# Generation of DFG for digital filter in HLL and algorithm optimization using MATLAB

R.Parameshwaran[#1],K.Hariharan[#2],C.Manikandan[#3,] T.Saipriyadharshini[*1], J.vasumathi[*2]
[#1,2,3]Assistant Professor , SASTRA University
[*1,2] PG SCHOLAR, SASTRA University
[#1] paramu32@gmail.com [#2] harikalyan87@ict.sastra.edu [#3] cmanikandan87@gmail.com
[*1]saipriyadharshini19@gmail.com [*2]vasucse90@gmail.com

**Abstract-This paper implements a tool that will integrate a HIGH LEVEL LANGUAGE and LOOP OPTIMIZATION TECHNIQUES. We can use this tool for IIR and FIR. This software tool is mainly used to calculate the critical path by utilizing the optimization techniques. The netlist output will be produced for each technique. Here we use RETIMING and UNFOLDING for optimization. Comparison of the netlist and critical path before and after the transformation (optimization) is illustrated.**

**Keywords- Retiming, unfolding, netlist, critical path, IIR-Infinite Impulse Response, FIR-Finite Impulse Response**

## I. INTRODUCTION

DSP- DIGITAL SIGNAL PROCESSING is used in various real time applications related with VLSI technology like wireless communication, multimedia, digital set up box, digital video, digital audio. For implementing IIR /FIR filter critical path calculation by using various techniques is most important to find feasible solution. But this will make some difficulties in manual calculations. To avoid those difficulties special software tools are needed. Here we proposed that software tool and estimate the critical path for different transformations[1].

## II. DESCRIPTION OF TECHNIQUES

IIR-Infinite Impulse Response Filter depends on present input, past input and past output. Phase distortion is introduced i.e. non- linear. It consists of sharp cutoff and throughput (success message delivery). It will work for infinite length.[2]

FIR-Finite Impulse Response filter produces output from the present input and past input. No need of feedback i.e. past output. No phase distortion is required i.e. linear. Recursive so it is Stable in nature. It will work for Fixed duration of time.[3]

LPM- Longest Path Matrix algorithm which is similar to Dijikstra's algorithm is used to obtain iteration bound. Retiming is used to find the critical path.

III. DATA FLOW DIAGRAM

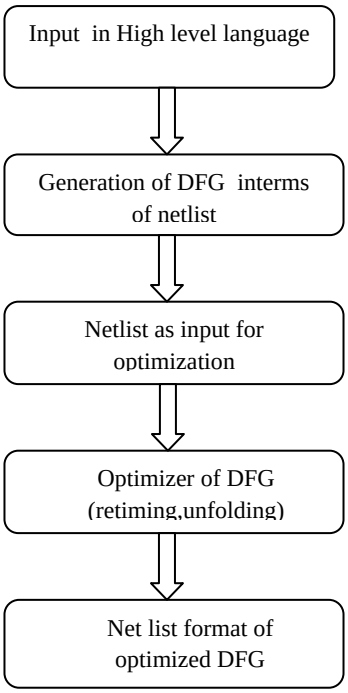


Fig1: Flow of project

Figure 1 shows the project flow. JAVA(HLL-HIGH LEVEL LANGUAGE) is used to get the user specification of filter equation as a input. This will produce the DATA FLOW GRAPH (DFG) in the form of netlist.

The output (netlist)of the HLL is given as a input for MATLAB (SIMULATION TOOL). Here we are applying the optimization techniques RETIMING and UNFOLDING. The final output will be the OPTIMIZED NETLIST and CRITICAL PATH.

IV. FUNCTIONAL DESCRIPTION

*A. Retiming*

Retiming is a transformation technique similar to pipelining. This optimization technique is used to reduce the critical path by changing the location of latches without affecting the functionality of input and output parameters. Additionally this technique reduces the power consumption and provides high speed. Reliable to do logic synthesis. This method is achieved through Floyd Warshall algorithm.[4]

*1) Retiming Inputs*

a)Graph matrix: This is the Edge Weight matrix. Matrix values obtained from the delay elements.

1. If there connection exists consider the delay value at each edge.
2. If there is a direct connection between two nodes consider delay value as zero
3. If there is no connection between two nodes consider -1.

b) Weight matrix: This is the node weight matrix. Values of this matrix are obtained form computation time of each node. This node should be in the order of obtained netlist

*2) Retiming outputs*

- Gfinal matrix: This is Edge weight matrix it gives value of delay at each edge after retiming.
- Success: Tells about the status of the retiming algorithm. Such as success or not.
- crit_orig: Before retiming-original critical path of the DFG
- target_crit: After retiming-new critical path of DFG
- Tinf: Iteration bound of the DFG

*B.Unfolding*

A great deal of research has been done attempting to optimize the various aspects of an application's execution by applying various graph transformation techniques to the application's flow graph. Unfolding is one of the most effective techniques which alter the graph by making multiple iterations visible simultaneously.

