Design and Implementation of a 64-Tap 16-Bit Fixed-Point FIR Filter with CDC FIFO

Linxiao Wu
Electrical Engineering department of Columbia University
Columbia Univeristy in the city of New York
New York, United States
lw3227@columbia.edu

***Abstract*—The project presents a design and implementation of a 64 tap 16bit finite impulse response (FIR) filter core. Implementing the fixed-point for calculation, the FIR core receives 16bits signed Q1.15 inputs and coefficients, implementing Q7.30 width of data path inside, and generates signed Q7.9 data output. The FIR core receives the input samples at a throughout of 10KS/s, and the computation core runs at a higher clock frequency inside. To convey the data across two asynchronous clock domains, a Clock Domain Crossing (CDC) FIFO receiving dual clock is implementing. The data path is based on a single MAC unit controlled by the Finite-State-Machine (FSM), balancing the hardware efficiency and the speed. A Matlab-generated golden model is used for functional verification and accuracy analysis. The post-synthesis analysis is implemented to further confirm its functionality and timing compliance.**

Keywords—FIR Filter, Fixed-Point Arithmetic, CDC FIFO, Asynchronous Clock Domains, FSM-Controlled MAC

I. Introduction

Finite impulse response (FIR) filters are fundamental building blocks in digital signal processing systems and are widely used in applications such as communications, audio processing, and data acquisition. Compared to infinite impulse response (IIR) filters, FIR filters offer inherent stability and linear-phase characteristics, making them well suited for hardware implementation in fixed-point digital systems.

The main calculation that the FIR is implementing can is an accumulation. The expression is shown below, x[n] is the input signal, y[n] is the output signal, N is the filter order (number of taps), bi is the filter coefficient. This computation is also known as discrete convolution [1].

$$y[n] = \sum_{i=0}^{N} (b_i \times x[n-i])$$

In practical digital systems, FIR filters are required to operate under constraints on throughput, area, and power consumption. Hence, fixed-point arithmetic is usually implemented to balance the hardware efficiency and computation accuracy.

This project focuses on the design and implementation of a 16-bit 64-tap FIR core, implementing Q7.30 [2] width of data path inside, and generates signed Q7.9 data output. The FIR core supports receiving data at a throughout of 10K/s, and can run the inner computation at a much higher frequency. The maximum inner clock frequency that the FIR can support is 21.275MHz. To minimizing the area while maintaining functionality, a single multiplier–accumulator (MAC) unit controlled by a FSM is implemented, sequentially performs the MAC operations over all taps[3].

To address the clock domain crossing issue due to the two asynchronous clocks the FIR receives, the CDC FIFO is designed the convey the input data from 10K/s throughout to the inner clock domain. Otherwise, direct data transfer would lead to metastability and functional errors.

The design of FIR core is developed using the Verilog RTL and verified through the Matlab golden module. The RTL implementation is functionally verified through simulation and further validated using post-synthesis timing analysis.

II. Architecture & Design Methodology

A. Architecture

The FIR filter core is controlled by a finite-state machine (FSM) that coordinates coefficient loading, input data buffering, MAC computation, and register shifting. The FSM ensures correct data flow while enabling a single MAC unit to be reused for all 64 taps.

After reset, the FSM enters the IDLE state, where all control signals are deasserted. When the coefficient load signal (cload) is asserted, the FSM transitions to the PRE_LOAD state and writes the FIR coefficients into the coefficient memory.

Once coefficient loading is complete and valid input data is available, the FSM moves to the WAIT_DATA state. In this state, input samples are written into the CDC FIFO as long as the FIFO is not full. When valid data is present and the FIFO is not empty, the FSM proceeds to the REG_WRITE state.

In the REG_WRITE state, one input sample is read from the FIFO and written into the shift register. The FSM then enters the MAC state, where the multiply–accumulate operation is performed. Each clock cycle computes one tap of the FIR filter, and this state repeats until all 64 taps are processed.

After completing the MAC operations, the FSM enters the SHIFT state, where the shift register is updated to prepare for the next input sample. The FSM then returns to the WAIT_DATA state to continue processing subsequent samples.

