ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 4, Issue 8, August 2015

VLSI MODELING OF FPU ARITHMETIC UNIT WITH FMA TECHNIQUE

[1]N·Jyothi
M.tech Student (VLSID), Department of ECE,Shree Institute of technical Education, Tirupathi.
[2]S.Sruthi
Assistant Professor, Department of ECE, Shree Institute of technical Education, Tirupathi.

***Abstract*— Scope displaying is one of the key errands of the confirmation stream in frameworks improvement. The subsequent model is regularly used to assess the advancement and nature of the check process; it likewise gives a valuable deliberation to the era of test vector designs. This work shows a heuristic methodology for scope model definition in view of the ideas of equality classes and limit esteem investigation to address the check of skimming point number juggling units. As a contextual investigation, a scope model was intended to confirm the ADD operation of a drifting point module for the binary16 number configuration characterized in the IEEE 754-2008 standard, and a System Verilog test seat was actualized to perform the check process. The adequacy of the heuristic and the nature of the subsequent model are broke down by measuring the scope acquired in the execution of an outsider test suite, and by creating an arrangement of test vectors from the model and invigorating an outline under confirmation (DUV) to recognize bugs for configuration survey.**

***Index Terms*—*Floating Point*, IEEE Standard, Floating Point Arithmetical Points, Functional Coverage, FPU, FMA Technique.**

I. INTRODUCTION

The check of drifting point units (FPU) it is a no doubt understood testing assignment. Many-sided quality emerges from the vast test space which incorporates a few corner cases hard to distinguish and that must be focused on. It has been tended to falling back on formal systems and reproductions [1] [2]. Formal routines are utilized to demonstrate execution accuracy in view of numerical systems, for example, model checking, identicalness checking and hypothesis demonstrating. In [3] a hypothesis demonstrates is utilized to affirm formal conditions for SRT division plan, both (outline and conditions) are demonstrated formally as an arrangement of arithmetical relations. The exponential algorithmic many-sided quality of these systems coming about because of the state space blast issue, confines its application to little pieces of the configuration or to a particular usage. Then again, recreation techniques can be connected in any case the many-sided quality of the configuration and its comparing confirmation space. In any case, they don't give a proof on the execution accuracy; despite the fact that they are viable to demonstrate the vicinity of bugs, it is unrealistic to guarantee their nonattendance unless the combinatorial test space, regularly not plainly characterized, is totally secured. Invigorating an outline under check (DUV) with each conceivable info vector is unfeasible in light of the fact that it includes an exponential multifaceted nature, similar to formal routines do. Along these lines, check engineers must select an agent test of the test space, check the right conduct by recreations and have certainty that this best exertion test is sufficiently capable to uncover any slip in the outline. Indeed, even with this inadequacy, recreations are still the primary workhorse in check building. Another significant contrast between formal check and reenactments is identified with the perceivability of the configuration. While formal confirmation requires to be mindful of the inside points of interest of the execution, known as white-box testing in the product building field, recreations can be utilized notwithstanding when the usage code is not accessible, checking the usefulness by just dissecting the accuracy of the yield, otherwise called discovery testing.

II EXISTING SYSTEM

A. Functional Coverage

Utilitarian confirmation includes the procedure of ensuring a right mapping between the determinations of a configuration and its usage. Disparities between the normal and watched conduct may come about because of mistakes presented in the outline or usage stages. While plan blunders are more often than not because of inadequate or vague particulars, wrong understanding of details, or an erroneous execution of the configuration errand, usage lapses are frequently identified with coding mistakes. At whatever point utilitarian confirmation in light of reenactments is received, the procedure obliges a metric to assess the nature of a test suite. This quality measure called scope, serves to assess the confirmation procedure development, showing whether the kind and measure of chose test vectors are sufficient to achieve a certain quality worth. There are two basic sorts of scope. The principal is called auxiliary scope and shows which divides of the outline have been practiced or not by the test seat. Normal basic measurements are line-scope, branch-scope and

switch scope. Line-scope distinguishes the lines of the source code that have been executed in a reenactment, branch-scope demonstrates which ways have been practiced and switch scope checks whether variables or signs have taken certain qualities.

III PROPOSED SYSTEM

A. Coverage Modeling

The methods utilized as a part of equipment confirmation building are like the ones utilized by the discovery procedure in programming testing [6]. These are comparability parceling and limit esteem examination. Equality parceling alludes to part the data information space in subsets or classes where all the inputs that empower the same area of code or rationale fit in with the same subset. This parceling is proposed to recognize proportional vectors in a check point of view so as to choose one example of every class, test the usage and state that all proportionate rationale has been confirmed, along these lines minimizing the quantity of experiments. Limit esteem examination is utilized to recognize corner cases identified with data and yield values or to shape estimations of the comparability parts, in light of the fact that bugs are prone to be in the corners. In test vector era, the fundamental reason for the scope model is to distinguish intriguing conditions to be tried (keeping in mind the end goal to discover blunder) and to lessen the quantity of tests without decreasing its scope. Comparability dividing and limit esteem examination points the development of the model with those qualities.

