Fixed Coefficient, Single-Channel FIR Filters

Executive Summary

This filter was composed of 2, 64-tap Square Root Raised Cosine (SRRC) Filters.

Module	FixdCoefFilt
Device	QuickDSP QL7180
Speed Grade	-7 (Worst Case)
Area (no buffers/ buffered)	3976/4031 of 4032
ECU Cells	4 of 18
Maximal Clock Frequency	42 MHz

This filter was composed of 47 randomly generated 12-bit coefficients.

Module	FixdCoefFilt
Device	QuickDSP QL7180
Speed Grade	-7 (Worst Case)
Area (no buffers/ buffered)	3930/4021 of 4032
ECU Cells	18 of 18
Maximal Clock Frequency	45 MHz

This filter was composed of a single 96-tap Square Root Raised Cosine (SRRC) Filter. The coefficients were symmetric.

Module	FixdCoefFilt
Device	QuickDSP QL7180
Speed Grade	-7 (Worst Case)
Area (no buffers/ buffered)	1970/2260 of 4032
ECU Cells	1 of 18
Maximal Clock Frequency	61 MHz

Features

Features of the Amphion Fixed Coefficient FIR (Finite Impulse Response) Filter Generator:

- Supports up to 128 Taps
- Input word length 8 - 10 bits
- Coefficient word length 8 - 12 bits
- Output word length 8 - 29 bits
- Two's complement input and output data
- Full precision output data - no rounding or overflows
- Filter is pipelined to give best performance
- Multiplier structure for each tap is optimized according to coefficient value
- Option of including the ECU in filter taps
- Supports Non-Symmetrical and Symmetrical Filters (Odd and Even)
- Estimates the resources required for filter implementation
- Optimized for QuickLogic FPGA technologies
- Potential applications include:
  - Wavelet transforms
  - Pulse shaping filters
  - General interpolating filters
  - Matched filters

General Description

FIR (Finite Impulse Response) filters are one of the most basic building blocks used in digital signal processing. The output Y of an N tap FIR filter is given by the equation below

$$Y_t = a_0X_t + a_1X_{t-1} + \cdots + a_{N-2}X_{t-(N-2)} + a_{N-1}X_{t-(N-1)} \quad [1]$$

Here X is the input to the filter and a0, a1… aN-1 are the filter's coefficients. If the coefficients exhibit such a symmetry that

$$a_0 = a_{N-1},\ a_1 = a_{N-2} \ \ldots \text{ etc.}$$

then equation [1] becomes

$$Y_t = a_0(X_t + X_{t-(N-1)}) + a_1(X_{t-1} + X_{t-(N-2)}) + \cdots \quad [2]$$

The filter represented by equation [2] is called an even-symmetric FIR filter due to the even number of coefficients.

If there are an odd number of coefficients and they exhibit such a symmetry that

$$a_0 = a_{N-1},\ a_1 = a_{N-2} \ldots \text{ etc. and } a_{N/2}$$

is an unpaired coefficient.

Then equation [1] becomes

$$Y_t = a_0(X_t + X_{t-(N-1)}) + a_1(X_{t-1} + X_{t-(N-2)}) + \cdots + a_{N/2}(X_{N/2}) \quad [3]$$

The filter represented by equation [3] is called an odd-symmetric FIR filter since all of the coefficients are symmetric and have a matching coefficient apart from the odd coefficient.

If the coefficients do not display these symmetric characteristics then the FIR filter represented by equation [1] is called a non-symmetric filter.

This FIR filter program has the ability to generate Non-Symmetric and both Odd and Even Symmetric FIR filters.

.

Fixed Coefficient Filters

This DSP Wizard Software Fixed Coefficient FIR Filter Graphical User Interface (GUI) creates fixed coefficient FIR filters; i.e. The coefficients are static and cannot be changed or programmed on-line. The fixed coefficient FIR filter program reads the coefficients an input file and generates the HDL accordingly. Since the coefficient values are static the program optimizes the multiplier structure in each tap according to the coefficient value. The resources required for each tap depend on the coefficient value, the input data width and the type of FIR filter chosen. Some filter configurations will require more resources than others but generally as more taps are required, the input data width increases and the coefficients become more complex more resources will be needed.

