COMPARATIVE STUDIES OF PIPELINED CIRCUITS

Fabian Klass and Michael J. Flynn

Technical Report No. CSL-TR-93-579

July 1993

This work was performed with support from the Delft University and Stanford University using facilities provided under NSF Contract No. MIP88-22961.

COMPARATIVE STUDIES OF PIPELINED CIRCUITS

by

Fabian Klass and Michael J. Flynn

Technical Report No. CSL-TR-93-579

July 1993

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science

Stanford University

Stanford, California 94305-4055

Abstract

Wave pipelining is an attractive technique used in high-speed computer systems to speed-up pipeline rate without partitioning a system into pipeline stages. Although recent implementations have reported very high-speed operation rates, a real evaluation of the advantages and disadvantages of wave pipelining requires a comparative study with other techniques, in particular the understanding of the trade-offs between conventional and wave pipelining is very important. This study is an attempt to provide approximate models which can be used as first-order tools for comparative study or sensitivity analysis of conventional and wave pipelined systems with different overheads. The models presented here are for subsystem-level pipelines. The product Latency $\times$ Cycle-Time is used as a measure of performance and is evaluated as a function of all the parameters of a design, such as the propagation delay of the combinational logic, the data skew resulting from the difference between maximum and minimum propagation delays through various logic paths, rise and fall time, the setup time, hold time, and propagation delay through registers, and the uncontrollable clock skew. In this way, an analytical basis is provided for a comparison between different approaches and for optimizations.

Key Words and Phrases: Wave pipelining, conventional pipelining, optimal pipelining, cycle time, latency, delay equalization

Contents

List of Figures

List of Tables

1 Introduction

Pipelining is one of the most attractive and widely used design techniques in high-speed computer systems as it offers a potential speed-up of n when n pipeline stages are used. However, as new applications require the achievement of extremely short cycle times – close to the physical limit of the technology – the potential performance increase of conventional pipelining is substantially reduced due to the practical limitations and overheads resulting from partitioning a system into several pipeline stages. One alternative technique, which avoids the partition of the system, is the so called technique of wave pipelining. Although proposed initially time ago [1], wave pipelining has received recently considerable attention from several research groups. Fig. 1 illustrates how the two pipelining approaches work. While conventional (or regular) pipelining requires the insertion of clocked elements to increase the clock rate of the system, wave pipelining achieves the same goal by clocking the system faster than its propagation delay, thus effectively achieving pipelining, but without the need of partitioning the system to insert clocked elements. If the number of sets of data – waves – between registers at any time instance is n, a speed-up of n is achieved with wave pipelining.

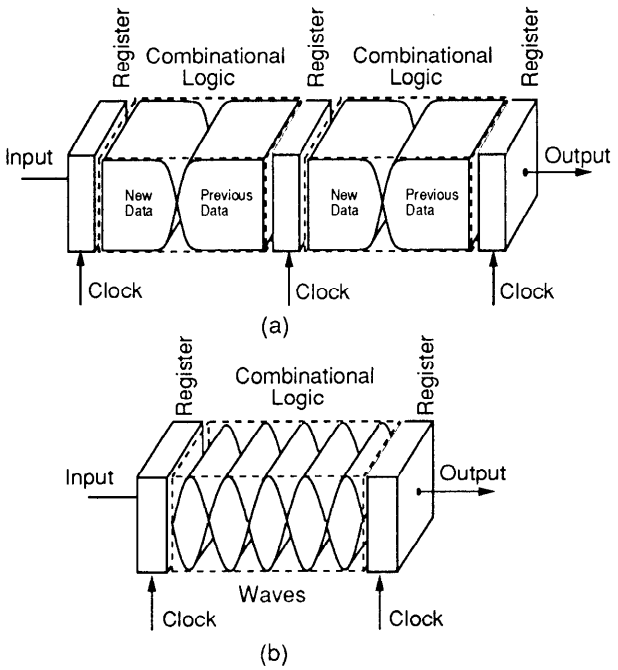

Figure 1: Pipelining techniques: (a) Regular pipelining and (b) Wave pipelining

Despite the inherent difficulties encountered in the design of wave pipelined circuits, a number of applications were implemented in bipolar and CMOS technologies [2], [3], [4], [5], [6]. Speed-up factors between 2-7 were reported in different circuits. Although performance numbers are encouraging, a real evaluation of the advantages and disadvantages of wave pipelining requires a comparative study with other techniques, in

particular the understanding of the trade-offs between conventional and wave pipelining is very important. For instance, is it possible to achieve the same high performance as reported for wave pipelining using conventional techniques? What would be the cost in this case? Is some degree of wave pipelining always convenient? What is the optimum combination of the two techniques?

This study is an attempt to answer some of the questions above by providing approximate models which can be used as first-order tools for comparative study or sensitivity analysis of conventional and wave pipelined systems with different overheads. The models presented here are for subsystem-level pipelines. The product Latency $\times$ Cycle-Time, as will be discussed below, is used as a measure of performance and is evaluated as a function of all the parameters of a design, such as the propagation delay of the combinational logic, the data skew resulting from the difference between maximum and minimum propagation delays through various logic paths, rise and fall time, the setup time, hold time, and propagation delay through registers, and the uncontrollable clock skew. In this way, an analytical basis is provided for a comparison between different approaches and for optimizations.

The rest of this work is organized as follows. In Section 2, the effect of the partitioning overhead is discussed. In Sections 4 and 5, Latency $\times$ Cycle-Time is evaluated using edge-triggered registers for regular and wave pipelining, respectively, and examples are provided of pipelined systems with different overheads. In Section 6, the two approaches are compared and an upper bound is found for the difference in path delay which guarantee a better performance for wave pipelining. In Section 7, Latency $\times$ Cycle-Time is evaluated using edge-triggered registers for optimal pipelining, a technique which combines both regular and wave pipelining. In Section 8, optimal pipelining is compared against regular and wave pipelining only. It is shown that optimal pipelining has always a better performance. In Section 9, the analysis from Section 2 to Section 8 is extended for transparent latches. In Section 10, a practical application of wave pipelining for 16$\times$16 multiplication is described and compared with a conventional implementation. In Section 11, a summary is provided.

2 Partitioning Overhead

As mentioned above, the major factor limiting performance in conventional pipelining when the goal consists of achieving the shortest possible cycle time, besides the area penalties due to the registers (or latches), is the partition overhead. This overhead has two main components: (a) register (or latch) overhead, due to the setup and hold time of registers, propagation delay through registers, and uncontrollable clock skew between different stages of the pipeline, and (b) quantization overhead, due to the fact that a pipeline stage consists of an integer number of gate levels and hence

the propagation delay is quantized. The register overhead limits the minimum clock period that can be achieved, while the quantization overhead increases the latency.

The following study presents an example of conventional pipelining where the clocking overhead becomes the dominating factor limiting the performance of a system and reducing its cost-effectiveness (the computation of the minimum clock period in this study will be explained in Section 3).

Study 1: Suppose a circuit has a logic depth of 10 gates. The delay of a gate is 0.3ns, and the total overhead due to the delay of registers, setup time, and clock skew is 0.8ns. It is required to speed up the circuit as much as possible using a conventional pipelining technique. If the pipeline is built as shown in Fig. 2, with only one gate level per stage, the minimum achievable clock period is $0.3ns + 0.8ns = 1.1ns$, and the latency is $10 \times 1.1ns = 11ns$. If no pipelining is used, the minimum clock period is $10 \times 0.3ns + 0.8s = 3.8ns$. The pipeline scheme achieves a speed-up of 3.4.

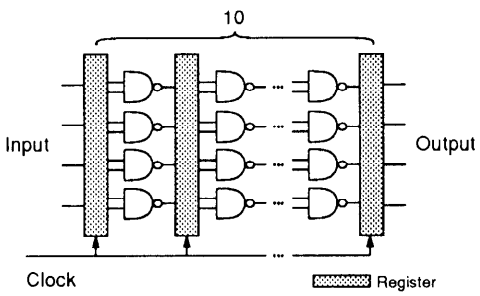

Figure 2: Fine-grain pipelined circuit.

If performance is measured as pipeline clock rate only, the scheme shown in Fig. 2 is probably the fastest possible solution. However, if other important factors such as latency increase, area penalties, and power consumption are included in the evaluation of performance, the efficiency of this approach is very poor. Note that despite 10 segments are used, the speed-up is only 3.4. Furthermore, the total latency is increased 2.9×, and the area about 5× (assuming the area required by a one-bit register is 4× the area of a gate). Also power would be increased proportionally to the area growth. Rather than pipelining, a more efficient approach in this case would be three logic units working in parallel, without pipelining at all. This scheme would provide a 3× speed-up, with relatively minimum latency penalties, and only 3× increase in area and power.