So that size of the graph is increased, but we can derive some benefit by creating more options for parallel execution. [5]

Steps to create unfolding homogeneous graphs:

1. Let f be a positive integer
2. To alter a graph f consecutive iterations are visible simultaneously
3. To do this need to create f copies of each node
4. Replacing node u in original graph by the nodes u1 through uf in new graph
5. Result in unfolded graph Gf=hVf,Ef,df,tf i
6. Vf is a vertex set is a union of f copies of each node in V
7. All are exact copies so that computation time remains same. Such as tf(Uf)=t(u)
8. Each edge of G is corresponds to f copies in the unfolded graph
9. Delay counts of the copies do not match that of the original edge.

Unfolding technique is employed to reduce the iteration bound to further reduce its critical delay path.

*1) Unfolding inputs*

- Graph matrix and Weight matrix is same as specified in the section retiming.
- Jfactor: This factor is specified by the user according to their needs

*2) Unfolding outputs*

- Unfolded matrix: This is the unfolded form of weight matrix
- Weight_new: New node matrix contains the new nodes

V. EXPLANATION FOR DFG NET LIST AND NODE LIST

*A. DFG:*

Data Flow Graph is used to represent the DSP program, which is a directed graph (i.e., each edge has a distinct direction) that describes the program. This consists of nodes and edges. Each edge in DFG describes a precedence constraint between two nodes.

For example y(n)=x(n)+a1*y(n-1)

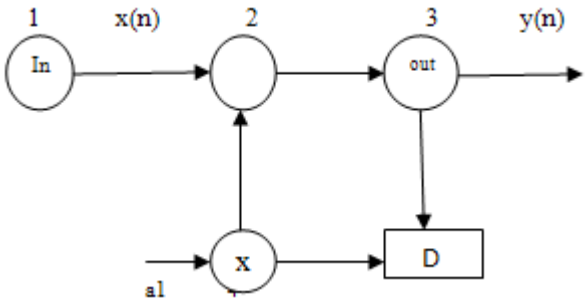


Fig 2: (DFG) Data Flow Graph of the nodelist and netlist given below

*B. Node list:*

This list consists of Node numbers,Node description and Computation time.[7]

Table 1: Node list of the given DFG

| Node Number | Node Description | Computation Time |
|---|---|---|
| 1 | Input | 0 |
| 2 | Adder | 1 |
| 3 | Output | 0 |
| 4 | Multiplier | 2 |

*C. Netlist*

Table 2: Net list of the given DFG

| Node Number | Connected with | Computation Time |
|---|---|---|
| 1 | 2 | 0 |
| 2 | 3 | 0 |
| 3 | 4 | 1 |
| 4 | 2 | 0 |

Table 2 gives the Net list of the given DFG Net list is a textual description of a circuit made of components. This net list consists of Node number, connected node number and Computation time. [7]

VI. OUTPUT

*1)Description of main function*

A. function main(unfolding, retiming,fir_irrbar,J_unfold)

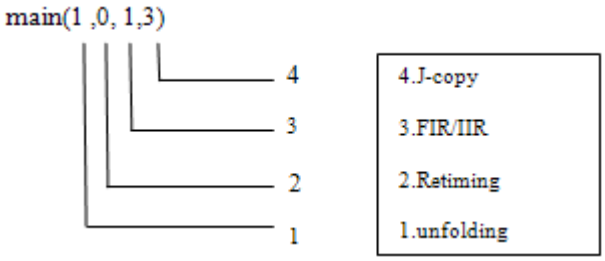


Fig 3: view of main function

Table 3: Main function description

| Unfolding | Retiming | Operation |
|---|---|---|
| 0 | 0 | Create the netlist of the given filter equation |
| 0 | 1 | Create netlist and retime of the filter |
| 1 | 0 | Create netlist and unfold of the filter |
| 1 | 1 | Create the netlist,unfold of the filter and retimed it |

Figure 3 gives the over view about the main function. Table 3 describes the main function briefly.

The numbers 1 and 0 are assigned to the main function. User can select the technique, filter and number of copies. This can be done by giving 1 to select and 0 to deselect the corresponding operation. First two numbers for unfolding and retiming. If we want to do the optimization for FIR filter number 1 should be given in the third place. 0 for IIR filter.

