

# FIR Filter for Ultra Wide-Band Communications

Albert H. Chang and Nurrachman 'Rach' Chih Yeh Liu

EE241 Midterm Report, Spring 2006

*chang@berkeley.edu, rachliu@berkeley.edu*

**Abstract – Power dissipation and silicon area have become important concerns for designing in recent years. Finding the optimal power and area product for a specific throughput requirement has been a big challenge. This paper explores and compares methods for efficient energy-area-performance optimization at both circuit and micro-architectural levels on an impulse-based ultra wide-band digital filter. Impulse-based ultra wide-band communication requires high speed (around 1GHz sampling) correlation in the digital baseband. Such a high throughput makes power efficient realization of digital baseband functionality critical. Due to the complexity involved, the design will be optimized in synthesis (Synopsis) environment using a 90nm CMOS process.**

The main building block will be built in Simulink using Xilinx Block sets. The FIR filter will have 79-tap complexity with each input of 4 bits with 6 bits Pulse Matched Filter (PMF) coefficients. The techniques that we will be exploring at micro-architecture level are parallelism, pipelining and time-multiplexing. At the circuit level, we will consider voltage, device scaling and different topologies for arithmetic circuit. The final goal of the project is to have a digital FIR filter optimized for power and area.

## I. INTRODUCTION

Ultra-wideband wireless communication is a relatively new technology for short-range high-speed applications. In particular, impulse-based ultra-wideband uses a series of very narrow pulses (often less than 1 nano-second), while the bandwidth of the pulses is very large. Ultra-wideband is advantageous because of its low power, and operation in existing frequency bands without significantly interfering with other wireless communications.

A pulse-based UWB system has been designed using a digital baseband and analog sampling with sub-nanosecond spacing [1]. The system uses a digital backend. This project focuses on the power and area minimization of the main building block of the digital backend, the FIR filter, on an architectural and circuit level. We will explore the power-area tradeoffs, and find an optimal design point. Our optimization uses a synthesis environment for 90 nm process. We begin our system design with Matlab Simulink, enabling logic synthesis from block level descriptions. The synthesized logic gives quick power and area estimates, which we use to revise our Simulink design. At this stage we optimize at the micro-architectural level: parallelism, time-multiplexing, and pipelining. After designing the optimal system in Simulink, we move onto the ASIC design flow. In the ASIC design stage, at the circuit level, we optimize transistor sizing, adder topology, and voltage scaling. At this stage we also perform register re-timing using the CAD tools. We may have to go back and forth many times between Simulink and Synthesis tool to find the most optimal point in the delay-energy-area space.

## II. PREVIOUS WORK

Previous work has explored the design of low power, low-cost ultra wideband systems. In addition to the system designed by Mike Chen [1], another low power, small area UWB baseband transceiver [9] has been designed with a core size of  $0.98 \times 1.071 \text{ mm}^2$  using  $0.18 \mu\text{m}$  CMOS technology. Their power dissipation with the supply voltage varying from 1.8V to 1.2V ranges from 15.6mW to 6.7mW with operating frequency ranging from 163MHz to 65MHz. Comparing this design to [1], it has 14.8% power reduction and ten times better area. But in [9], it only has 32-taps for Pulse position detection as compared to the 128-taps in [1].

In our work, we will try to minimize power and area of the filter subject to a throughput constraint of 1GS/s. We will compare our result at the end to both previous works to evaluate potential improvements. The key is in using low supply voltage to maximize power efficiency.

## III. ULTRA WIDE-BAND FILTER

### *Filter Operation:*

The filter we are optimizing is 79 taps, with a size of 4 bits. The throughput is targeted at 1 GHz. The data bits from the ADC are input to the pulsed-matched filter (PMF) block, which essentially multiplies all the input data bits with the waveform we are detecting. We sample the encoding waveform with 64 input samples of 5-bit resolution. The PMF block generates 16 outputs by multiplying the first 64 taps with the 64 input samples of our encoded waveform; then it multiplies the next 64 samples with the encoding by shifting the input signal by one at a time. As a result, it generates 15 (79-64) more outputs. These 16 outputs are then fed into the Correlation block. The Correlation block generates an accumulated number for each of the 16 outputs from the PMF block; the accumulated number result is based on the waveform pattern specified by PN generator. The Correlation block then outputs 16 more outputs for the Peak detector, which outputs the address of the maximum value (the address is thus the offset 1-16 having maximum value), and the maximum value itself. The maximum value and its address are then passed to a threshold detector; above the threshold means the pattern was detected.

