Optimum Power/Performance Pipeline Depth

A. Hartstein and Thomas R. Puzak
IBM - T. J. Watson Research Center
Yorktown Heights, NY 10598
hart@watson.ibm.com
trpuzak@us.ibm.com

Abstract

The impact of pipeline length on both the power and performance of a microprocessor is explored both theoretically and by simulation. A theory is presented for a wide range of power/performance metrics, BIPS^m/W. The theory shows that the more important power is to the metric, the shorter the optimum pipeline length that results. For typical parameters neither BIPS/W nor BIPS^2/W yield an optimum, i.e., a non-pipelined design is optimal. For BIPS^3/W the optimum, averaged over all 55 workloads studied, occurs at a 22.5 FO4 design point, a 7 stage pipeline, but this value is highly dependent on the assumed growth in latch count with pipeline depth. As dynamic power grows, the optimal design point shifts to shorter pipelines. Clock gating pushes the optimum to deeper pipelines. Surprisingly, as leakage power grows, the optimum is also found to shift to deeper pipelines. The optimum pipeline depth varies for different classes of workloads: SPEC95 and SPEC2000 integer applications, traditional (legacy) database and on-line transaction processing applications, modern (e. g. web) applications, and floating point applications.

1. Introduction

In the very early design stages of a microprocessor, the architects must decide on the pipeline structure. Frequently, this has to be done without the benefit of any accurate modeling since it occurs so early in the concept phase of the design. Researchers have tried to understand the inherent tradeoffs to be made in the choice of pipeline depth in order to change this major decision from an art into a science. Recently, power has become a major design issue. Now both performance and power must be considered in making this key microarchitectural decision.

The optimum pipeline depth based on a performance metric has been the subject of numerous studies over the years. The first such study, of which we are aware, is Kunkel and Smith [1]. They considered the impact of pipeline depth, in the context of gate delay/segment, on the performance of a scalar processor, specifically addressing the effect of latch delays. Dubey and Flynn [2] treated this problem, considering pipeline overhead parameters and branch misprediction pipeline stalls. Recently, Agarwal, et.al. [3] in analyzing microarchitecture scaling strategies, studied this problem, but only considered combinations of pipeline and microarchitectural changes, which did not allow them to elucidate the various dependencies involved. The most coherent early treatment was given by Emma and Davidson [4]. They provided a theoretical treatment for an in-order scalar processor, but without detailed simulations to test their theoretical predictions.

Recently, several authors [5, 6, 7] revisited this problem of the optimal pipeline depth using a performance only metric. Hartstein and Puzak [5] studied the optimum pipeline depth both theoretically and with simulation. Their treatment included both out-of-order and superscalar processes, and provided detailed simulations of the z-Series mainframe architecture to confirm their predictions. Hrishikesh et. al. [6] treated the question of logic depth per pipeline stage empirically for an Alpha 21264-like machine. The Pentium 4 machine was studied by Sprangle and Carmean [7] using IPC degradation factors for adding cycles to critical processor loops. All of these studies report an optimum pipeline depth in the range of 8 to 10 FO4 inverter delays, including both logic delay and latch overhead. The theory of Hartstein and Puzak [5] is able to quantitatively account for the simulated optimal pipeline depths obtained for four different microarchitectures: z-Series mainframe, Alpha 21264, Intel's Pentium 4, and PowerPC.

Consideration of power constraints [8, 9, 10] in microprocessor design has been a much more recent concern. As power becomes an important consideration, two possible design strategies present themselves. One strategy is to design for the best possible performance, subject to the constraint that the power be just below some maximum value, which can be effectively dissipated by the packaging environment. The other strategy, which we study here, is to define an appropriate power/performance metric, and optimize the design to that metric. This latter approach has proven to be much harder than one might think, in that the best metric to use still remains an open question. Various authors [8, 9, 10, 11] have argued for metrics ranging through the choices: $BIPS^3/W$, $BIPS^2/W$ and $BIPS/W$, where BIPS (Billion Instructions / sec) is the performance measure and W (watts) is the power dissipated. As was done in Srinivasan et. al. [12], we study all of these metrics together.

The recent study by Srinivasan et. al. [12] provides the first comprehensive look at the power/performance optimization problem, and in many ways our treatment follows from their work. They formulate a pipelined processor power model, which allows them to study the relative importance of power and performance in the optimization problem. They go on to vary many of the parameters to explore the design space. What we have done is to combine the performance model of Hartstein and Puzak [5] with the power model of Srinivasan et. al. [12], and solve the resulting power/performance model for a range of metrics analytically. Our simulations, being for a different machine architecture, both extend their results and highlight different features of the optimization problem. We solve the complete theoretical problem analytically, and can therefore analyse a more robust set of conditions than were treated in the previous work. We were also able to perform detailed comparisons of theory and simulation for all of the metrics studied in a far more comprehensive manner than was previously possible.