The reason behind the low efficiency of the scheme proposed in Study 1 is because the delay of one stage in the pipeline is less than the total clocking overhead. In cases like this, wave pipelining seems to be a better choice. Since no clocked elements are required, the cycle time can be decreased while the latency remains practically the same, minimizing in this way the area penalties and power consumption increase incurred

in conventional systems. However, the reduction in cycle time for wave pipelining is limited by the difference between maximum and minimum delay. Although techniques were developed for the equalization of path delays [7], [8], differences cannot be completely eliminated in practice. Circuit structures, data-dependencies, process variation, and temperature changes are among the factors which prevent the achievement of a perfect balance. If the difference in delay cannot be reduced to small values, the potential advantages of wave pipelining over conventional pipelining are limited.

The study above suggests that a better way of characterizing the performance of a pipeline system is the product Latency $\times$ Cycle-Time. This metric, rather than the cycle time only, can provide an indication of how the latency of a circuit is affected by pipelining. This is of particular concern if a design is aimed to achieve a cycle time in the order of a few gate delays. Latency $\times$ Cycle-Time can also be correlated, although indirectly, to the effects of pipelining on area increase and power consumption. In the following sections, Latency $\times$ Cycle-Time is evaluated for different pipelining techniques.

3 Definitions

- T Propagation delay of the longest path in the combinational logic
- t_S Setup time for the registers
- t_H Hold time for the registers
- t_D Propagation delay for the registers
- t_{RF} Worst-case rise or fall time in the logic
- ΔT Difference between maximum and minimum path delay
- ΔC Worst-case clock skew
- t_{CK} Clock period

4 Regular Pipelining Using Edge-Triggered Registers

The product Latency $\times$ Cycle-Time is first evaluated for regular pipelining. The following analysis is for single-phase clocking and edge-triggered registers.

Suppose a pipeline is constructed by dividing a combinational logic circuit with initial delay T into n stages, where each stage has delay $T_n = T/n$, as shown in Fig 3.

Assume that the segmentation of the circuit does not add overhead by itself, i.e., all stages have equal delay T_n (zero quantization overhead).

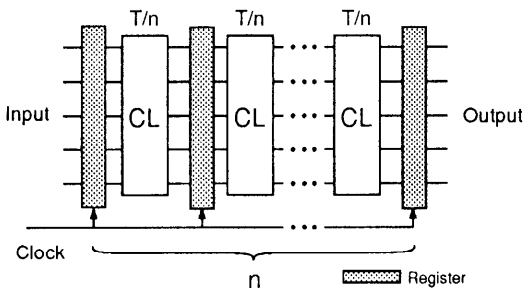

Figure 3: A regular pipelined circuit with n stages

The total latency of the system, denoted as L, can be formulated as:

$$L = n \times t_{CK} \tag{1}$$

where t_{CK} is the clock period at which the pipeline is clocked.

The product Latency $\times$ Cycle-Time, denoted as LC, can be computed as:

$$LC = n \times {t_{CK}}^2 \tag{2}$$

It is required to find the value of n that minimizes the product Latency $\times$ Cycle-Time, under the assumption that for any given n the minimum clock period is achieved.

The clock period in a regular pipelined system must be such that data are stored into the output register only after all data have arrived. This timing constraint can be formulated as:

$$t_{CK} > T_{max} \tag{3}$$

where $T_{max} = T_n + t_D + t_{RF} + t_S + \Delta C$, and T_n is the delay of one stage of the pipeline previously defined as T/n.

A timing diagram of the clocking scheme is shown in Fig. 4 including all the circuit parameters. Note that the definition of T_{max} includes all the overheads due to the clock. Although parameter t_{RF} is determined by the logic and not by the registers, it is considered part of the clock overhead, since a signal transition must become fully steady before it can be stored, which results in some overhead.

According to constraint (3), the minimum clock period is achieved when $t_{CK} = T_{max}$. For notation convenience, all the overheads resulting from the partition of the logic

are grouped into a single parameter H, referred to as the clocking overhead, which is defined as $H = t_D + t_{RF} + t_S + \Delta C$. In this way, the minimum clock period is expressed as:

$$\min\{t_{CK}\} = T_n + H \tag{4}$$

The use of a single paramcter H not only contributes to make the notation more concise but also shows in better way the effect of the total overhead in the performance of the system. For the rest of this study, parameter H is assumed to be constant independently of the number of pipeline stages into which the logic is partitioned. This and the assumption above that the quantization overhead is zero provide a simplified model which is used as a first-order approximation in order to find an optimum number of pipeline segments such that the product Latency $\times$ Cycle-Time is minimized. The result will be compared against the performance of wave pipelining, as will be discussed in Section 5.

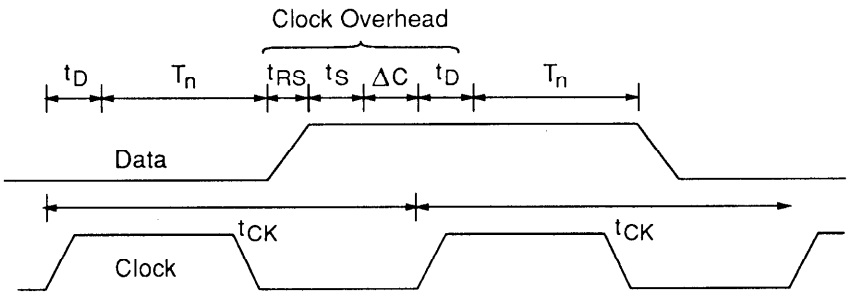

Figure 4: Timing diagram for regular pipelining using edge-triggered registers

Combining Eqs. (4) and (2) and replacing T_n with T/n, the final expression for LC is obtained:

$$LC = \frac{(T + nH)^2}{n} \tag{5}$$

Now, the value of n which minimizes function LC can be computed by solving $\partial LC/\partial n = 0$ for n. The minimum LC is found for:

$$n_o = T/H \tag{6}$$

Since n must be integer, a possible heuristic is to pick $n = \lceil n_o \rceil$, if $(n_o - \lfloor n_o \rfloor) \geq 1/2$, or $n = \lfloor n_o \rfloor$ otherwise. Thus, if n_o is replaced in Eq. (5), the function LC is bounded by:

$$LC > 4\ HT \tag{7}$$

The maximum number of stages into which the logic can be divided is limited by the race-through constraint [7]. Assuming that the delay of the shortest path of a segment can be expressed as $T/n - \delta T/n$, where $0 < \delta < 1$, it can be shown that the maximum n is given by:

$$n_{max} = (1 - \delta)T/G, \tag{8}$$

where G is defined as $G = t_h + \Delta C - t_d$.

Depending upon the value of δ, partitioning the system into n_o segments might be feasible or not. In any case, constraint (7) holds. In the following study, $n_{max} \gg n_o$ is assumed.

Study 2: Suppose a pipelined system can be built with three types of registers. The delay of the logic is $T = 10ns$, and depending upon the type of register used, the clock overhead is $H = 1ns$, $H = 1.5ns$, and $H = 2ns$. It is required to compute the value of n which minimize the product Latency $\times$ Cycle-Time for each different type. The values of n are computed using Eq. (6), which yields $n = 10$, $n = 7$, and $n = 5$, respectively. Fig. 5 plots LC as a function of n in each case.

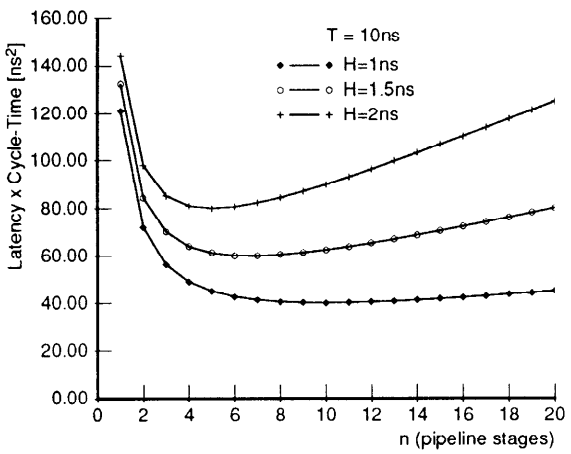

Figure 5: Latency $\times$ Cycle-Time as a function of the number of pipeline stages

The analysis above shows that independent of how the logic is partitioned in a conventional pipelined system, the product Latency $\times$ Cycle-Time has a lower bound provided by constraint (7). For values of n beyond the minimum LC, although

shorter cycle times could be achieved, the latency of the system would be increased substantially due to the preponderant effect of the clock overhead.

