Compact Clock Skew Scheme for FPGA based Wave-Pipelined Circuits

U. Vijayalakshmi

Communication Systems, Anna University,
Tirichirappalli, India

Abstract

FPGAs have advantage over ASICs such as rapid prototyping, in circuit programmability, lower NRE costs and results in more economical designs. In conventional pipelining technique, the operating frequency is increased by dividing the combinational logic into number of stages and registers are introduced between the stages. All the registers are fed with a global clock. This improves the speed of the system but at the cost of increased number of registers, area, latency, power and clock routing complexity. Wave-pipelining technique is an approach which improves the speed of the circuit with less area and clock loads. A compact clock skew scheme is proposed in this paper to improve the performance of the wave pipelined circuits. The proposed technique is evaluated by implementing 4 tap FIR filters using Distributed Arithmetic Algorithm (DAA) by using 3 different schemes: non-pipelining, pipelining and wave pipelining on XILINX platform. The power, area and latency comparison between the 3 different schemes are shown. It is found that wave pipelining is best when compared to area and power.

Keywords: Wave pipelining, FPGAs, DAA, FIR filter.

Introduction

In classical pipelines, registers not only increment the chip area, but also limit the minimum size of the stages, because of their setup time and propagation delay. Pipelining allows the designer to increase the throughput N times by dividing the circuit in N stages running concurrently. However, a concrete fine-grain implementation exhibits a more modest value of speedup, usually between N/2 and N/3, in spite of the technology utilized or the skill of the designers. In actual high
speed pipelines, the number of gates corresponding to extra registers is several times the original gate count of the circuit; the clock period is mainly limited by the setup and propagation delays of the registers; and the latency is significantly increased.

As a consequence, the increment of speed in a classic pipeline follows a law of diminishing returns. Each additional logic depth reduction implies that the number of registers is almost duplicated to obtain a small speedup. This in turn increases the area required and hence higher power also. Ordinary pipeline systems operate at a frequency that corresponds to the maximum logic path delay between any two stages. Theoretically, it is possible to increase this frequency by an amount of time equal to the minimum path delay that might occur in the worst-case stage.

As an alternative to pipelining, it provides a method for significantly reducing clock loads and the associated area, power and latency while retaining the external functionality and timing of a synchronous circuit.

In Section II wave pipelining is discussed. Section III discusses the feasibility of FPGA s for wave pipelining .Section IV discusses the sources of delay variation which affect the timing constraints and presents methods for minimizing their impact. Section V discusses the FIR filter using Distributed Arithmetic Algorithm used to demonstrate wavepipeling and Section VI reviews the compact clock skew scheme for all the three methods.

Wave Pipelining
The idea of wave-pipelining was originally introduced by Cotton, who named it maximum rate pipelining. Cotton observed that the rate at which logic can propagate through the circuit depends not on the longest path delay but on the difference between the longest and the shortest path delays.

As a result, several computation “waves,” i.e., logic signals related to different clock cycles, can propagate through the logic simultaneously. Testing the wave-pipelining ideas is given in [1], [3], [4]. A comparative study of existing methods in wave-pipelining can be found in [2]. Previous papers implemented on wave pipelining circuits used micro controllers, CAD and complex circuitry for adjusting the clock skews in offline condition.

Figure 1 shows a typical combinational logic circuit along with input and output registers [4]. Figure 2 depicts the data flow graph through the combinational circuit [6]. The skew between the input and output registers is denoted as δ. At the beginning of each clock cycle, data is fed into the combinational logic block through the input register. A number of paths may exist between the inputs and output of a logic block. A change in the input causes the output to change after a delay of \([D_{\text{min}} D_{\text{max}}]\) through the shortest and longest path, respectively. The shaded regions (bounded by \(D_{\text{min}}\) and \(D_{\text{max}}\)) depict the periods where the logic levels of the logic block vary with time. The nonshaded areas depict the stable duration of the logic block.
In the conventional system, the output register is clocked in the non-shaded region and the minimum clock period, $T_{clk}$, is chosen to be greater than $D_{max}$. In the wavepipelined system, the clock period is chosen to be $(Dmax - D_{min}) +$ clocking overheads such as setup time, hold time, etc. To ensure correct operation, $\delta$ should be adjusted so that the active clock edge occurs in the stable period. As the shaded region increases with increase in the logic depth, while the operating clock frequency should be reduced with increase in logic depth. Moreover, to maximize the frequency of operation of the wavepipelined system, the difference $(D_{max} - D_{min})$ is minimized by equalizing the path delays. Hence, adjustment of the clock period, clock skew ($\delta$), and equalization of path delays are the three tasks required for maximizing the operating speed of the wave-pipelined circuit. All three tasks require the delays to be measured and altered if required. Layout editors, such as the FPGA editor from Xilinx, Floor planner from Altera are used for this purpose.

