Architecture Evaluator's Work Bench and its Application to Microprocessor Floating Point Units

Steve Fu, Nhon Quach and Michael Flynn

Technical Report: CSL-TR-95-668

June 1995

This work was supported by NSF under contract MIP93-13701.

Architecture Evaluator's Work Bench and its Application to Microprocessor Floating Point Units

by

Steve Fu, Nhon Quach and Michael Flynn

Technical Report: CSL-TR-95-668

June 1995

Computer Systems Laboratory

Departments of Electrical Engineering and Computer Science

Stanford University

Stanford, California 94305-4055

pubs@shasta.stanford.edu

Abstract

This paper introduces Architecture Evaluator's Workbench(AEWB), a high level design space exploration methodology, and its application to floating point units(FPUs). In applying AEWB to FPUs, a metric for optimizing and comparing FPU implementations is developed. The metric – **FUPA** incorporates four aspects of AEWB – latency, cost, technology and profiles of target applications. FUPA models latency in terms of delay, cost in terms of area, and profile in terms of percentage of different floating point operations. We utilize sub-micron device models, interconnect models, and actual microprocessor scaling data to develop models used to normalize both latency and area enabling technology-independent comparison of implementations. This report also surveys most of the state of the art microprocessors, and compares them utilizing FUPA. Finally, we correlate the FUPA results to reported SPECfp92 results, and demonstrate the effect of circuit density on FUPA implementations.

Key Words and Phrases: Floating point units, microprocessors, SPECfp92, profiling, die area, technology scaling, delay modeling, interconnect modeling

Copyright © 1995
by
Steve Fu, Nhon Quach and Michael Flynn

Contents

1 Introduction **1**

2 Discussion of AEWB **5**

2.1 Area Modeling 7
2.2 Latency Modeling 10
2.2.1 Unloaded Gate Delay 10
2.2.2 Interconnect Modeling 10
2.2.3 Unifying Gate and Interconnect Delays 14
2.2.4 Computing the Latency Compensating Factor 15
2.3 Application Profiling 17
2.4 FUPA Computation 17
2.5 FUPA2 Computations 18
2.6 Verifying FUPA 19

3 Floating Point Unit Comparisons **19**

3.1 Effective Latency 19
3.2 Die Area Usage 21
3.3 FUPA 22
3.4 Correlating FUPA to SPECfp 23

4 Limitation of Metric **25**

5 Conclusion **25**

List of Figures

1 Area Versus Power Statistic for Microprocessors 3
2 Freq Versus Power Statistic for Microprocessors 3
3 Area X Speed Versus Power Statistic for Microprocessors 4
4 Declining Yield with Increasing Circuit Density 5
5 Pentium Scaling Results 8
6 Measured or Predicted I_{dsat} per Width (um) 9
7 Measured or Predicted I_{dsat} with Width Scaling 10
8 Trends and Projections of CMOS Scaling for High-Speed 11
9 Trends and Projections of CMOS Scaling for Low-Power 11
10 Critical Length where C_{int}=C_g and R_{int}=R_{tr}. Local and global interconnect length as predicted by length model 12
11 Interconnect length distribution of VLSI chip in 1982 13
12 Fanout of 4 model allows for technology independent evaluation of delay partitioning and optimization 14
13 Delay Model 15
14 Gate delay of FO4 inverter driving local interconnect. 16
15 Scale Factor Determination for Latency 16
16 Dynamic floating point instruction mix of SPECfp92 18
17 Latencies versus Normalized Latencies of Floating Point Units 20
18 Die Area of Floating Point Units 21
19 Normalized Area and Latency of Microprocessor FPUs 22
20 Floating Point Unit Implementation Comparisons for Microprocessors 23
21 Eliminating Circuit Density Factor from FUPA 24
22 Correlating FUPA to SPECfp92 24

List of Tables

1 Past Trend of MOS Scaling(Hu[4,12]) 6
2 Projected Future Trend of MOS Scaling(Hu[4,12]) 6
3 Ideal MOS Scaling 7
4 Ideal Interconnect Scaling 8

1 Introduction

Architecture Evaluator's Workbench(AEWB) is a high level design space exploration methodology focusing on evaluating processor implementations based on its resource allocations in a technology-independent way. The resources are time in terms of latency and throughput, and costs in terms of die-area and power consumption. Optimality of designs is judged based on best cost-performance in executing relevant benchmarks. AEWB incorporates the powerful concept of technology-independent comparisons using models predicting the latency and density gains from progressing technology. By removing the gains from technology, we enable comparisons among implementation of different processes, the ability to measure the progress of both algorithms and implementations among generations of processors, and whether designs have made best use of the current technology. This methodology also projects the effective life time of a given process as the performance increase resulting from implementation and algorithmic improvements are reduced.

To prove the relevance of AEWB, we apply its concepts in developing a FPU metric—FUPA—*F*loating Point *U*nit *P*erformance Cost *A*nalysis Metric. Even though the development applies specifically to FUPA, the associated concepts also apply to floorplanning other functional units of microprocessor.

Beside the desire to prove AEWB's concepts, the emergence of a floating point metric is overdued. With the added integration afforded through technology scaling, microprocessor designers have gone from software emulations of floating point (FP) operations, to dedicated FP chips (80387), to on-chip FPU, and finally to multiple FPUs on chip. Meanwhile, the latencies of most FP operations have gone from hundreds of cycles, to two or three cycles. The basis of these design efforts is the fundamental need of applications for fast floating point executions. Despite these efforts, allocation of die area to floating point units is an art based on engineering intuition, and past experiences. FUPA enables quantitative trade-offs between performance and cost.

Traditionally, researchers have published papers proclaiming the superiority of their FP architecture and implementation to those preceding them. Comparisons are made on algorithmic levels in terms of number of cycles, and on circuit level in terms of number of transistors and its inherent critical path gate delay. However, most of the arguments have been unconvincing or inconclusive due to lack of attention to anyone of 4 important aspects –latency, cost, technology, and profile of applications.

Of the 4 aspects, the most attention has been paid to latency by microprocessor designers. Previous researches have studied delay modeling of MOS circuitry [14], timing analysis at the chip level [20], and performance optimization [19]. For this paper, latency is defined as the delay incurred from the time input data is applied to the input of FP units till the time the result is available at the output of the FP units, and is usually expressed as either delay (ns) or the number of cpu cycles. FP operations can be pipelined into sub-operations with latencies less than the cycle time of the cpu leading to faster throughput, but pipelining also adds synchronizing delay as well as die area for storage. FP algorithms also lead to iterating implementations [21] where additional result bits are generated each cycle providing savings in die area, but adding to both the overall latency of the operation and communication overhead. One element of latency is often ignored – technology. Often, comparisons have

been made between implementations of different technologies. This is obviously inequitable since both implementations could have benefited from advance processes. We introduce a model which permits technology-independent comparisons of latencies based on results from BSIM3 spice models [3] , empirical data [10], and interconnect and delay modeling [2, 8, 9].

Substantially less attention has been paid to die area in implementation comparisons. The general problem of optimizing area is arduous because area usage is a function of transistor size, placement, and routing. In fact, two dimensional pacement of transistors into the smallest rectangle, and the three dimensional routing of transistors with the smallest routing area are both NP complete [16] [17]. Decisions at the algorithmic level directly impact the layout size and regularity of the routing. An example of this is the choice of Wallace tree reduction [18] versus an reduction array. On the surface, Wallace tree implementation with $log_x(2n)$[1] stages of reduction delay seems superior when compared to an array structure with $O(n)$ stages of reduction where n is the number of operand bits. However, after investigating the routing requirements of Wallace tree in terms of wire length, and additional tracks, the initial impression may not be true. Decision at the circuit level implementation also impacts the layout area. Although dual rail circuit design [11] reduces delay, it also doubles the number of wiring tracks at each bit slice, and this may be unacceptable for the given technology's routing density and the design's bit-slice pitch limitations.

To motivate the close relationship between die-area, operating frequency, and power, we collect power consumption, die-area, and operating frequencies of various low power and high speed microprocessors from the past 4 years. Figure 1 exhibits the total die-area versus power consumption of microprocessors. The general trend exhibited is higher power for processors using larger die-area, although the lack of consideration for operating frequency in the plot, and power supply voltage voltage levels contributes to the scattering of the data points.

Figure 2 shows the operating frequency versus power consumption of microprocessors. Higher operating frequency consistently leads to higher power consumption with the DEC21164 leading the pack at 50 watts and 300 MHz. Figure 3 shows the plot of Die-Area x Operating frequency versus power consumption. Interesting enough, a lot of the scattering exhibited by the previous data have being removed from this graph, an indication of the close relations of Die-Area, Operating Speed, and Power.