Specifically, in this paper we show that as dynamic power increases in importance, the optimal pipeline depth shifts to shorter pipelines. In a similar vein clock gating moves the optimum position to deeper pipelines. Increases in leakage power, which is becoming increasingly important in processor design, also pushes the optimum pipeline depth to larger values. Finally, we show that the optimum pipeline depth is highly dependent on the two exponents governing this problem, the exponent in the power/performance metric, and the exponent governing the growth of the number of latches in a design with increasing pipeline depth.

2. Theory

We begin our theoretical treatment of the power/performance optimization problem by starting with expressions for performance and power derived in previous papers [5, 12]. For the performance metric we utilize Eq. 8 from Hartstein and Puzak[5]. This expression

$$T/N_I = \left(\frac{t_o}{\alpha} + \frac{\gamma N_H}{N_I} t_p\right) + \frac{t_p}{\alpha p} + \frac{\gamma N_H t_o}{N_I} p \ . \qquad (1)$$

gives the time (T) per instruction (N_I) as a function of numerous parameters: the total logic delay of the processor (t_p), the latch overhead for the technology (t_o), the number of pipeline stages in the design (p), the number of pipeline hazards (N_H), the average degree of superscalar processing (α) and the weighted average of the fraction of the pipeline stalled by hazards (γ). Finding the optimum pipeline depth with performance as a metric, involves taking the derivative of Eq. 1 with respect to p, and setting it equal to zero. The optimum pipeline depth is then given by:

$$p_{opt}^2 = \frac{N_I t_p}{\alpha \gamma N_H t_o}. \qquad (2)$$

We derive our expression for power from Srinivasan, et. al. [12] as

$$P_T = (f_{cg} f_s P_d + P_l) N_L p^\eta \ , \qquad (3)$$

where the total power (P_T) is composed of dynamic (P_d) and leakage (P_l) power. In this expression we assume that the majority of the power consumed in the processor is associated with latches, including clocking; and that the dynamic and leakage power factors are per latch. The dynamic power is multiplied by two factors: the degree of clock gating (f_{cg}), which can alter latches from switching on a given cycle, and the processor frequency (f_s). Other factors are assumed to be subsumed in our definition of P_d. Since P_d and P_l are per latch powers, we must multiply by the number of latches, $N_L p^\eta$, where N_L is the number of latches per pipeline stage. If the number of latches grows linearly with the number of pipeline stages, $\eta = 1$. However, Srinivasan, et.al.[12] have argued that the number of latches in a pipeline stage grows super linearly with the number of pipeline stages. Following them, we take $\eta = 1.1$, but keep η as a parameter throughout our calculations. It should also be noted that our definition of P_d is somewhat different from Srinivasan, et.al.[12] in that we have explicitly taken the factors f_{cg} and f_s out of their expression for P_d. The units for P_d and P_l are therefore different as well.

Rather than choose a particular power/performance metric for study, we have chosen to develop an expression that encompasses all of the metrics commonly considered, $BIPS^3/W$, $BIPS^2/W$, and $BIPS/W$. Now, within a scale factor, $BIPS = (T/N_I)^{-1}$, so the general power/performance metric ($Metric_{P/P}$) we have chosen is

$$Metric_{P/P} = ((T/N_I)^m \cdot P_T)^{-1}, \qquad (4)$$

where the exponent, m, covers all of the metrics needed. Following the methodology of Hartstein and Puzak [5], we form the derivative of our general power/performance metric, set it equal to zero, and solve for the optimum pipeline depth, p_{opt}.

In order to carry out this calculation, we need to specify how some of the factors in Eq. 3 change as pipeline stages are added to a design. We have already explicitly considered how the number of latches grows with pipelining. However, both the frequency and clock gating factors change with pipelining. Frequency is expressed in terms of cycle time, $f_s = 1/t_s = (t_o + t_p/p)^{-1}$ [5]. If we do no clock gating, then $f_{cg} = 1$. Partial clock gating leads to a fractional value for f_{cg}. However, a particularly interesting case arises if we have complete clock gating on a fine grained scale. The circuits and latches only switch with work changes; to first order, pipelining itself does not lead to additional switching. However, it has previously been shown that pipelining can lead to better performance. This secondary effect can lead to changes in the switching behavior. The net result is that the combined effects, of the increased frequency and reduced clock gating switching with pipelining, become proportional to the performance, $(T/N_I)^{-1}$. The effective switching frequency is proportional to the number of instructions executed per unit time. That is we can make the substitution, $f_{cg}f_s \propto (T/N_I)^{-1}$. We will apply this approximation in a later portion of this paper.