### *Filter Implementation:*

The digital system implementation is shown in Figure 1. In this project, we are optimizing the PMF (Pulse-Matched Filter), Correlation Block, and Peak Detector sections of the system.

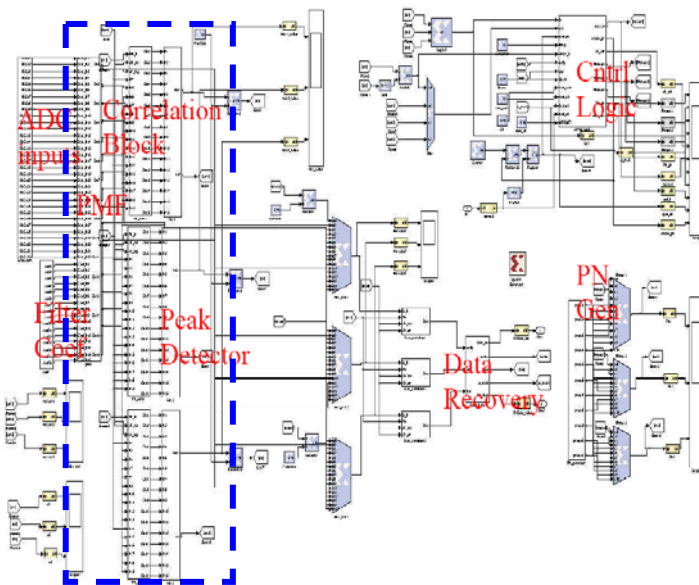


Figure 1: UWB Filter

**1. The PMF Block:**

The Pulsed-Matched Filter (PMF) performs many multiplications and additions. Since the PMF takes in 79 inputs of 4 bits and multiplies it 64 of the 79 inputs with 64 encoders of 5 bits. This will result in a 15 bit output for each of the 16 offsets.

The PMF architectural optimization lies in how the multiplications and additions are performed. One extreme is a design in which an adder is realized in a chain with addition being done serially after parallel multiplication of all 79 taps as shown in Fig 2. This design is a good reference for the architectural exploration since the design has the worst-case delay due to serial addition. It also efficiently utilizes the area by interleaving multiple data streams. Another extreme point is a design with the worst-case area, but best delay where processing is done in parallel by having 16 long add and multiply chains as in Fig 3. Even though the serial addition saves the area from not having extra adders and multipliers, but it requires extra control logics and careful timing. Between these two extremes, there is a lot of room for improvement. For example, we can implement the adder using a tree structure so we can improve our area and speed with only  $O(\log)$  stages. We also want to name the bit-width and latency of many blocks to be variables, so we can easily turn them in order to get us to the optimal point of operation.

