução abordada foi calcular previamente, via software, os *a's* e *b's* das matrizes A e B. Estes valores são gerados através dos (x_i, y_i), de modo que o cálculo prévio dos mesmos eliminou a necessidade do IP de conhecer estes (x_i, y_i), sendo a sua entrada formada apenas pelos *a's* e *b's*.

Desenvolvimento de expressão única e simplificação da mesma

Para implementar a multilateração no IP, toda a fórmula apresentada anteriormente foi desenrolada numa grande expressão única para *x* e *y*.

Analisando esta mesma expressão, chegamos a conclusão de que ela poderia ser simplificada, de modo a diminuir o número de cálculos matemáticos a serem efetuados pelo IP.

Esta simplificação deu-se pela identificação e substituição de três expressões comuns encontradas ao longo de todos os cálculos:

```
exp1 = (a11*a11) + (a21*a21) + (a31*a31);
exp2 = (a12*a12) + (a22*a22) + (a32*a32);
exp3 = -((a11*a12) + (a21*a22) + (a31*a32));
```

Os três exp's

Logo, todo o cálculo foi simplificado e, posteriormente fatorado, como segue:

```
num_x = ((exp2*a11) + (exp3*a12))*b11
      + ((exp2*a21) + (exp3*a22))*b21
      + ((exp2*a31) + (exp3*a32))*b31;

num_y = ((exp3*a11) + (exp1*a12))*b11
      + ((exp3*a21) + (exp1*a22))*b21
      + ((exp3*a31) + (exp1*a32))*b31;

div_factor_calc = (exp1 * exp2) - (exp3 * exp3);

x = num_x / div_factor_calc;
y = num_y / div_factor_calc;
```

A simplificação da expressão para cálculo da muliletaração

Modelo de referência

Um modelo de referência, também chamado de *golden model*, foi desenvolvido numa linguagem de alto nível.

Este modelo foi modularizado pensando na hierarquia do hardware do IP, ou seja, foi dividido em partes independentes, pensadas nos futuros componentes do IP. Essa modularização ocorreu em conjunto com a simplificação da expressão mencionada acima.

O modelo de referência – tanto no todo, como em suas partes – serviu para a validação do IP e seus componentes. A mesma entrada era aplicada tanto no modelo como num determinado módulo do IP, e após o processamento, seus valores eram comparados. Caso fossem iguais para um conjunto de valores de entrada com uma boa cobertura do domínio do problema, isso garantiria a funcionalidade do IP.

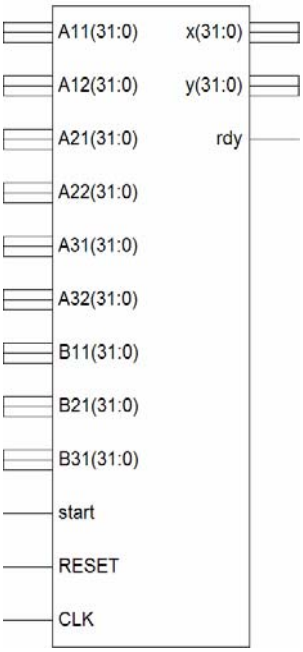
Particionamento

A arquitetura do IP foi divida em módulos, simplificando o entendimento do problema e a máquina de estados do IP. O IP foi dividido nos seguintes módulos:

- multilateration;
- exps_calc;
- div_factor_calc;
- numerators_calc;
- xy_calc.

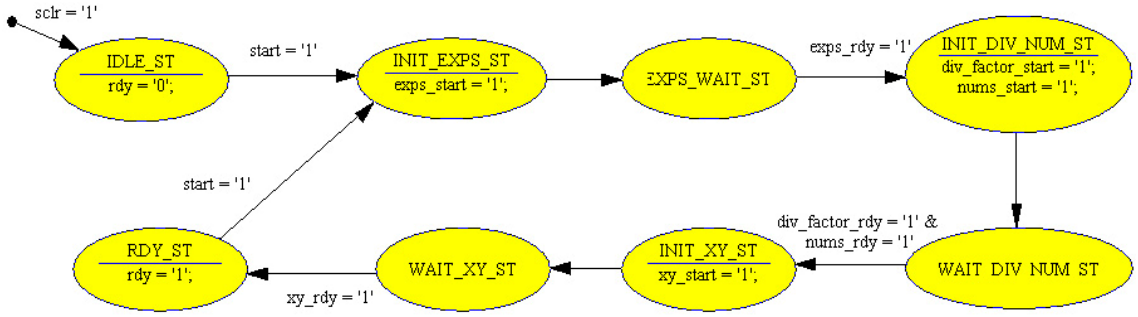
Componente multilateration

O módulo *top level* do IP, é a multilateração em si, excetuando-se o cálculo dos *a's* e *b's (os valores das matrizes A e B precisam ser pré-processados)*. Sua interface é definida da seguinte maneira:

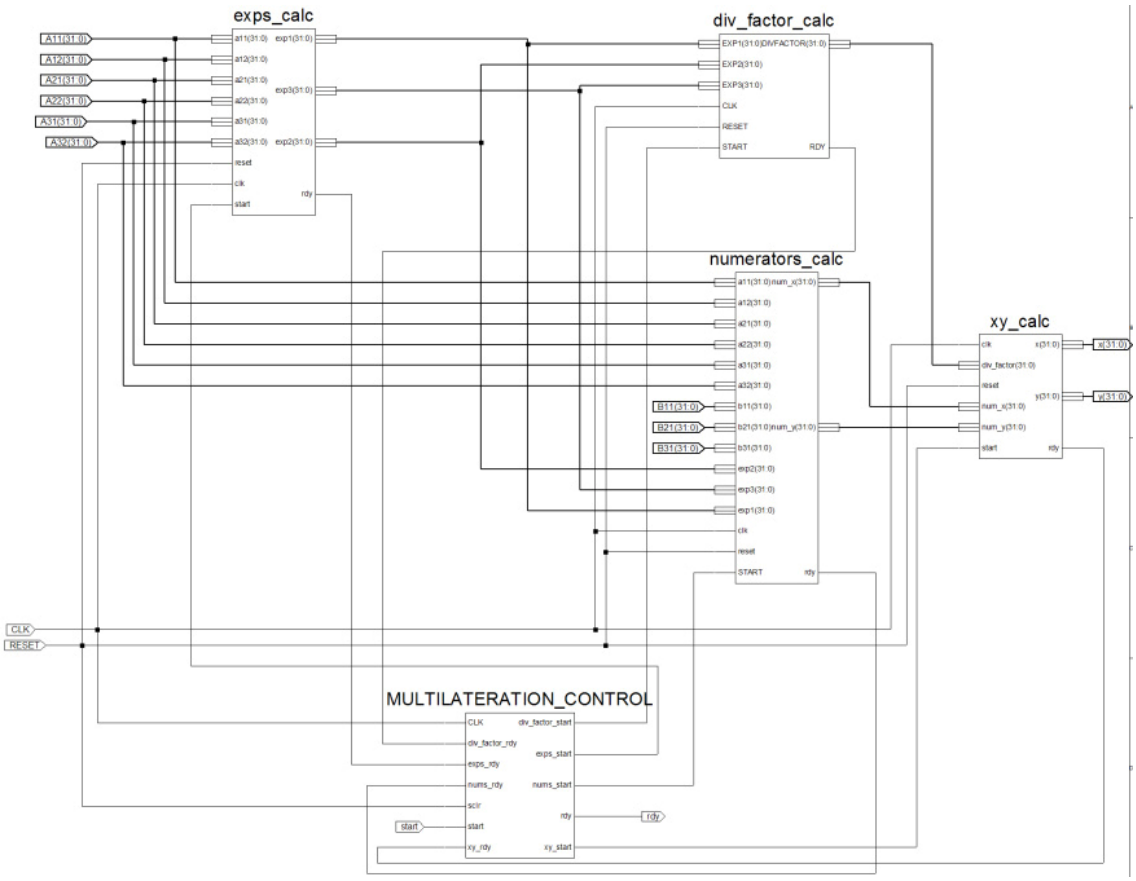


multilateration - Interface

Este módulo é formado pela ligação portas de saída e entrada dos submódulos criados, e por uma máquina de estados simples para controlar tudo.