Taking the derivative of our power/performance metric with respect to p and setting the result equal to zero yields a quartic equation in p.

$$A_4p^4 + A_3p^3 + A_2p^2 + A_1p + A_0 = 0. \qquad (5)$$

Although quite straightforward to derive, writing the expressions for all of the A_n terms is not very instructive because they are very lengthy. However, the term $A_0 = (\eta - m)at_p^3P_l$ is particularly important. Most of the A_n terms are positive, and in order for Eq. 5 to have a solution with a positive value for p (a necessary requirement for a real physically meaningful solution), A_0 must be negative. This requires that $m > \eta$. Therefore, for the case $m = 1$, corresponding to the metric $BIPS/W$, no solution is possible. This means that the optimum design point is guaranteed to be a single stage pipeline, or rather no pipelining. If A_0 cannot be negative, for instance if leakage is negligible, then there are even more restrictive conditions on m that can be derived from other A_n terms. The next more restrictive condition, derived from A_3, is that $m > 2\eta$, which leads to similar considerations for a $BIPS^2/W$ metric as were just discussed for the $BIPS/W$ metric, i.e. neither the $BIPS^2/W$ metric nor the $BIPS/W$ metric yield pipelined solutions.

2.1 Solutions for a Limiting Case ($m \to \infty$)

Before attempting to solve Eq. 5 for the general case, it is instructive to recover the solution, Eq. 2, for the case of optimizing only on performance. This can be done by noting that for the metrics we have been considering, as m gets larger, performance becomes a more and more important part of the metric at the expense of power. Indeed in the limit $m \to \infty$, we should recover Eq. 2 as a solution.

Taking the limit of Eq. 5 with $m \to \infty$ simplifies the A_n terms, but still leaves a quartic equation, and the terms are still too large to conveniently include here. However, with the assumption that the two solutions, represented by Eq. 2, are valid, it becomes relatively easy to factor Eq. 5 and obtain the four solutions. Eq. 2 gives two of the solutions, with only the positive one being physically meaningful, and the other two are:

$$p = -t_p/t_o \qquad (6a)$$

and

$$p = -\frac{t_pP_l}{P_d + t_oP_l}. \qquad (6b)$$

Both of these are negative solutions and not physically meaningful. These extra solutions were introduced in the process of forming Eq. 5 and then taking the limit as $m \to \infty$, rather than formulating the problem to only have a performance metric as was done previously [5].

More guidance in attacking the general problem can be obtained by plotting Eq. 5 using typical values of the parameters. A typical case is shown in figure 1. Solutions are obtained at the points that the function crosses the x axis on the graph. One can see that there are four zero

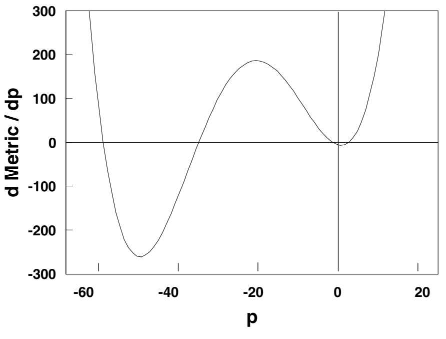

Fig. 1 is a plot of Eq. 5 as a function of p. The zero crossings are the solutions to Eq. 5.

crossings, but only one of these is positive. This demonstrates that all four of the solutions are real, and not complex, a condition by no means guaranteed for a quartic equation. This also shows that there is only one physically meaningful solution. Additionally, it should be noted that the solution at $p = -55$ corresponds to the solution Eq. 6a of the limiting case; and that the solution near $p = -0.5$ corresponds to Eq. 6b. By varying the parameters and replotting figure 1, it becomes clear that these two solutions are largely stationary and not dependent on the other parameters, not contained in Eqs. 6. This leads one to surmise that Eqs. 6a and 6b might also be solutions of the general Eq. 5, as well as solutions of the limiting case.

2.2 General Solution

With this guidance, we were able to factor the solution in Eq. 6a out of Eq. 5, leaving a cubic equation to be solved. Further analysis shows that Eq. 6b is only an approximate solution to Eq. 5. This term does not factor exactly out of Eq. 5, but numerical analysis shows that the deviation from the true solution is less than 5%. Therefore, we take both Eqs. 6a and 6b to be solutions to the general case within this accuracy. Eq. 6a is an exact solution, and Eq. 6b is an approximate solution. This leaves the following approximate quadratic equation, which contains the solution of interest.

$$B_2 p^2 + B_1 p + B_0 = 0 \tag{7}$$

where

$$B_2 = (\eta + m)\gamma \tfrac{N_H}{N_I} t_o$$

$$B_1 = \eta\gamma \tfrac{N_H}{N_I} t_p + \eta\alpha t_o + \frac{P_d \gamma \frac{N_H}{N_I} t_p}{P_d + t_o P_l} \tag{8}$$

$$B_0 = (\eta - m)\alpha t_p + \frac{\alpha P_d t_p}{P_d + t_o P_l}$$

Writing the solutions to this quadratic equation, one of which is positive and the other negative, is little better than substituting the B_n coefficients into the well known quadratic formula. Since it is not possible to simplify the expression very much, we have chosen not to do so here. However, several important results can be obtained from the solution to Eq. 7, that can still be seen from the equation itself. Note that the condition for positive solutions, $m > \eta$, is clearly preserved in this approximate solution. We also note that while $m > \eta$ is a necessary condition for a real solution, it is not sufficient because of the additional term in B_0.