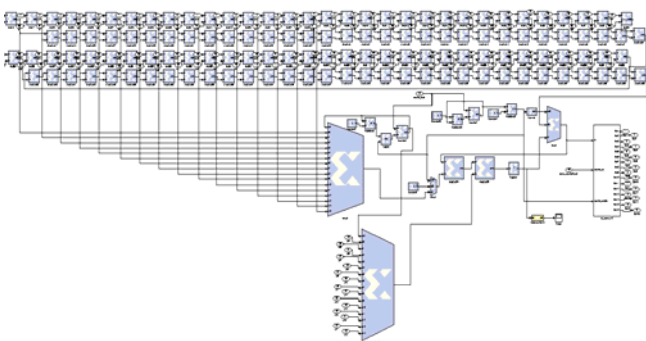


Figure 2: Serial addition

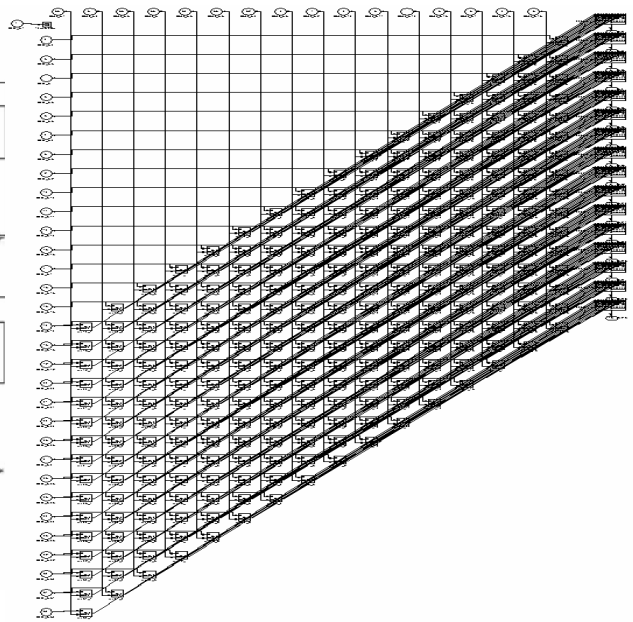


Figure 3: Parallel addition

**2. Correlation Block:**

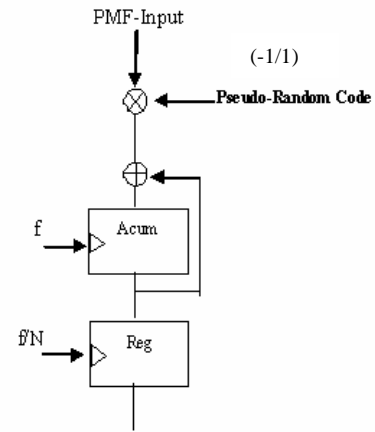


Figure 4: Correlation Block

The correlation block takes in 16 PMF inputs. Each input gets multiplied by a Pseudo-Random Code (either 1 or -1) before going into the accumulation block. The Pseudo-Random Code corresponds to the encoding pattern we have at our transceiver side. Our goal is to accumulate enough outputs from the PMF block (let's say we accumulate  $N$  samples) to enhance the process of pattern matching. Since we have to wait for  $N$  cycles before we let go of the signal into the peak-detector, the register after the accumulation block has a frequency (or symbol rate) of  $f/N$ .

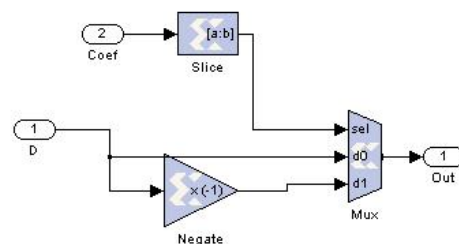


Figure 5: Bit-Flip Block

In order to implement the multiplication block, we don't want to have a full multiplier because we will only use it to multiply 1 or -1. We can simplify it by just doing a bit-flip on the inputs as shown in figure 5 above.

### 3. Peak Detector:

The peak detector accepts 16 15-bit inputs, and outputs the maximum 15-bit input value and the address of that input value. Optimization of the peak detector at the micro-architecture level means balancing the number of gates we use to do parallel comparisons versus the area/power they will consume.

#### IV. OPTIMIZATION TECHNIQUE

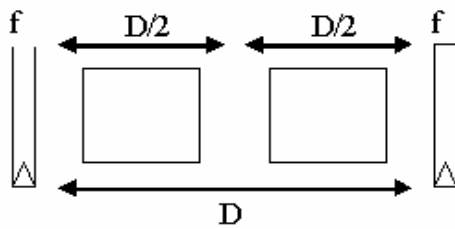


Figure 6: Reference Design, Throughput = f

With the three main blocks (PMF, Correlation, Peak Detector) of the FIR filter in hand, our plan is to optimize each block individually and then put them together to see how the optimal point is shifted on the delay-energy plot. By optimizing, we want to minimize the energy-delay product (EDP) of the circuit.