Optimizing die area usage positively impacts dynamic power consumption, delay, as well as chip yield. The dynamic power dissipated in a CMOS circuit is modeled by,

$$P_d = 1/2 f_{Ave} C_T V_{dd}^2, \tag{1}$$

where f_{AVE} is the average circuit switching frequency, C_T is the sum of the driving transistor's self load, the load transistors' gate and parasitic capacitance, and the wiring capacitances. Dynamic power dissipation due to transistor capacitance is decreased by minimizing the width of the transistor, and the dynamic power dissipation due to interconnect capacitance is decreased by minimizing the length , width, and thickness of the metal.

[1] $(x = \frac{CSA_{IN}}{CSA_{OUT}})$

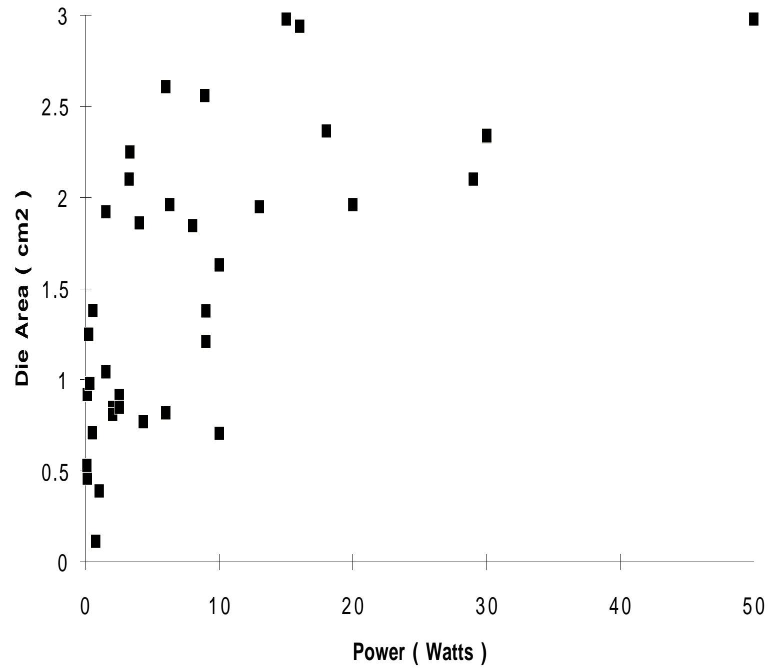

Figure 1: Area Versus Power Statistic for Microprocessors

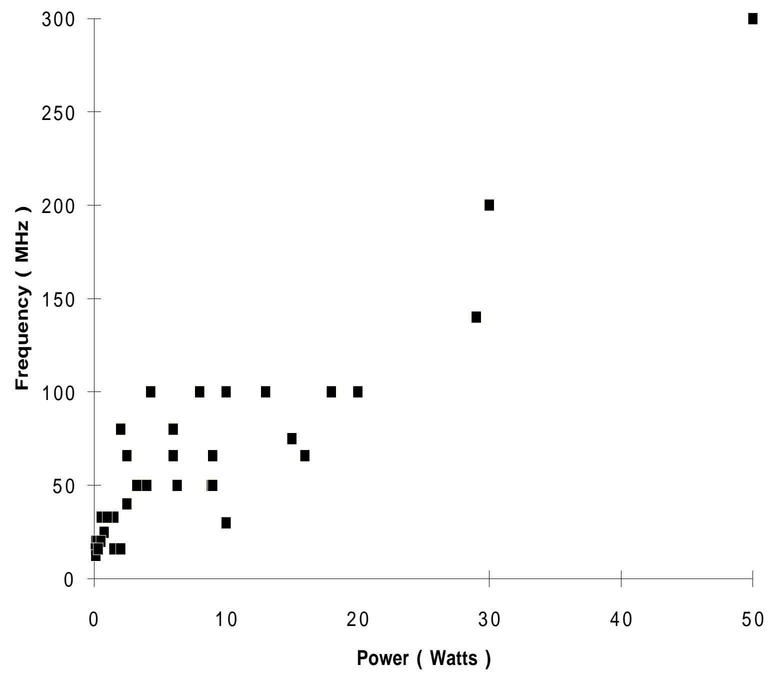

Figure 2: Freq Versus Power Statistic for Microprocessors

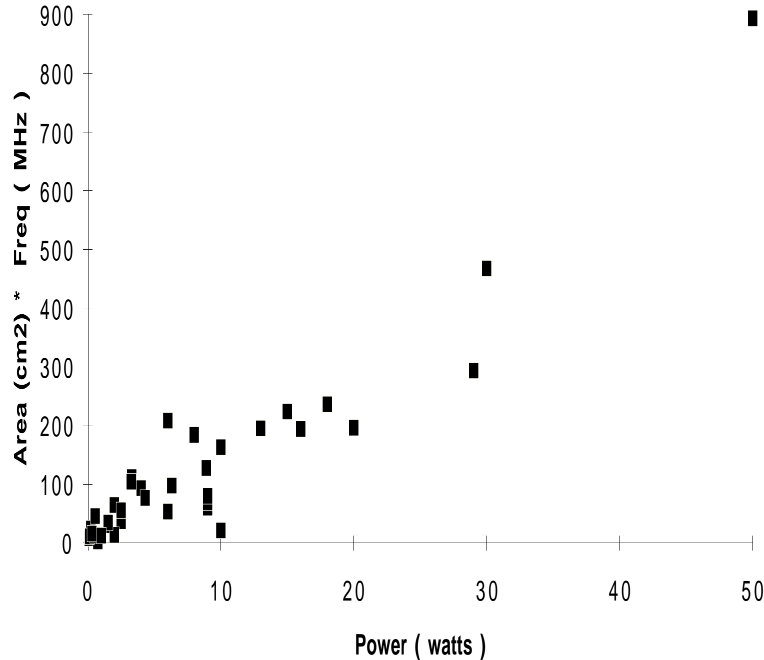

Figure 3: Area X Speed Versus Power Statistic for Microprocessors

The impact of reduce die-area and lower total capacitance on delay is also a positive one despite limiting effects of velocity saturation in transistors, and rising resistance in the interconnect as features in IC and interconnect design shrink. RC models are used in this paper whereby scaling the effective length of the channel leads to decreases in both on-resistance of the transistor and the transit time of carriers across the channel. Optimizing die-area usage also shortens the length of the local interconnect lines thereby reducing both capacitance and resistance. We explore these aspects in more detail in section 2.

Minimizing die area not only minimizes power and reduces delay, it also improves chip yield and manufacturing cost. Many studies have shown the direct relationship between transistor area and chip yield, yet another advantage of die area usage reduction. As discussed in [7], figure 4 shows net probe yield has steadily declined since 1Mb DRAM generation. Even though decline has been compensated by introducing redundancy circuits so gross yield has declined at a slower rate, unless IC process breakthrough occurs in the near future, die size increases will be slowed due to the inevitable yield decreases. As in the case of latency, AEWB facilitates equitable comparisons among design of different technologies by employing a model to normalized the area data prior to comparisons.

AEWB incorporates another key aspect – applications' profile to FUPA. Customer applications are constantly evolving, and in turn, microprocessor designers are constantly attempting to optimize the microprocessor to execute these applications faster, and with more efficiency. FUPA uses application profiles to measures the marginal utility of die area investment and design effort for FPUs, an idea often used in instruction set design. If a instruction is only executed 0.1% of the time, it would be unwise to dedicate 20% of the die

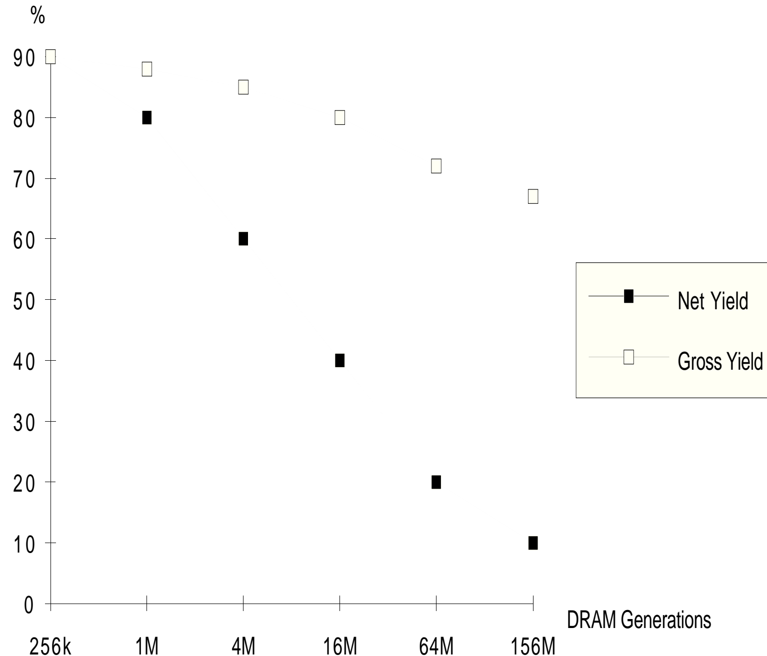

Figure 4: Declining Yield with Increasing Circuit Density

area to reduce the latency of executing that instruction. The same idea can be applied to FPUs where applications use FP-Adds and Multiplies more than FP-Divide, and SQ-Root. In this paper, we profiled the SPECfp92 [5, 6] suite to obtain the dynamic FP operation distributions used in the metric.