Looking carefully at Eq. 7, we can recover all of the sensitivities of the optimum solution, which have been discussed for the performance only optimization [5]. If the number of hazards, N_H, is increased, the coefficients, B_1 and B_2, get larger. This means that p_{opt} must get smaller, in agreement with previous findings. The optimum design point moves to shorter pipelines. Similarly, if γ increases, it leads to a decrease in p_{opt}. The analysis for α, t_p, and t_o, is more difficult because there are competing dependencies to be considered. However, in the final analysis, all of the dependencies discussed in previous work [5] are still valid when power is also included. As the degree of superscalar processing, α, increases, the optimum pipeline depth decreases; and as the ratio t_p/t_o increases, there is more opportunity for pipelining.

3. Simulation Methodology

In order to compare our optimum power/performance theory to an actual microprocessor design, we have used a proprietary simulator. The simulator uses design parameters that describe the organization of the processor and a trace tape, as inputs. It produces a very flexible cycle accurate model of the processor. With this tool we are able to model numerous pipeline designs, various issue width superscalar designs, and either in-order or out-of-order execution processing. We also had the availability of 55 traces, encompassing traditional (legacy) workloads, “modern” workloads, SPEC95 and SPEC2000 workloads. The traditional workloads include both database and on-line transaction processing (OLTP) applications. The modern workloads were real, substantial workloads, written in either C++ or Java. These traces were carefully selected to accurately reflect the instruction mix, module mix and branch prediction characteristics of the entire application, from which they were derived. This tool has mainly been used for work on IBM zSeries processors.

In order to build a model to explore the above theory it is useful to be able to expand the processor pipeline in a uniform manner. The theory assumes that the processor logic can be uniformly divided into p stages. In modeling, it is not practical to repartition the pipeline for each new design. Instead we used the pipeline shown in figure 2.

This is the pipeline model of a 4 issue superscalar, out-of-order or in-order execution machine. The model can handle zSeries code, so that register only instructions (RR), as well as register / memory access instructions (RX), must be executed efficiently. The pipeline has 2 major instruction flow paths. The RR instructions go sequentially through Decode - Register Rename - Execution Queue - Pipelined E-Unit - Completion - Retire.

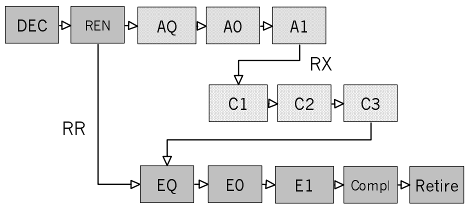

Fig. 2 shows the pipeline modeled in this study. The stages include: Decode, Rename, Agen Q, Agen, Cache Access, Execute Q, Execute, Completion and Retire.

The RX instructions, including loads and stores, add to this pipeline the sequence: Address Queue - Pipelined Address Generation - Cache Access, between the register rename and execution queue stages. For an in-order model the register rename stage is skipped.

This is the base pipeline we have modeled. In testing the theory we utilize the flexibility of our simulator model. In particular we expand the pipeline depth by adding stages "uniformly". We insert extra stages in Decode, Cache Access and E-Unit Pipe, simultaneously. This allows all hazards to see pipeline increases. Hazards, whose stalls cover a larger fraction of the pipeline, see larger increases due to the increased pipeline depth. In addition to expanding the pipeline for this study we needed to contract the pipeline in as uniform a manner as possible. We did this by first combining multiple stages of the same unit, execution, address generation, cache access, etc. We then started combining units, such as decode and address generation on the same cycle. Using this procedure, we constructed models ranging from 2 pipeline stages up to 25 pipeline stages, measured between the beginning of decode and the end of execution (for the shortest pipelines the depth from fetch to retirement was the same as the depth from decode through execution).

Since a 2 or 3 stage pipeline model really won't process instructions out of order to any great extent, we have used an in-order execution model in this study. Hartstein and Puzak [5] explored both in-order and out-of-order models and found only minor differences in the pipeline depth optimization. These differences could be accounted for by changes in the superscaling parameter, α, and the pipeline hazard parameter, γ. By using an in-order model, we avoid additional effects of changing the degree of out of order processing with changing pipeline depth, particularly at very short pipeline lengths.

As well as monitoring performance on a cycle by cycle basis, we have constructed a power model, to model the power usage on a cycle by cycle basis. We monitor the usage of each microarchitectural unit of the processor every cycle, and use this information to calculate the related power. Each unit is assigned a power factor, and we calculate power for both a complete clock gating model and a non-clock gating model. In the clock gating model we utilize the usage information for each unit, whereas in the non-clock gated model, we assume that all units are active each cycle.

Since in the model we pipeline individual units, we assume that the power of each individual unit scales according to the latch scale factor p^{η}, with p being the pipeline depth of the actual unit, not the overall pipeline depth. When we combine two separate units into the same cycle, we assume that the intervening latches can be eliminated. Therefore, the power assigned is the greater of the power requirement for each unit. We have assumed that whichever unit uses more power, also needs to preserve more state for later pipeline stages.