As shown below is the three optimization methods that we are mainly exploring in our project. The goal is to generate the delay-energy tradeoff plot and see where we want to operate our circuit at so that we can minimize our EDP. If the point is on the right of the optimal point, we want to shift it to the left and vice versa.

With the given throughput requirement, we know how fast the input signals are coming into our FIR filter, let's called that frequency  $f$ . Figure 6 is the reference block where we want our throughput of the new design to be the same as. The reference block has delay  $D=1/f$  between the two registers. Each logic block inside has a delay of  $D/2$ . This means we want our design to be able to take inputs with the frequency  $f$  and output it also with frequency  $f$  also with possibly some latency.

#### Case 1: Pipelining

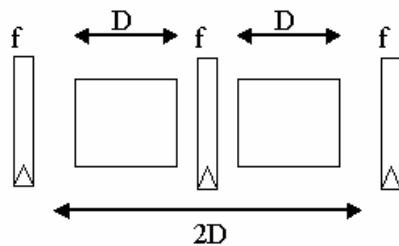


Figure 7: Pipelining with same throughput

We want the throughput to be constant, and so the clock frequency is kept at the same value  $f$ . This means we

always want to keep the delay between two registers equal to a constant  $D = 1/f$ . So through pipelining, which is inserting a register between the two logic blocks, the delay required for each logic block increases from  $D/2$  to  $D$ . Thus, we can relax the delay constraints on each logic block, resulting in decreased energy or area (the operational point moves rightwards on the energy-delay graph). Relaxing the delay is accomplished through decreasing the supply voltage (energy saving) or decreasing transistor sizing (area saving).

#### Case 2: Parallelism

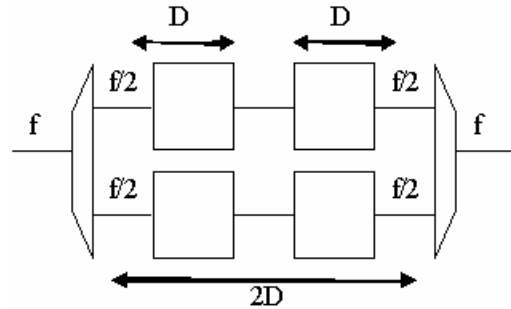


Figure 8: Parallelism with same throughput

By adding another set of duplicate hardware, we can take advantage of parallelism. The 2<sup>nd</sup> path allows double the computation, thus doubling the throughput. But if the input data comes at the same constant frequency  $f$  and the output data is read at  $f$ , then the computational paths only need to take  $2D$  instead of  $D$ . In a sense, parallelism while muxing the inputs and outputs and keeping the input and output frequency the same  $f$  is equivalent to using pipelining to double the clock frequency.

Even though pipelining and parallelism achieve the same goal: moving the operating point to the right by increasing the delay within the block, parallelism uses double the area compared to pipelining. One main reason people use parallelism is because it is hard to insert a register in the middle of two logic block once it has been designed. It takes a lot of extra and complex rewiring to re-layout the existing design. Therefore, we usually sacrifice area to make the design easier by using parallelism.

#### Case 3: Time-Multiplexing

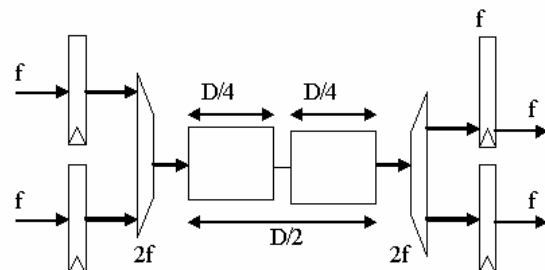


Figure 9: time-multiplexing with same throughput

In time multiplexing, the same one set of hardware has to do double the work by effectively doubling the input data frequency and output frequency. This, however, means that the total delay of the two logic blocks can only be  $T=1/(2f)=D/2$ , and so each logic block must have delay =

$D/4$ . Thus, the delay needs to be halved, and the operational point moves left on the energy-delay graph.