The GUI which generates the HDL will also automatically add pipeline delays to the FIR filter. The circuit complexity will increase as more taps are required, with the use of larger coefficients and with larger input data. As complexity increases more pipelines will be added to shorten critical paths and ensure that the best performance is obtained from the fixed coefficient FIR filter.

This GUI can also include the ECU in a filter tap to reduce the number of logic cells required.

The different types of FIR filters produced by the program will be explained in the following paragraphs.

Non-Symmetric FIR Filters

A non-symmetric FIR filter performs the filtering function defined by equation [1]. The filter program will allow up to 128 taps in the fixed coefficient FIR filter. The non-symmetric filter will process a new two's complement data word in every clock cycle. For a non-symmetric FIR filter of this architecture a multiplier is required for every tap. In each tap the coefficient value will be multiplied by the input data in the delay chain. The products from the taps are added together to give the filter output.

The architecture of an non-symmetric FIR filter is illustrated below.

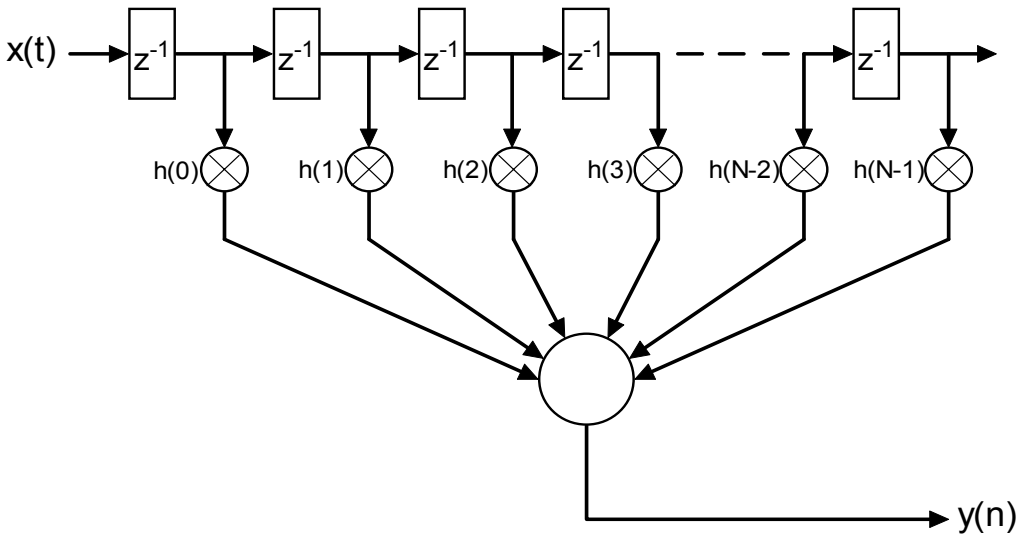


FIGURE 1: Architecture of a Non-Symmetric FIR Filter

Symmetric FIR Filters

A symmetric FIR filter takes advantage of the symmetry exhibited by the coefficients to reduce the circuit size. An even-symmetric FIR filter performs the function defined by equation [2] and on odd-symmetric FIR filter performs the function defined by equation [3]. The filter program will allow up to 128 taps in the symmetric fixed coefficient FIR filter. Like the non-symmetric filter, the symmetric filter will also process a new two's complement data word in every clock cycle. For a symmetric FIR filter of this architecture some resources can be saved by implementing equation [2]. Since some taps share the same coefficient, the multiplier structure can be further reduced to take advantage of this property.

The products from the taps are added together to give the filter output.

For an even-symmetric FIR filter there is perfect symmetry of coefficients (i.e. Each tap has the same coefficient as another) but an odd-symmetric FIR filter will have one unpaired coefficient. This is the only difference between an odd-symmetric and even-symmetric FIR filter.

This is the architecture of an even symmetric FIR filter.