Under the above premises, we apply our metric to collected FPU area usage of most of the recent microprocessor design with their associated operation latencies and process technology. Section 2 describes the different aspects of AEWB and FUPA in more detail. Section 3 presents and analyzes the results from applying FUPA to industrial microprocessors. Section 4 discusses the limitations of FUPA, and the observations gained from the results. Finally, in section 5, we conclude with a summary of the contributions of this work.

2 Discussion of AEWB

Past impact and future trend of CMOS scaling have been detailed in [1, 12, 3, 7]. These and other research efforts provide accurate modeling of sub-micron devices, and discuss some of the challenges of future scaling. AEWB incorporates BSIM3 empirical spice models that have shown good conformance with experimental data up to and including the sub-quarter micron range. We summarize the results of these studies here, and discuss how they are incorporated into the metric. Table 1 shows the past trend of transistor scaling which has allowed the rapid growth in circuit complexity and speed in the past 2 decades. Table 2 projects the future scaling trend based on physical limitations such as T_{ox}, leakage current,

Table 1: Past Trend of MOS Scaling(Hu[4,12])

	1977	1979	1982	1985	1988	1991
Minimum Feature(um)	3	2	1.5	1	0.7	0.5
Gate Length(um)	3	2	1.5	1.1	0.9	0.6
Channel Length(um)	2	1.5	1.2	0.9	0.7	0.4
Gate Oxide(nm)	70	40	25	25	20	13.5
Vcc(volt)	5	5	5	5	5	5
Gate Delay(ps)	800	350	250	200	160	90

Table 2: Projected Future Trend of MOS Scaling(Hu[4,12])

	1991	1994	1997	2001	2005	2009
Minimum Feature(um)	0.5	0.35	0.25	0.18	0.13	0.09
Channel Length(um)	0.4	0.30	0.25	0.2	0.15	0.13
Gate Oxide(nm)	13.5	9	8	7	4.5	4
Vcc(volt)	5	3.3	3.3	3.3	2.2	2.2
Gate Delay(ps)	90	79	68	56	49	43

and reliability constraints.

In this paper, we focus on technology independent comparisons of microprocessor implementation, and with this requirement, we develop models that remove the effects of scaling. For latency, the transistor saturation current I_{dsat} is of key importance since it determines the time required to charge and discharge the capacitive loads that it drives. Instead of the classical MOSFET model,

$$I_{DSAT} = \mu C_{ox} W (V_g - V_t)^2 / L, \tag{2}$$

a more realistic model accounts for the fact that the inversion layer charge carrier moves with the saturation velocity, v_{sat}, at the pinch off point in the channel. Such a model is,

$$\begin{aligned} I_{DSAT} &= v_{sat} W C_{ox} (V_g - V_{DSAT} - V_t), &(3)\\ \frac{1}{V_{DSAT}} &= \frac{1}{V_g - V_t} + \frac{\mu}{2 v_{sat} L_{eff}}, &(4)\\ \mu_n &= 240[0.06 T_{ox}/(V_g + V_t)]^{1/2}, &(5) \end{aligned}$$

where V_{DSAT} is the potential at the pinch-off in the channel, and vsat = 8x 10^6 cm/s for electron and 6.5x10^6 cm/s for holes. The only dependence on L_{eff} is V_{DSAT} (approximately $1X$ increase in current for $5X$ decrease in L_{eff}) , but this dependence is much weaker than the linear dependence suggested by the classic model. Only reduction in T_{ox} accompanied

Table 3: Ideal MOS Scaling

Dimensions (W, L, T_{ox}, X_j)	$1/\alpha$
Substrate Doping (N_{SUB})	α
Voltage (V_{dd}, V_{tn}, Vtp)	$1/\alpha$
Device Current (I_{ds})	$1/\alpha$
Gate Capacitance (C_g)	$1/\alpha$
R_{out} (V_{dd}/I_{ds})	1
Intrinsic Delay (τ)	$1/\alpha$
Device Area ($A = WL$)	$1/\alpha^2$

by decreasing L_{eff} leads to more significant increase in I_{DSAT} [12]. Equation 5 models the reduction of surface mobility due to increasing vertical field. The above equations models the driving capability of CMOS with scaling.

The general goal of scaling is two-fold – to increase transistor current, and to increase circuit density. Transistor current increases by (1) decreasing L_{eff}, (2) decreasing T_{ox}, and (3) increasing W. The last technique goes against the goal of increasing circuit density, and is usually used sparingly to buffer heavily loaded nodes, or to speed up critical paths. Reducing the oxide thickness, which increases the vertical field and the inversion charge density leads to greater I_{dsat} increases. However, the reliability constraint of oxide thickness due to time-dependent dielectric breakdown (TDDB) leads to the following model,

$$\begin{aligned} E_{ox} &= (1.1V_{dd})/T_{ox} < 7MV/cm, \text{which requires} \\ T_{ox} &> 8\text{nm for } V_{dd} = 5V, \\ T_{ox} &> 5\text{nm for } V_{dd} = 3.3V, \text{and} \\ T_{ox} &> 4\text{nm for } V_{dd} = 2.5V. \end{aligned}$$

The above oxide thickness is reflected in the projected scaling trend in table 2.

2.1 Area Modeling

Ideal scaling rules for devices, local, and global interconnect are shown in Table 3 and 4. Results from [2] show interconnect capacitance and resistance are become dominating factors in VLSI design in both area utilization and delay modeling as devices and interconnects are scaled. Without accounting for velocity saturation, the need to drive the interconnect, $O(n^2)$ worst case growth in communication area overhead, and most importantly the constant desire to reduce the cycle time, $1/\alpha^2$ reduction in die-area is possible. However, after taking these factors, die-area scaling is less than quadratic. In order to make technology independent comparisons, we derive a area compensating factor, $Area_{cf}$, based on empirical data. Figure 5 demonstrate the scale factors of various parameters of the Intel Pentium processor. Metal-4 pitch has a zero scale factor since Pentium process had no metal-4. Instead of the $1/\alpha^2$ reduction in area, the scaling resulted in $1/\alpha^{1.4}$ reduction in area without

Table 4: Ideal Interconnect Scaling

Parameter	Local Interconnect	Global Interconnect
Thickness (H_{int})	$1/\alpha$	$1/\alpha$
Width (W_{int})	$1/\alpha$	$1/\alpha$
Separation (W_{sp})	$1/\alpha$	$1/\alpha$
Insulation (T_{ox})	$1/\alpha$	$1/\alpha$
Length (L_{int})	$1/\alpha$	α_{DS}[a]
Resistance (R_{int})	α	$\alpha^2\alpha_{DS}$
Capacitance (C_{int})	$1/\alpha$	α_{DS}
RC Delay (T)	1	$\alpha^2\alpha_{DS}^2$

[a] Die Size Increase

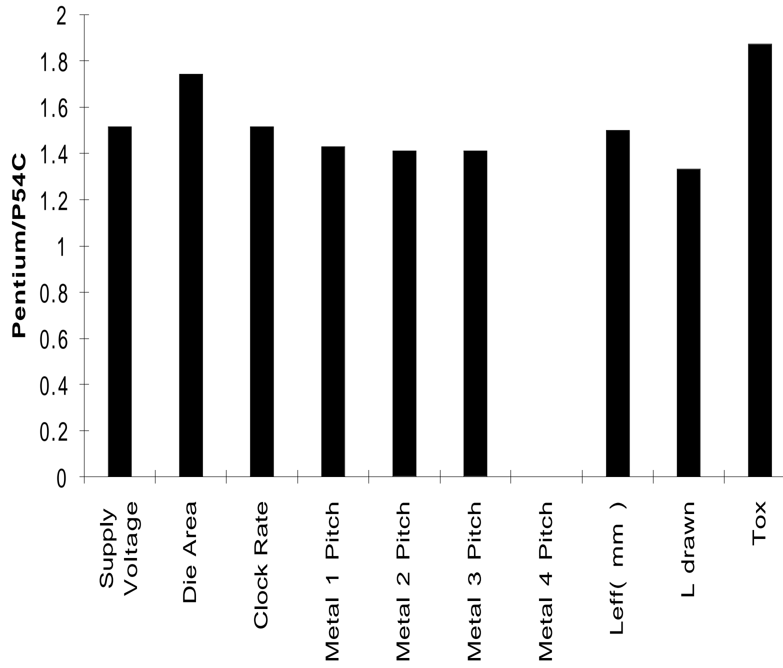

Figure 5: Pentium Scaling Results

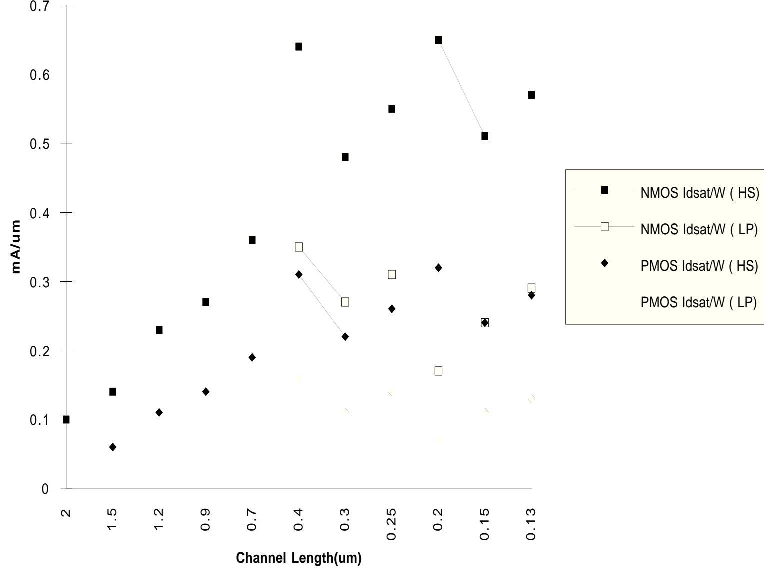

Figure 6: Measured or Predicted I_{dsat} per Width (um)

eliminating the effect of an additional fourth layer metal which the designer claimed to have reduced the area by 10%.