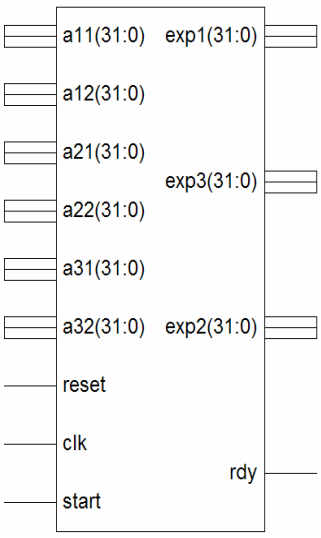
multilateration – Máquina de estados



multilateration – Esquemático

Componente exps_calc

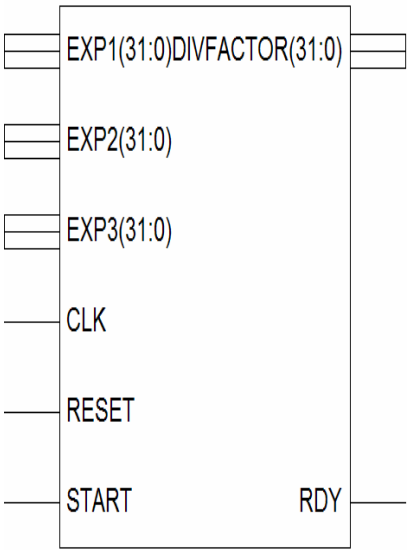
Este módulo calcula os três exp's, usados tanto no cálculo dos numeradores como no do denominador das duas divisões finais.



exps_calc - Interface

Componente div_factor_calc

Este módulo calcula o fator de divisão (denominador) das divisões finais.



div_factor_calc - Interface

O cálculo efetuado pelo componente é o seguinte:

```
div_factor_calc = (exp1 * exp2) - (exp3 * exp3);
```

div_factor_calc - Cálculo efetuado

Componente numerators_calc

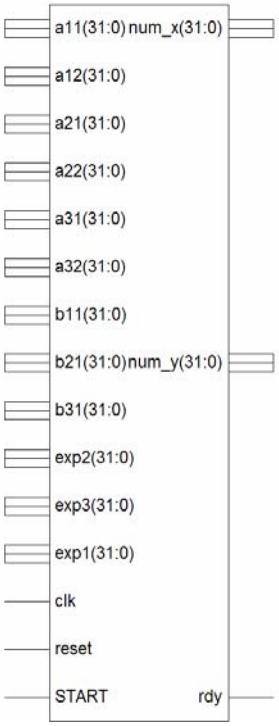
Este módulo calcula os numeradores (para o cálculo de x e y) das divisões finais.

O cálculo efetuado pelo componente é o seguinte:

```
num_x = ((exp2*a11) + (exp3*a12))*b11
      + ((exp2*a21) + (exp3*a22))*b21
      + ((exp2*a31) + (exp3*a32))*b31;

num_y = ((exp3*a11) + (exp1*a12))*b11
      + ((exp3*a21) + (exp1*a22))*b21
      + ((exp3*a31) + (exp1*a32))*b31;
```

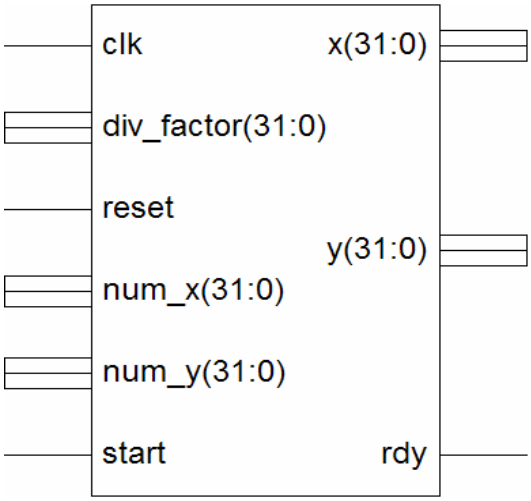
numerators_calc - Cálculo efetuado



numerators_calc - Interface

Componente xy_calc

Este módulo calcula o resultado da multilateração – *x* e *y*.



xy_calc - Interface

O cálculo efetuado pelo componente é o seguinte:

```
x = num_x / div_factor_calc;
y = num_y / div_factor_calc;
```

xy_calc - Cálculo efetuado

Verificação do IP

A verificação do IP (e seus submódulos) foi feita através de testbenchs, statebenchs e comparação dos resultados com o *golden model*.

Os testbenchs foram feitos através de wave forms estimulando os componentes com alguns valores de entrada.

Os statebenchs foram realizados através da ferramenta *StateCAD*, fornecida juntamente com o *ISE WebPack*. Esta ferramenta foi usada para modelar as máquinas de estados, gerando automaticamente o VHDL correspondente a máquina desejada. Esse modelo criado pode ser usado para simulação e verificação da máquina.

Integração

Após desenvolvimento do IP, entrou-se na etapa de integração com a plataforma desenvolvida pelo grupo Beta.