Proportionality dividing system includes:
1) Identify the space of conceivable inputs.
2) Split the space in segments with the basis characterized by the details and usefulness of the DUV.
3) Identify substantial and invalid classes.
4) Select delegate vectors of each legitimate subset.
5) Simulate and check the outcomes.

As an initial illustration, the scope model for a basic configuration is depicted. The DUV executes an immersion operation, if the 16-bit number info is lower than 564(dec) then the yield is equivalent to the data, generally the yield is equivalent to 564(dec). Let Cs be the scope space characterized by the 16-bit conceivable numbers, then Cs = {0 . . . 65535}. The DUV should exercise an area of rationale when the data is lower than 564(dec) and another when it is higher or measure up to than 564(dec). Taking after this suspicion, the scope space is isolated in two proportionality classes, $C1$ = {0 . . . 563} and $C2$ = {564 . . . 65535}, and the scope model Cm is characterized as a two component set, Cm = {$C1$, $C2$}. One arrangement of conceivable examples that totally covers the model is S = {24(dec), 1524(dec) }, the first speaks to the 16- bit numbers under 564(dec) class and the second the higher or equivalent 564(dec). For this situation, comparability parceling decreases the quantity of 65535 conceivable test vectors to 2 having a full scope of the model. Despite the fact that equality parceling is exceptionally proficient in lessening the quantity of test vectors, it introduces a few inadequacies, for instance in number juggling DUV upper limits can bring about an undesired flood.

IV CASE STUDY

In this segment, we exhibit the heuristic depicted in area III connected to the ADD operation of a coasting point unit that actualizes the binary16 design. The skimming point unit has two inputs A and B and an outcome F, inputs and yields are in binary16 position. In this work just the include operation is confirmed yet the same examination should be possible to the subtract and increase operations of any FPU. To begin with comparability dividing, a care full and cognizant investigation of the data and yield sort must be done so as to recognize the space of conceivable inputs.

A The binary16 format

Binary16 are 16-bit gliding point numbers otherwise called Half Precision skims in view of the examination with the ordinarily received single and twofold accuracy (32 and 64 bits) representations. This configuration is valuable for applications that need to handle littler extents with a lower cost on memory than the single and twofold exactness, however number juggling reckonings have not to be basic in light of the fact that its detriments are the loss of accuracy and reach. The binary16 configuration is characterized by the IEEE 754-2008 Standard for Floating Point operations [7]. The organization is as per the following.

- Sign bit: 1 bit
- Exponent width: 5 bits (encoded using an offset binary representation, with zero offset equals to 15)
- Significand precision: 11 bits (10 explicitly stored)

And the real number is obtained following the function f:
Note that zero has a twofold representation positive and negative. The situations when the concealed bit is 1 (00001 $\leq e \leq$ 11110) are called typical numbers and when breaks even with 0 are called subnormal numbers. Another critical trademark is the adjusting method of the DUV. For this situation, truncation is chosen. B. Comparability dividing and limit esteem investigation Clearly, the data scope space

are all the two 16-bit numbers mixes, that is an aggregate of 232 −1 ∼ 4.29×109 focuses in the space, this number is the cardinality of the set. The best possible elucidation of binary16 organization in ordinary, subnormal, zero, unendingness and NaN sets the paradigm to distinguish the inputs and yield classes, and part the scope space in disjoint subsets. Due to the proportionality apportioning strategy, as well as we can derive that these definitions makes the architect think the diverse treatment (and rationale ways) when including quantities of equivalent or distinctive subsets; for instance the rationale when including two typical, or when including a subnormal and an ordinary, or when including two subnormal's and the outcome is an ordinary.