B. S*imulation results*

1) >>main(0,1,1,3)

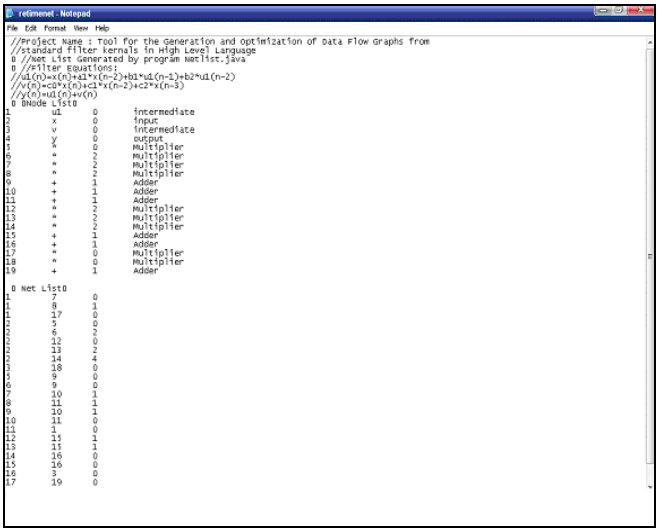


Fig 4: Retiming for FIR with copy of 3

2) >>main(1,0,0,3)

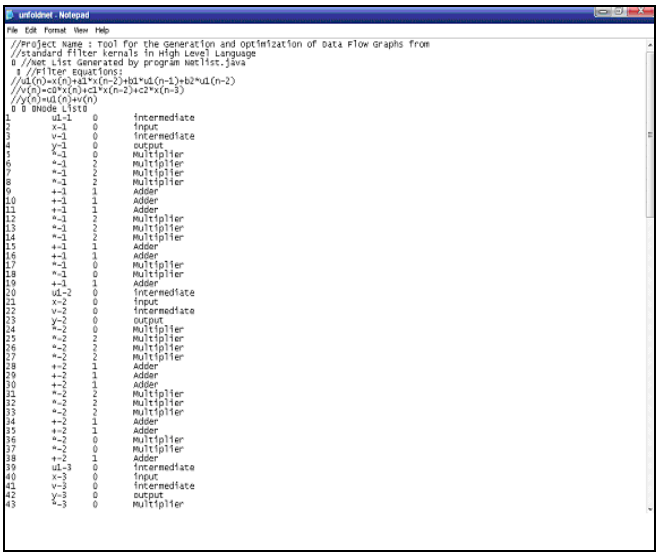


Fig 5: Unfolding for IIR with copy of 3

3) >>main(0,0,1,3)

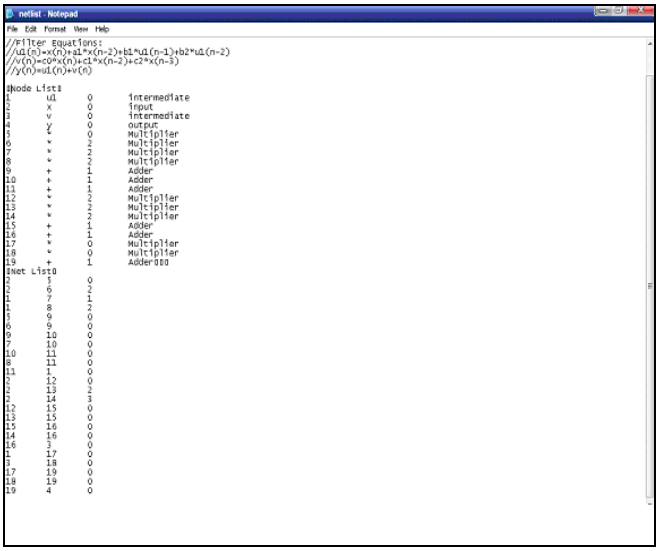


Fig 6: Nodelist and netlist for the given FIR filter Without retiming and unfolding

4) >>main(1,1,1,3)

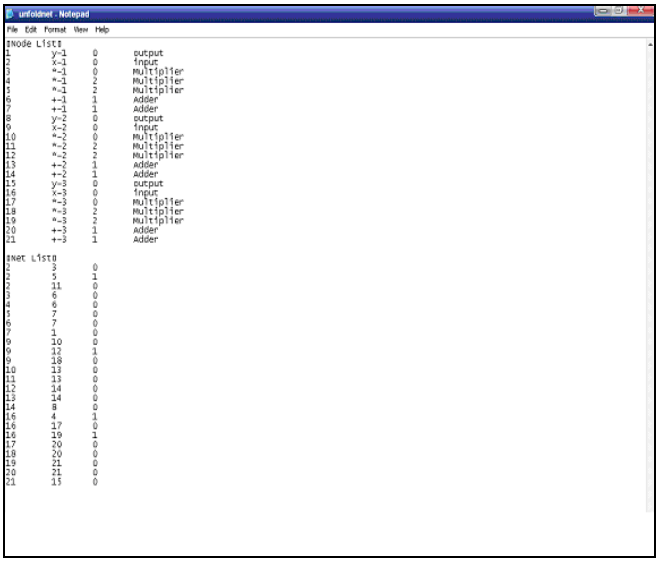


Fig 7: Unfolding and retiming for FIR with copy of 3

*C.Functions*

1) crit.m

**function [lenOfCrit,path ,lpm] = crit(graph, weight)**

This critical path function is used to compute

- lenOfCrit : The critical path of the original DFG
- path : This matrix gives the nodes on the critical path
- lpm : This is the input matrix to the lpm algorithm