When we run a simulation, we monitor the power, and can therefore determine how the power usage and latch count actually scale with overall pipeline depth. Figure 3 shows the latch count scaling. In order to get the overall latch count to scale with an exponent, $\eta = 1.1$, the individual unit latch counts needed to grow faster, over the range of our simulations, with an exponent, $\eta = 1.3$. In comparing simulation with the theory we utilize this observed value of the exponent, $\eta = 1.3$.

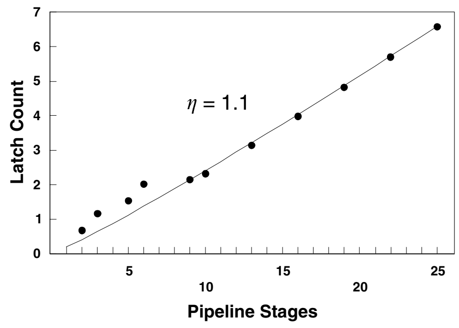

Fig. 3 shows the growth in the number of latches as the pipeline depth increases. The best fit power law is shown.

4. Simulation Results

Each of the 55 workloads was simulated with pipeline depths ranging from 2 stages to 25 stages. Even though we model the complete processor pipeline, we only refer to the stages between decode and execution as the pipeline length. We determine the entire logic delay time in the same way. To compare with theory, we use the detailed statistics obtained from a simulator run at one particular pipeline depth for each workload to determine the parameters in Eq. 4. Two of the parameters, N_I and N_H, are simply enumerated, but α and γ require more extensive analysis of the details of the pipeline and the particular distribution of instructions and hazards in each simulation run. The parameters, $t_p = 140$ FO4 and $t_o = 2.5$ FO4, were chosen to represent a particular technology.

Figures 4a, 4b and 4c show simulated results for three typical workloads, one a "modern" workload, another a SPEC95 integer workload and the last a SPEC2000 floating point workload. In these figures we plot $BIPS^3/W$ for various pipeline depths. Data are plotted for both the case of fine grained clock gating and no clock gating. The non-clock gated data fall below the clock gated data because of the larger power usage in the latter case. The scatter in the data is indicative of the fact that the real pipeline boundaries chosen give discontinuous results, particularly for short pipelines. The solid curves are plots of Eq. 4 with the only adjustable parameter being the overall scale factor.

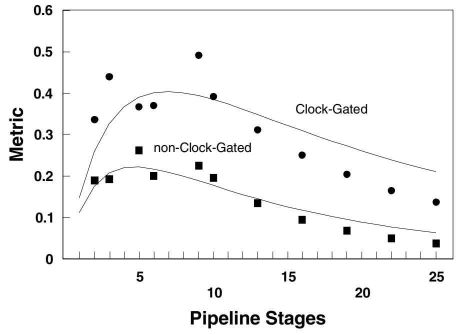

Fig. 4a shows a comparison of theory and simulation for a "modern" workload, both with and without clock-gating.

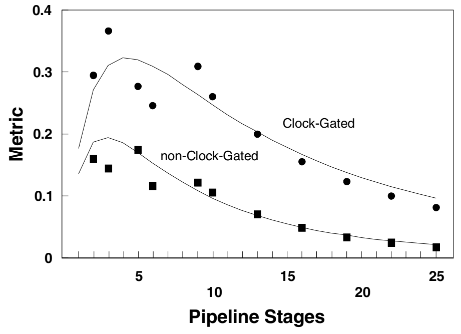

Fig. 4b shows a comparison of theory and simulation for a SPECint workload, both with and without clock-gating.

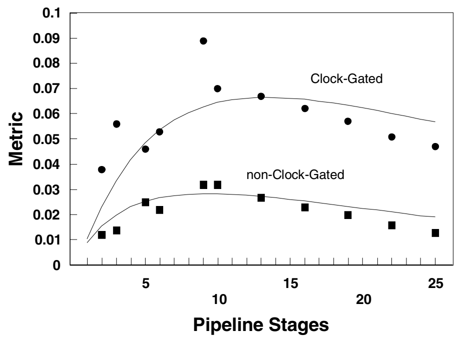

Fig. 4c shows a comparison of theory and simulation for a floating point workload, both with and without clock-gating.

The theory with clock gating utilizes the approximation, $f_{cg} f_s \propto (T/N_I)^{-1}$, discussed in a previous section. The non-clock gated theory assumes $f_{cg} = 1$. As can be seen from the agreement between the theories and the simulated results over a wide range of parameters, the theory gives a reasonable account of the simulations. As such we have obtained an important understanding of the power/performance optimization problem and the relevant approximations. Deviations between the theory and simulations highlight the fact that the theory is only an approximation. We observe that clock gating pushes the optimum pipeline depth to larger values, tending toward the rather large values obtained from performance only optimization. This is a natural trend, in that the larger the role of power in the optimization problem, the shorter the pipeline depths predicted and observed. In fact as we have already seen, for a $BIPS/W$ metric, a single stage design is optimal.