Figure 6 shows measured and predicted values of I_{dsat} per um width for NMOS and PMOS for both high speed(HS) and low power(LP) scenarios based on [1]. Splitting of high speed and low power processes is a trend currently practiced in the industry with processes targeted for either high-speed or low power applications. A trend of slowing of slowing decreases in I_{dsat} is exhibited with drops during years where power supply reductions are predicted. The situation is worsen once we take into account the scaling of width as shown in figure 7. Two consequences can be concluded from these results– (1) unless load capacitance are equally scaled, the decrease in delay will be lessen, and (2)designers would have to increase the $\frac{W}{L}$ ratios of transistors as technology is scaled. It has been shown in [2] that for local interconnect paths, capacitance have decreased, but not at sufficiently rate to keep $\frac{W}{L}$ constant in order continue the trend of clock rate doubling every two process generations.

Based on the above data, we normalized the die-area as,

$$NArea = Area/(L_{eff})^{Area_{cf}} \tag{6}$$

$Area_{cf}$ is derived empirically, and took account of:

- Velocity Saturation's implication on density is due to the fact that I_{dsat}/W slowing increase as L_{eff} approaches zero. To drive identical load with equal delay, W cannot be scaled at the same rate as L_{eff} is scaled.
- The increase routing density required with increasing circuit density is apparent with the increasing number of metal layers employed. The end result is more area is taken by routing.

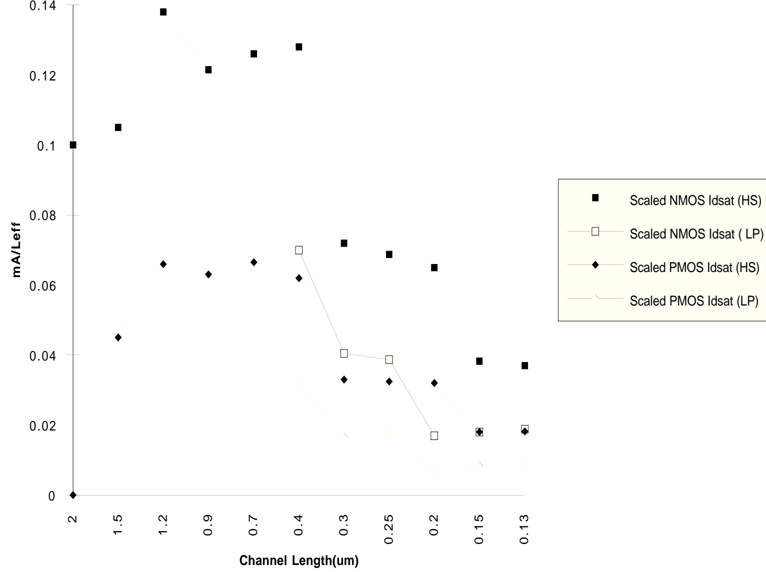

Figure 7: Measured or Predicted I_{dsat} with Width Scaling

- To keep interconnect resistance from dominating both local and global interconnects and to keep coupling noise within reasonable levels, ideal scaling of interconnect cannot be used. For local interconnect, constant R scaling can be used by $1/\sqrt{\alpha}$ or keeping the thickness H_{int} constant. For global interconnect and constant delay scaling, the situation is much worse – we need to scaled up all dimensions by α_{DS}. Both scenarios take away from the ideal area reduction of $1/\alpha^2$.

2.2 Latency Modeling

2.2.1 Unloaded Gate Delay

Figure 8 and 9 displays the trend and projection of high-speed and low-power CMOS. It is apparent that after 1994 the unloaded gate delay is declining at less than linear rate with respect to the scaling of effective channel length, but prior to 1994, the relationship between gate delay and L_{eff} is almost exactly linear. This trend of slowing decline is attributed to the reduction in V_{cc} required for oxide reliability, and the slowing increase of I_{dsat} per unit width. The unloaded gate delay is modeled as,

$$\tau_d = \frac{C_{self}V_{cc}}{4}\left(\frac{1}{I_{dsat-n}} + \frac{1}{I_{dsat-p}}\right) \tag{7}$$

2.2.2 Interconnect Modeling

As further proof that the importance of interconnect capacitance(C_{int}) and resistance(R_{int}) to delay cannot be unestimated. Figure 10 from [2] shows precisely this fact by calculating