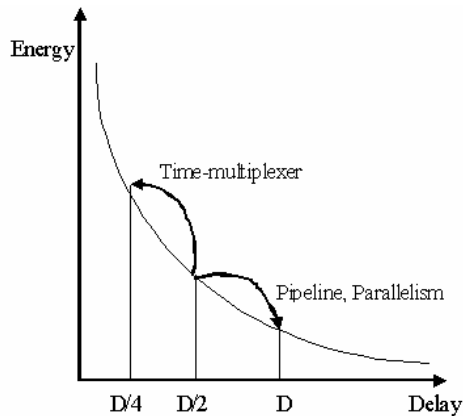


Figure 10: Energy-Delay Tradeoff Plot

With these sets of optimizing methods in mind, once we generate our delay-energy tradeoff plot for our design, we can easily move the operating point to either left or right depending on where the current operating position.

## V. DESIGN FLOW

The complexity of the project design is handled by a synthesis environment in a 90 nm technology. The steps of the design flow are as follows:

1. Matlab Simulink: use Xilinx-provided FPGA logic synthesis blocks to construct the overall system architectural design.
2. Logic synthesized from Simulink gives rough power, area, and delay estimations, using Simulink to generate hardware description (HDL) and BWRC INSECTA tool to translate the code into Synopsys dialect. If design specifications or goals are not met, go back to step 1) and re-design.
3. Map the synthesized logic into gate-level circuit descriptions using CAD tools and the standard-cell library.
4. Optimize at the circuit level, including topology choices, transistor sizing, and voltage scaling. Translate timing constraints from low voltage to an equivalent set of timing constraints at the nominal voltage used by synthesis tools.

## VI. CONCLUSION

We plan to optimize the proposed filter design, by designing the three main digital components: the pulsed-match filter, the correlation block, and the peak detector. The design involves analysing the energy-delay design space and choosing the optimal operating point for both metrics (energy and area go hand in hand). Our work will be completed in a synthesis environment for 90 nm technology, after which we will fabricate the system on a chip for further testing.

## REFERENCES

- [1] M. Chen "Ultra Wide-band Baseband Design and Implementation," *MS Thesis*, 2002
- [2] R. Blazquez, P.P. Newaskar, F.S. Lee, A.P. Chandradasan, "A Baseband Processor for Impulse Ultra-Wideband Communications," *IEEE J. Solid-state Circuits*, vol. 40, no. 9, pp. 1821-1828, Sept 2005.
- [3] K-S Kim, K. Lee, "Low-Power and Area-Efficient FIR Filter Implementations," *IEEE Trans VLSI*, vol. 11, no. 1, pp. 150-153 Feb 2003.
- [4] D. Markovic, V. Stojanovic, B. Nikolic, M.A. Horowitz, R. W. Brodersen, "Methods for True Energy-Performance Optimization," *IEEE J. Solid-State Circuits*, vol. 39, no. 8, pp. 1282-1293, Aug 2004.
- [5] T. Gemmeke, M. Gansen, H.J. Stockmanns, T.G. Noll, "Design optimization of Low-Power High Performance DSP Building Blocks," *IEEE J. Solid-State Circuits*, vol. 39, no. 7, pp. 1131-1139, July 2004.
- [6] J. Rabaey *et al*, *Digital Integrated Circuits: A Design Perspective*, 2<sup>nd</sup> edition, Prentice Hall, 2003.
- [7] Y. Yi, R. Woods, L.K. Ting, C.F. Cowan, "High Sampling Rate Retimed DLMS Filter Implementations in Virtex-II FPGA," *SIPS'02*.
- [8] Y. Yi, R. Woods, "FPGA-based System-level Design Framework based on the IRIS Synthesis Tool and System Generator," 2002.
- [9] C. Yang, K. Chen, T. Chiueh, "A 1.2V 6.7mW Impulse-Radio UWB Baseband Transceiver," in *Proc. IEEE International Solid-State Circuit Conference*, Feb, 2005.