These considerations are further born out by the data in figure 5. Here we plot the various metrics as a function of pipeline depth for the clock gating example of the same workload as shown in figure 4a. In the figure we see optimum pipeline depths (peaks) for $BIPS$ and $BIPS^3/W$, but no optima for $BIPS^2/W$ or $BIPS/W$. The latter cases show the optimum metric for a 1 stage design. Even though it was theoretically possible to get an optimum for $BIPS^2/W$, the particular parameters have moved this optimum point below 1.

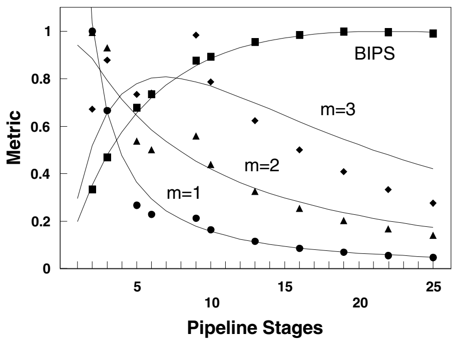

Fig. 5 shows four different metrics as a function of pipeline depth. Both theory and simulation are shown.

It is important to note that without considering power for the optimal design point, the optimum for this workload gives a pipeline depth of about 20 stages, corresponding to a design of 9.5 FO4, including latch overhead. When power is taken into account the optimum has moved to 7 stages, or a 22.5 FO4 per stage design.

This latter result is from the best theoretical fit to the simulation results. If instead we take the simulation results on their own, do a blind least squares fit to a cubic function and find the peak, the optimum occurs at 9 stages, corresponding to an 18 FO4 design point. For these numerical results we have assumed that leakage power accounts for 15% of the power usage. We note that Srinivasan et. al. [12] find an optimum design point of 18 FO4 from simulations, based on a PowerPC processor.

This analysis can be carried one step further by performing our least squares fit on the data from all of the workloads simulated. We do a least squares fit to a cubic equation, verify that the fit is a smooth curve through the data points, and then obtain the maximum of this equation. We take this to be the optimum design point for a microprocessor running the particular workload. The distribution of optimum design points, obtained in this way, for all of the workloads is shown in figure 6.

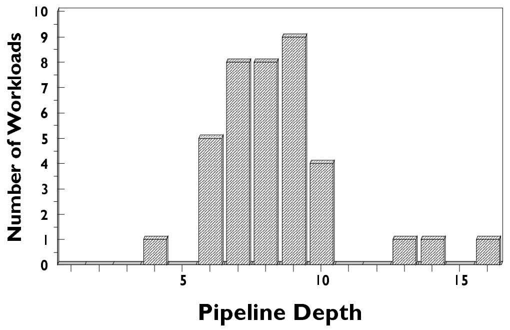

Fig. 6 shows the distribution of observed optimum pipeline depths for all workloads.

We get a distribution of optimum pipeline depths centered around a pipeline depth of 8 stages. This corresponds to a per stage cycle time of 20 FO4. The optimum pipeline depth for a performance only optimization [5] was found to be 22 stages, for a cycle time of 8.9 FO4. Clearly, accounting for power in the optimization has a profound effect on the optimum design point.

One can gain further insight into the nature of this distribution by sorting the various types of workloads and replotting this distribution. The results are shown in figure 7. Different types of workloads clearly have different characteristics. Traditional workloads (programmed in Assembler), show a peak at 9 stages (18 FO4). SPECINT95 and SPECINT2000 workloads show the peak at 7 stages (22.5 FO4). Modern workloads (programmed in C++ or Java) show the peak between 7 and 8 stages (about 21 FO4). Floating point workloads cover the whole range between 6 and 16 stages for the optimum. These very large pipeline depths come from

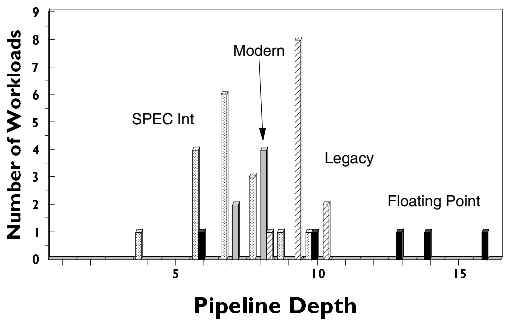

Fig. 7 shows the distribution of optimum pipeline depths for different classes of workloads.

workloads with a large number of floating point operations and fewer pipeline hazards. In our model floating point instructions are assumed to execute individually and take multiple cycles to complete. This greatly reduces the degree of superscalar processing, leading to large optimum pipeline depths.

5. Discussion

Including power along with performance in the pipeline optimization problem has a profound impact on the depth of the pipeline that is optimum. In the method of obtaining that optimum discussed above, that design point, averaged over all of the workloads, was found to be at 8 stages (20 FO4) as opposed to an optimum design point of 22 stages (8.9 FO4) for a performance only optimization. This result depends on the selection of the relevant metric as $BIPS^3/W$. It also depends on the method of extracting the optimum depth. If instead of fitting the data to a smooth curve drawn through the data points, the least squares fit, one were to pick the best fit of our theoretical curves, the optimum depth would be about 20% shorter. The predicted optimum would be 6.25 stages, for a cycle time of 25 FO4. Of course one could not design a pipeline with 6.25 stages, one would have to choose either 6 or 7 stages.