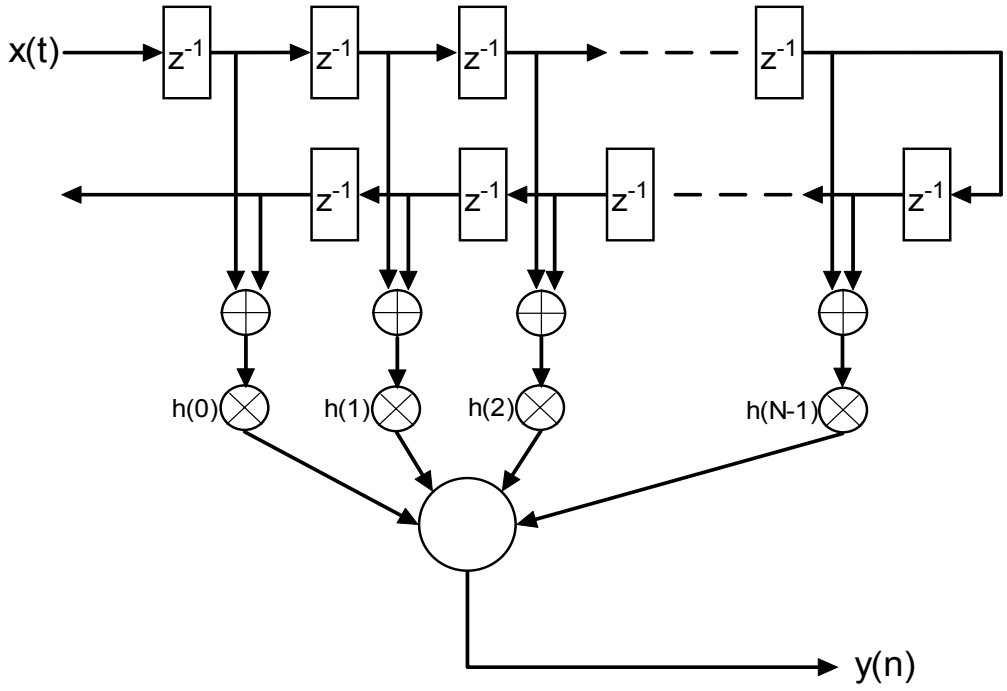


FIGURE 2: Architecture of an Even Symmetric FIR Filter

This is the architecture of an odd symmetric FIR filter. Note the unpaired tap.

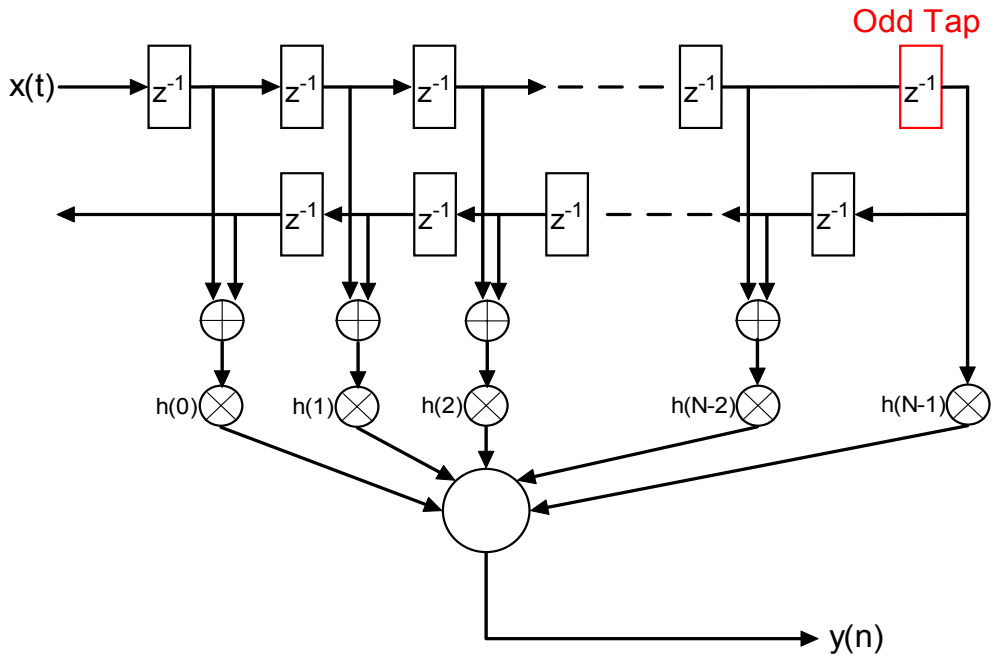


FIGURE 3: Architecture of an Odd Symmetric FIR Filter

I/O Descriptions

Unless stated otherwise all signals are active high and bit(0) is the least significant bit.