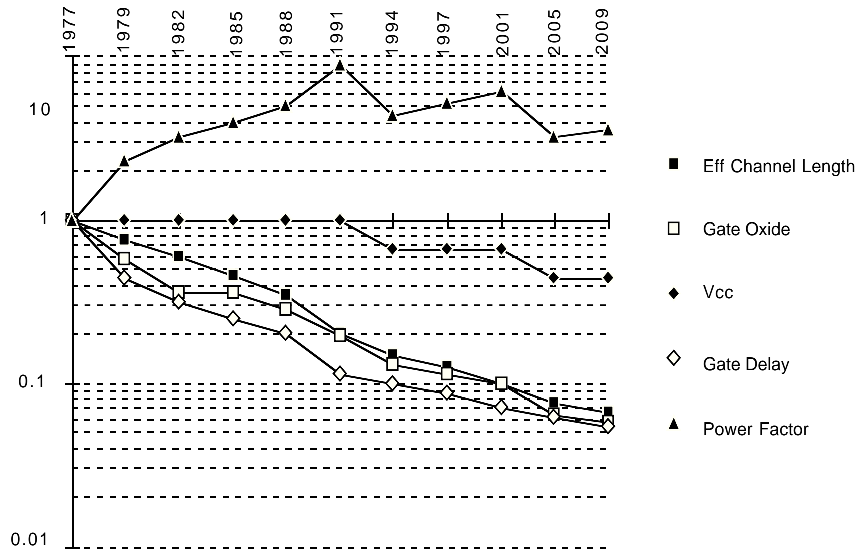

Figure 8: Trends and Projections of CMOS Scaling for High-Speed

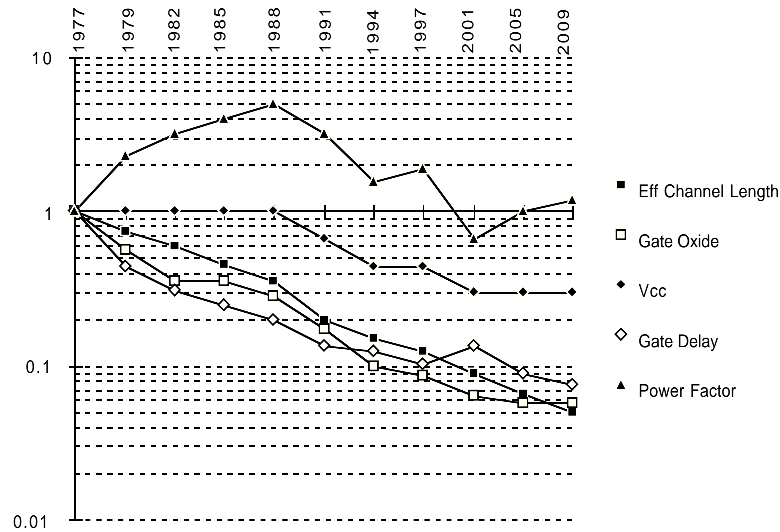

Figure 9: Trends and Projections of CMOS Scaling for Low-Power

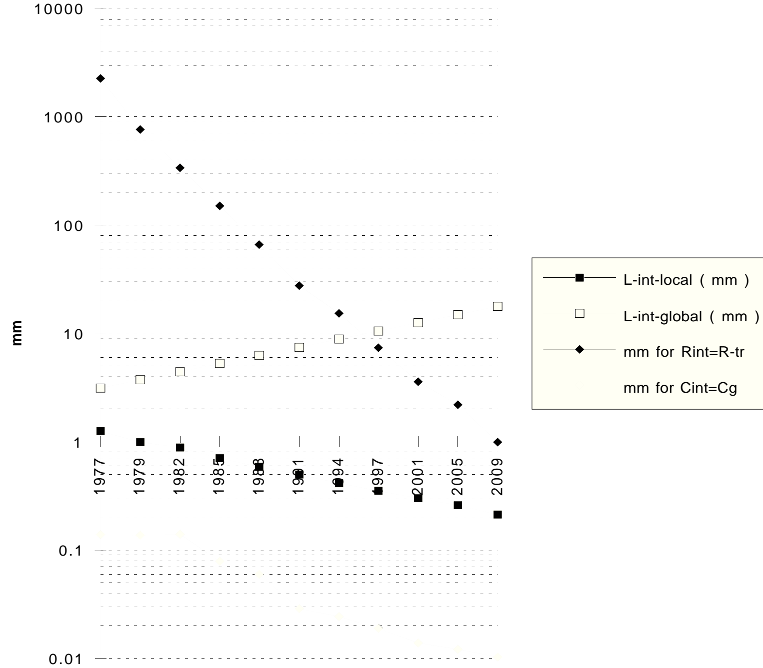

Figure 10: Critical Length where C_{int}=C_g and R_{int}=R_{tr}. Local and global interconnect length as predicted by length model

the critical length of metal when both interconnect capacitance and resistance is equivalent to that of transistor resistance(R_{tr}) and capacitance(C_g). On the same figure, we plot both local and global interconnect length as predicted by our interconnect length model. From this plot, we can predict when interconnect capacitance and resistance dominates as feature sizes are scaled. Capacitance have always been determined by interconnect for both local and global connections, whereas resistance is still dominated by the driving gate's on-resistance (R_{tr}) until 1996 when the global interconnect resistance exceeds it. Since a reasonable analysis of delay cannot exclude interconnect, we incorporate results from [2] which has modeled scaling effects on the delay of CMOS gate driving both interconnect and load capacitance. We summarize the model and the results here.

The interconnect model consists of:

- Aluminum metallization with resistivity of 3.0 $\mu\Omega - cm$.
- Capacitance model that includes all 3-dimensional effects such as coupling and plate capacitances.
- Minimum feature size as described in table 12..
- Wire spacing(W_{sp}) and wire width(W_{int}) are equal to 2.5X of minimum feature size.
- Metal thickness(H_{int}) and insulation thickness(H_{ox}) are equal to 1.25X of minimum feature size.

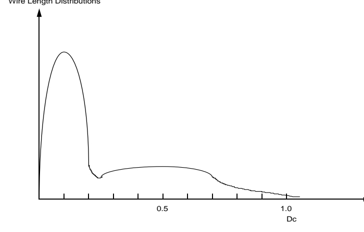

Figure 11: Interconnect length distribution of VLSI chip in 1982

- Ideal local interconnect scaling is used described in table 4.
- Die size is doubled every two generation of technology.

We developed a local interconnect model based on the work of [4] in 1982 with wire length distributions shown in figure 11. The two peaks represent two groups of interconnect: local and global interconnects, with the local representing intra-module connections, and the global representing shared data-paths, and inter-module connections. For our analysis, a local interconnect length scaling model is used where,

$$l_{int} = 0.1(\frac{MinFeature}{MinFeature_{1982}})(\sqrt{A_c}) \tag{8}$$

The model is based on the assumption that as circuit density is increased, local interconnect lengths scale linearly with minimum feature size. Another factor is the increasing die-size(A_c) as technology is scaled.

Interconnect resistance is modeled as

$$R_{int} = \frac{\rho_{AL} l_{int}}{W_{int} H_{int}}, \tag{9}$$

where ρ_{AL} is the resistivity of the conductor material, W_{int} is the width of the interconnect, and H_{int} is the thickness of the interconnect. Note that ideal scaling increases R by greater that α once we take into account die-area increases.

Interconnect capacitance is modeled as

$$C_{int} = \epsilon_{ox} l_{int}(1.15(\frac{W_{int}}{H_{ox}}) + 2.8(\frac{H_{int}}{H_{ox}})^{0.222} \tag{10}$$

$$+ [0.6(\frac{W_{int}}{H_{ox}}) + 1.66(\frac{H_{int}}{H_{ox}}) - 0.14(\frac{H_{int}}{H_{ox}})^{0.222}](\frac{T_{ox}}{W_{sp}})^{1.34}), \tag{11}$$

where ϵ_{ox} is the dielectric constant of oxide, H_{ox} is the thickness of the insulating layer between adjacent interconnect layers or between interconnect layer and ground plane, and

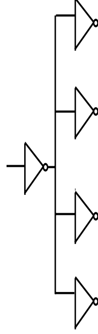

Figure 12: Fanout of 4 model allows for technology independent evaluation of delay partitioning and optimization

W_{sp} is the spacing between adjacent wires on the same interconnect layer. This capacitance model is from [8], and takes into account both parallel plate capacitance and fringing capacitive effects as well as both coupling capacitance between adjacent wires, and plate capacitance to the ground plane. As compared to rigorous numerical analysis of wire capacitance, this model reports less than 10% error.

2.2.3 Unifying Gate and Interconnect Delays

The $zero_{th}$ order delay modeling is the response of a lumped RC network to a step input, and is expressed as,

$$V_{out}(t) = 1 - e^{-\frac{t}{RC}}, \tag{12}$$

where R is the total resistance including R_{int} and R_{tr}, and C is the total capacitance including C_{int} and C_g. This lumped model is easy to use, but overestimates the actual delay. A distributed network can be used, but has no close form solution in the time domain. However, many methods have been proposed to approximate the distributed RC delay. For this paper, the formulation for $T_{50\%}$ from [9] is:

$$C_L = (k_n + k_p)L_{eff}C_{ox}FO, \tag{13}$$

$$C_{ox} = \frac{\epsilon_{ox}}{T_{ox}}, \tag{14}$$

$$R_{tr} = \frac{V_{dsat}}{I_{dsat}}, \tag{15}$$

$$T_{50\%} = 0.4R_{int}C_{int} + 0.7(R_{tr}C_{int} + R_{tr}C_L + R_{int}C_L), \tag{16}$$

where C_L is the loading introduce by a fanout=FO of identical gates, k_n and k_p are the W/L ratios of the NMOS and PMOS, and R_{tr} is the on-resistance of the transistor. The I_{dsat} and V_{dsat} make use of the BSIM3 as a predictive model with technology scaling. The gate level delay model is shown in figure 12. Fanout of 4 inverter delay is chosen because of

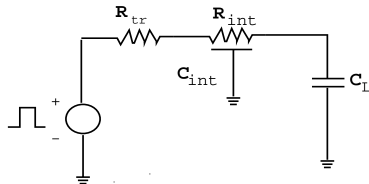

Figure 13: Delay Model

its technology independence. In general the cycle time of a processor or delay of a functional unit can be mapped to the delay of a number of fanout of 4 inverters in series. Even as the delay of gates decreases with technology, this number can be used to indicate the level of partitioning or delay optimization of the designs. Switch level modeling [14, 15] which is both computationally efficient, and flexible is used in our gate delay modeling where each transistor is modeled as a perfect switch in series with a resistor. The final delay model shown in figure 13 takes into account all the previously discussed parameters. Figure 14 shows the constitution of total fanout of 4 gate delay connected via a local interconnect. The delay is divided into three components–T-interconnect, T-gate1, and T-gate2 where T-gate1 is simply the FO4 delay without interconnect, T-gate2 is the delay of transistor driving interconnect capacitance, and T-interconnect is the distributed RC delay of the interconnect alone.

2.2.4 Computing the Latency Compensating Factor

Finally, based on the above models, we can calculate the compensating factor(Lat_{cf}). The idea behind Lat_{cf} is by predicting the local delay variation as technology is scaled, we can derive its relations to L_{drawn} scaling, and produce Lat_{cf} which effectively nullify the effects of scaling.

