Implementation of Reconfigurable Constant Multiplication

B.RANJITH[1], RUPA KUMAR DANAVATH[2]
[1]*P.G Student, VLES*
[2]*Assistant Professor, Department of Electronics and Communication Engineering, Nagole Institute of technology and science*

Abstract- This work introduces a new heuristic to generate pipelined run-time reconfigurable constant multipliers for FPGAs. It produces results close to the optimum. It is based on an optimal algorithm which fuses already optimized pipelined constant multipliers generated by an existing heuristic called RPAG. Switching between different single or multiple constant outputs is realized by the insertion of multiplexers. The heuristic searches for a solution that results in minimal multiplexer overhead. Using the proposed heuristic reduces the run-time of the fusion process, which raises the usability and application domain of the proposed method of run-time reconfiguration.

Keywords- **FPGA, CONSTANT**

I. INTRODUCTION

The augmentation with consistent coefficients is a fundamental activity in advanced flag handling. At first one reason to put inserted multipliers or DSP obstructs into the texture of field-programmable door clusters was to decrease the execution hole between application particular coordinated circuits (ASICs) and FPGAs. By the by, the cost to pay for those settled coarse-grained squares is their resoluteness in word size and constrained amount. Restricted amount is especially basic in modern applications, when less expensive and rather little FPGAs with just few DSP squares must be picked due to value weight. Along these lines, rationale based consistent increase techniques are required. Advancing the usage of this task is all around considered. Exchanging between a given arrangement of constants of such multipliers amid run-time as opposed to utilizing bigger nonexclusive multipliers is vital to acknowledge equipment proficient run-time versatile channels [1], [2], [3], DCT and FFT usage [4] and additionally multi-arrange channels for obliteration or introduction like polyphase FIR channels [5]. A reconfigurable consistent multiplier is a duplication circuit in which the scaling steady can be looked over a constrained predefined set of constants amid run-time. For the given application spaces two to six of such flexible coefficient sets are normal. The exchanging amid run-time is accomplished by embeddings multiplexers into a few consistent augmentation circuits, to accomplish a reuse of excess halfway circuits and hence a decrease of required assets. The issue is to locate the most ideal arrangement while embeddings the combining multiplexers. Rather than ASICs, this is particularly prudent for FPGA plans [6], because of their inalienable execution burden. A case for such a pipelined reconfigurable consistent multiplier which can be exchanged between the constants 1912, 1111, 1331 can be found in Fig. 1. Pipeline registers are embedded after each phase which incorporates enlists in the multiplexer stages. The wires can be related with a left move and a sign. The esteem vector noted other than every activity relates to the moderate or yield factors for a particular multiplexer design. A switchable snake/subtractor is portrayed as a viper with an extra sign vector input.

II. LITERATURE SURVEY

Finding the run-time reconfiguration of SCM and MCM snake charts is a speculation of the essential SCM/MCM issue and accordingly additionally NP-finish. In any case, arrangements were introduced which can discover reconfigurable SCMs (RSCM). As a matter of first importance there are diverse arrangements focusing on ASICs, all concentrating on multiplexer-based reconfiguration. In the technique for Tummelt shammer et al. [17] a few enhanced SCM diagrams are combined by a recursive calculation called DAG combination. Two SCM charts are intertwined with negligible equipment exertion by embeddings multiplexers to switch between the distinctive constants. Promote coefficients can be incorporated by recursively adding the related SCMs to the current RSCM. Similitudes between various coefficients in the authoritative marked digit portrayal of constants are abused by Chen et al. [18] to acknowledge RSCMs. Indistinguishable examples in the CSD portrayal of constants are sought and combined utilizing multiplexers to have the capacity to switch between the distinctive movements and interconnections to understand a particular consistent. Faust et al. [5] utilize a snake chart construct approach with unique concentration in light of insignificant rationale profundity. Notwithstanding the techniques depicted before, their calculation does give answers for RSCM as well as for reconfigurable various consistent increase (RMCM). Such RMCM are additionally given by ORPHEUS [2] which can intertwine MCM arrangements given by Hcub [12] a heuristic. Alterative ideas to acknowledge RMCMs are assessed amid the calculation run-time and the best generally speaking arrangement is chosen. The displayed calculations for RSCM and RMCM, separately, don't consider pipelining or other FPGA particular issues as their attention is on ASIC usage. There is a FPGA-particular calculation called ReMB technique [1] which was additionally dissected and reached out in [19] by our gathering. A RSCM is developed from essential structures that fit into the fundamental rationale components (BLE) of FPGAs. This strategy is restricted to little issue sizes because of a high memory utilization [19] and does not consider pipelining. As pipelined arrangements are required for rapid applications on