Signal	I/O	Size	Description
CLK	1	1	Clock - rising edge active
RST	1	1	Asynchronous Reset
X	1	8 - 10	Input data for the fixed coefficient FIR filter. The input data should be in two's complement format.
Y	0	8 - 29	Output data for the fixed coefficient FIR filter. The output data will be in two's complement format.

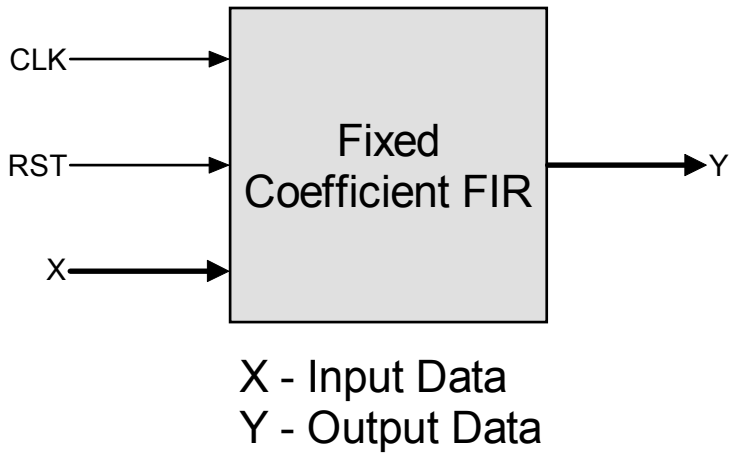


FIGURE 4: I/O Diagram of Fixed Coefficient FIR Filter

Functional Description

A functional description of the fixed coefficient FIR filter is given in the following paragraphs.

Clock and Reset

All flipflops in the fixed coefficient FIR filter core operate on the rising edge of the input clock CLK. All flipflops in the fixed coefficient FIR filter are reset on the rising edge of the input reset RST.

Data Loading

Input data is loaded into the fixed coefficient filter through port X. The size of port X can vary from 8 to 10 bits and the data should be in two's complement format. Data can be loaded continuously into the filter at a rate of one new sample every clock cycle.

Outputs

Port Y is the output port for the fixed coefficient filter. The size of port Y can vary from 8 to 29 bits and the data will be in two's complement format. The filter latency depends on the number of taps, coefficient values and input data width. The latency will be at least one cycle and can be as many as 10.

Using the Fixed Coefficient FIR Filter

The fixed coefficient filter design files are generated by configuring the DSP Wizard Fixed Coeffiicient FIR filter GUI. One of the files is the coefficicent file (*.cof). The coefficients must be in hexadecimal format. If any coefficient is out of range, the log file will contain a message reporting this. The input data for the test vectors can also be entered via the GUI. If the user wishes to use their own input data it should be in a data file (*.dat). Otherwise, the sampling data test vectors will be generated randomly and automatically. In other words, the data file is optional.

Format of the coefficient file (*.cof) and data file (.dat) is shown below. The first hexadeciamal is the first coefficient or first input data to the FIR filter.

029
FDD
F42
784
D1F
A94
B2A

Using the Filter GUI

The user will be required to enter:

1. The number of taps (1 to 128)
2. The Input Data Width (8 to 10 bits)
3. The Coefficient Width (8 to 12 bits)
4. The Filter Type (0 for Non-Symmetric and 1 for Symmetric)
5. The number of ECU's available (Depends on the Device used - 10 to 18)
6. The Filter top-level module name (Use letter and numbers only)
7. The name of the coefficient file (Type the *.cof extension)
8. The name of the log file (Type the *.log extension)
9. Source of input data (From a file or randomly generated)
10. The name of the input file (Type the *.dat extension)
11. The number of data samples in the input file

Items 9-11 are entries to generate the testbench for design file. If the coefficient file or input data file cannot be found, or if there is a problem with the information in any of the fields a message will be displayed. More detailed information regarding any problem can also be found in the log file.