The resulting latency compensating factor(Lat_{cf}) unifies both device and interconnect scaling analysis and is shown in figure 15. The latency compensating factor factor(Lat_{cf}) computes as following,

$$Delay_{sc} = \frac{T_{50\%}Generation_{n-1}}{T_{50\%}Generation_{n}}, \quad (17)$$

$$Technology_{sc} = \frac{L_{drawn}Generation_{n-1}}{L_{drawn}Generation_{n}}, \quad (18)$$

$$Lat_{cf} = \frac{Delay_{sc}}{Technology_{sc}}. \quad (19)$$

Therefore, to normalize floating point unit performance among processors of different technologies, the Normalized Effective Latency(NEL) is,

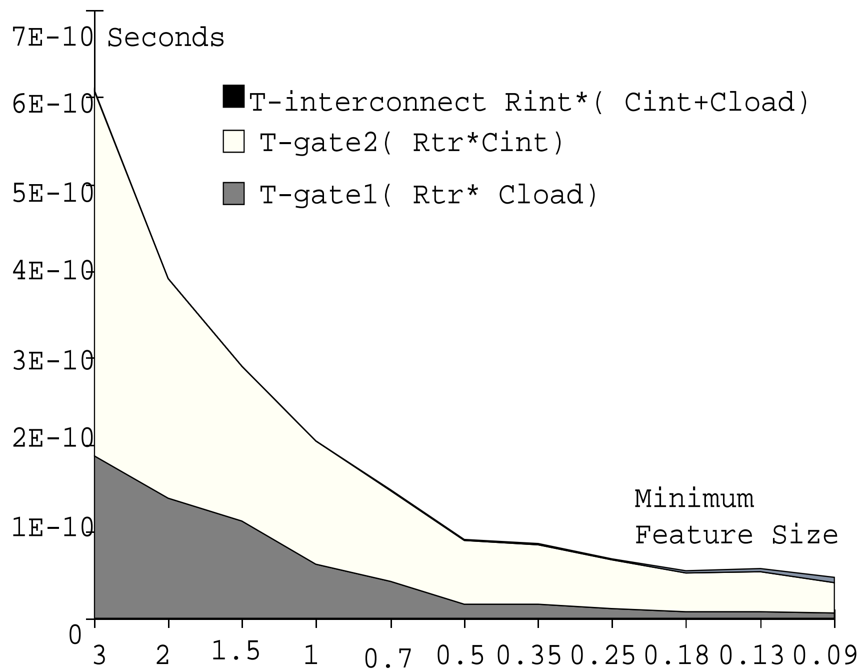

Figure 14: Gate delay of FO4 inverter driving local interconnect.

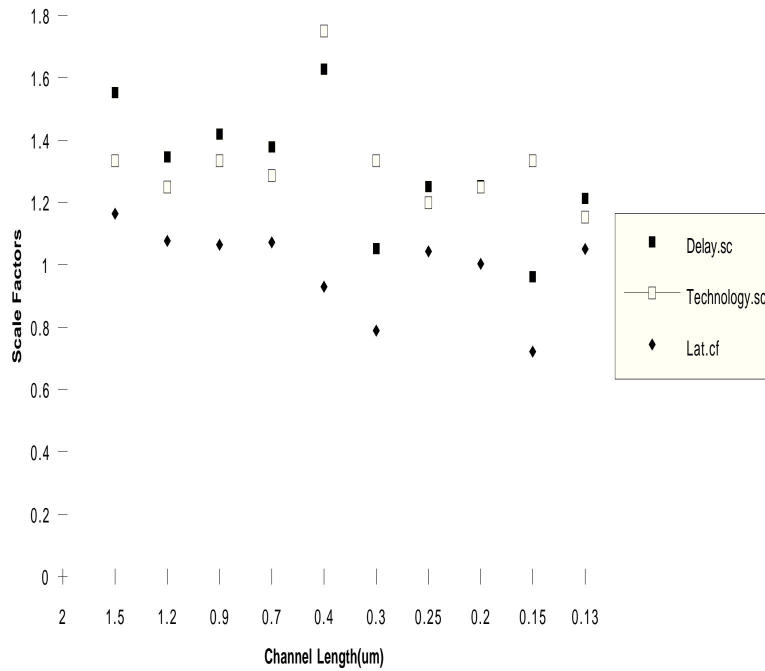

Figure 15: Scale Factor Determination for Latency

$$NEL = \frac{EL}{L_{drawn}Lat_{cf}}, \tag{20}$$

where EL represents Effective Latency introduced in the next section. Verifying the results from [2], the analytical model matches the of pentium cycle time scaling results precisely as figure 5 shows.

2.3 Application Profiling

Computer benchmarking has always been a troublesome, but necessary step to the design of computers. Benchmarks provide a common ground for comparing processor performance where a suite of selected program and data set is run on all the different vendor machine. The ultimate results of these runs provide a indication of the overall performance of the system taking into account instruction set efficiency, latency and throughput of key operations, as well as the capability of machines to supply needed memory and I/O bandwidths to its underlying functional units.

In this study, we have incorporated the distributions of floating instruction reported in [22] into the metric as effective latency (EL). We profiled the SPECfp92 benchmark, a industry standard for comparing workstation systems. The benchmarks were compiled on DECstation 5000 using the MIPS C and Fortran compilers. The binaries are then instrumented using *pixie* which partitions a program into its basic blocks, and adds extra code to tracks the execution of each basic block. *Pixstat* then gathers the dynamic distribution of FP instructions shown in figure 16.

Dynamic distribution from the profiling is used to formulate the Effective Latency as following,

$$EL = \sum_{i=1}^{n} OP_{i-latency} OP_{i-distribution}. \tag{21}$$

EL is normalized as described in the previous section. The $OP_{i-distribution}$ serves as a weighting function for the different operation latencies. $OP_{i-latency}$ is the latency in nsfor processing each of the operations.

2.4 FUPA Computation

Now that all aspects of FUPA have been discussed, we can summarize with the computation of FUPA as following,

$$FUPA = \frac{(NEL)(NArea)}{100} \tag{22}$$

FUPA is the unweighted product of NEL and NArea. The assumption is that area and latency are of equal importance. This can be easily adjusted for applications where the designer wishes to emphasized either parameter. In general, since latency is related to

Figure 16: Dynamic floating point instruction mix of SPECfp92

distance(interconnect) and area (loading), it can be reasoned that increasing area beyond an optimal point increases the overall latency of an operation, and this is proven in later sections where data shows the lowest latency implementation is not the largest implementation.

The interpretation of FUPA rating is trivial– the smaller the rating, the more optimum the implementation. Designer can use FUPA to discover inadequacies in their FPU such as poor choice of algorithms, inefficient utilization and allocation of available die-area and integration, too much resource sharing leading to long timing paths, and ineffective partitioning of timing budget into pipeline stages by comparing to previous implementations of same processor family or other processor families. FUPA also permits alteration for different and evolving applications. By changing the distributions of the FP operations in EL, we can weigh each operation according the profiles of target applications.

2.5 FUPA2 Computations

One element of process technology ignored in FUPA2 is the difference in interconnect density afforded by the specified metal pitches as well as the number of metal layers. It is obvious, however that additional interconnect densities could lead to algorithms requiring more interconnect overhead, and overall all circuit density.

Instead of normalizing area with scaling technology, it is possible to normalize area with circuit density which would have accounted for the increased interconnect density as well. The argument against this method is that designers should not be penalized for using algorithms or implementation that take advantage of available circuit and interconnect

densities. In fact, this type of optimization lead to lower FUPA ratings, thus encouraging designers to evolves with available technologies.

One interesting observation gained from normalizing area with circuit density is that by comparing the results with FUPA it demonstrate whether the designs utilizes the available density well. We define a different FUPA which normalize area with circuit density as,

$$FUPA2 = \frac{(NEL)(NArea2)}{100} \tag{23}$$

$$NArea2 = (Area)(LogicCircuitDensity)/1000, \tag{24}$$

$$LogicCircuitDensity = \frac{Transistor_{total} - Transistor_{storage}}{Area_{total} - Area_{storage}}. \tag{25}$$

Logic density is used excluding storage density since include storage densities would skewed results for processors with different size caches.

2.6 Verifying FUPA

To verify FUPA, we have compared the results of EL to the reported SPECfp numbers. Direct comparison between FUPA and 1/SPECfp is not made since SPECfp is a pure compute time metric and does not take into consideration die-area consumption. SPECfp metric also depends heavily on other aspects of system such as memory and I/O bandwidth as well as software aspects such as compiler optimization.

By showing correlation between EL and SPECfp, we show the effectiveness of the latency aspect of FUPA. The advantage of FUPA over a metric like SPECfp is the incorporation of cost as well as technology, and allows for realistic comparison of FP algorithms and implementation regardless of technology, system configuration, and compiler tricks.

3 Floating Point Unit Comparisons

3.1 Effective Latency

Figure 17 shows the results of applying equation 20, 21 to industrial microprocessor FPU's operation latencies. From the EL data, there is a general trend of reducing floating point unit latencies within each processor families. It can also be observed that FPU latency is highly dependent upon the intended application of the processor with low-end and embedded processors exhibiting higher EL. Processors deficient in their operating frequency such as the SuperSparc suffers since the per cycle latency of floating point units are restricted by the cycle time. SuperSparc with its 3-cycle latency Adds and Multiplies receives a higher EL rating than that of DEC21164 which has 4-cycle Adds and Multiplies. The pentium processor announced in March of 1993 shows quite an improvement over a period of one generation in the x86 architecture as its EL rating shows a 5 times improvement over i486DX2 announced in March 1992.