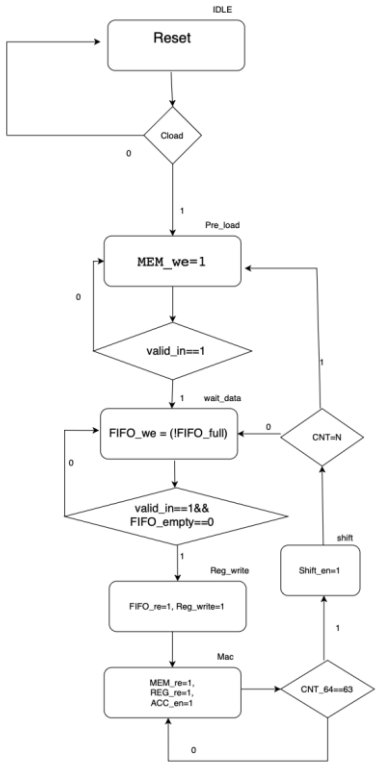

Fig. 1 ASM figure for the FSM

B. *ALU module*

The ALU in this project implements a single-MAC (Multiply–Accumulate) unit used by the FIR core to compute one convolution output every 64 cycles. It performs signed fixed-point multiplication (Q1.15 × Q1.15 → Q2.30), accumulates the 64 partial products with an extended internal precision (37 bits), and outputs a truncated 16-bit Q7.9 result. The ALU is designed as a sequential MAC engine, reusing one multiplier and one adder to minimize area and power, consistent with the project guideline requiring "one computational unit". It receives input samples (Din) and coefficients (Cin) each cycle from the data path(the data come from the register and memory in the system) and forms a multiply-accumulate operation when enabled, and signals one valid output every 64 MAC operations.

C. *Coefficient Memory and Shift Register*

The CMEM module stores 64 coefficient values required by the FIR core. Coefficients are written via Din, addressed by caddr, and activated by cload. Read operations provide the coefficient value for the ALU based on the same address.

The shift register module implements a 64×16 shifting buffer. It supports three operations: Write enable (we): Load new 16-bit input to D[0]; Shift enable (se): Shift all stored values upward; Read enable (re): Output the oldest stored value D[addr], therefore all register can be selected to connect to the output port.

D. *CDC FIFO*

The FIFO support two clock signal inputs and a reset. The width of input and output signals are 16bit (Q1.15). Two control signals (write enable and read enable) are supported. The output of the FIFO are data, empty flag and full flag.

In this asynchronous, the write pointer is generated under clk1, but the read pointer is generated under clk2, the two clocks are independent. Therefore, any signal transferred directly from one clock domain to another may cause set-up/hold time violation, leading to the metastability.

The read and write pointer inside are transformed to gray code before they are implemented to generate the full and empty signal. Because the gray changes only one bit at a time, using the gray code can help to prevent the metastability. Full and empty conditions depend on comparing synchronized pointers. Stable, monotonic Gray code ensures these comparisons remain correct even when the clocks are asynchronous

To reduce the metastability to an acceptably low probability, the pointers need to go through two flip-flops, thus the synchronized pointer w_ptr_2 and r_ptr_2 are considered as safe for functional logic.

III. FUNCTIONAL VERIFICATION

To verify the functionality of the FIR core, a MATLAB-generated golden module is implemented. In addition, the output txt file is sent to the MATLAB to conduct error analysis.

A. *RTL simulation.*

Implementing the Matlab to generate the random 64-tap 16-bit coefficients for the FIR core. For testing, a random generated 10000 data file is implemented to send the data input to the FIR core at a throughout of 10K/s.

In the test case, the 64 tap coefficient is loaded to FIR core initially before the computation. At this stage, the state of the FSM is “PRE_LOAD”.

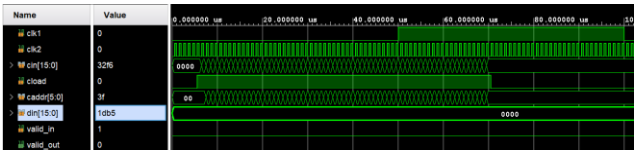

Fig. 2 RTL waveform

After finishing the loading of coefficients, valid_in is set to high and the data begins to be sent to the FIR core. The sequentially computation runs in the inner clock frequency of 1MHz, and the FIR core receives input and generates data output at the throughout of 10KHz.

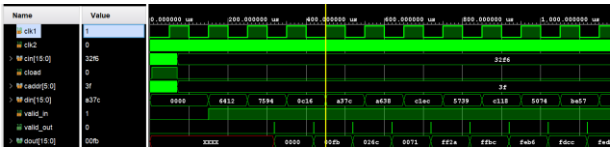

Fig. 3 output waveform

B. *Logic Synthesis*

Logic Synthsis is implemented to generate the netlist file, and the total are of the FIR core is 150462.

```
Number of ports:                          103
Number of nets:                         11753
Number of cells:                        11483
Number of combinational cells:           9045
Number of sequential cells:              2437
Number of macros/black boxes:               0
Number of buf/inv:                       4237
Number of references:                      51

Combinational area:              69755.041295
Buf/Inv area:                    21772.800865
Noncombinational area:           80707.677399
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:                150462.718694
Total area:                 undefined
1
Information: Propagating switching activity (medium effort zero delay simulation). (PWR-6)
Warning: Design has unannotated primary inputs. (PWR-414)
Warning: Design has unannotated sequential cell outputs. (PWR-415)
```