2) fir_retiming.m

**function [gfinal, success, target_crit] = retiming(graph, weight)**

This function is used to perform retiming to achieve the least possible critical path for a given FIR filter. This is done by iterating till a feasible solution reached. Output of this function as follows:

- gfinal : This is the edge matrix which gives the value of the delay at each edge connecting two nodes after retiming
- success : Indicates whether the retiming algorithm was successful or not
- target crit : The new critical path of DFG achieved after retimimg

**Sprime = floyd_warshall(Gprime);**

Flyod warshall algorithm is used to compute the shortest path from node U to V.

This matrix is called Sprime and is computed from matrix Gprime

3) flyod_warshall.m

**function [path, success] = floyd_warshall(graph)**

This function implements the floyd_warshall algorithm which is a type of shortest path algorithm. If no negative cycles exist in the constraint graph the algorithm computes the shortest path between all possible pairs of nodes. It returns the following information.

a) Path : This is the matrix that gives the shortest path between pairs of nodes

b) Success : This variable indicates the successful execution of the algorithm. Success is 0,if negative cycles exist.

4) mat2net.m

**function mat2net( newnetlist, graph, weight,symbol, nodeinfo,header)**

This function is used to convert the different matrices representing the DFG into a netlist and used to recover the netlist format after loop optimization have been carried out using various algorithm. Output of this function is a netlist that is written into a text file.

5) net2mat.m

**function [graph, weight, symbol, nodeinfo, header] = net2mat(filename)**

This function is used to convert the netlist into different matrices that are used by various matlab functions to optimize the filter.

6) retiming.m

**function [gfinal, success, crit_orig, target_crit, Tinf ] = retiming(graph, weight)**

This function is used to perform retiming to achieve the least possible critical path for a given DFG.

Output function consists of

- Crit_orig : The original critical path of the DFG before retiming
- Target_crit: The new critical path of the DFG achieved after retiming
- Tinf : Iteration bound of the DFG
- Gfinal and success as same as described above

7) Unfold.m

**function[unfolded,weight_new]=unfold(graph,Weight,Jfactor)**

This function implements unfolding for a DFG. The unfolding factor is specified by the user. Unfolding reduces the iteration bound of DFG

a) Output of this function:

- Unfolded : The unfolded edge weight matrix
- Weight_new : This is the new node weight matrix that contains the new node

## VII. CONCLUSION

The software suite for calculating the critical path which plays a vital role in designing of IIR/FIR filters All calculations has been done by automation. This kernel tool can be used for various types of filter configurations. Retiming and unfolding techniques were applied on a nodelist, and then optimized nodelist has been generated.

## VIII. FUTURE WORKS

We implemented this tool using retiming and unfolding techniques.Further work can proceed with other optimizationtion techniques. we generated the output in the form of nodelist and netlist. Future work may be generating the output in the forrm of DFG(Data Flow Graph). This software tool can be expanded usimg GUI-GRAPHICAL USER INTERFACE.

## XI. REFERENCES


[1] VLSI Digital Signal Processing Systems – *Design and Implementation* – Keshab K.Parhi, John Wiley & Sons, Inc,1999

[2] **"**Digital IIR filters with minimal group delay for real-time applications"-IIR Filters,IEEE by Skogstad, Ståle Andreas 2012

[3] "Short length FIR filters and their use in the non recursive filtering"-*FIR filters*,IEEE Trans by,Z.L.Mou and P.Duhamel.

[4] "Exhausting scheduling and retiming digital signal processing systems"- *Retiming*, IEEE Transaction on circuits and system,by T.C.Denk,K.K.Parhi ,Inc 1998.

[5] "Rate optimal and DSP synthesis by pipelining minimum unfolding"-*Unfolding*,IEEE Transactions on signal processing by L.G.Jeng and L.G.Chen,Inc 1994.

[6] S.H. Gerez, S.M.Heemstra deGroot, and O.E.Herrmann, "A polynomial - time algorithm for the computation of the iteration-period bound in recursive dataflow graphs",-*DFG* IEEE Trans on Circuits and Systems - I: Fundamental Theory and Applications, Jan 1992.

[7] Functions to generate node list and net list in matlab - http://www.mathworks.in/matlabcentral.