A Fine-Grain Local Clock Generator Architecture to Enable Dynamic Frequency Scaling in MPSoCs

Guilherme Heck, Leandro Heck, Matheus Moreira, Fernando Moraes, Ney Calazans
Pontifical Catholic University of Rio Grande do Sul
Faculty of Computer Science – Porto Alegre, Brazil
{guilherme.heck, leandro.heck, matheus.moreira}@acad.pucrs.br, {fernando.moraes, ney.calazans}@pucrs.br

Abstract—Synchronous MPSoCs are widespread today, but the evolution of technology into the deep submicron domain leads to increasingly complex timing closure problems. One natural solution to these is to resort to the globally asynchronous, locally synchronous (GALS) paradigm. This work proposes a novel architecture for clock generators to support GALS systems. The generator can drive individual modules of an MPSoC, e.g., NoC routers and processing modules. The main design objectives were low power and area overheads. This generator produces 16 distinct operating frequencies between 90 MHz and 1 GHz. It supports clock gating, glitch-free frequency switching and is robust to PVT variations.

Keywords – Local Clock Generator, Robustness to PVT Variations, DFS, NoC-based MPSoCs, GALS.

I. INTRODUCTION

The MPSoC paradigm emerged as a trend to deal with evolving requirements of the semiconductor market. High modularity and massive parallel processing capability, like multiprocessor systems-on-chip (MPSoCs), provide an efficient solution for the growing need for high performance systems and increasingly constrained time-to-market windows. Most often, an MPSoC comprises a set of intellectual properties (IPs) cores interconnected through some intrachip communication architecture. The limitations of classic bus infrastructures led to the development of networks-on-chip (NoCs). These present higher levels of scalability and communication parallelism than busses. In fact, the use of NoCs is already well established for creating effective MPSoC communication architectures. Moreover, in current technologies, the adoption of the synchronous paradigm at system level for MPSoCs leads to complex timing closure problems [1]. These make the design of MPSoCs easier if implemented with multiple frequency domains.

A straightforward approach to control a system with different frequency domains is to use the globally asynchronous, locally synchronous (GALS) design paradigm, where IPs communicate asynchronously. In such a system, two new optimization problems arise: the partitioning of the system into frequency domains and the choice of the clock generation scheme that feeds each domain [2]. This works addresses the later problem. Both problems are directly related to performance figures, such as power consumption and operating speed, and to physical aspects, such as silicon area and average wire length. In this context, efficient local clock generation schemes, as those proposed here, enable fine-grain partitioning of frequency domains and the independent regulation of the operating frequency of each IP and/or communication module in an MPSoC. Of course, the approach allows eliminating global clock trees altogether.

This work proposes a novel architecture for local clock generators designed for GALS NoC-based MPSoCs. The proposed architecture was specified to present low power and area compared to IPs and NoC routers, thus conducting to low overheads at the system level. In this way, it is possible to have one clock generator per MPSoC module, either an IP or even a NoC router. A case study implementation of the generator digitally controlled oscillator (DCO) can produce 16 distinct operating frequencies between 90 MHz and 1 GHz. Frequencies evolve in close to linear frequency steps, and the DCO supports clock gating and glitch-free frequency switching. Also, the design is robust to processing, voltage and temperature (PVT) variations. The DCO was implemented in a 65nm CMOS technology from STMicroelectronics (STM) and takes only 828 minimum size transistors. Its total power is very small, 22 µW and 93µW at minimum and maximum frequencies, respectively.

The rest of this paper is organized in five sections. Section II discusses related work. Section III presents the MPSoC target architecture. Section IV explores the design of the proposed local clock generator architecture. Section V discusses some experimental results. Finally, Section VI draws conclusions and directions for further work.

II. RELATED WORK

Despite the growing need for local clock generation, few works have addressed implementation of such modules so far. In [1], the authors propose a standard cell DCO with clock gating support using inverters and NANDs. Results show a 22% uncertainty in the generated frequency but process variations can produce 70% deviation on this value. In [4] and [5], authors propose circuits to suppress some clock pulses to effectively reduce a base frequency. However, the main clock must be the maximum operating frequency supported, which is not reasonable for MPSoCs (since the longest path determines the maximum frequency). In [6], a clock generator uses eight 2 GHz frequencies, 45 degrees apart in phase from each other, to generate 83 distinct frequencies between 83 MHz and 666 MHz for IPs and 2 GHz and 4 GHz frequencies for routers. Although it allows different frequencies for IPs and routers, this circuit shows high complexity, which increases its area. Furthermore, routers working faster than IPs suggest that the communication is underutilized at all times and may waste power.
III. ARCHITECTURAL ASSUMPTIONS

The local clock generator proposed here aims GALS NoC-based MPSoCs, such as [2], [7] or [8]. Fig. 1 presents an example of such an architecture, composed by processing elements (PEs), where each one has a router (R) and an IP. In this architecture, each router and/or IP may work at a different operating frequency, which suggests the use of fine-grained clock domains. In such a scenario, routers and IPs require synchronization interfaces in all interconnection points, as in the case of all ports of a synchronous router. Other clock schemes are possible, where groups of routers and/or IPs are in a same frequency island. This relaxes the requirements on the use of synchronization interfaces, but prevents fine grain frequency domains design space exploration, and is usually application specific.