The NEL is intended to allow for technology independent comparisons of FPU latencies. The NEL of processor with more advance processes such as the the 0.5μ L_{drawn}

Figure 17: Latencies versus Normalized Latencies of Floating Point Units

DX4 and DEC 21164 are normalized to remove their technology advantage. NEL shows improvements or regressions in FPU latencies within the same processor family. Looking at the x86 family, we see improvements across processor generations from 486 to Pentium. However, within each generation, process upgrades have not done as well. This is shown by the processor pairs of [486DX2, DX4] and [Pentium, P54C]. The lack of improvement in these processor pairs is due to the absence of optimization for lowering latency. For the MIPs processor family, NEL ratings shows improvement between [R4000, R4400], and degradation between [R4400, R8000]. The degradation can be attributed to the relatively high cycle time of 13.5 nsof the R8000. The relative high NEL rating of R4200 is due to its merged integer/fp data-path as well as its intended application in low power and embedded environment. For the Sparc family, SuperSparc2 showed an improvement from SuperSparc. This can be attributed to the removal of several critical path in SuperSparc2. On the other hand, HP-PA family, PA7200 did not showed much improvement from PA7100 due to the relatively modest improvement in cycle time despite the more advance process. However, this is consistent with P54C where the FPU remained unchanged from the previous generation. Both the IBM Power family, and the PPC family showed improvement over processor generations.

Overall, it is concluded that simply counting on technology scaling alone to reduce latency results in degraded NEL ratings between generations. In the case of R8000, the degradation is due to bad management of the cycle time, but in general improvements were in NEL was achieved by each family of processors.

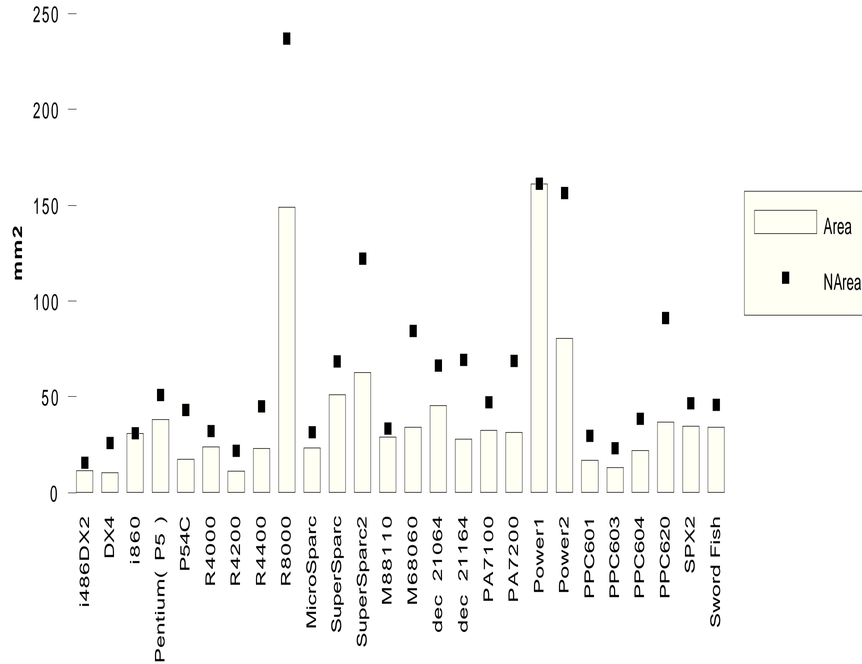

Figure 18: Die Area of Floating Point Units

3.2 Die Area Usage

Figure 18 shows the FPU die-area(Area) usage, and the normalized area(NArea) usage for technology independent comparisons. The Power2 and R8000 FPU area is half of their actual area since there are two independent FPU units in each. Three interesting trends can be observed in the *Area* data.

1. An upward trend in processor family where initial area allocation was low. (Ex. PPC family, x86 family, PA7x00 family)

2. An downward trend in processor family where initial area allocation was high. (Ex. Power family)

3. Looking at figure 19, we can easily see that dedicating the most area to FPU does not lead to the best latency rating.

From the above observation, we can see a general trend of optimization both in terms of area allocation and latency minimization. It can be inferred that there is an optimal area for each technology/architecture where above which latency increases, and FUPA is intended to provide insights into this.

The *NArea* allows for technology independent comparison of implementations. Single-chip FPU implementation such as R8000, Power1, and Power2 uses the highest die-area although both R8000 and Power2 possesses dual FPU units. The P54C FPU which is simply a process shrink of the Pentium benefited from the shrink since the *NArea* decreased. On

Figure 19: Normalized Area and Latency of Microprocessor FPUs

the other hand, SuperSparc2, a shrink from SuperSparc, demonstrated a $NArea$ almost double that of the SuperSparc. The additional area is due to the addition of a separate unit for divides and square roots, which solves a critical timing path of SuperSparc. The DEC21164 $NArea$ is slightly higher than that of the DEC21064, but gets reduced latency and split multiply/add data path in return. In comparison, the PPC620 has more than doubled the $NArea$ of the PPC604 in return for reduced latencies, and a hardware square-root functional unit which will increase performance of applications like Spice.

3.3 FUPA

Figure 20 shows both the product of EL and Area, and FUPA. In most cases, the data sets tracks each other with one technology dependent the other technology independent. However, the advantage of FUPA is allowing designers to evaluate whether optimum use of current technology in terms of area, delay, and density has been accomplished. For example, in comparing the i486DX2 to the DX4 processors form Intel, even though by comparing the product of EL and Area, it appears the designers have improved upon the design from DX2 to DX4. However, FUPA shows otherwise, with DX4 almost doubling the rating of DX2. It can be pointed out based on FUPA rating that more optimization is available for the DX4 FPU. Similar argument apply toward the processor pairs of [R4000,R4400], [SuperSparc,SuperSparc2], [PA7100,PA7200], and [PPC604,PPC620]. Examples where designs have become more optimal as processor family have progressed are [Pentium, P54C], and [DEC21064,DEC21164].

Figure 20: Floating Point Unit Implementation Comparisons for Microprocessors

FUPA is more general than allowing for comparisons within a single processor family. It also permit designers to compare their designs with the rest of the industry, and perhaps learn from those who are doing better.

As discussed in section 2.5, the authors hope to compensate for the differing interconnect densities due to processes with different metal pitches, and number of available metal layers. Figure 21 shows the results of FUPA2 as it compares to FUPA.

Processors with FUPA2 ratings greater than FUPA ratings made good use of the available circuit density since after normalizing for circuit density, the ratings are degraded. This situation applies to the Intel family, DEC21164, as well as the IBM PPC families.

Processors with FUPA2 ratings less than FUPA, inefficient use of available circuit densities has occurred. This situation applies to the R8000, DEC21064, HP PA family, and the Power1 processor. For the R8000 case, 298 mm^2 was consumed by less than a million transistors, obviously underutilizing the available circuit density. For these processors, the designers should consider methods of reduce their die-area utilizations in their respective implementations to achieve better ratings.

3.4 Correlating FUPA to SPECfp

Figure 22 correlates the latency aspects of FUPA to SPECfp as discussed in section 2.6. Although the SPECfp metric also measure both system as well as compiler performance, we show our EL model tracks with it very well.

For cases where EL is greater than $\frac{1}{SpecFP}$, the execution of the SPECfp92 benchmarks