Thus, an info and yield of an experiment can have a place with one of this subsets (classes): +/ - ordinary, +/ - subnormal, zero, +/ - vastness, and NaN. As the DUV has two inputs, the info space can be splitted by the cross-result of the past subsets. Be that as it may, likewise must be considering the usefulness of the DUV, i.e. the yield as an element of the inputs. The situation when including two subnormal and the outcome is a typical open the need to include the yield classes as it induces diverse conduct for equivalent data classes pair. Presently the yield classes are considered as the cross-result of the inputs. However, there are a few aftereffects of the item that can never be because of F equivalents An or more B, for instance, An and B has a place with typical numbers and F fit in with negative ordinary. The subsequent parts can be communicated in a table like the accompanying

TABLE I. EQUIVALENCE PARTITIONS

P.Number	A	B	F
1	Subnormal	Subnormal	Subnormal
2	Subnormal	Subnormal	Normal
3	Normal	Subnormal	Normal
4	Subnormal	Normal	Normal
5	Normal	Normal	Normal
6	Normal	Normal	Normal

It can also be shown in a two dimensional graphic. To visualize the partitions, lets first draw the lower and upper bounds for each class (Zero, +/-Subnormal, etc.) and for each variable (A, B and F) in the A,B plane. Fig. 1 shows these straight lines.

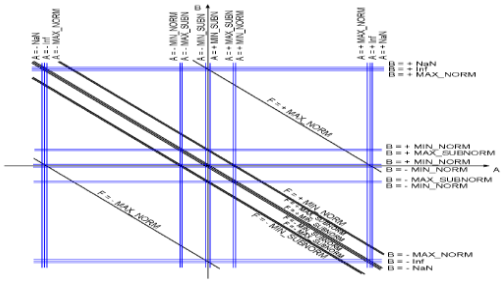


Fig. 1 Lower and upper bounds in the A,B plane.

Now, identify the valid regions, points and lines that represent a partition. For example, the region defined by the lines $A = MIN\ NORM$, $B = MIN\ NORM$ and $F = MAX\ NORM$, the point (A = 0,B = 0, F = 0), and the line B = 0 between $A \in Normals$ and $F \in Normals$ are all valids. And it should be noticed that the position of the line $F = MAX\ NORM$ is the infinite precision representation of the ADD operation, in our case this operation has a rounding mode that moves up this line since adding a number x in the vicinity of the maximum normal with a relatively small number s gives as the result x. Finally, lets colour those spaces, lines and points with blue and erase the limit lines: Fig. 2. Fig. 2 represents graphically the partitions obtained as the equivalences defined by the binary16 format and the DUV's add operation. The numbers inside the partitions follows the subsets numbers of the previous table.

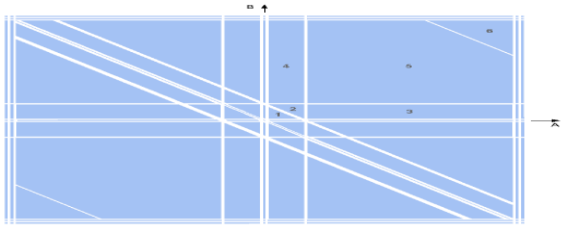


Fig. 2 Equivalence classes.

B. Equivalence class impact on coverage space

To be able to understand the impact on testing a sample of a partition, is valuable to know for a given test vector, the number of others test vectors represented by it (equivalents), and the percentage covered by the partition. Cardinality of the coverage model for the first quadrant is $|CMfirst\ quadrant|$ = 1, 073, 741, 824. Each $|Partitioni|$ is shown in Table II. Thus, a test vector ti covers the percentage defined by the division between the cardinality of the partition and the cardinality of the input space (in this case only the first quadrant is considered). $\%Coveredti = 100 \times |Partitioni|/|CMfirst\ quadrant|$ This percentage is the same for all its equivalents, and

therefore the percentage covered by the partition (third column of Table II).

Table II shows the cardinality (*|Partition|*) and the percentage that represents the first six subsets (excluding 0-subset). +

TABLE II. Cardinality of Quivalence Partitions

P.Number	Partition	Percentage Represented
1	522,753	0.048685%
2	523,776	0.048780%
3	31,426,560	2.926826%
4	31,426,560	3.926826%
5	939,537,408	87.501240%
6	4,180,992	0.389385%

V IMPLEMENTATION

Even though the common usage of a coverage model is as a metric for verification progress, it was used for test vector generation as, knowing the coverage model; we already know the possible values that fully cover it. For the sake of simplicity, only the generation of the test vector generation for the first quadrant of the coverage model was implemented, but the same logic can be applied to the rest.

A Detailed Subsets in First Quadrant

Accordingly, the first quadrant of the coverage model has 70 subsets. Equivalence partitioning defines 26 and boundary value analysis adds 44 subsets, as can be seen in Fig.

B Verification environment and test vector generation

The check environment includes a System Verilog test seat that incorporates the assets to create a test vector and understand its imperatives, and the reference model. The dialect System Verilog is the present reception for test seat improvements. As an item situated programming (OOP) dialect, it permits to fabricate very much organized and extensible check segments.