It is unclear which analysis is better in this case. The one we presented first is solely based on the simulation data and drawing the best smooth curve through that data. On the other hand, the second analysis utilizes a first principles theory, which predicts the major features of the results quite well, but contains necessary approximations. For these reasons, it is not completely clear which method is the most trustworthy, and we have presented both results here. It is reassuring that the differences between the two approaches are fairly small, leading to increased confidence in the whole analysis.

The details of how the analysis is done does not change the fact that the optimum pipeline depth is found to be much shorter when power is taken into account. This seems intuitive, in that shorter pipelines use less power. The more important power is to the design, the shorter the pipeline that should be used. This also helps to explain why metrics that are more dependent on the power, $BIPS/W$ and $BIPS^2/W$, tend to optimize at only a single stage. The metrics, which are less dependent on power, $BIPS^3/W$ and $BIPS$, optimize at larger depths. The latter metric has no dependence on power at all and optimizes at the largest depth.

This also explains why clock gating pushes the optimum to greater pipeline depths. Clock gating reduces the power for a given performance. Therefore, one can push the pipeline to larger depths before the power requirement becomes too onerous. The theory quantifies all of these dependencies.

The theory does much more than just explain the simulation results. It can be used to predict the correct design point when new technologies, new workloads, or just changed microarchitectures are involved. All of the input parameters to the theory can be obtained with either no simulations or at most the simulation of a single pipeline depth. In calculating our theoretical curves we obtained N_H/N_I, α, and γ from a single simulation of a workload. This allowed for the calculation of the entire curve. As was seen, this gave a good account of the simulations for all of the other pipeline depths. If one also needs to understand the detailed deviations from this general theory more detailed simulation would be required.

The theory allows us to explore the consequences of greater and greater leakage power on the optimum design point. Figure 8 shows normalized theoretical curves for a particular SPEC95 integer workload with varying amounts of assumed leakage power in the processor. To obtain this figure, the leakage power was increased, while the dynamic power was held constant. The leakage power was varied between 0% and 90% of the total power dissipation. It can clearly be seen that as leakage power grows, the optimum pipeline depth increases. In this particular case the optimum design point changes from 7 stages all the way up to 14 stages as the leakage power increases from a negligible amount to 90% of the total. For this workload the optimum design point had been at 8 stages for a leakage power of 15%. This argues that it is dynamic power that pushes the optimum pipeline depth to shorter pipelines, whereas leakage power has the opposite effect.

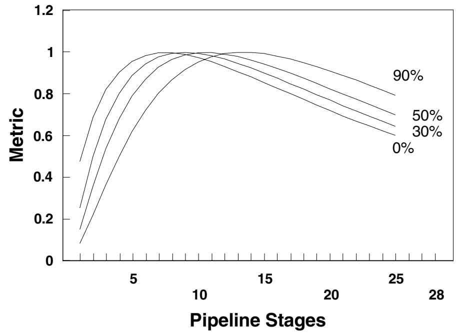

Fig. 8 shows how the optimum pipeline depth varies with increasing leakage power.

We can understand this result as a complex trade-off between power and performance. Leakage power scales linearly with the number of latches, whereas dynamic power scales much faster because, as the pipeline depth increases, both the frequency and the latch count increase. If dynamic power is dominant, the power/performance trade-off has a higher power component, leading to shorter optimal pipeline depths; whereas if leakage power is dominant, the power component is lessened somewhat, favoring deeper pipelines. This complex interplay leads to the observed behavior.

The theory also allows us to explore the critical dependence of the optimum design point on the latch growth factor, η. Figure 9 shows the metric, $BIPS^3/W$, for various values of η, for the same workload as shown in figure 8. It is abundantly clear that the optimum design point is a strong function of η. In fact if η becomes larger than 2, the theory points to the optimum as a single stage design. In this paper we have chosen a value of $\eta = 1.3$, since that translated into an overall latch count scaling of $p^{1.1}$, which was the factor used by Srinivasan, et. al. [12].

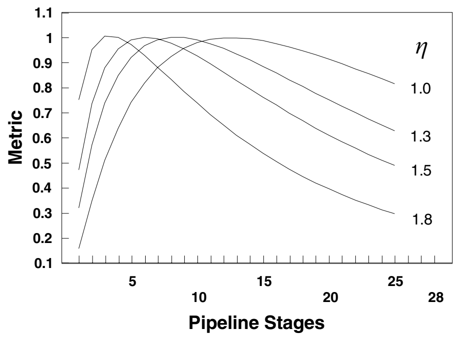

Fig. 9 shows how the optimum pipeline depth varies with changes in the latch growth exponent.