FPGAs, there is an ideal viper chart based calculation for RSCM and RMCM with center around pipelined acknowledge proposed by our gathering [20]. The possibility of DAG combination [17] is gotten as of now upgraded pipelined snake diagrams (PAGs) are melded. Rather than melding just two PAGs in one improvement run, all PAGs of the required constants are considered in one single streamlining hurried to create a superior, multiplexer-mindful pipelined acknowledgment. Nonetheless, the ideal approach must be utilized for little issues on account of the intricacy of a full hunt over all conceivable melding arrangements. Thus, a great heuristic strategy is required which gives arrangements near the ideal.

III. PROPOSED RELIABLE ARCHITECTURES

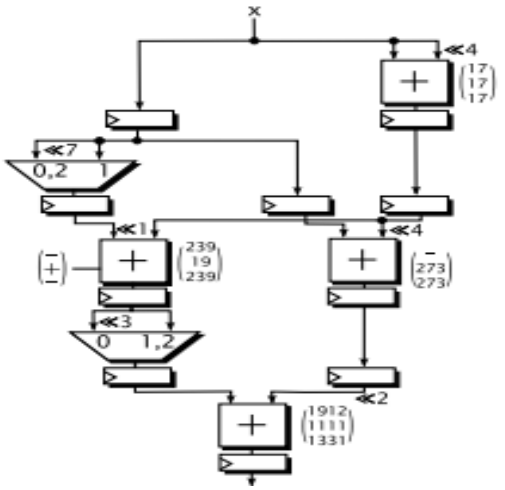


Fig.1: Reconfigurable single constant multiplier which can be switched between the constants 1912, 1111, 1331.

Just like RPAG, the proposed pipelined adder graph fusion is backward-exploring. Starting with the constant mapping of the output stage all PAGs are fused stage by stage. The basic idea is to combine those intermediate values in the respective preceding stage to share the same adder, which leads to a minimal overhead of possibly necessary multiplexers or switchable adder/subtractors. To do so, all combinations of intermediate values are evaluated and their costs are calculated separately and stored in a cost matrix. Multiplexers can appear at the inputs of the successive stage in the following cases:

1) input has a different shift value
2) input has a different source
3) both of 1) and 2)

As described before, the target is to select the overall best mapping M for the specific stage s. This selection will be the source for the determination of the next preceding stage s − 1. The procedure is repeated until the input (stage 0) is reached. A simplified pseudo-code of the generalized fusion process is given in Listing 1. It assumes that the overall best solution and costs are globally known. It is started with the constant mapping M of the output stage, the preceding stage s, the search width w (unlimited for the optimal search) and the costs of the current path current cost, which is zero in the beginning. Compared to the algorithm presented in [20] the algorithm was generalized, such that it can be used both as heuristic and in an optimal way. In contrast to an arbitrary search through the whole search space, which was done in the former version, the search is now improved and based on a sorted cost matrix.

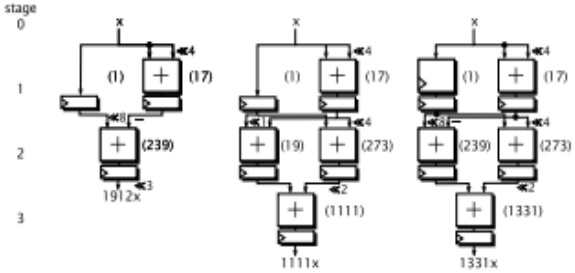
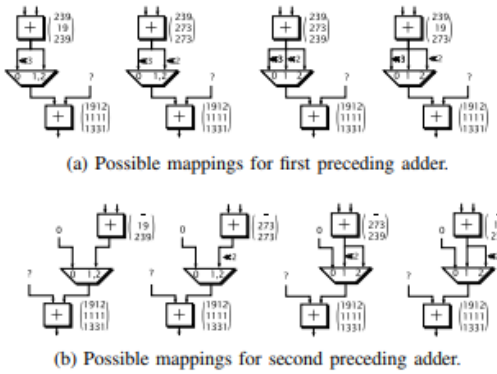


Fig.2: RPAG solutions for the constants 1912, 1111, 1331.