Fig. 4 area report

C. *Post-Synthesis Verification and accuracy analysis*

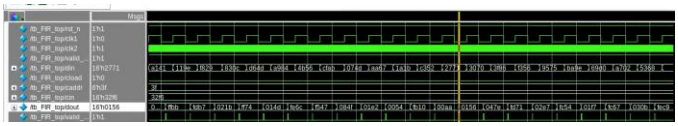

Fig. 5 post-synthesis waveform

Latency can be observed for the post-synthesis verification, and the waveform is the same as the pre-synthesis. For accuracy analysis, the Q7.9 output data txt file is sent the matlab. The accuracy is analyzed by comparing the results with the output file generated by MATLAB golden module.

```
0000000011111011
0000001001101100
0000000011110001
1111111100101010
1111111110111100
1111111010110110
1111110111001100
1111111011011001
0000000001111100
0000011011100110
0000000011101111
```

Fig. 6 output file

Error Analysis:

RMSE= 0.228299

NRMSE= 1.2120%

MAX error= 0.398438

```
Samples compared: 9000
RMSE   = 0.228299
NRMSE = 0.012120 (1.2120%)
Max |error| = 0.398438
```

Fig. 7 error analysis

D. *Power and Timing Analysis*

Total power:

```
                          Internal  Switching  Leakage    Total
Power Group               Power     Power      Power      Power   (     %)  Attrs
--------------------------------------------------------------------------------
clock_network          3.403e-05    0.0000     0.0000 3.403e-05 (83.20%)  i
register               6.538e-07 7.104e-08 9.611e-08 8.209e-07 ( 2.01%)
combinational          3.981e-06 1.999e-06 7.150e-08 6.051e-06 (14.79%)
sequential                0.0000    0.0000     0.0000    0.0000 ( 0.00%)
memory                    0.0000    0.0000     0.0000    0.0000 ( 0.00%)
io_pad                    0.0000    0.0000     0.0000    0.0000 ( 0.00%)
black_box                 0.0000    0.0000     0.0000    0.0000 ( 0.00%)

  Net Switching Power  = 2.070e-06   ( 5.06%)
  Cell Internal Power  = 3.866e-05   (94.53%)
  Cell Leakage Power   = 1.676e-07   ( 0.41%)
                         ---------
Total Power            = 4.090e-05  (100.00%)

X Transition Power     = 4.121e-11
Glitching Power        = 2.561e-08

Peak Power             =    0.1912
Peak Time              = 13019999.999
```

Fig. 8 power report

Maximum frequency:

23.2812ns*2 is the shortest period for clock2, so the maximum delay for the FIR is 21.475MHz.

Max path:

```
data required time                                    499.5372
data arrival time                                     -23.2815
--------------------------------------------------------------
slack (MET)                                           476.2557
```

Fig. 9 Max path

Min path:

```
Path Group: clk1
Path Type: min

Point                                                   Incr       Path
--------------------------------------------------------------------------
clock clk1 (rise edge)                                0.0000     0.0000
clock network delay (ideal)                           0.0000     0.0000
input external delay                                  0.0500     0.0500 r
valid_in (in)                                         0.0183     0.0683 r
U1991/Y (CLKBUFX2TS)                                  0.1790     0.2474 r
U3091/Y (OAI31X1TS)                                   0.1161     0.3635 f
U2023/Y (INVX2TS)                                     0.1069     0.4704 r
U3100/Y (OAI21XLTS)                                   0.1019     0.5723 f
u_fifo_w_ptr_gray_reg_1_/D (DFFRXLTS)                 0.0000     0.5723 f
data arrival time                                                0.5723

clock clk1 (rise edge)                                0.0000     0.0000
clock network delay (ideal)                           0.0000     0.0000
clock reconvergence pessimism                         0.0000     0.0000
u_fifo_w_ptr_gray_reg_1_/CK (DFFRXLTS)                           0.0000 r
library hold time                                    -0.0390    -0.0390
data required time                                              -0.0390
--------------------------------------------------------------------------
data required time                                              -0.0390
data arrival time                                               -0.5723
--------------------------------------------------------------------------
slack (MET)                                                      0.6113
```

Fig. 10 Min path

IV. Conclusion

This project successfully designed, implemented, and verified a 64-tap 16-bit fixed-point FIR filter core with an integrated CDC FIFO for reliable operation across asynchronous clock domains. The FIR core supports signed Q1.15 input samples and coefficients, employs a Q7.30 internal data path for improved numerical precision, and produces signed Q7.9 output data suitable for low-bandwidth signal processing applications.

Overall, this project demonstrates a complete and robust FIR filter implementation, covering architectural design, fixed-point analysis, CDC handling, RTL development, synthesis, and verification. The design methodology and modular structure provide a solid foundation for future extensions, such as higher-order filters, pipelined MAC architectures, or dynamic coefficient updates.

V. Reference

[1] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing, 3rd ed. Upper Saddle River, NJ, USA: Pearson, 2010.

[2] S. K. Mitra, Digital Signal Processing: A Computer-Based Approach, 4th ed. New York, NY, USA: McGraw-Hill, 2011.

[3] U. Meyer-Baese, Digital Signal Processing with Field Programmable Gate Arrays, 4th ed. Berlin, Germany: Springer, 2014.