Register Transfer Level and High Level Synthesis Implementation of Correlation measure focused on small number of features

Jardel Silveira, Pedro Lima, Jarbas Silveira e César Marcon

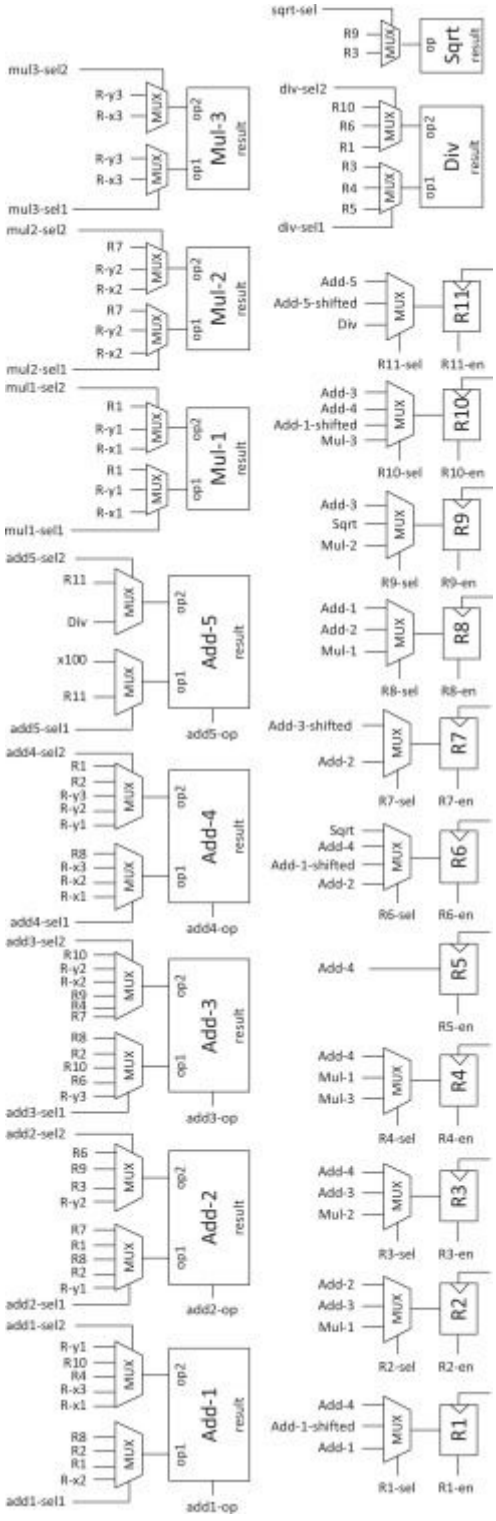
Abstract

Keywords: DDoS attack detection; FPGA; Correlation measure

Introduction

Review

A real-time DDoS attack detection method should identify attacks with low computational overhead. Although a large number of statistical methods have been designed for DDoS attack detection, real-time statistical solution to detect DDoS attacks in hardware is only a few. In this paper, a real-time DDoS detection method is proposed that uses a novel correlation measure to identify DDoS attacks. Effectiveness of the method is evaluated with three network datasets, viz., CAIDA DDoS 2007, MIT DARPA, and TUIDS. Further, the proposed method is implemented on an FPGA to analyze its performance. The method yields high detection accuracy and the FPGA implementation requires less than one microsecond to identify an attack.



Methodology

Materials HLX

In order to evaluate the effectiveness of the proposed DDoS detection method, an experiment was carried out on a workstation with 12GB main memory, 2.26 GHz Intel Xeon processor and 64-bit Windows 7 operating system. The proposed DDoS detection

framework was initially implemented and evaluated in software. Additionally, to evaluate the throughput of the proposed method, a prototype of the hardware based attack detection module was implemented on a Xilinx Virtex-5 FPGA device (XC5VLX50T).