5 Wave Pipelining Using Edge-Triggered Registers

The product Latency × Cycle-Time is now evaluated for wave pipelining. As in the case of regular pipelining, Latency × Cycle-Time is evaluated for single-phase clocking and edge-triggered registers. The clock at the input and output registers is the same, as shown in Fig. 6. In this way, a fair comparison can be established between regular and wave pipelining, since both techniques use the same clocking scheme.

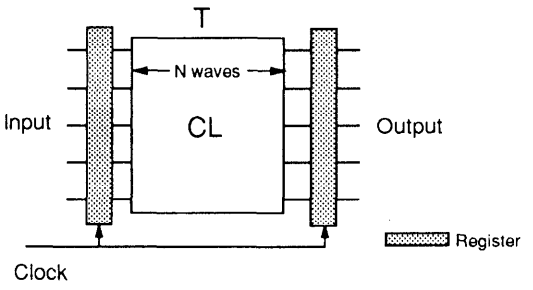

Figure 6: A wave pipelined circuit with N effective stages

The product Latency × Cycle-Time in this case can be formulated as:

$$LC = N \times {t_{CK}}^2 \tag{9}$$

where t_{CK} is a valid clock period and N is the number of effective pipeline stages of the circuit. The number N can be physically interpreted as the number of waves in the system at any time instance.

It is required to find the number of effective pipeline stages, or waves, and the corresponding clock period, which minimizes the product Latency × Cycle-Time. A timing analysis for wave pipelining is presented below.

In contrast to conventional pipelining, double sided timing requirements must be met in this case in order to clock safely a circuit. The clock period must be such that the data emerging from the last stage of the logic are stored into the output register only after the latest data of the current wave has arrived and before the earliest data from the coming wave arrives. Assuming that $N \times t_{CK}$ is the time elapsed between the occurrence of the clock at the input and the occurrence of the clock at the output

which stores the wave after it has propagated through the logic, then the double timing constraint can be formulated as:

$$T_{max} < N \times t_{CK} < T_{min} + t_{CK} \tag{10}$$

where $T_{max} = T + t_D + t_{RF} + t_S + \Delta C$, $T_{min} = T + t_D - t_H - \Delta C - \Delta T$, and N is a positive integer number. Like before, the definition of T_{max} and T_{min} includes all the overheads due to the clock. Fig. 7 shows a timing diagram in this case.

The time interval $[T_{max},\ T_{min} + t_{CK}]$ is called the valid clock interval, because the occurrence of the leading edge of the clock, without accounting for the clock skew, must fall within this interval in order to store safely the data emerging from the pipeline into the output register (see Fig. 7). The product $N \times t_{CK}$ is defined as the latency of the circuit.

Constraint (10) can be formulated in a different way as:

$$\frac{T_{max}}{N} < t_{CK} < \frac{T_{min}}{N-1} \tag{11}$$

If $N = 1$, as in conventional pipelining, the expression above becomes $T_{max} < t_{CK} < \infty$ [9]. Consequently, constraint (10) is meaningful for wave pipelining as long as $N > 1$.

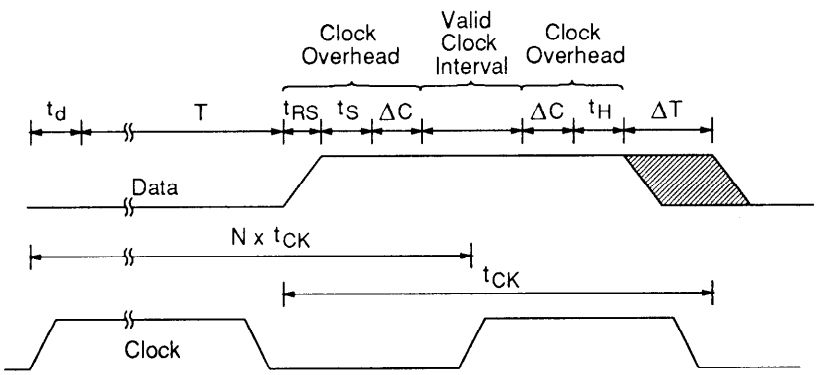

Figure 7: Timing diagram for wave pipelining using edge-triggered registers

Constraint (10) can be used to compute a lower bound on the clock period. This lower bound can be derived from the condition that the length of the valid interval is null, that is $T_{max} = T_{min} + t_{CK}$. The result is given by:

$$\min\{t_{CK}\} = \Delta T + H' \tag{12}$$

where H' is the total clocking overhead for wave pipelining defined as $H' = t_{RS} + t_S + t_H + 2 \times \Delta C$ (see Fig. 7).

The lower bound given by Eq. (12) guarantees that a wave does not interfere with the next wave at the end of the combinational logic. Additional constraints on the clock period can be derived from the requirement that waves do not interfere inside a section of the combinational logic [7], [10]. In this study, the constraint at the output register is assumed to prevail over the constraints inside the logic. Therefore, the minimum clock period is given by Eq. (12). The analysis can be generalized however if internal constraints prevail, as discussed below.

The number of waves in the pipeline is related to the clock period as expressed by constraint (10). Suppose now that the leading edge of the clock can be adjusted to be exactly in the middle of the valid interval. Then, the latency of the wave-pipelined circuit can be computed as $L = (T_{max} + T_{min} + t_{CK})/2$, which yields:

$$L = T + t_D + (t_{RF} + t_S - t_H + t_{CK} - \Delta T)/2 \tag{13}$$

For notation convenience, Eq. (13) is rewritten using parameters H and H' defined above in the following way:

$$L = T + H - H'/2 + t_{CK}/2 - \Delta T/2 \tag{14}$$

Now, replacing L with $N \times t_{CK}$ in Eq. (14) gives:

$$N \times t_{CK} = T + H - H'/2 + t_{CK}/2 - \Delta T/2 \tag{15}$$

Eq. (15) can be used to compute N, given a valid clock period t_{CK}, or conversely, to compute t_{CK} given a valid number of waves N. If N is given, t_{CK} can be computed as:

$$t_{CK} = \frac{T - \Delta T/2 + H - H'/2}{N - 1/2} \tag{16}$$

Finally, combining Eqs. (16) and (9), the product Latency $\times$ Cycle-Time for wave pipelining can be formulated as:

$$LC = [T - \Delta T/2 + H - H'/2]^2 \frac{N}{(N - 1/2)^2} \tag{17}$$

Since function LC above decreases as N increases, the minimum Latency $\times$ Cycle-Time is achieved when the number of waves is maximum. The maximum N can be derived from Eq. (16), considering that the clock period should be greater than a minimum:

$$\frac{T - \Delta T/2 + H - H'/2}{\max\{N\} - 1/2} > \min\{t_{CK}\} \tag{18}$$

Then, $\max\{N\}$ can be computed as:

$$\max\{N\} = \left\lfloor 1/2 + \frac{T - \Delta T/2 + H - H'/2}{\min\{t_{CK}\}} \right\rfloor \tag{19}$$

The minimum clock period given by Eq. (12) can be used to computed $\max\{N\}$ above. If the minimum clock period is limited by constraints inside the logic, an expression similar to Eq. (12) can be used where parameters ΔT_i is defined as the difference between maximum and minimum delay for an internal node of the circuit, and $H'_i = t_{RS} + t_{ST} + \Delta C$, where t_{ST} is defined as the minimum time that the signal at the internal node must be stable [7].

Study 3: Suppose a system has parameters $T = 11.5ns$, $H = H' = 1ns$, and $\Delta T = 2ns$. It is required to compute the best performance of the system using wave pipelining. According to Eq. (12), the minimum clock period is $2ns + 1ns = 3ns$. The maximum number of waves, computed with Eq. (19), is 4. The valid clock period is computed using Eq. (16), which yields $3.14ns$. The latency is $4 \times 3.14ns = 12.57ns$, and the product Latency $\times$ Cycle-Time is $39.5ns^2$.

Notice that the maximum value of N which minimizes the product Latency $\times$ Cycle-Time also minimizes the clock period, thus maximizing the pipeline rate of the system.

6 Wave Pipelining Versus Regular Pipelining

One of the problems of using conventional pipeline techniques when a very short cycle time is required in a design is that the latency of the system is substantially increased due to the overhead introduced by clocked elements. Instead, the use of wave pipelining can decrease the cycle time while the latency of the system remains almost unchanged – an attractive advantage. However, the reduction in cycle time is limited by the difference between maximum and minimum path delay. If the difference in delay cannot be reduced to a small fraction of the total delay due either to the practical

limitations of the balancing techniques being used or to the uncontrollability of some sources of delay variation, such as data-dependent delay and process deviation, the number of waves that a system can support are substantially reduced, blurring the the potential advantages of wave pipelining over conventional systems.