Figure 4 outlines the test seat construction modeling for the discovery check segment. The Top module is the primary element of the confirmation program. It instantiates the FLP center under test and the practical check assets, join them through an interface question and characterizes the clock signal. The depiction of fundamental segments of a confirmation domain (driver, screen and checker) can be found in [8].

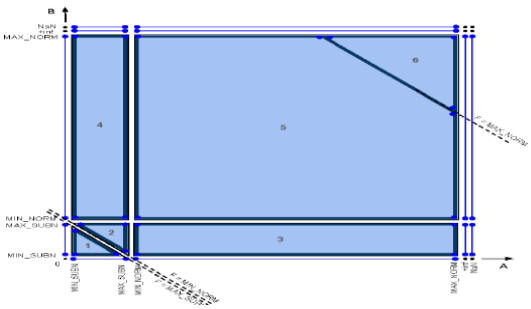
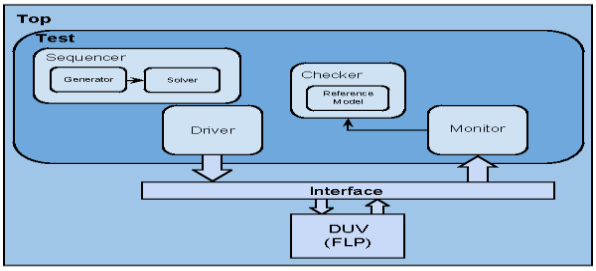
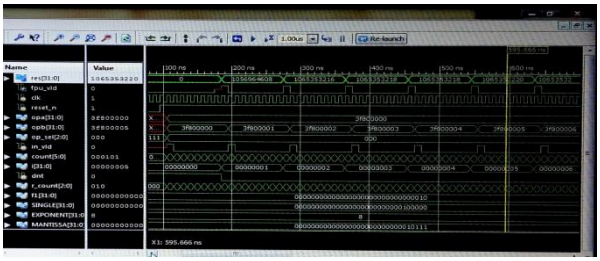


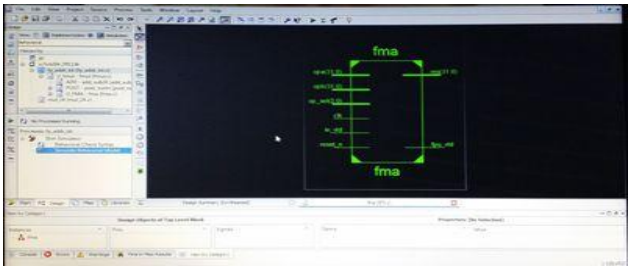
Fig. 3 Coverage model - first quadrant.

The reason is obvious, the binary16 data type, comparison ("higher or equal than" and "lower or equal than"), and the ADD operation are not supported for this floating point format. Our solution is to treat *A*, *B* and *F* as 16-bit integers. This allows to solve the random generation for one of the inputs (for example, *A*) and the result (*F*) for the first quadrant of the coverage model, having the restriction that the input must be lower than the result (*A* lower than *F*).



VI SIMULATION RESULTS





VII CONCLUSION

This paper displays a heuristic methodology for scope model definition for drifting point number-crunching units to perform useful confirmation. The heuristic proposed results to be sensible contrasted with a solid test suite created by FPGen. The model proposed is utilized as a test vector generator for a recreation based confirmation. In like manner, it effectively discovered specific bugs in a FPU binary16 IP (Intellectual Property). Thusly, the proposed confirmation execution helps to check the usefulness of a FPU binary16 IP. Furthermore, the heuristic can be utilized to the subtract and reproduce operations or the scope model can be stretched out to different arrangements characterized by the IEEE-754-2008.

REFERENCESS

[1] E. Guralnik, M. Aharoni, A. Birnbaum, and A. Koyfman, "Simulationbased verification of floating-point division," *IEEE Transactions on Computers*, vol. 60, no. 2, pp. 176–188, Feb 2011.
[2] O. Goni, E. Todorovich, and O. Cadenas, "Generic construction of monitors for floating point unit designs," *VIII Southern Conference on Programmable Logic (SPL), 2012*, pp. 1–8, March 2012.
[3] E. Clarke, S. German, and X. Zhao, "Verifying the srt division algorithm using theorem proving techniques," *Formal Methods in System Design*, vol. 14, no. 1, pp. 7–44, 1999.
[4] V. Jerinic, J. Langer, U. Heinkel, and D. Muller, "New methods and coverage metrics for functional verification," *Design, Automation and Test in Europe, 2006. DATE '06. Proceedings*, vol. 1, pp. 1–6, March 2006.