Materials HLS

Vivado 2017.1
xc7a100tcsg324-1

HLS Implementation

```
#include "fxp_sqrt_top.h"
#include <ap_fixed.h>
#include <stdlib.h>
#include <cmath>

#define NAHID_BW   20
#define NAHID_IW    12

typedef ap_ufixed<NAHID_BW,NAHID_IW>   nahid_data_t;

nahid_data_t fxpnahid(unsigned short x1,unsigned short x2,unsigned short x3,
                      unsigned short y1, unsigned short y2, unsigned short y3);

nahid_data_t fxp_nahid_top(unsigned short x1,unsigned short x2,unsigned short x3,
                           unsigned short y1, unsigned short y2, unsigned short y3)
{
    nahid_data_t result;

   result=fxpnahid(x1,x2,x3,y1, y2, y3);
   return result;
}

nahid_data_t fxpnahid(unsigned short x1,unsigned short x2,unsigned short x3,
                      unsigned short y1, unsigned short y2, unsigned short y3)
{

    unsigned short ax1,ay1,Mx,My,SDx,SDy,MSDx,MSDy,DX1,DX2,DX3,DY1,DY2,DY3,D1,D2,D3,N1,N2,N3;

    nahid_data_t Q1,Q2,Q3,aQ1,aQ2,NaHiDverc;
    unsigned long mx1,mx2,mx3,my1,my2,my3,Mx2,My2,amx1,amy1,M2x,M2y,Vx,Vy;


    ap_ufixed<22,22> VxFxp;
    ap_ufixed<12,12> raizVx;

    ap_ufixed<22,22> VyFxp;
    ap_ufixed<12,12> raizVy;

    ax1=x1+x2;
    ay1=y1+y2;
    Mx=(ax1+x3)/4;
    My=(ay1+y3)/4;
    mx1=(unsigned long)x1*(unsigned long) x1;
    mx2=(unsigned long)x2*(unsigned long)x2;
    mx3=(unsigned long)x3*(unsigned long)x3;
    my1=(unsigned long)y1*(unsigned long)y1;
    my2=(unsigned long)y2*(unsigned long)y2;
    my3=(unsigned long)y3*(unsigned long)y3;
    Mx2=(unsigned long)Mx*(unsigned long)Mx;
    My2=(unsigned long)My*(unsigned long)My;
    amx1=mx1+mx2;
    amy1=my1+my2;
    M2x=(amx1+mx3)/4;
    M2y=(amy1+my3)/4;
    Vx=abs((long)Mx2-(long)M2x);
    Vy=abs((long)My2-(long)M2y);

    VxFxp = ap_ufixed<22,22>(Vx);
    fxp_sqrt(raizVx, VxFxp);
    SDx = raizVx.to_uint();

    VyFxp = ap_ufixed<22,22>(Vy);
    fxp_sqrt(raizVy, VyFxp);
    SDy = raizVy.to_uint();

    SDy=(short) raizVy;
    MSDx=abs((short)Mx-(short)SDx);
    MSDy=abs((short)My-(short)SDy);
    DX1=abs((short)MSDx-(short)x1);
    DX2=abs((short)MSDx-(short)x2);
    DX3=abs((short)MSDx-(short)x3);
    DY1=abs((short)MSDy-(short)y1);
    DY2=abs((short)MSDy-(short)y2);
    DY3=abs((short)MSDy-(short)y3);
    D1=DX1+DY1;
    D2=DX2+DY2;
    D3=DX3+DY3;
    N1=abs((short)x1-(short)y1);
    N2=abs((short)x2-(short)y2);
    N3=abs((short)x3-(short)y3);


    Q1=((nahid_data_t)N1)/((nahid_data_t)D1);
    Q2=((nahid_data_t)N2)/((nahid_data_t)D2);
    Q3=((nahid_data_t)N3)/((nahid_data_t)D3);
    aQ1=Q1+Q2;
    aQ2=(aQ1+Q3)/((nahid_data_t)4);

        NaHiDverc=((nahid_data_t)1.0)-aQ2;

    return NaHiDverc;
}
```

Results

Gráfico de Utilização:

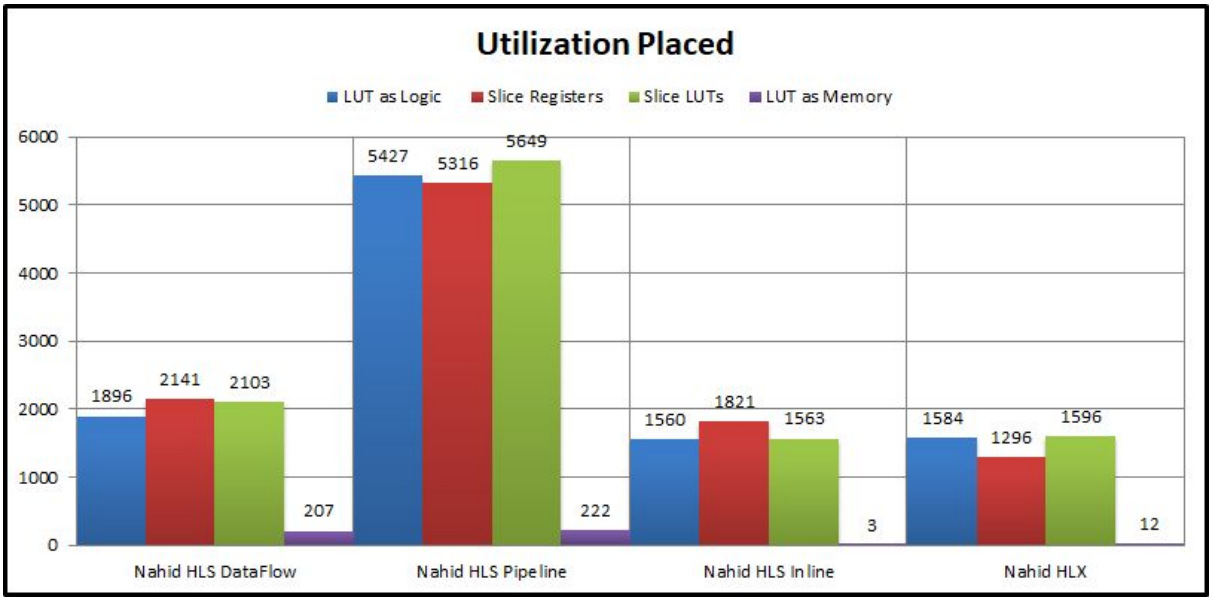


Gráfico de ?