Assuming that a system must be designed to achieve a very short cycle time, the use of wave pipelining is justified depending upon the parameters of the system. In particular, the difference in path delay must be small enough to allow a minimum number of waves propagate in the combinational logic. If the conditions for wave pipelining are not favorable, the use of a conventional approach, or a combination of both, should be considered alternatively.

In order to evaluate the most suitable technique for a particular design, it is required to find a relation between the parameters of a system which guarantee that the performance achieved with wave pipelining is better than in regular pipelining. Like in previous sections, the product Latency $\times$ Cycle-Time is used as a measure of performance.

Problem Formulation: For a generic system, find a relation between its parameters to satisfy the constraint:

$$LC(wave\ pipelining) < \min\{LC(regular\ pipelining)\} \tag{20}$$

A relation which is necessary but not sufficient is derived as follows. First, the maximum clock period which satisfies constraint (20) is determined. Then, the maximum allowed ΔT required to achieve this clock period is computed.

Instead of using Eq. (17) for the Latency $\times$ Cycle-Time of wave pipelining, the following expression, which has t_{CK} explicited, is preferred:

$$LC = (T + H - H'/2 + t_{CK}/2 - \Delta T/2) \times t_{CK} \tag{21}$$

Then, constraint (20) can be formulated as:

$$(T + H - H'/2 + t_{CK}/2 - \Delta T/2) \times t_{CK} < 4\,HT \tag{22}$$

Solving constraint (22) for t_{CK} gives:

$$t_{CK} < \sqrt{(T - \Delta T/2 + H - H'/2)^2 + 8\,HT} - (T - \Delta T/2 + H - H'/2) \tag{23}$$

Therefore, if a system can be wave-pipelined with a clock that satisfies constraint (23),

the performance achieved is guaranteed to be better than the best regular implementation.

According to Eq. (12), the clock period is bounded by the following constraint:

$$\Delta T + H' < t_{CK} \tag{24}$$

Combining constraints (23) and (24), a necessary condition to satisfy initial constraint (20) can be found. As a result, the maximum ΔT is bounded by:

$$\Delta T < \frac{4\,H}{1 + H/T} - H' \tag{25}$$

Notice that constraint (25) is a necessary condition only to satisfy constraint (20). However, if the valid clock period is close to the minimum, which is desired in practice, it can be considered a sufficient condition as well. Besides, constraint (25) was derived assuming that the product Latency $\times$ Cycle-Time in the case of regular pipelining is equal to $4\,HT$, which is a lower bound. The fact that n must be integer in Eq. (2) prevents regular pipelining from achieving this minimum. Furthermore, breaking down a logic circuit into small pieces all with equal delay is difficult in practice due to quantization problems. If the slowest segment is used in the evaluation of the product Latency $\times$ Cycle-Time, a result less close to the theoretical minimum of $4\,HT$ is achieved.

Finally, assuming $T \gg H$ and $H' = H$, constraint (25) can be approximately formulated as:

$$\Delta T < 3\,H \tag{26}$$

Study 4: Suppose a system has parameters $T = 9.5ns$, $H = 1ns$, and $\Delta T = 3ns$ (assume $H' = H$). It is required to determine whether wave pipelining or regular pipelining is the best approach to increase the pipeline rate of the system. Since $\Delta T = 3\,H$, according to constraint (26), the two approaches would achieve about the same performance. For instance, a wave-pipelined implementation would support only 2 waves and achieve a clock period of $5.66ns$, yielding a latency of $2 \times 5.66ns = 11.33ns$. Instead, a regular pipeline with two stages could be clocked at $6ns$, with a latency of $2 \times 6ns = 12ns$. And a regular pipeline with three stages could be clocked at $4.33ns$, with a latency of $3 \times 4.33ns = 13ns$. The ratio $LC(w.p.)/LC(r.p.)_{2-stage}$ is $64.22/72 = 0.89$, while the ratio $LC(w.p.)/LC(r.p.)_{3-stage}$ is $64.22/56.33 = 1.14$. Notice that the 3-stage regular pipeline performs better than the wave-pipelined design, although it

requires two internal registers.

Another useful expression for ΔT can be derived from the condition $LC(w.p.) = 1/2 \min\{LC(r.p.)\}$. This condition means that the latency of a regular pipelined circuit will be increased at least by a factor of 2 with respect to wave pipelining if both circuits were designed to achieve the same cycle time. The solution in this case is given by:

$$\Delta T < \frac{2\,H}{1 + H/T} - H' \tag{27}$$

which can be formulated approximately, under the same assumptions $T \gg H$ and $H' = H$, as:

$$\Delta T < H \tag{28}$$

The condition $LC(w.p.) = 1/2 \min\{LC(r.p.)\}$ can be used as a heuristic for the design of optimal pipelined systems, as will be discussed in Section 8. This requires that the difference in path delay must be less than the clock overhead of the system, as given by constraint (28).

Study 5: Suppose the delay variation in Study 4 is reduced from $3ns$ to $1ns$ through improved balancing techniques. It is required to evaluate the performance of both techniques is this case. According to constraint (28), wave pipelining can achieve a performance about $2\times$ better than the best regular implementation, since $\Delta T = H$. For instance, a wave-pipelined circuit would support 5 waves at $2.11ns$, and have a latency of $5 \times 2.11ns = 10.55ns$. Instead, a regular pipeline with 10 stages, where each stage has $1ns$, would achieve a clock cycle of $2ns$, and a latency of $10 \times 2ns = 20ns$. The ratio $LC(w.p.)/LC(r.p.)$ in this case is $22.3/40 = 0.56$, which reflects the better performance of wave pipelining.

6.1 Wave Pipelining Versus Regular Pipelining Assuming Latency Increase

In Section 6, it was assumed that techniques used in wave pipelining do not affect the maximum propagation delay of the combinational logic. In some cases, however, although the techniques used for balancing delays should ideally increase the length of short paths only, they might slow down the longest path delay as well. In other cases, the choice of a circuit structure which is best suitable for wave pipelining might result in longer delays. For instance, a circuit whose structure has inherently balanced delays – well suited for wave pipelining – might be slower than a circuit which is not designed

for wave pipelined applications, i.e., a circuit with no minimum delay constraints. If the latency of a wave-pipelined system is increased due to design constraints required for equalizing delays, this effect must be taken into account when the performance of wave pipelining is evaluated and compared with regular pipelining.

Assuming that the maximum delay of a wave-pipelined circuit is increased to αT, where T is the delay for non pipelining, and $\alpha > 1$ is the factor of latency increase due to wave pipelining, in order to satisfy constraint (20), it is required that:

$$(\alpha T + H - H'/2 + t_{CK}/2 - \Delta T/2) \times t_{CK} < 4\,HT \tag{29}$$

The maximum ΔT, which is derived in the same way as in Section 6, is now bounded by the following expression:

$$\Delta T < \frac{4\,H/\alpha}{1 + H/\alpha T} - H' \tag{30}$$

And the approximate relation, assuming $\alpha T \gg H$ and $H' = H$, is given by:

$$\Delta T < (4 - \alpha)H/\alpha \tag{31}$$

Notice that constraint (31) becomes tighter as α increases. For instance, if $\alpha = 2$ (100% latency increase due to wave pipelining), $\Delta T < H$ is required to achieve a performance similar to regular pipelining. If $\alpha = 1$, constraints (30) and (27) are equivalent.

Study 6: Suppose a system has initial parameters $T = 10ns$, $H = H' = 1ns$, and $\Delta T = 4ns$. After balancing the circuit for wave pipelining, the difference in path delay is reduced to $1ns$, but the latency is increased by 15% ($\alpha = 1.15$). It is required to compare the performance of the two techniques. Since $T = 11.5ns$ for wave pipelining, the system could work with 6 waves at $2.1ns$, resulting a latency of 12.54ns. If the regular pipeline is built with the initial circuit ($T = 10ns$), the clock cycle would be $2ns$, and the latency $20ns$ (as in Study 5). The ratio $LC(w.p.)/LC(r.p.)$ is now $26.2/40 = 0.66$, which shows a performance degradation for wave pipelining compared with the value of 0.56 obtained in Study 5, as a consequence of the 15% increase in the latency of the circuit.

7 Optimal Pipelining

As discussed in Section 6, the difference between maximum and minimum delay through various path in the logic has to be approximately less than three times the clocking overhead of the system in order to achieve a good performance with wave pipelining. However, as the length of a combinational logic increases, the difference in path delay increases proportionally due to the accumulative effect of delay variations which cannot be completely eliminated by balancing techniques. Therefore, if the logic depth of a system is of considerable length, the performance of wave pipelining might be substantially degraded. In this case, a combination of wave pipelining and conventional pipelining techniques seems to be the best approach to optimize the overall performance, as shown in Fig. 8.