In the running example used here, the three SCM graphs generated with RPAG (see Fig. 2) are fused starting with the desired output mapping M = {1912; 1111; 1331}, meaning that the resulting circuit can be switched between these three values. This will be called an SCM circuit with three configurations in the following. The enumeration of all adder combinations of the second last stage consisting of {239} for the first, {19},{273} for the second and {239},{273} for the third SCM solution, respectively, is given in Fig. 3. For the constant 1912 only one adder is required in stage two, but two adders are required in the other SCM circuits. This fact is considered by the insertion of a don't care "–". Due to a separate cost calculation for a specific combination, some of the multiplexer inputs are unknown from a local point of view. These are marked with a question mark. They do not have any contribution to the currently considered adders' multiplexer costs, which is a main advantage of the proposed method as the costs for each combination can be calculated and evaluated separately.



(a) Possible mappings for first preceding adder.

(b) Possible mappings for second preceding adder.

Fig.3: All combinations of the adders in the second last stage for the given output mapping and the given RPAG SCM solutions in Fig. 2.

As a zero input can be realized by resetting the succeeding register, these inputs are not considered as multiplexer inputs,

as our implementation targets pipelined implementations. The multiplexer cost for each mapping is stored in a multidimensional cost matrix C (line 3 in Listing 1). The cost matrix for the combinations of the current stage can be found in TABLE I in a two dimensional representation. For example, the first entry in the first row (1.33) corresponds to the leftmost mapping in Fig. 3 (a) in which two multiplexer inputs, each with a cost MUX of 2 3 are used.

To get a valid solution a selection of one mapping for the first preceding adder in Fig. 3 (a) will directly force the selection of the corresponding mapping for the second adder (Fig. 3 (b)) or reduces the selectable possibilities for other adders in a more general case. This means each valid mapping solution for a specific stage consists of selections with a unique row and column index. Thus, finding the cheapest mapping solution M for a specific stage reduces to finding the valid solution with the lowest sum of costs. An example for such a selection is given in Fig. 3. It corresponds to the highlighted with a total cost contribution of 1.33 + 0.67 = 2 2:1 multiplexers. The cheapest solution for a specific stage is not necessarily the best overall choice as it affects the costs in the preceding stages. So, to find the optimal solution, a full search over all possibilities is necessary. The search space can be illustrated as a decision tree, which consists of the decision itself as node and the cost of the decision as edge.

IV. SIMULATION RESULTS

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	91	16640	0%
Number of Slice Flip Flops	112	33280	0%
Number of 4 input LUTs	150	33280	0%
Number of bonded IOBs	69	309	22%
Number of GCLKs	1	24	4%

Fig.4: DESIGN SUMMARY

```
==================================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'Clock'
  Total number of paths / destination ports: 6272 / 32
----------------------------------------------------------------------------------
Offset:              12.868ns (Levels of Logic = 6)
  Source:            OB_0 (FF)
  Destination:       YC1<7> (PAD)
  Source Clock:      Clock rising

  Data Path: OB_0 to YC1<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q              5    0.591   0.776  OB_0 (OB_0)
     LUT2:I0->O           1    0.648   0.452  SC/Mxor_S0_0_xo<0>21 (N111)
     LUT4:I2->O           1    0.648   0.563  EAB/e026 (EAB/e026)
     LUT4:I0->O          36    0.648   1.406  EAB/e0206 (e0)
     LUT4:I0->O           8    0.648   0.900  SEC/Mmux_YC31211 (SEC/N2)
     LUT4:I0->O           1    0.648   0.420  SEC/Mmux_YC39 (YC3_2_OBUF)
     OBUF:I->O                 4.520          YC3_2_OBUF (YC3<2>)
    ----------------------------------------
    Total                     12.868ns (8.351ns logic, 4.517ns route)
                                       (64.9% logic, 35.1% route)

==================================================================================
```