IV. THE LOCAL CLOCK GENERATOR ARCHITECTURE

This work defines the local clock generator (LCG) as a parameterizable clock signal source able to provide clock to one or more MPSoC modules. As Fig. 2 displays, the proposed LCG architecture receives a reference clock signal (ref clk) from the external world, as well as frequency selection (freq sel) and clock gating signals (clk_pause), all provided by an IP or router, to produce the desired clock signal.

This work focuses attention on the choice of frequency only. Internally, the LCG structure comprises an Oscillator, controlled by an Actuator, which together form a Digitally Controlled Oscillator (DCO). The actuator takes a digital value as input and produces an analog signal to control the frequency of oscillation. Besides, the actuator compensates PVT variations effects. The Controller manipulates the generated frequency to provide precision.

In general, the router need not to share the same clock as its local IP on PEs and may even be an asynchronous module [9]. There are clear advantages in decoupling processing from communication, and using different clocks for both facilitates this. A fundamental observation of this work is that synchronous NoC routers used in GALS systems do not need a precise clock frequency to operate correctly and/or efficiently, since their role is to deliver a certain quality of service (QoS) in fulfilling communication tasks. In the case of the NoC employed here, Hermes-GLP [10], a set of frequencies is available to each router from the LCG, so that it can select one of these at any moment to achieve its requested QoS from it. Thus, the LCG for routers can be simple, to reduce its power and area overhead, making it small enough to be used in individual NoC routers. The rest of this paper discusses the DCO implementation case study and its use as a complete LCG for modules which do not require precision in the frequency generation process.

A. The Digitally Controlled Oscillator

The DCO converts a digital input from the Controller block or from the external world to a periodic Clock signal. Here, the DCO focus on low-power and low-area overhead exactly to make viable for each router or IP to have a dedicated LCG. The case study implementation employs low-power transistors with standard threshold voltage (LPSVt) from the STM 65nm technology.

Low power current embedded synchronous processors may execute at operating frequencies close to 1 GHz. This frequency was assumed as the maximum output frequency specification for the case study DCO, while the minimum frequency was stated around 90 MHz. The 910 MHz span of frequencies determines a target separation between successive steps of frequency around 57 MHz, since 16 different frequencies are to be generated. The number of distinct frequencies, sixteen, resulted as a maximum value from an analysis of the noise and supply voltage margins offered by the chosen technology. In addition, this DCO is designed with clock gating capabilities that indeed stop the clock to reduce the IP/router power dissipation. Also, the DCO contains a compensator block for PVT variations.

1) The LCG Oscillator

The LCG Oscillator is the DCO part directly responsible for generating the clock. It comprises two modules: the oscillator itself and the Pause circuit.

Fig. 3 depicts the oscillator used in this work. It is based on the current-starved ring oscillator [11] and is called here CsRO. Besides reduced area, this structure allows manipulating the oscillation time constants of the ring delay elements using current instead of voltage control, which improves stability [12]. Effectiveness of the CsRO operation and the number of delay stages (five) was determined experimentally, based on extensive electrical simulations, and is not discussed here for lack of space. Simulations employ circuit level parameters and consider: (i) the range and (ii) number of produced frequencies; a basic assumption is the use of (iii) minimum size LPSVt transistors and one important goal is to achieve (iv) linearity of frequency steps.

The most energy-efficient way to stop a clock consists in opening the oscillating ring. To enable glitchless clock gating we improved the current-starved architecture. The Controller produces a signal called clk_dis (see block at the bottom right of Fig. 3). To avoid stability problems, the last stage of the oscillator operates as a latch for clock gating, as
Fig. 3 shows. When the ring is opened, the latch formed by inverters I1-I2 holds the output logic level. Besides, a NAND gate filters the clk_dis signal, propagating it only after it is synchronized with the input of I1 and the output of I2. Also, there are adaptations to support a global reset signal with the goal to eliminate possible bubbles in the ring.