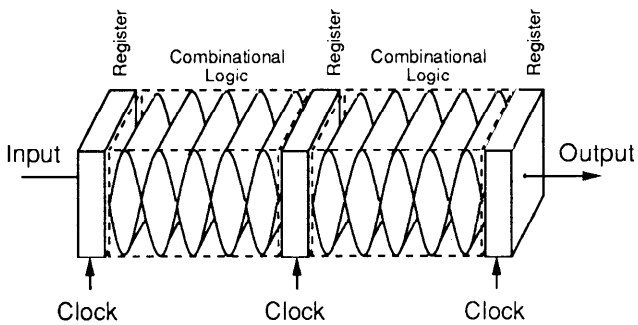

Figure 8: Optimal Pipelining: a combination of Regular and Wave Pipelining

The insertion of a few registers between sections of logic provides a way of synchronizing data in a conventional way, keeping the difference in path delay within acceptable margins, in such a way that wave pipelining can work efficiently between registers. This approach has the advantages of the high pipeline rates achievable with wave pipelining while it minimizes the disadvantages of conventional pipelining, such as latency increase and area penalties, since fewer register are required. This combined technique will be called hereafter optimal pipelining.

7.1 Optimal Pipelining Using Edge-Triggered Registers

Latency $\times$ Cycle-Time is now evaluated for optimal pipelining. Suppose a system with initial delay T is divided into n regular-pipelined stages, where each stage has delay $T_n = T/n$, as shown in Fig. 9. Each stage is wave-pipelined and the number of waves between registers is N. Therefore, the total number of effective stages in the pipeline is $n \times N$.

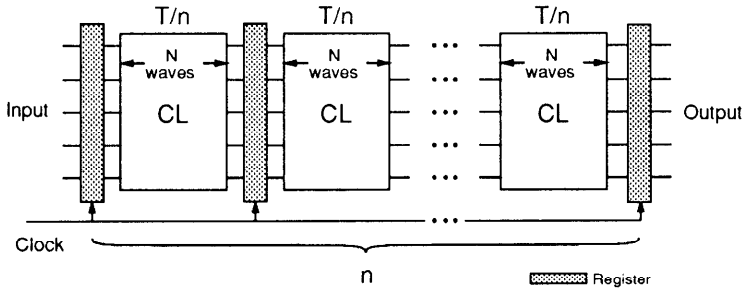

Figure 9: An optimal pipelined circuit with n regular stages, where each stage has N effective stages (N waves)

The total latency of the system can be expressed as:

$$L = n \times N \times t_{CK} \tag{32}$$

And the product Latency $\times$ Cycle-Time as:

$$LC = n \times N \times {t_{CK}}^2 \tag{33}$$

The latency of one stage of the regular pipeline, $N \times t_{CK}$, is given by:

$$N \times t_{CK} = T_n + H - H'/2 + t_{CK}/2 - \Delta T_n/2 \tag{34}$$

where ΔT_n is the difference between maximum and minimum delay in one pipeline stage (see Eq. (15) for similarity). The model assumptions are that ΔT_n is the same for every segment and that parameters H and H' are constant (same assumption as in Section 4).

The same steps followed in the derivation of Eq. (17) are repeated here, except for the fact that now ΔT_n depends upon the logic length of one pipeline stage resulting from the partition. A model which seems to agree with the practice assumes that the difference in path delay increases directly proportional with the delay of the logic. This is formulated as $\Delta T = \delta\, t$, where $t \leq T$ is the delay of one section of the logic, and $\delta < 1$ is a positive number which depends upon the structure of the circuit and the technology being used (parameter δ was introduced in Section 4 to find the maximum number of segments in a regular pipeline; although the definition is the same, in this case minimum δ is required to maximize performance).

Replacing T_n with (T/n) and ΔT_n with $(\delta T/n)$ in Eq. (34) yields:

$$N \times t_{CK} = T/n + H - H'/2 + t_{CK}/2 - \delta\,(T/n)/2 \tag{35}$$

From Eq. (35), the clock period t_{CK} can be derived as:

$$t_{CK} = \frac{(1 - \delta/2)\, T/n + H - H'/2}{N - 1/2} \tag{36}$$

And the maximum number of waves in one section of the pipeline as:

$$\max\{N\} = \left\lfloor 1/2 + \frac{(1 - \delta/2)\, T/n + H - H'/2}{\min\{t_{CK}\}} \right\rfloor \tag{37}$$

where $\min\{t_{CK}\}$ is the minimum clock period at which one section of the pipeline can be wave pipelined. Assuming that the constraint at the output prevails, it is defined as $\min\{t_{CK}\} = \Delta T_n + H'$, where $\Delta T_n = \delta\, T/n$.

Combining Eqs. (36) and (33), the product Latency $\times$ Cycle-Time can be finally formulated as:

$$LC = \frac{[(1 - \delta/2)\, T + n(H - H'/2)]^2}{n} \times \frac{N}{(N - 1/2)^2} \tag{38}$$

where N is computed using Eq. (37).

Latency $\times$ Cycle-Time for optimal pipelining is given by Eq. (38) as a function of the parameters of the system and the number of regular pipeline stages. Notice the correspondence of Eq. (38) with the expressions for regular pipelining in Eq. (5), and for wave pipelining in Eq. (17). If $n = 1$, Eqs. (17) and (38) are equivalent, except that the latter uses δ instead of ΔT.

It is required to find the optimum number of segments such that the product Latency $\times$ Cycle-Time is minimized for optimal pipelining. How to compute analytically the values of n which minimize the function LC above is described in Appendix A. Due to the intricate math involved, the following case study is considered more instructive.

Study 7: Suppose a design is implemented in three different technologies. For the three technologies, $T = 100ns$, $H = H' = 1ns$, but ΔT depends upon the technology being used and is 10% ($\delta = 0.1$), 20% ($\delta = 0.2$), and 30% ($\delta = 0.3$) of the total delay in each case. It is required to find the best implementation in each technology using optimal pipelining. Fig. 10 plots the number of waves per section, the clock period, and the Latency $\times$ Cycle-Time as a function of n in each case, according to Eqs. (37), (36), and (38), respectively. The values of n which minimize LC are summarized in Table 1. While the absolute minimum is found for $n = 35$ in the case where $\delta = 0.1$, notice that near optimal results can be achieved for $n = 20$ or $n = 12$,

Delay Variation	Stages	Waves	Clock	Latency × Cycle-Time
10%	35	3	1.28ns	$173ns^2$
20%	60	2	1.33ns	$213ns^2$
30%	40	2	1.75ns	$245ns^2$

Table 1: Optimal pipelining: minimum Latency × Cycle-Time

saving a considerable number of registers. The maximum value of n in each case (80, 60, and 40, respectively) is determined as the maximum n which allows a minimum number of waves between registers, which is 2 in this case. Beyond this point, the system behaves as a regular pipeline since $N = 1$.

8 Optimal Pipelining Versus Regular and Wave Pipelining

As discussed in Section 6, depending upon the difference in path delay, in some cases regular pipelining can achieve a better Latency × Cycle-Time than wave pipelining, but at a cost of a large number of registers. If the difference in path delay is greater than three times the clocking overhead, rather than the costly solution of regular pipelining, optimal pipelining seems to be the best approach to decrease the product Latency × Cycle-Time while minimizing the number of required registers.

Based on the analytical models derived in the previous sections, a comparison is established for a given system between the performance of optimal pipelining with respect to both regular and wave pipelining. Independently of the parameters of the system, it is demonstrated that the use of optimal pipelining, if achievable, can always yield a better product Latency × Cycle-Time compared with the other two approaches.