The use of a single-phase clock on wavepipelining gives rise to an important drawback: a set of frequency bands where the circuit does not work. This
phenomenon occurs every time the operation mode that is, the number of waves that run together inside the circuit is changed. Moreover, each new operation band is narrower than the previous one. As a consequence, parameters that affect propagation delay such as power supply voltage, process variations, or temperature establish a limit in the practical number of waves. The intentionally skewed clock strategy [5] was proposed by Gray, Liu and Cavin as an alternative to overcome the above problem. In this scheme, the output data is registered with a new edge, that lags behind the input clock a time equal to the longest datapath delay plus the setup time. This guarantees that the results will be correctly captured: a clock edge will always follow the arrival of every wave. This method leads to more robust WP circuits as well as facilitating the multiple-stage WP synchronization. However, it violates the synchronous design principle by introducing a new clock signal.

Use of FPGA in Wavepipelining
FPGA’s exhibit several characteristics suitable for wave pipelining: 1) LUT’s hide the delay of different logic functions, and also have been designed with similar rise and fall times to improve simulation accuracy. 2) the architecture exhibits a high regularity that leads to delay equalization. 3) the knowledge a priori of each FPGA element delay (wire segments, LUT’s, and other interconnection resources) makes the equalization task possible. 4) powerful layout editors exist, and finally, 5) the fast design cycle and reprogramability of this technology allows prototypes to be built, measured, and adjusted without significant cost, since registers and clock distribution elements are included in the chips, they are “free” components and their use does not produce an area penalty, in contrast to other VLSI technologies.

The high throughput/latency ratio attainable, and the potential power reduction, considering that the equalization not only eliminates registers and clock lines, but also diminishes spurious activity. In addition, an unexpected WP effect can be produced in an FPGA if the circuit has nearly the same number of LUTs in all paths.

Timing Constraints in Wavepipelining
For a proper operation with wave-pipeline system, the system clocking is very much important designing task. It should be in such a manner that output data is clocked after the latest data arrives at the output register and before the earliest data from the next clock cycle arrives at the output register. So, first we will derive the expression for latest and earliest data available at output register. These parameters indicate the degree of wave pipelining. The data should be clocked at time TL from the rising edge of the output register N clock cycle after it has been clocked by the input register.

Considering
the possible constructive skew between the output and the input register is given by \( \Delta = \Delta_i - \Delta_o \). Then TL = NTck + \( \Delta \)
A system designer has the following design goal when designing a circuit using wave pipelining.

1. Building the fastest possible circuit that can be handled as many waves as possible. The designer will fit the rest of the circuit after $T_{\text{max}}$ and $T_{\text{ck}}$ are determined.

2. Building a circuit that has a delay $T_{\text{max}}$ determined by the design of the rest of the system. In this case desired delay $T_{\text{max}}$ is determined by external constraint. The objective is to minimize the delay between longest and shortest path ($T_{\text{max}} - T_{\text{min}}$) delay.

**FIR FILTER DAA**

![FIR Filter Diagram](image)

**Figure 4: L tap FIR filter**
The conventional single-rate FIR version of the core computes the convolution sum defined in Equation 1, where \( N \) is the number of filter coefficients. The conventional tapped delay line realization of this inner-product calculation is shown in Fig 4. Although the figure is a useful conceptualization of the computation performed by the core, the actual FPGA realization is quite different. A distributed arithmetic (DA) realization is employed. This approach employs no explicit multipliers in the design, only look-up tables (LUTs), shift registers, and a scaling accumulator.

\[
y(k) = \sum_{n=0}^{N-1} s(n) x(k-n) \quad k = 0, 1, \ldots
\]

The exponential growth in the ROM size can be avoided by splitting the \( N \) address bits to the ROM into blocks of \( K \) address bits each. Now, only \( K \) inputs DALUTs are required and hence the individual ROM size becomes \( 2K \). Totally \( N/K \) such DALUTs are required for computing the output corresponding to a particular bit of the input samples. To get the correct output, the outputs of the \( K \) input DALUTs have to be added.

In the scheme shown in Fig. 6, the minimum sampling rate or the maximum clock frequency for the input register of the DAA block is determined by the processing time in the combinational logic block consisting of the ROM and the adders. The clock frequency can be increased by introducing pipeline registers at the output of the ROM and at the output of the adders. This scheme is also referred to as synchronous pipelining. In this case, the maximum operating frequency is determined by the largest critical path delay between any of the two registers. Pipelining increases the operating frequency at the cost of increase in the number of registers, increase in routing complexity and power dissipation. The basic operations required are a sequence of table look-ups, additions, subtractions and shifts of the input data sequence.