Files Created by the Filter Generator

After the configuration is done via the DSP Wizard GUI, it will read the coefficients and data from the *.cof file and *.dat file to produce 6 basic files (plus 2 other files, a *.prj file and a *.v toplevel-template file).The 6 basic files are sufficient for evaluation and simulation. The *.prj file is for Synplify project file and the *.v file only adds an additional hierarchy to the toplevel_FixdCoefFilt.v file. The name of the files will be determined by the filter name (e.g. FixdCoefFilt) and the log file name (e.g. fixcfir.log).

The files created by the program are -

■ Top Level Verilog source code	toplevel_FixdCoefFilt.v
■ Verilog source code	FixdCoefFilt.v
■ Verilog testbench	tb_FixdCoefFilt.v
■ Test data	FixdCoefFilt.vec
■ C Model of filter	FixdCoefFilt.c
■ Log file	fixcfir.log

Verilog Source Code

All the source code needed for the fixed Coefficient FIR filter is in the verilog file produced by the filter program. The user will need to add an " `include " line for the ecu.v file. This code can also be synthesized.

C Model of Fixed Coefficient FIR Filter

The C model of the fixed coefficient FIR filter is bit-accurate and can be called as a function within any C program. Input data should be passed into the function and it will return the output value. The function will have the same name as the top level Verilog module. Also included in the C model file will is a function which will generate random two's complement numbers. The size of the input data should be passed into the function which is called "randvec". This function will then return a random two's complement number.

The C Model uses static integers which means that if the function is used more than once within the same program it must be renamed each time.

e.g. If the function is called "FixdCoefFilt".

The first time the function is called it can be named "FixdCoefFilt".

If it was required a second time the name must be changed, for example "FixdCoefFilt1".

Log File

The log file will display all the fixed coefficient FIR filter parameters and coefficient values. This file will report any problems found in the *.cof file and will confirm all entered parameters and coefficients.

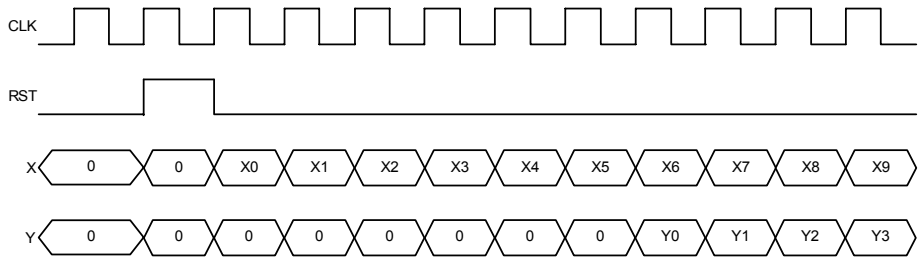
It will report exactly how many ECU's were used in the design and which coefficient used them and estimate how many logic cells are required for the filter.

(Note: The Logic Cell estimate is not guaranteed to be accurate but will provide the user with an idea of the resources required for the filter. Accurate utilization of logic cells can only be confirmed in the report file generated by QuickWorks SpDE after placed & routed).

Timing Diagrams

The following timing diagrams represent functional operation, since the actual timing is technology dependent.

This is the timing diagram for a 10 tap, non-symmetric filter with 6-cycle latency. The first output will appear 6 clock cycles after the first input.



Verilog Module Definition

The Verilog Module declaration for the core is given below. The user chooses the module name when they run the executable (e.g. "FixdCoefFilt"). The program will create a top level Verilog file which will instantiate the Fixed Coefficient Filter.

The top level module will be called toplevel_filtername and it will instantiate a module called filtername.

This is an example and the size of input port X and the output port Y are determined by the information entered by the user.

```
`include "FixdCoefFilt.v"

module toplevel_FixdCoefFilt (CLK, RST, X, Y);
  input CLK /*synthesis syn_isclock=1 */;
  input RST /*synthesis syn_isclock=1 */;
  input [9:0] X;
  output [14:0] Y;

FixdCoefFilt FCFIR1 (CLK, RST, X, Y);

endmodule
```

Support Contact Details

QuickLogic Corp.
1277 Orleans Dr.
Sunnyvale, CA 94089
USA

Tel: 408-990-4100
Fax: 408-990-4040
Email: support@quicklogic.com
URL: http://www.quicklogic.com