The LCG architecture also allows interrupting the clock without opening the ring through a Clock Retain (CR) circuit. This is just a copy clock gating circuit in Fig. 3 at the output of the LCG oscillator, substituting the clk_dis signal by the clk_retain signal and eliminating the ring feedback wire. The activation of the clk_retain signal does not stop the ring oscillation but disconnects it from the clock output. This allows, for example, the adjustment of the PVT compensator (see next Section) by the Controller, without directly interfering on the CsRO.

2) The LCG Actuator

The Actuator is responsible for converting the digital input from the external world or from the Controller block into the analog signal required by the CsRO. Two digital to analog converters (DACs) form the Actuator: (i) one that selects the operating frequency and (ii) another for compensating PVT variations. Since the CsRO assumes the use of current control, the employed DACs work by manipulating current values as well. See in Fig. 4 the Frequency Selector DAC (FSDAC), which produces the vctrl input to the CsRO. It receives as input a 15-bit unary or thermometric code through signal fsel, and based on it generates one of sixteen distinct current values at vctrl. The use of this code guarantees a monotonic conversion process. Starting from a basic reference current source, as many current mirrors as specified by the fsel value are connected in parallel to define the required current.

To produce the required operating frequencies, the classical drain-based switching of current mirrors is substituted by a transistor gate-based switching as observable in Fig. 4. In this way, it is guaranteed that the output current will vary between the minimum and nominal values, never exceeding this later. Consequently, it is possible to limit the maximum frequency generated below or at the specified value. Transistor M2 will be always on, to keep a minimum oscillation frequency. The FSDAC has as inputs the output of the reference current source (the structure to the left of transistor M2 in Fig. 4) and fsel, a 15-bit bus using unary code. The output is vctrl, a voltage defined by the current driven from the CsRO.

The PVT Compensator DAC (PCDAC), on the other hand, enables fine tuning of the current reference, to compensate PVT variations. Since the CsRO is controlled by current, directly manipulating the reference current provides precision in this control process. The PCDAC allows defining 256 distinct levels to adjust the FSDAC current reference. The PCDAC has a circuit topology that is similar to that of the FSDAC. However, differently from the FSDAC it employs a binary code input and a current mode output. Since the supply voltage in the case study implementation is quite low (1.2V) and currents are consequently small, it is necessary to resort to the use of parallel association of transistors, duplicating the current for each bit. Just as in the FSDAC, the PCDAC also has a transistor dedicated to keep a minimum reference current. Before manipulating the PCDAC binary input, the LCG has to disconnect from the driven IP or NoC router, to avoid that transient frequencies occurring during PVT adjustment propagate to these modules. PVT compensation typically occurs at startup (to compensate process variations and lock operation at a given level) and occasionally, when some external subsystem detects that significant changes occurred in supply voltage and/or temperature.

3) The Overall DCO Structure

The block diagram of Fig. 5 details the DCO overall structure. The PCDAC comprises 629 transistors and the FSDAC contains 93 transistors. The CsRO has 74 transistors and the CR circuit just 32. All these 828 transistors are minimum size.

V. EXPERIMENTS

The presented results are all focused in the DCO simulation, due to the fact that this is the main module of an LCG and has a full-custom design. The conducted experiments point to high robustness even at severe PVT variations.

Three scenarios were employed in the simulation of the DCO: worst, nominal and best cases. Table I presents PVT
parameters for each scenario. Note that according e.g. to Bhasker and Chadha [13], deviations regarding the temperature of worst and best case scenarios are common in deep-submicron technology nodes.

Table I. Variables, Values and Corner in Evaluated Cases.

<table>
<thead>
<tr>
<th>Variables / Cases</th>
<th>Worst Case</th>
<th>Nominal</th>
<th>Best Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corner</td>
<td>Slow-Slow</td>
<td>Typical-Typical</td>
<td>Fast-Fast</td>
</tr>
<tr>
<td>Voltage Source (Vdd)</td>
<td>1.08V</td>
<td>1.2V</td>
<td>1.32V</td>
</tr>
<tr>
<td>Reference Source (ν ref)</td>
<td>4.5μA</td>
<td>5μA</td>
<td>5.5μA</td>
</tr>
<tr>
<td>Temperature</td>
<td>-55°C</td>
<td>25°C</td>
<td>125°C</td>
</tr>
</tbody>
</table>

Fig. 6 presents the charts obtained when setting the DCO to operate at its maximum frequency and varying the compensator input along all possible values, during simulation time. The point where this frequency is achieved identifies the necessary compensator value, to guarantee a maximum frequency and varying the compensator input along all possible values, during simulation time. The maximum frequency gap for a given PVT variation is easily covered by the PCDAC.

Currently, two versions of the Controller block are under design, a complex and a simple one. Second, the analog design of the current source for the PVT Compensator DAC has been concluded and its physical layout is under implementation. Ongoing work still includes the design of the physical layout for the DCO.

REFERENCES