![Figure 5: DAA FIR filter](image-url)
**Compact Clock Skew Scheme for FPGA based Wave-Pipelined Circuits**

**Inner-product computation**
The TSB is itself constructed using a cascade of shorter bit–serial shift registers. The nodes in the cascade connection of TSBs are used as address inputs to a look-up table. This LUT stores all possible partial products over the filter coefficient space.

In a conventional multiply-accumulate (MAC)-based FIR realization, the sample throughput is coupled to the filter length. With a DA architecture, the system sample rate is related to the bit precision of the input data samples. Each bit of an input sample must be indexed and processed in turn before a new output sample is available. For $B$-bit precision input samples, $B$ clock cycles are required to form a new output sample for a nonsymmetrical filter, and $B+1$ clock cycles are needed for a symmetrical filter. The rate at which data bits are indexed occurs at the *bit-clock* rate. The bit-clock frequency is greater than the filter sample rate ($f_s$) and is equal to $Bf_s$ for a nonsymmetrical filter and $(B+1)f_s$ for a symmetrical filter. In a conventional instruction-set (processor) approach to the problem, the required number of multiply-accumulate operations are implemented using a time-shared or *scheduled* MAC unit. As the filter length is increased, the system sample rate is proportionately decreased. This is not the case with DA-based architectures. The filter sample rate is decoupled from the filter length.

**Clock Skew Scheme**

![Control circuit](image)

*Figure 6: Control circuit*

![Timing diagram](image)

*Figure 7: Timing diagram*
Algorithm for Control Logic Design
Steps:
1. Tin = 0
2. Lclose = Tin + Dmin
3. Lopen = Tin + Dmax
4. Tin = Tin + Tclk
5. Goto step 2

Where, Tin is the instant at which input data arrives; Tclk defines the clock period. Lopen is the instant at which output latch is opened and Lclose is the instant at which output latch is closed. Dmin and the Dmax are minimum and maximum delay through the combinational block. The proposed control circuit for the online clock skew scheme consists of one T flip-flop driven by common clock ‘clk’ and T input is always made high as shown in Fig. 6 so that T flip flop toggles for every clock.

Assuming that a combinational block is having maximum delay of Dmax and minimum delay of Dmin, the outputs of flip-flops are given to the delay blocks- one to a delay block representing Dmax and another to delay block representing Dmin. The outputs of both delay blocks are XORed to generate the enable signal ‘en’, which is ideally the difference of Dmax and Dmin. This enable signal is given to the output latch of the combinational block (Fig.6) as a control signal to open/close the latch. Initially the Q1, Q3 and Q4 are ‘0’. The ‘en’ signal is ‘0’. After clk is applied, T flip-flop toggle and Q1 and Q3 become ‘1’. After delay of Dmin, Q3 becomes high making ‘en’ high. The latch is closed (opaque) and does not allow the data to pass through it. After delay of Dmax, Q4 becomes high in making ‘en’ low. The latch is transparent allowing the data to pass through it. This process continues for every input data and the latch is open between Dmax and Dmin which safely latching of data during stable period.

Table 1: Area and speed comparison

<table>
<thead>
<tr>
<th></th>
<th>Frequency Mhz</th>
<th>Logic Utilisation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non pipelined</td>
<td>73.63</td>
<td>114</td>
</tr>
<tr>
<td>Pipelined</td>
<td>173.37</td>
<td>150</td>
</tr>
<tr>
<td>Wave pipelined</td>
<td>175.37</td>
<td>136</td>
</tr>
</tbody>
</table>

Speed Area comparison
Table 2: Power dissipation

<table>
<thead>
<tr>
<th></th>
<th>Power dissipation mw</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non pipelined</td>
<td>59</td>
</tr>
<tr>
<td>Pipelined</td>
<td>60</td>
</tr>
<tr>
<td>Wave pipelined</td>
<td>56.05</td>
</tr>
</tbody>
</table>

Conclusion

The experiments presented in this work are focused on single-stage WPs implemented on XC3S50 Spartan 3 FPGAs. In this case, the longest path is not modified; just the delay of the shorter ones are increased to achieve the balance. Thus, the minimum latency characteristic of the WP technique can be fully exploited. Considering that WP is just another way of trading speed for additional area and design time as well as exhibiting lower reliability than conventional Pipelines. The wave-pipelined DA filters implemented with the proposed control circuit for online clock skew can operate at higher frequency than that of non-pipelined but less than that of pipelined. The gain in speed in pipelined compared to that of wave-pipelined is at the cost of increased area. The power dissipation was smaller for wave pipelining than pipeline and non pipeline. Dmax and Dmin reported by the Xilinx Timing Analyzer tool 12.248 ns and 6.216 ns respectively. This was used in implementing the circuit and the results are obtained.

References