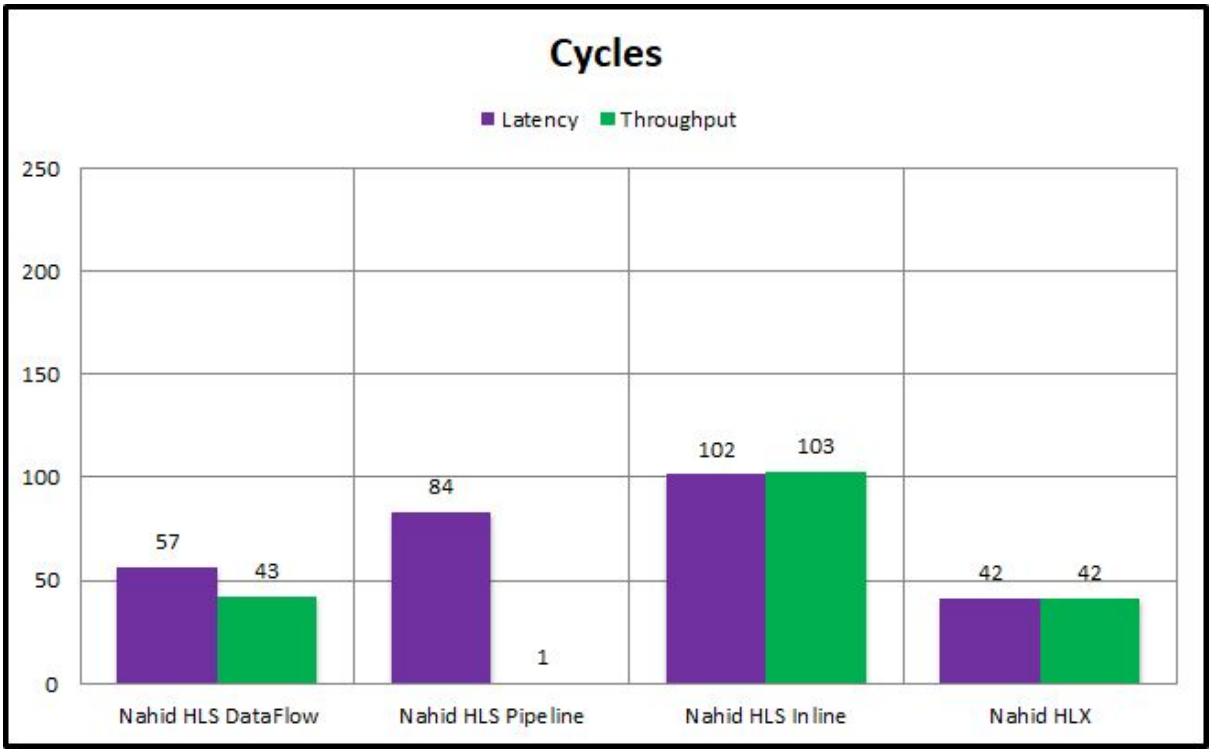


Gráfico da frequência máxima :

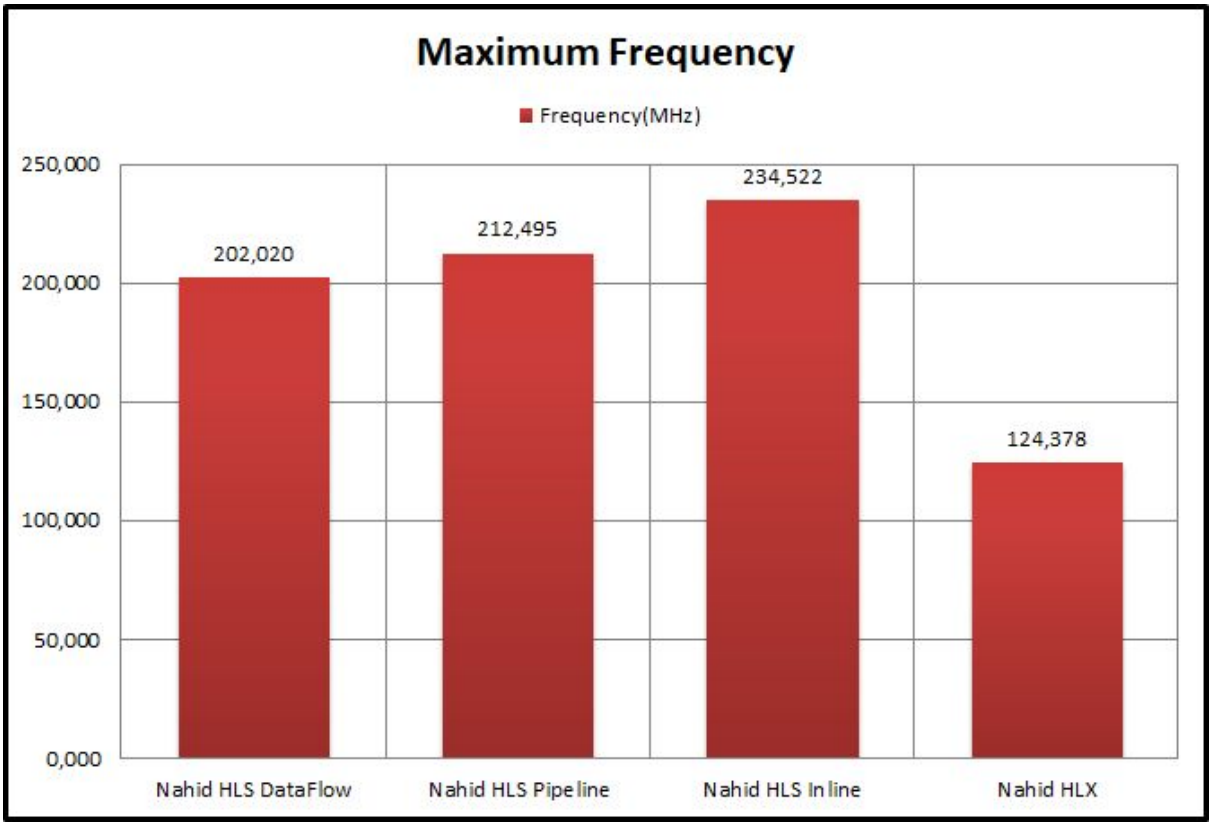


Gráfico de ?:

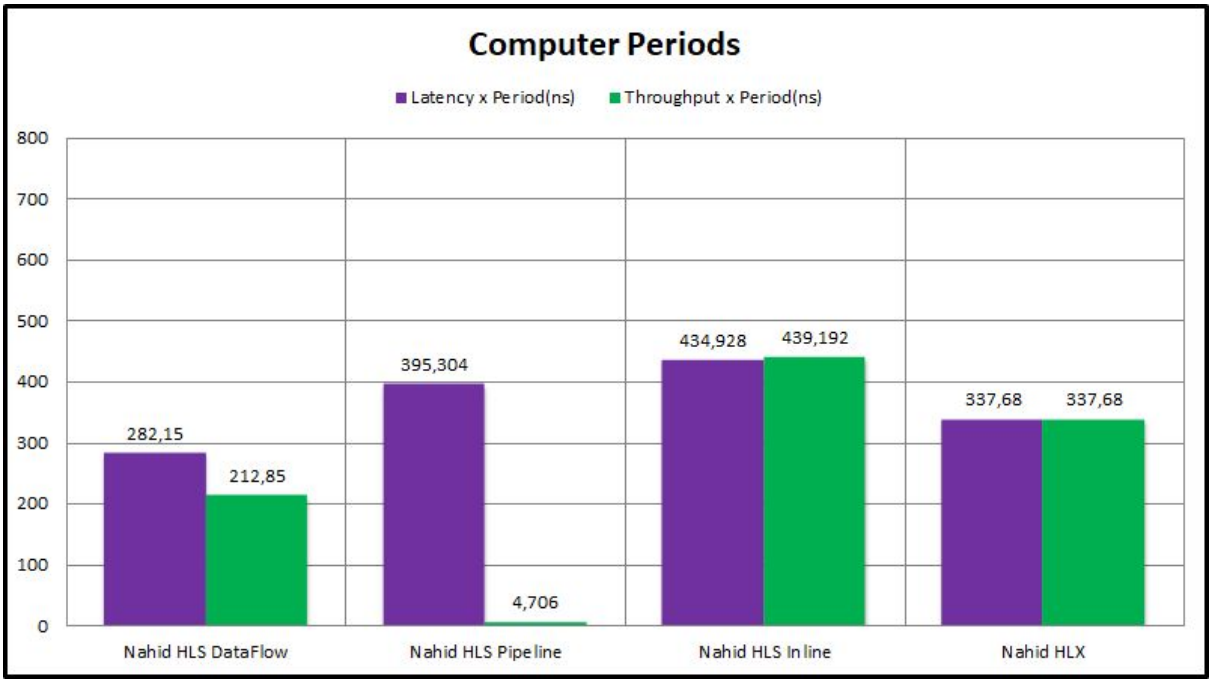


Gráfico de potência:

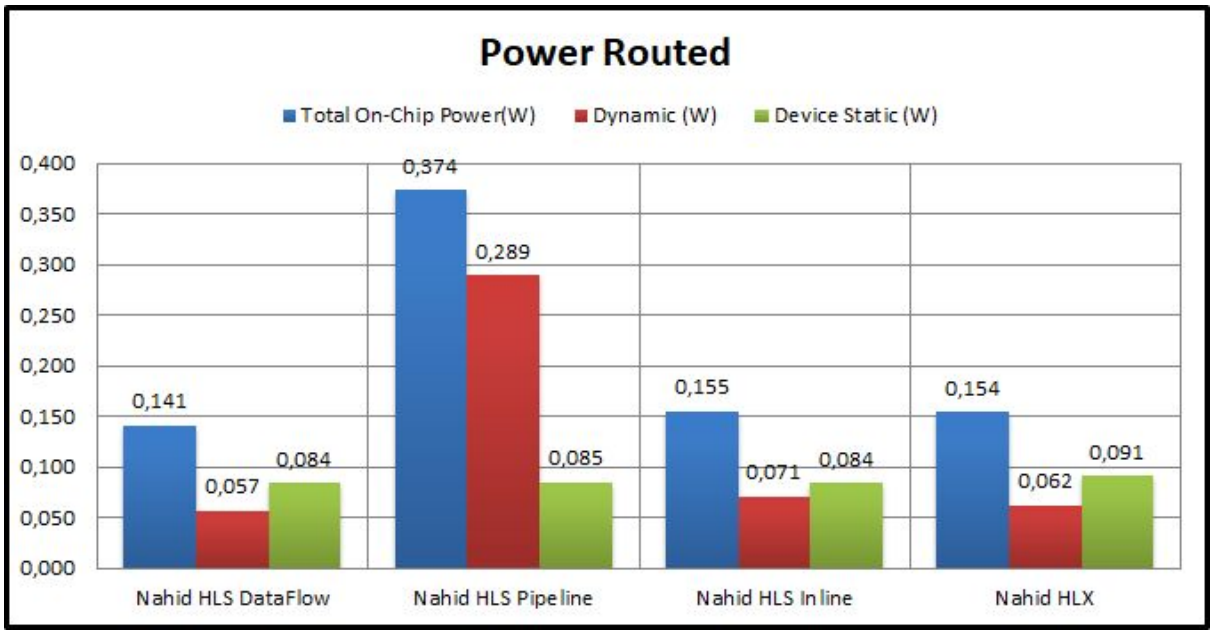
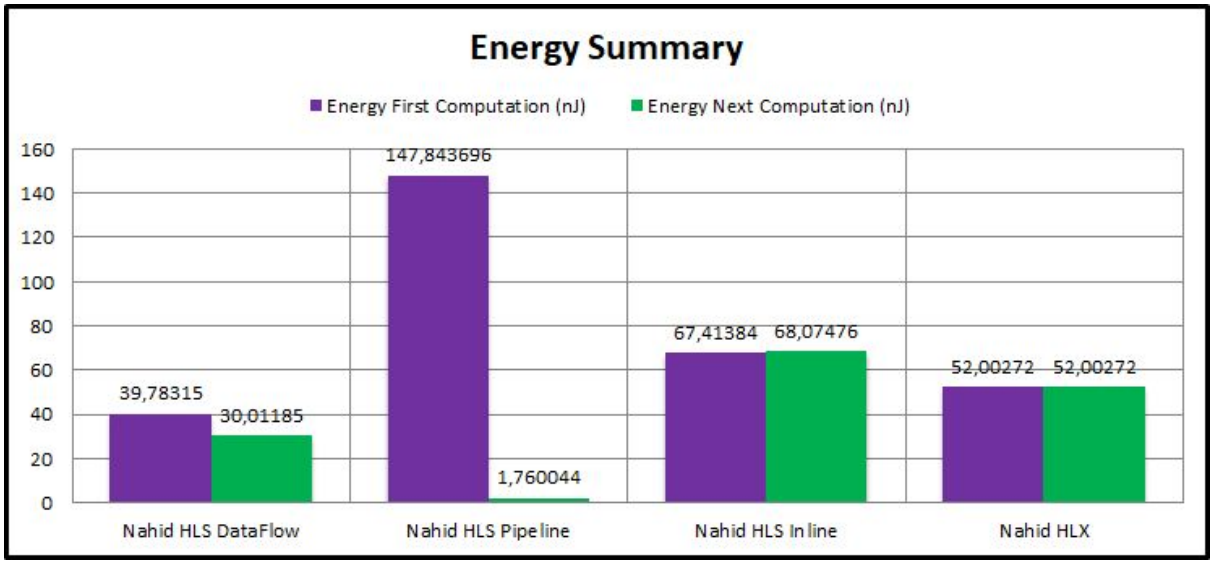