On-Chip Peripheral Bus

- Totalmente síncrono
- Address bus até 64 bits
- Data bus: 32 ou 64 bits
- Ciclo único de transferência entre master OPB e slave OPB
- Mapeamento de registradores em memória

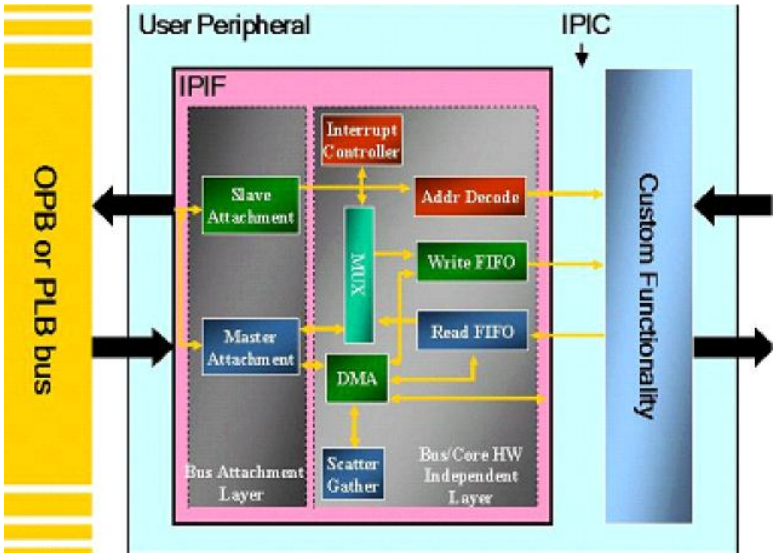
Integração da Plataforma

O IP deveria ser plugado ao barramento OPB (On-Chip peripheral bus). O desenvolvimento do Wrapper para conectar o IP ao barramento foi facilitado graças ao utilitário '*CIP Wizard' do EDK.*

Esse utilitário gera um bloco IPIF (*intellectual property interface)* bastante customizável. Esse bloco implementa funcionalidades comuns a vários periféricos.

O IPIF lida com os sinais mais complicados do barramento OPB e fornece uma interface mais simplificada de sinais para o IP que se deseja plugar. Essa interface simplificada é chamada de IPIC (IP Interconnect).

A figura abaixo ilustra o funcionamento do IPIF:



Depois do desenvolvimento do wrapper, o IP foi inserido no projeto do grupo Beta (no *EDK*) e começaram as tentativas de síntese da plataforma.

Problemas com a Integração

Infelizmente, apesar de termos um IP validado funcionalmente, ocorreram problemas na geração do *bistream* da plataforma e, portanto, nosso IP não será utilizado pelo grupo Beta.

```
ERROR:NgdBuild:604 - logical block
   'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/exps_calc_1'
with type 'exps_calc' could not be resolved. A pin name misspelling can cause this,
a missing edif or ngc file, or the misspelling of a type name.
   Symbol 'exps_calc' is not supported in target 'virtex4'.
ERROR:NgdBuild:604 - logical block

'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/divfaccalc_1/sub
1' with type 'sub_fp' could not be resolved. A pin name misspelling can cause this,
a missing edif or ngc file, or the misspelling of a type name.
   Symbol 'sub_fp' is not supported in target 'virtex4'.
ERROR:NgdBuild:604 - logical block

'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/divfaccalc_1/mul
t_1' with type 'mult_fp' could not be resolved. A pin name misspelling can cause
this, a missing edif or ngc file, or the misspelling of a type name. Symbol
'mult_fp' is not supported in target 'virtex4'.
ERROR:NgdBuild:604 - logical block

'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/xy_calc_1/div_fp
_1' with type 'div_fp' could not be resolved. A pin name misspelling can cause
this, a missing edif or ngc file, or the misspelling of a type name. Symbol
'div_fp' is not supported in target 'virtex4'.
ERROR:NgdBuild:604 - logical block

'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/num_calc_1/mult1
' with type 'mult_fp' could not be resolved. A pin name misspelling can cause this,
a missing edif or ngc file, or the misspelling of a type name. Symbol 'mult_fp' is
not supported in target 'virtex4'.
ERROR:NgdBuild:604 - logical block

'multilateration_ip_0/multilateration_ip_0/USER_LOGIC_I/multilatIP/num_calc_1/add1'
with type 'add_fp' could not be resolved. A pin name misspelling can cause this, a
missing edif or ngc file, or the misspelling of a type name.
   Symbol 'add_fp' is not supported in target 'virtex4'.
```

Mensagem do erro que ocorre durante a geração do bitstream

Aparentemente o EDK não consegue localizar os módulos gerados pelo *Core Generator* utilizados no IP de multilateração (add_fp, sub_fp, div_fp, mult_fp).