> **Theorem 1** *For any given system, if the logic is divided into n stages of equal length, the product Latency × Cycle-Time using optimal pipelining is always less than the product Latency × Cycle-Time using regular pipelining if at least two waves are allowed per stage.*

Proof: According to Eq. (5): $LC(r.p.) = \frac{(T+nH)^2}{n}$

According to Eq. (38): $LC(o.p.) = \frac{[(1-\delta/2)T+n(H-H'/2)]^2}{n} \times \frac{N}{(N-1/2)^2}$

To prove that $LC(o.p.) < LC(r.p.)$, it is sufficient to verify:

(a) $1 - \delta/2 < 1$,

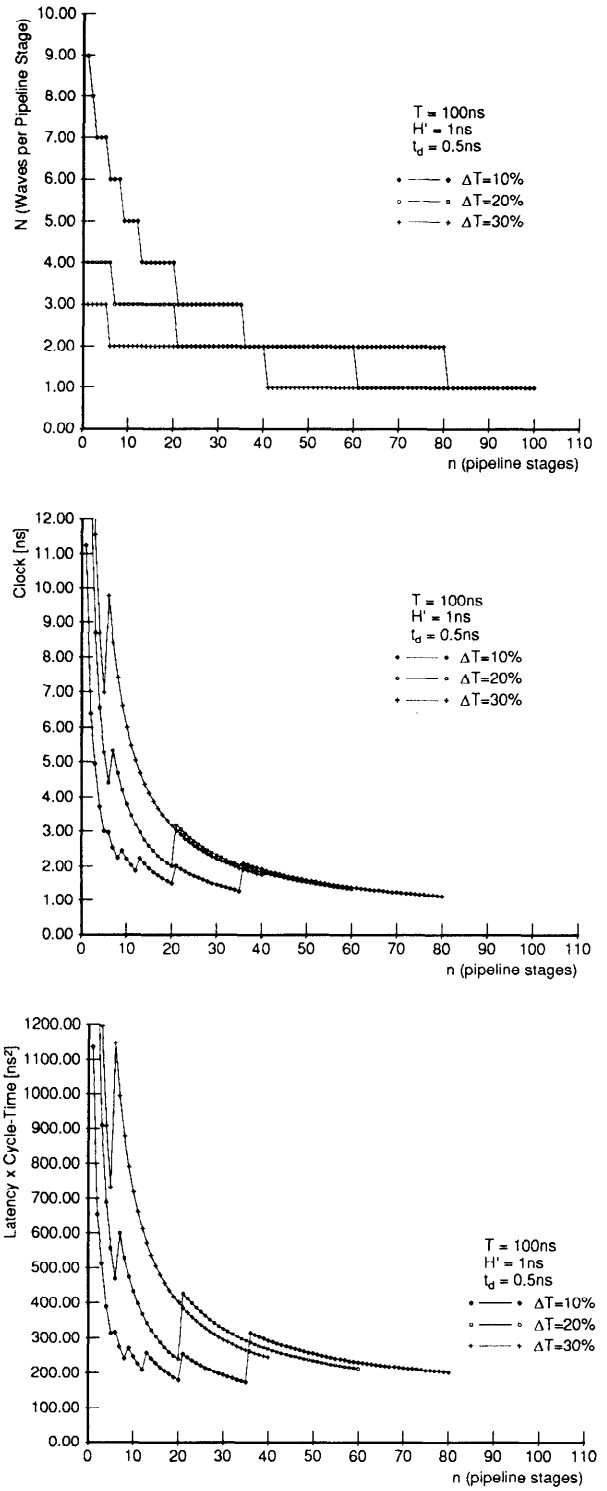

Figure 10: Number of waves at each stage, clock cycle, and Latency $\times$ Cycle-Time as a function of the number of pipeline stages n

(b) $H - H'/2 < H$, and

(c) $N/(N - 1/2)^2 < 1$

Since $0 < \delta < 1$, (a) is true. Since $H' > 0$, (b) is true. Since $N/(N - 1/2)^2$ is monotonically decreasing for $N \geq 2$, and less than 1 for $N = 2$, then (c) is true for all $N \geq 2$. □

Theorem 2 *For any given system, the product Latency × Cycle-Time using optimal pipelining is always less than the product Latency × Cycle-Time using wave pipelining, assuming that the optimal pipeline is divided into n stages of equal length, where n is less than a maximum defined by the system.*

Proof: According to Eq. (38): $LC(o.p.) = \frac{[(1-\delta/2)\,T+n(H-H'/2)]^2}{n} \times \frac{N(n)}{(N(n)-1/2)^2}$,

where $N(n)$ denotes the value of N computed for a given n using Eq. (37).

For wave pipelining, $LC(w.p.) = [(1 - \delta/2)\,T + H - H'/2]^2 \times \frac{N(1)}{(N(1)-1/2)^2}$ (since $n = 1$).

To prove that $LC(o.p.) < LC(w.p.)$, it is sufficient to verify:

(a) $N(n)/(N(n) - 1/2)^2 \leq N(1)/(N(1) - 1/2)^2$ for $n > 1$, and

(b) $[(1 - \delta/2)\,T + n(H - H'/2)]^2/n < [(1 - \delta/2)\,T + H - H'/2]^2$ for $1 < n < n_l$

Since the function $N/(N - 1/2)^2$ is monotonically decreasing, and $N(1) \leq N(n)$ for all $n > 1$, then (a) is true. Since function $[(1 - \delta/2)\,T + n(H - H'/2)]^2/n$ is monotonically decreasing for $1 < n < (1 - \delta/2)\,T/(H - H'/2)$, and monotonically increasing for $(1 - \delta/2)\,T/(H - H'/2) < n < [(1 - \delta/2)\,T/(H - H'/2)]^2$, then (b) is true for $1 < n < n_l$, where $n_l = [(1 - \delta/2)\,T/(H - H'/2)]^2$. □

In practice, n_l is relatively a very large number. The validity of the theorem is of interest for small values of n.

Study 8: Suppose a circuit has parameters $T = 100ns$, $H = H' = 1ns$, and $\Delta T = 10\%\ T$ ($\delta = 0.1$). It is required to compare the performance of three implementations: regular pipelining, wave pipelining, and optimal pipelining. Fig. 11 plots the Latency × Cycle-Time as a function of n for regular pipelining, according to Eq. (5), and for optimal pipelining, according to Eq. (38). Notice that wave pipelining is a special case of optimal pipelining when $n = 1$. If wave pipelining only were used, the clock period would be $11.2ns$, the latency $101.1ns$, and the Latency × Cycle-Time $1136ns^2$. Instead, for $n = 100$, regular pipelining could achieve a clock period of $2ns$, a latency of $200ns$, and a Latency × Cycle-Time of $400ns^2$. For $n = 11$, however, optimal pipelining can achieve a clock period of $2.03ns$, a latency of $111.66ns$, and a Latency × Cycle-Time of $226.7ns^2$. The results are summarize in Table 2. Observe that

Pipeline	Clock Period	Registers	Latency	Lat. $\times$ Cycle-Time
Optimal	$2.03ns$	10	$111.7ns$	$226.7ns^2$
Regular	$2.0ns$	99	$200ns$	$400ns^2$
Wave	$11.23ns$	–	$101.1ns$	$1136ns^2$

Table 2: Three pipelined circuits

while optimal pipelining can achieve the same pipeline rate as regular pipelining, it has about half the latency and saves 89 registers (the poor performance of wave pipelining is due to the fact that $\Delta T = 10ns$).

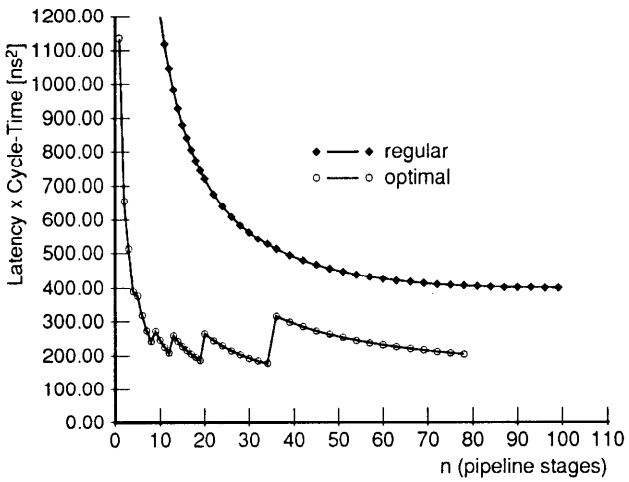

Figure 11: Regular and Optimal Pipelining as a function of the number of pipeline stages using edge-triggered registers

A heuristic approach based on constraint (28) can be used in optimal pipelining to find a partition of the logic without need of computing all possible values of function LC. According to constraint (28), $\Delta T_n < H$. Since $\Delta T_n = \delta T/n$, n can be computed as:

$$n > \delta T/H \tag{39}$$

However, due to the discontinuity of LC, this approach might yield inefficient partitions in some cases. For example, if $T = 100ns$, $H = 1ns$, and $\delta = 0.2$, constraint (39) yields $n > 20$. According to Fig. 10, values of n between 12 and 20 can achieve the same performance saving a number of registers. If $\delta = 0.1$, constraint (39) yields $n > 10$, which is a good approximation.

9 Regular, Wave, and Optimal Pipelining Using Transparent Latches

In this section, the analysis described in Section 4 to Section 8 is re-evaluated assuming transparent latches are used.