Gráfico de energia:



Conclusion

References

@article{HOQUE201748,
title = "Real-time DDoS attack detection using FPGA",
journal = "Computer Communications",
volume = "110",
pages = "48 - 58",
year = "2017",
issn = "0140-3664",
doi = "https://doi.org/10.1016/j.comcom.2017.05.015",
url = "http://www.sciencedirect.com/science/article/pii/S0140366416306442",
author = "N. Hoque and H. Kashyap and D.K. Bhattacharyya",
keywords = "DDoS attack detection, FPGA, Correlation measure"
}

@ARTICLE{7368920,
author={R. Nane and V. M. Sima and C. Pilato and J. Choi and B. Fort and A. Canis and Y. T. Chen and H. Hsiao and S. Brown and F. Ferrandi and J. Anderson and K. Bertels},
journal={IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems},
title={A Survey and Evaluation of FPGA High-Level Synthesis Tools},
year={2016},
volume={35},
number={10},
pages={1591-1604},
doi={10.1109/TCAD.2015.2513673},
ISSN={0278-0070},
month={Oct}}

@inproceedings{zhao2017accelerating,
  title={Accelerating binarized convolutional neural networks with software-programmable fpgas},
  author={Zhao, Ritchie and Song, Weinan and Zhang, Wentao and Xing, Tianwei and Lin, Jeng-Hau and Srivastava, Mani and Gupta, Rajesh and Zhang, Zhiru},
  booktitle={Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays},
  pages={15--24},
  year={2017},
  organization={ACM}
}

@ARTICLE{7859319,

author={F. B. Muslim and L. Ma and M. Roozmeh and L. Lavagno},
journal={IEEE Access},
title={Efficient FPGA Implementation of OpenCL High-Performance Computing Applications via High-Level Synthesis},
year={2017},
volume={5},
number={},
pages={2747-2762},
keywords={field programmable gate arrays;high level synthesis;parallel processing;FPGA-based accelerators;FPGA-specific OpenCL programming;HLS-based FPGA implementation;OpenCL high-performance computing;high-level languages;high-level synthesis;Algorithm design and analysis;Computer architecture;Field programmable gate arrays;Graphics processing units;Hardware;Kernel;Random access memory;FPGA;GPU;High-level synthesis (HLS);OpenCL;low-power low-energy computations;parallel computing},
doi={10.1109/ACCESS.2017.2671881},
ISSN={},
month={},}

Mackay D. (2017) C-Based Design. In: Churiwala S. (eds) Designing with Xilinx® FPGAs. Springer, Cham, Print ISBN 978-3-319-42437-8

@article{KOUTRAS201791,
title = "On supporting rapid prototyping of embedded systems with reconfigurable architectures",
journal = "Integration, the VLSI Journal",
volume = "58",
pages = "91 - 100",
year = "2017",
issn = "0167-9260",
doi = "https://doi.org/10.1016/j.vlsi.2017.02.007",
url = "http://www.sciencedirect.com/science/article/pii/S0167926017301001",
author = "I. Koutras and K. Maragos and D. Diamantopoulos and K. Siozios and D. Soudris",
keywords = "Reconfigurable architectures, Rapid prototyping, Genetic algorithm"
}

Figuras

(1)

$$NaHiD(X,Y) = 1 - \frac{1}{n}\sum_{i=1}^{n}\frac{|X(i)-Y(i)|}{\|meanX-SDX|-X(i)|+\|meanY-SDY|-Y(i)|}$$

where, *n* is the dimension of the two objects *X* or *Y*.

For simplicity, Eq. (1) can be written as:

$$NaHiD\,(X,Y) = 1 - \frac{1}{n}\sum_{i=1}^{n} D\,(i)$$

where,

(2)

$$D\,(i) = \frac{|X(i)-Y(i)|}{\|meanX-SDX|-X(i)|+\|meanY-SDY|-Y(i)|}$$

kd o codigo da raiz quadrada
?