Fig.5 TIME SUMMARY

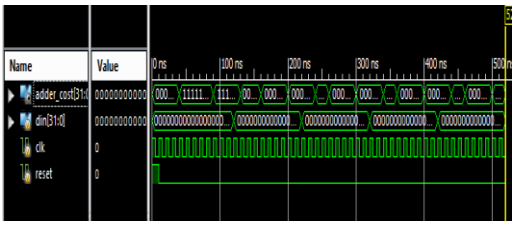


Fig.6: OUTPUT for 1912

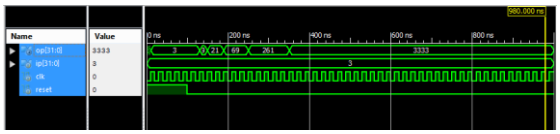


Fig.7: OUTPUT for 1111

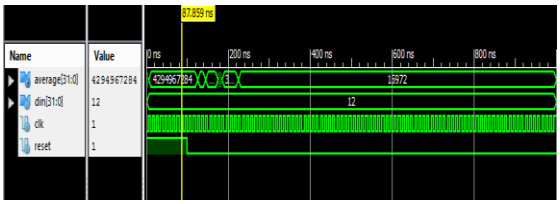


Fig.8: OUTPUT for 1331

V. CONCLUSION

This work presented a new heuristic to generate pipelined run-time reconfigurable constant multipliers based on an optimal algorithm. The heuristic was motivated by a complexity consideration of the search space. With the heuristic problems with a larger size become solvable. An extensive benchmark evaluation showed superiority over previous work, as we could show a 9-26% slice reduction on average. Additional extensions to the algorithm were presented which further reduce the slice consumption of the resulting solutions. These were the support of ternary adders, and optimized multiplexer and switchable adder/subtractor mapping. Finally it could be shown by RMCM and FIR filter experiments that the heuristic is raising the solvable problem size and the application domain of the proposed fusion method. Compared to other reconfiguration approaches our method provides the fastest reconfiguration time with a low resource consumption for a limited number of configurations.

VI. REFERENCES

[1]. Massoud Pedram, "Power minimization in ic outline: Principles and applications," ACM Trans. Des. Autom. Electron. Syst., vol. 1, no. 1, pp. 3– 56, Jan. 1996.

[2]. Qing Wu, M. Pedram, and Xunwei Wu, "Clock-gating and its application to low power outline of successive circuits," Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on, vol. 47, no. 3, pp. 415– 420, Mar 2000.

[3]. G.E. Tellez, A. Farrahi, and M. Sarrafzadeh, "Action driven clock outline for low power circuits," in Computer-Aided Design, 1995. ICCAD-95. Process of Technical Papers., 1995 IEEE/ACM International Conference on, Nov 1995, pp. 62– 65.

[4]. E. Lee and A. Sangiovanni-Vincentelli, "Looking at models of calculation," in Proceedings of the 1996 IEEE/ACM global

meeting on Computer-supported outline. IEEE Computer Society, 1997, pp. 234– 241.
[5]. Gilles Kahn, "The Semantics of Simple Language for Parallel Programming," in IFIP Congress, 1974, pp. 471– 475.
ssss
[6]. Edward A. Lee and David G. Messerschmitt, "Static booking of synchronous information stream programs for computerized flag handling," IEEE Trans. Comput., vol. 36, no. 1, pp. 24– 35, 1987.
[7]. E.A. Lee and T.M. Parks, "Dataflow process systems," Proceedings of the IEEE, vol. 83, no. 5, pp. 773 – 801, may 1995.

[8]. Syed Suhaib, Deepak Mathaikutty, and Sandeep Shukla, "Dataflow structures for GALS," Electronic Notes in Theoretical Computer Science, vol. 200, no. 1, pp. 33– 50, 2008.
[9]. Tzyh-Yung Wuu and Sarma B. K. Vrudhula, "Union of offbeat frameworks from information stream determination," Research Report ISI/RR-93-366, University of Southern California, Information Sciences Institute, Dec 1993.
[10]. Behnam Ghavami and Hossein Pedram, "Superior offbeat outline stream utilizing a novel static execution examination technique," Comput. Electr. Eng., vol. 35, no. 6, pp. 920– 941, Nov. 2009.
[11]. S.C. Brunet, E. Bezati, C. Alberti, M. Mattavelli, E. Amaldi, and J.W. Janneck, "Parceling and enhancement of abnormal state stream applications for multi clock space designs," in Signal Processing Systems (SiPS), 2013 IEEE Workshop on, Oct 2013, pp. 177– 182.
[12]. Simone Casale-Brunet, Analysis and enhancement of dynamic dataflow programs, Ph.D. proposition, STI, Lausanne, 2015.
[13]. Endri Bezati, High-level combination of dataflow programs for heterogeneous stages, Ph.D. postulation, STI, Lausanne, 2015.
[14]. S. Casale-Brunet, M. Mattavelli, and J.W. Janneck, "Cushion improvement in view of basic way investigation of a dataflow program configuration," in Circuits and Systems (ISCAS), 2013 IEEE International Symposium on, May 2013, pp. 1384– 1387.
[15]. Xilinx, Analysis of Power Savings from Intelligent Clock Gating, August 2012, XAPP790.