9.1 Definition

- t_{ON} Length of the transparent period of the clock signal

9.2 Regular Pipelining

Since all sections have equal delay in the model here considered, there is no 'borrowing' of time from the transparent period. Thus, the timing constraint for the minimum clock period is given by Eq. (4) and the product Latency $\times$ Cycle-Time by Eq. (5).

However, due to race-through constraints, the maximum number of segments is constrainted by the length of the transparent period. In this case, parameter G used in Eq. (8) is defined as as $G = t_{ON} + t_h + \Delta C - t_d$. The longer the transparent period, the shorter the maximum number of segments.

9.3 Wave Pipelining

A timing diagram in this case is shown in Figure 12. The minimum clock period that can be achieved is also limited by the length of the transparent period t_{ON}. Parameter H' is defined as $H' = t_{RS} + t_{ON} + t_{II} + 2 \times \Delta C$.

Assuming $t_{ON} > t_S$, it can be demonstrated that the maximum number of waves is given by:

$$\max\{N\} = \left\lfloor 1/2 + \frac{T - \Delta T/2 + H - H'/2 + t_{ON} - t_S}{\min\{t_{CK}\}} \right\rfloor \tag{40}$$

and the product Latency $\times$ Cycle-time:

$$LC = [T - \Delta T/2 + H - H'/2 + t_{ON} - t_S]^2 \frac{N}{(N - 1/2)^2} \tag{41}$$

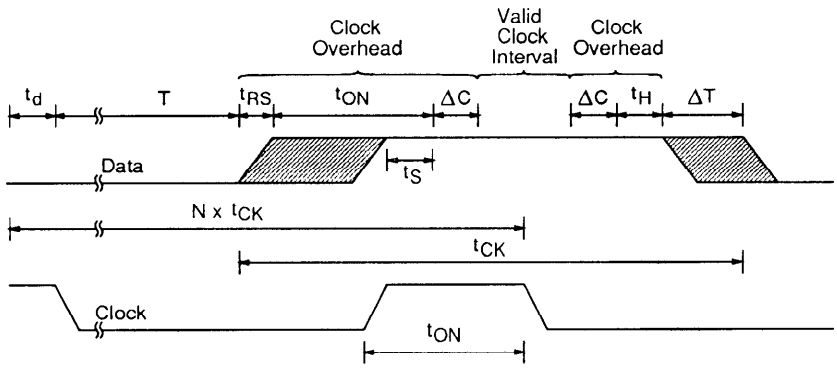

Figure 12: Timing diagram for wave pipelining using transparent latches

9.4 Optimal Pipelining

For a given partition n, the number of waves and the product Latency $\times$ Cycle-time can be computed, respectively, as:

$$\max\{N\} = \left\lfloor 1/2 + \frac{(1-\delta/2)\,T/n + H - H'/2 + t_{ON} - t_S}{\min\{t_{CK}\}} \right\rfloor \tag{42}$$

$$LC = [T - \Delta T/2 + H - H'/2 + t_{ON} - t_S]^2 \frac{N}{(N-1/2)^2} \tag{43}$$

Note that parameter H is the same as in the case of edge-triggered registers, while H' is redefined in Section 9.3 including t_{ON}.

9.5 Comparison between Regular, Wave, and Optimal Pipelining Using Transparent Latches

In order to satisfy constraint (20), now the maximum ΔT is bounded by:

$$\Delta T < \frac{4\,H}{1 + (H + t_{ON} - t_S)/T} - H' \tag{44}$$

Since H' increases as t_{ON} increases, constraint (44) is more restrictive as the transparent period becomes longer.

In can be proved that Theorem 1 holds if $H'/2 > t_{ON} - t_S$ (sufficient condition). This can be rewritten as:

$$t_{ON} < t_{RF} + t_H + 2 \times t_S + 2 \times \Delta C \tag{45}$$

Theorem 2 holds, but n_l is defined as $n_l = [(1 - \delta/2)\, T/(H - H'/2 + t_{ON} - t_S)]^2$.

Study 9: Suppose a system based on transparent latches has parameters $T = 100ns$, $H = 1.5ns$, $H' = 3.9ns$, $t_{ON} = 2.5ns$, $t_S = 0.3ns$, and $\Delta T = 10\%\ T$ ($\delta = 0.1$). It is required to compare the performance of regular, wave, and optimal pipelining in this case. Fig. 13 plots the Latency $\times$ Cycle-Time as a function of n for regular pipelining, according to Eq. (5), and for optimal pipelining, according to Eq. (43) (wave pipelining is for $n = 1$). For $n = 5$ and $n = 8$, optimal pipelining has the best performance. For n between 10 and 19, the performance of optimal pipelining is close to regular pipelining. This is due to the fact that the overhead for optimal pipelining is affected by the length of the transparent period. For $n > 19$, the number of waves for optimal pipelining is one, as in regular pipelining. Compare the plots of Fig. 13 with Fig. 11.

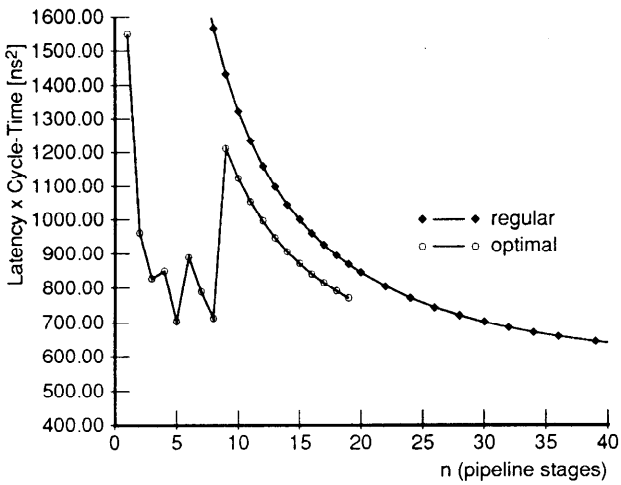

Figure 13: Regular and Optimal Pipelining as a function of the number of pipeline stages using transparent latches

10 Example Application: a 16×16 multiplier

In order to evaluate the performance of wave pipelining in a real application, a 16×16 wave-pipelined integer multiplier was built in CMOS technology. Fig. 14(a) shows a block diagram of the circuit including the main building blocks. For details on the

Pipeline	Clock-Period	Speed-up	Stages	Latency	Latency × Cycle-Time
No	$10.1ns$	$1\times$	1	$10.1ns$	$102ns^2$
Regular	$2.7ns$	$3.7\times$	6	$16.2ns$	$43.7ns^2$
Wave	$1.45ns$	$7.0\times$	7	$10.2ns$	$14.8ns^2$

Table 3: Performance comparison between regular and wave pipelining

construction of the layout, equalization of path delays, and simulation results, see [4]. Fig. 14(b) shows the scheme of a regular pipelined version of the same multiplier where the logic is divided into 6 stages. One stage of this structure was built and simulated, so its overall performance can be predicted and compared with wave pipelining. In both designs, edge-triggered registers were used.

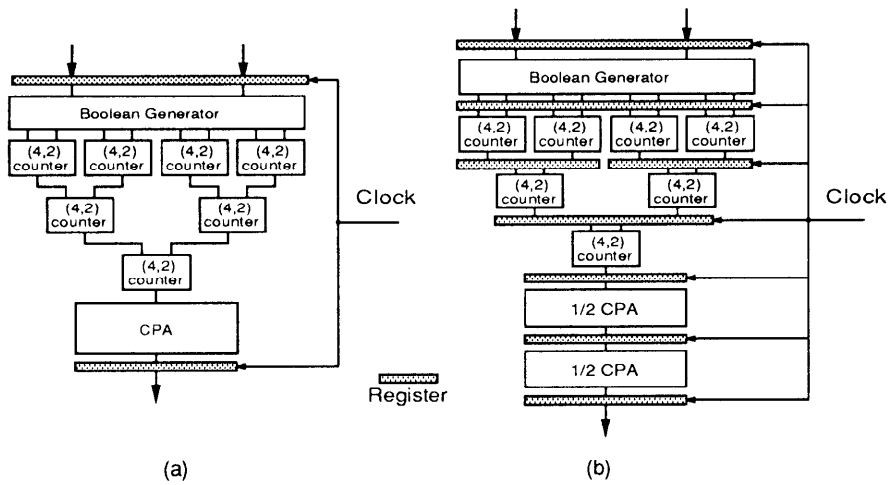

Figure 14: 16x16 Multiplier: (a) wave pipelining, (b) regular pipelining

The parameters of the system, measured through circuit simulation, are $T = 9.0ns$, $H = 1.2ns$, $H' = 0.7ns$, and $\Delta T' = 0.7ns$. Since $\Delta T < H$, according to constraint (28), a factor of 2 at least is expected for the Latency × Cycle-Time in favor of wave pipelining.

