# All-domain fine grain dynamic speed/voltage scaling for GALS processors

Anand Eswaran, Shelley Chen

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
Email: {aeswaran, schen1}@andrew.cmu.edu

## Abstract

Dynamic voltage scaling (DVS) has emerged as a successful and scalable solution to deal with the growing power consumption associated with increased chip complexity. We describe two schemes that allow the extension of DVS across multiple clock domains specific to GALS out-of-order superscalar processors. One scheme addresses the issues involved in the commonly shared front end of the pipeline. The other enhances the effectiveness of voltage scaling within the various functional units of a superscalar processor by addressing dependency issues. We plan to implement our design on simGALS [1].

## 1 Introduction

Every generation, CMOS transistors are getting smaller in area, allowing processor designers to increase the complexity of a processor chip. A detrimental fall-out of this increasing complexity and integration in modern microprocessors is the fact that power density is rising at a significant rate. One significant component of the power budget is clock power.

As processors become more and more complex, the complexity of the interconnects increases significantly. Since most processors today are single-clock driven, the clock signal must be propagated to the furthest parts of the chip without increasing clock skews. However, chips today have become so complicated and clocks frequencies are so high that the effects of clock skew, though small, would have a significant effect on the functionality of the processor.

Fortunately, GALS, Globally Asynchronous, Locally Synchronous, processors have emerged as a good solution to the clock skew problem. The processor chip is divided into smaller clusters, each cluster having its own clock.

An added incentive of having the processor split into separate clusters, each running on their own clock, is that this gives each cluster the freedom to have its clock frequencies and voltage sources independently manipulated. These conditions will be discussed later in this paper.

This paper is organized according to the following sections. Section 2 will talk about some previous research that has been done concerning dynamic voltage scaling in GALS processors. Section 3 will describe what we intend to do in order to make power consumption in GALS more efficient. Section 4 will list out the schedule that we hope to follow concerning the testing and implementation of our new design. Finally, Section 5 will list out previous publications that we used as references to this area of research.

## 2 Previous or Related work

In all data flow paths other than the critical path, the locally-synchronous blocks can be slowed down , thus producing significant power savings. This is because the energy consumption in CMOS is proportional to the square of the Vdd. Our work aims at the development of a GALS architecture that can dynamically adapt the supply frequency and voltage delivered in a particular window of execution in order to achieve maximal utilization of power savings without significantly affecting performance of the machine.

The implementation issues involved in power saving mechanisms using dynamic clock management for high-performance GALS out-of-order superscalar processors have been discussed in [1] and [2] . There exists an inverse relation between supply voltage and logic delay due to switching capacitances involved. Thus any change in voltage needs to be accompanied by a proportional change in the frequency. Power dissipated in the chip is quadratically proportional to the supply voltage and hence reduction of Vdd would result in significant power savings.

The implementations of current architectures of GALS out-of-order superscalar processors[1],[2],[3],[4] performs no voltage (and hence frequency ) scaling for the fetch, decode and other front-end stages on the assumption that slowing down these stages might be detrimental to net throughput of non-blocking stages further down the pipeline. Most current implementations are based on some mechanism by which the queuing buffers and asynchronous FIFOs that lie in between two differently-clocked stages of a GALS processor informs the producer or receiver to slow down the clock frequency.

## 3 Technical Description

We plan to investigate two solutions for reducing power consumption in the pipeline. One solutions targets the commonly shared front end of the pipeline (the fetch and decode stages). The second is an extension of [1], to accommodate the cross-cutting issues like data dependencies between functional units in the execute stage.

### 3.1 Front End Voltage Scaling

For particular windows of execution, the issue rate and the commit rate might vary considerably. This, in turn, implies that the pipeline as a whole is not balanced with respect to throughput. Ideally, the commit rate should be equal to the issue rate. In practice, the various stages of the pipeline can be clocked so as to ensure that the difference is minimal. When this is accomplished by reducing clock frequency, power savings rises. The issue rate is primarily governed by the clock frequency of the fetch and decode stages, while the commit rate is dependent on the write back stage. Therefore, we propose to equalize these two rates through timely feedback. It is to be noted that this feedback is fed only to the commonly shared fetch and decode stages.

### 3.2 Data Dependencies among Functional Units

The queue algorithm for scaling the clock frequencies of the functional units mentioned in [1] and [2] does not account for data cross-dependencies among the various functional units themselves. This may actually lead to scenarios where the queue length may not be an accurate estimate for the power scaling requirements of the functional unit. In fact, queue length based estimates may result in vastly reduced throughput for the same power usage.

## 4 Schedule, Milestones, and Deliverables

Milestone 1 (10/3):
- Identify the dependencies among functional units that will, in turn, be exploited by 3.1.
- Familiarize ourselves with simGALS environment.
- Create module for determining issue and commit rates.

Milestone 2 (10/17):
- Implement 3.1.

Milestone 3 (10/31):
- Implement 3.2.

Milestone 4 (11/4):
- Integration of 3.1 and 3.2.

Final Milestone (12/2):
- Simulation/verification of proposed schemes.
- Drawing inferences.

## 5 References


[1]A. Iyer and D. Marculescu. Power-Performance Evaluation of Globally Asynchronous, Locally Synchronous Processors. *Intl. Symposium on Computer Architecture (ISCA)*. May 2002.

[2]A. Iyer and D. Marculescu. Power Efficiency of Multiple Clock, Multiple Voltage Cores. *IEEE/ACM Intl. Conference on Computer-Aided Design (ICCAD).* Nov. 2002.

[3]G. Semeraro, D.H. Albonesi, S.G. Dropsho, G. Magklis, S. Dwarkadas, and M.L. Scott. Dynamic Frequency and Voltage Control for a Multiple Clock Domain Microarchitecture. *35th International Symposium on Microarchitecture*. November 2002.

[4]G. Semeraro, G. Magklis, R. Balasubramonian, D.H. Albonesi, S. Dwarkadas, and M.L. Scott. Energy-Efficient Processor Design Using Multiple Clock Domains with Dynamic Voltage and Frequency Scaling. *8th International Symposium on High-Performance Computer Architecture*. pp. 29-40, February 2002.