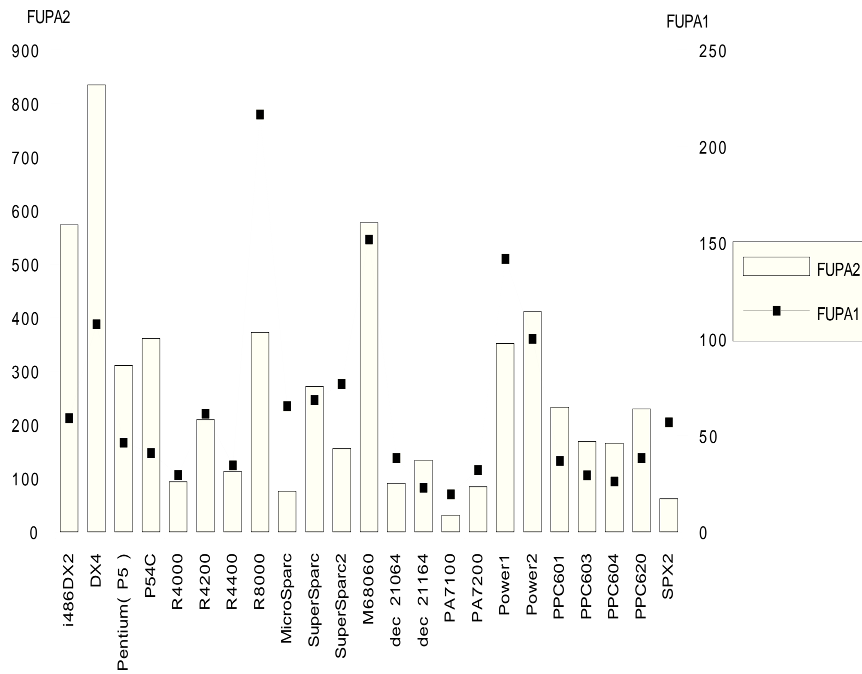

Figure 21: Eliminating Circuit Density Factor from FUPA

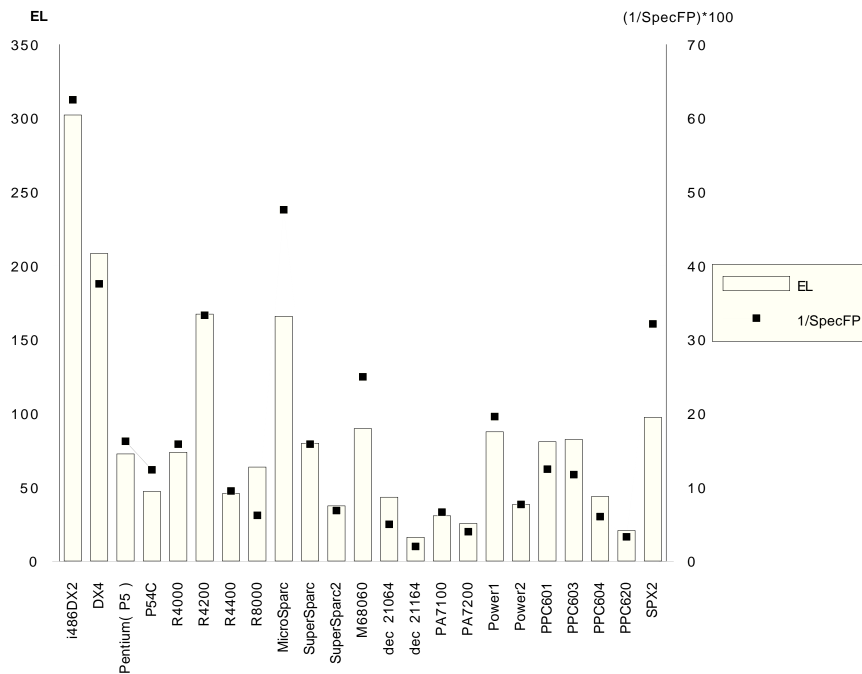

Figure 22: Correlating FUPA to SPECfp92

benefited from either data bandwidth or compiler optimization to compensate for its inherent operation latencies. This applies to DX4, R8000, DEC21064, DEC21164, PA7200, and the IBM PPC family of processors.

The reverse conclusion can be reached for processors where EL is lower than $\frac{1}{SpecFP}$. In this cases, the processor is either not supplying sufficient bandwidth to the FPU , or the code is not optimized sufficiently for the FPU architecture. The processors under this category are the Pentium, P54C, MicroSparc, M68060, Power1, and the SPX2. The designers of these processors can either improve the data bandwidth and compiler optimization, or if FPU performance is not the focus, reduce the area allocated to FPU.

4 Limitation of Metric

The intent of $FUPA$ is to provide insights into the evolution of FPU designs. A key aspect of $FUPA$ is that the underlying data are non-proprietary. We initially plan to include implementation details to suggest how to achieve optimal $FUPA$. However, companies usually treat detailed implementations as confidential. To quote partially released information would only serves to confuse the issue further.

As a result, we derive $FUPA$ based on— area, latency, and technology— public information announced during microprocessor introductions, easing the future application of $FUPA$. To that end, since $FUPA$ uses area measured from die photos, we introduce errors during measurements; however, the error introduced should not prevent $FUPA$ from serving its purpose of providing high-level comparisons and cost/performance tradeoffs of FPU design.

As with any metric, the desire for simplicity and ease of use imposes limitations on $FUPA$. Specifically, $FUPA$ does not account for:

- different circuit design styles, and
- throughput of FPUs.

Circuit design style affects latency, area, power, as well as noise margins, and reliability of operation. Processor designs have typically used static CMOS for its ease of design although dual-rail and dynamic design styles are attractive for switching speed. Even though $FUPA$ does not explicitly take circuit design style into account, it is still applicable since the latency gains by varying circuit design style are counter-balanced by losses in noise margin, power, or area.

Throughput of FPUs is the number of FP operations per cycle performed, or the number of FPU operations initiated every cycle. Faster throughput enhances performance during consecutive FP operations and multiple-issue of FP operations. However, most recent microprocessor FPUs are fully pipelined and have a throughput of one. Other design choices that affect throughput are: 1) fused operations such as the popular multiply-and-add, 2) shared data path for different FP ops, and 3) limited number of FP register ports. We deem these design choices to be second order effects, and as a result, we choose not to include them in $FUPA$.

5 Conclusion

In this paper, we have introduce the general concepts of Architecture Evaluator's Workbench as well as its application in deriving FUPA. In FUPA, we have reached our intended goal of incorporating cost, performance, technology, and application– the four main elements of AEWB.

The idea behind FUPA has general applicability to computer architecture. An equitable comparison of architectures requires cost analysis in terms of area, latency, and power consumption. FUPA facilitate technology independent comparisons–a powerful concept that separates advancements in technology from algorithms, and implementation efficiency. This concept allows realistic comparison between implementations using different technologies, and between implementation of different generations. Design with lower FUPA also exhibits more optimal use of current technology in terms of circuit and interconnect densities.

FUPA promotes the concept of marginal utility of die-area in that it suggest optimal partitioning of die-area based on dynamic distribution of individual operations. Adjusting parameters in FUPA allows it to evolve with changing applications.

In this paper, FUPA, a metric for comparing FPU implementations is introduced and applied to recent microprocessor FPUs. FUPA achieved the goal of incorporating latency, cost, technology, and profiles of target application in a seamless way.

References

[1] Chenming Hu. Low-Voltage CMOS Device Scaling. *IEEE Solid-State Circuit Conference*, pages 86, 1994.

[2] Steve Fu. A Predictive Model for Future Microprocessor Performance.

[3] Ping Ko. Performance and Reliability Design Issues of Deep Submicron Mosfet. *IEEE Transaction of Electron Devices*, vol.38, pages 38, March 1991.

[4] K. C. Saraswat. Effect of Scaling of Interconnections on the Time Delay of VLSI Circuits. *IEEE Transaction of Electron Devices*, pages 645-50, April 1982.

[5] SPEC Benchmark Suite Release 2/92.

[6] Brian Case. Updated SPEC Benchmarks Released.. *Microprocessor Report*, pages 14, September 1992.

[7] M. Ogirima. Process Innovation for Future Semiconductor Industry. *IEEE VLSI Technology Symposium*, pages 17-19, May 1993.

[8] T. Sakurai. Simple Formulas for the Estimation of the Capacitance of Two-Dimensional Interconnects in VLSI Circuits. *IEEE Electron Device Letters*, vol-3, pages 391-3, Dec 1982.

[9] T. Sakurai. Approximation of Wiring Delay in MOSFET LSI. *IEEE Journal of Solid State Circuits*, vol-18, pages 418-26, Aug 1983.

[10] Joseph Schutz. A 3.3V 0.6μ BICMOS Superscalar Microprocessor. *IEEE International Solid-State Circuits Conference*, page 202, 1994.

[11] Hamid Partovi. A Regenerative Push-Pull Differential Logic Family.. *IEEE International Solid-State Circuits Conference*, page 294, 1994.

[12] Chenming Hu. Future CMOS Scaling and Reliability. *Proceedings of IEEE*, 81-5, May 1993.

[13] Donald Stanley Gardner. Layered and Homogeneous Aluminum Alloys For Interconnection Devices in Integrated Circuits. Technical Report G815-1, ICL, Stanford University, February 1988.

[14] Mark Alan Horowitz. Timing Models For MOS Circuits. Thesis. Stanford University, January 1984.

[15] John. Ousterhout. A Switch-Level Timing Verifier for Digital MOS VLSI. *IEEE Transactions on CAD*, page 249-62, July 1985.

[16] S. Sahni. The Complexity of Design Automation Problems. *Proceedings of DAC*, page 402-411, June 1980.

[17] T. G. Szymanski. Dogleg Channel Routing is NP-Complete. *IEEE Transaction on CAD/ICAS*, page 31-41, 1985.

[18] Gary W. Bewick. Fast Multiplication: Algorithms and Implementation. Technical Report CSL-TR-94-617, Stanford University, April 1994.

[19] David P. Marple. Performance Optimization of Digital VLSI Circuits. Technical Report CSL-TR-86-308, Stanford University, October 1986.

[20] Norman P. Jouppi. Timing Verification and Performance Improvement of MOS VLSI Designs. Technical Report CSL-TR-84-266, Stanford University, October 1984.

[21] Mark Ronald Santoro. Design and Clocking of VLSI Multipliers. Technical Report CSL-TR-89-397, Stanford University, October 1989.

[22] S. Oberman. Design Issues in Floating-Point Division. Technical Report: CSL-TR-94-647, Stanford University, December 1994.

[23] H.B. Bakoglu. Circuits, Interconnections, and Packaging for VLSI. Innovative Technology Series. Prentice Hall, 1990.