Table 3 summarizes the main features of each design. The numbers for the non-pipeline implementation correspond to the same circuit used in wave pipelining but clocked in a conventional way (only one wave at a time). They are included to show the speed-up of the two pipelined designs. The ratio $LC(w.p)/LC(o.p.)$ is $43.7/14.8 = 2.95$, which shows the better performance of wave pipelining and confirms what is predicted by the theory. In practice, considering effects not accounted for in simulations, a ratio close to 2 is expected.

Waveforms obtained in the simulation are shown in Fig. 15 for wave pipelining. The latency and the clock period are indicated in the figure. Compare the waveforms with

the timing diagram shown in Fig. 7. The number of waves in the pipeline is 7, thus the latency is equal to 7 clock cycles.

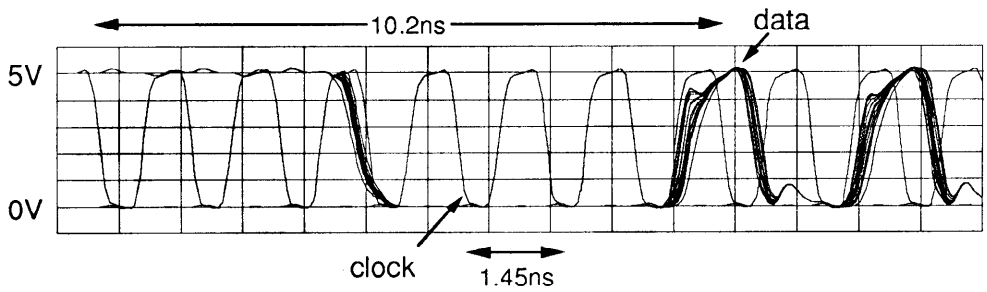

Figure 15: Waveforms for the wave-pipelined multiplier obtained through simulation (16 waveforms from 16 outpus of the multiplier and the clock signal are superimposed)

In the regular pipelined design, the partition of the logic into 6 stages was chosen for practical reasons, since in this way nearly all stages have one functional block with equal delay, except for the last two stages where the Carry Propagate Adder is split into two parts to equalize the delay of each section [4]. Theoretically, this is not the optimum partition. Since $T = 9.0ns$, and $H = 1.2ns$, according to Eq. (6), the optimal partition would be $\lfloor 9.0/1.2 \rfloor = 7$. In this case the Latency $\times$ Cycle-Time could be reduced from $43.7ns^2$ to $40.8ns^2$ (7% improvement). However, a partition into 7 stages would complicate the optimization of each stage, resulting in additional quantization overhead.

11 Concluding Remarks

This paper provides approximate models which can be used as a first-order approximation for the optimization of pipelined systems and as a way of understanding the trade-offs between conventional and wave pipelining. A comparative study and sensitivity analysis was presented of conventional and wave pipelined circuits with different overheads.

Pipeline optimization was achieved by minimizing the product Latency $\times$ Cycle-Time of a circuit. This product was first evaluated for regular and wave pipelining using edge-triggered registers. An upper-bound was derived for the difference between maximum and minimum delay which is necessary (although not sufficient) to guarantee that wave pipelining achieves a better Latency $\times$ Cycle-Time than regular pipelining. If for practical limitations, the difference in delay cannot be reduced below this upper bound, it was also demonstrated that the combination of the two techniques yields optimal results. The results were extended for the case of transparent latches. The same conclusions apply, except that the wave pipelining operation is constrained by

the length of the transparent period. Finally, an application of wave pipelining, a 16×16 multiplier, was described and compared with a regular design.

In practice, assumptions such as constant register overhead and zero quantization overhead are not true in general. Once a technique is chosen and a number of segments is found using approximate methods, further optimizations can be performed. For instance, non-uniform partitions and intentional clock skew can be used [10].

12 Acknowledgements

The authors would like to thank Kevin Nowka for his comments and for reviewing a first draft of this paper. Thanks also to Prof. A.J. van de Goor for his active support. This work was performed with support from the Delft University and Stanford University using facilities provided under NSF Contract No. MIP88-22961.

13 Appendix A: Optimum Partition for Optimal Pipelining

The relative minimums of the function LC given by Eq. (38) can be computed analytically considering that function LC is monotonic between discontinuities (see Fig. 10). If the interval of n for a given N is known, one limit of the interval yields a relative minimum for LC.

According to Eq. (37), values of n and N are related as follows:

$$N \leq 1/2 + \frac{(1-\delta/2)\,T/n + H - H'/2}{\min\{t_{CK}\}} < N+1 \tag{46}$$

Replacing $\min\{t_{CK}\}$ with $\delta\,T/n + H'$, the above inequalities can be expressed as:

$$N \leq \frac{T/n + H}{\delta\,T/n + H'} < N+1 \tag{47}$$

Solving each side of the above inequalities, the interval for n can be derived. The result is given by:

$$\frac{[1-(N+1)\delta]\,T}{(N+1)H' - H} < n \leq \frac{(1-N\delta)\,T}{NH' - H} \tag{48}$$

N	n	Latency × Cycle-Time
2	80	$202.5ns^2$
3	35	$173.6ns^2$
4	20	$180.0ns^2$
5	12	$209.9ns^2$
6	8	$243.0ns^2$
7	5	$315.0ns^2$
8	2	$655.4ns^2$
9	1	$1136.1ns^2$

Table 4: Partitions which yield relative minimums in the product Latency × Cycle-Time for optimal pipelining

Since LC is monotonically decreasing, the right limit of the interval yields the minimum LC. From constraint (48), n is computed as:

$$n = \left\lfloor \frac{(1 - N\delta)\,T}{NH' - H} \right\rfloor \tag{49}$$

The value of N to compute n above is defined as the maximum number of waves allowed in one section of the logic for a given partition. The range of possible values of N is determined as follows. From the left-hand side of constraint (47), for $n = 1$, $\max\{N\} = \lfloor (T + H)/(\delta\, T + H') \rfloor$, and for $n = \infty$, $\max\{N\} = \lceil H'/H \rceil$. Thus, n is computed between these two limits.

In the case where $T = 100ns$, $H = H' = 1ns$, and $\delta = 0.1$, the maximum number of waves per section range from $\lceil 1/1 \rceil = 1$ to $\lfloor (100 + 1)/(10 + 1) \rfloor = 9$. The relative minimums, computed with Eq. (49), are given in Table 4 for $N \geq 2$. The absolute minimum if found for $n = 35$.

References

[1] L. Cotten. "Maximum Rate Pipelined Systems." 1969 AFIPS Proceeding of Spring Joint Computer Conference, pp. 581-586.

[2] D. Wong, G. De Micheli, M. Flynn, and R. Huston. "A Bipolar Population Counter Using Wave Pipelining to Achieve 2.5× Normal Clock Frequency", *IEEE J. Solid-State Circuits*, Vol. 27, May 1992, pp. 745-753.

[3] D. Fan et al. "A CMOS Parallel Adder Using Wave Pipelining", *MIT Advanced Research in VLSI and Parallel Systems*, Providence, RI, March 1992.

[4] F. Klass, M. Flynn, and A. J. van de Goor: "Fast Multiplication in VLSI Using Wave Pipelining", to appear in *Journal of VLSI Signal Processing*

[5] V. Nguyen et al. "A CMOS Signed Multiplier Using Wave Pipelining", *Proc. of Custom Integrated Circuits Conference*, San Diego, CA, May 1993

[6] W. H. Lien and W. P. Burleson. "Wave-Domino Logic: Theory and Applications". *Proc. of TAU'92*, 1992 (a short version appears in *Proc. of ISCAS'92*)

[7] D. Wong, G. De Micheli, and M. Flynn. "Designing High-Performance Digital Circuits Using Wave Pipelining: Algorithms and Practical Experiences", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 12, No. 1, January 1993.

[8] F. Klass. "Balancing Circuits for Wave Pipelining.", Technical Report, Stanford University, October 1992, CSL-TR-92-549.

[9] W. Lam, R. Brayton, and A. Sangiovanni-Vincentelli. "Valid Cloking in Wavepipelined Circuits.", *Proc. of ICCAD '92*, Santa Clara, CA, Nov. 1992.

[10] C. T. Gray, W. Liu, and R. K. Calvin III. "Timing Constraints for Wave Pipelined Systems.", Technical Report, North Carolina State University, December 1992, NCSU-VLSI-92-06.