An equally compelling choice could have been $\eta = 1.1$, the growth factor they obtained by studying individual units. This gives a nearly linear scaling with p. As is evident in figure 9, that has a large effect on the optimum design point. This seemingly small change in η is enough to shift the overall average optimum from 22.5 FO4 to 17 FO4. Much more work needs to be done to accurately pin down the value of this extremely important parameter.

6. Summary

A theory has been presented of the optimum pipeline depth for a microprocessor taking into account both power and performance. It was found that only one optimal solution exists for this problem and the nature of that solution has been characterized. For some power/performance metrics one finds that the optimum design contains only a single stage in the processor, whereas for other metrics, including the performance only metric, a pipelined design results. Consideration of power in the optimization problem always leads to shorter pipelines than if power were no consideration.

Simulations were performed for 55 workloads, covering traditional, SPEC, and modern workloads on a 4 issue, in-order superscalar microprocessor model. The theory and simulations were found to be in good agreement. If one takes an average over the different analysis methods and workloads, the optimum pipeline depth was found to be 7 stages, corresponding to a 22.5 FO4 design point. Significant differences are seen between different types of workloads, which can be attributed to the different processing requirements of the different workloads. The SPEC integer workloads are seen to be less stressful of the processor than real workloads of various types.

The parameters, which have the greatest impact on the optimum design point, are the two exponents, m and η. The theory is so sensitive to these parameters because they occur in the exponent. Unfortunately, these parameters are perhaps the least well characterized. The value of the latch growth factor depends on details of the logic and circuit design of the particular pipeline and microarchitecture being considered. Which particular exponent, m, to use in the power/performance metric is still an open question. Indeed these considerations led us to formulate a theory, which kept these values as parameters. The theory is applicable for all values, but the numerical results depend sensitively on the chosen values for these parameters.

Finally, it has been shown that as leakage power grows in a processor, the optimum design favors deeper pipelines. Clock gating has also been seen to have a similar effect.

This theory can be used to investigate numerous dependencies as new microarchitectures, workloads, or new technologies arise. This can be done without the need for the detailed simulations, which were used to verify the theory.

8. Acknowledgments

The authors would like to thank Pradip Bose for help in defining the relative power requirements of the individual microarchitectural units in our model. We would also like to thank him, Viji Srinivasan and Phil Emma for many stimulating discussions.

9. References

[1] S. R. Kunkel and J. E. Smith. "Optimal pipelining in supercomputers", *Proc. of the 13th Annual International Symposium on Computer Architectures*, pp. 404 - 411, 1986.

[2] P. Dubey and M. Flynn. "Optimal pipelining", *J. of Parallel and Distributed Computing **8***, 10 - 19, 1990.

[3] V. Agarwal, M. S. Hrishikesh, S. W. Keckler and D. Burger. "Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures", *Proc. of the 27th Annual International Symposium on Computer Architectures*, pp. 248 - 259, 2000.

[4] P. G. Emma and E. S. Davidson. "Characterization of Branch and Data Dependencies in Programs for Evaluating Pipeline Performance", *IEEE Transactions on Computers* **C-36**, 859 - 875, 1987.

[5] A. Hartstein and T. R. Puzak. "The optimum pipeline depth for a microprocessor", *Proc. of the 29th Annual International Symposium on Computer Architectures*, pp. 7 - 13, 2002.

[6] M. Hrishikesh, N. Jouppi, K. Farkas, D. Burger, S. Keckler and P. Shivakumar. "The optimal logic depth per pipeline stage is 6 to 8 FO4 inverter delays", *Proc. of the 29th Annual Int. Symposium on Computer Architectures*, pp. 14 - 24, 2002.

[7] E. Sprangle and D. Carmean. "Increasing processor performance by implementing deeper pipelines", *Proc. of the 29th Annual International Symposium on Computer Architectures*, pp. 25 - 35, 2002.

[8] R. Gonzalez and M. Horowitz. "Energy dissipation in general purpose processors", *IEEE Journal of Solid-State Circuits **31***, 1277 - 1284, 1996.

[9] M. J. Flynn, P. Hung and K. Rudd. "Deep-submicron microprocessor design issues", *IEEE Micro **19***, 11 - 22, 1999.

[10] D. Brooks, P. Bose, S. E. Schuster, H. Jacobson, P. N. Kudva, A. Buyuktosunoglu, J. D. Wellman, V. Zyuban, M. Gupta, and P. W. Cook . "Power-aware microarchitecture: design and modeling challenges for the next-generation microprocessors", *IEEE Micro **20***, 26 - 44, 2000.

[11] V. Zyuban and P. Strenski. "Unified methodology for resolving power-performance tradeoffs of the microarchitectural and circuit levels", *Proc. of the International Symposium on Low-Power Electronics and Design*, pp. 166 - 171, 2002.

[12] V. Srinivasan, D. Brooks, M. Gschwind, P. Bose, V. Zyuban, P. N. Strenski and P. G. Emma. "Optimizing pipelines for power and performance", *Proc. of the 35th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 333 - 344, 2002.