



Deliverable – D24

Advantages of GALS-based design

Grant Agreement No:	214364
Project acronym:	GALAXY
Project title:	GALS InterfAce for CompleX Digital System Integration
Funding Scheme:	STREP
Date of latest version of Annex I against which the assessment will be made:	28.10.2008.
Contractual Date of Delivery to the EC:	31. May. 10
Actual Date of Delivery to the EC:	30. Jul. 10
Author(s):	Milos Stanisavljevic (EPFL), Milos Krstic (IHP), Davide Bertozzi (UNIBO)
Participant(s):	EPFL, IHP
Work Package:	WP7
Security:	Public
Nature:	Report
Version:	1
Total number of pages:	61

Abstract:

GALS systems despite having well-known drawbacks such as non-mature design-flow, lack of testing methods and no well-established target application, may offer significant advantages like better EMI characteristics, power reduction and process variability mitigation. These advantages have been explored in detail with theoretical research and chip fabrication and measurements. In addition, a systematic evaluation of GALS methodology benefits in NoC design process is performed.

Keyword list: asynchronous design, chip fabrication, GALS, Networks-on-Chip, low-EMI design, low power design, process variability, subthreshold design



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

<i>Function</i>	<i>Responsibility</i>	<i>Date</i>	<i>Signature</i>
Written by:	Milos Stanisavljevic	15.05.2010	
Checked by:			
Approved by:			

Reserved to EC

Approved by:			
---------------------	--	--	--



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA





GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

CHANGE RECORDS

<i>ISSUE</i>	<i>DATE</i>	<i>§ : CHANGE RECORD</i>	<i>AUTHOR</i>
1	15-May-10	First version	Milos Stanisavljevic
2	25-May-10	Update with low-EMI results	Milos Krstic
3	30-May-10	GALS advantages in NoC design	Davide Bertozzi
4	30-Jul-10	Final version	



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

BIBLIOGRAPHIC RECORD

Project Number:	214364
Project Title:	GALAXY
Deliverable Type:	Report
Deliverable Number:	D24
Contractual Date of Delivery:	31. May. 2010
Actual Date of Delivery:	30. Jul. 2010
Title of Deliverable:	Advantages of GALS-based design
Work package contributing to the Deliverable:	WP7
Authors:	Milos Stanisavljevic, Milos Krstic, Davide Bertozzi
Abstract	GALS systems despite having well-known drawbacks such as non-mature design-flow, lack of testing methods and no well-established target application, may offer significant advantages like better EMI characteristics, power reduction and process variability mitigation. These advantages have been explored in detail with theoretical research and chip fabrication and measurements. In addition, a systematic evaluation of GALS methodology benefits in NoC design process is performed.
Keywords	asynchronous design, chip fabrication, GALS, Networks-on-Chip, low-EMI design, low power design, process variability, subthreshold design
Confidentiality Level	Public
Name of Client:	EC
Distribution List:	GALAXY, EC, internet
Authorised by:	
Issue:	1
Document ID:	D24
Total Number of Pages:	61
Contact Details:	krstic@ihp-microelectronics.com



TABLE OF CONTENTS

1	INTRODUCTION	8
2	REFERENCES	9
2.1	ACRONYMS	9
2.2	REFERENCE DOCUMENTS.....	9
3	EVALUATING GALS CONCEPT IN RESPECT TO DESIGN EFFORTS FOR SYSTEM INTEGRATION AND IMPLEMENTATION	11
3.1	GALS DESIGN FLOW	11
3.2	EVALUATING THE EFFORT FOR EXAMPLE DESIGNS.....	14
3.2.1	WLAN GALS Processor (2005)	14
3.2.2	GALS FFT Processor (2009).....	15
3.2.3	Moonrake OFDM Processor (2009).....	15
4	GALS ADVANTAGES IN SYSTEM IMPLEMENTATION OF DIGITAL LOGIC DESIGN	16
4.1	EMI REDUCTION USING GALS APPROACH.....	16
4.1.1	Introduction.....	16
4.1.2	EMI in Digital Circuits	17
4.1.3	Software for Modeling EMI in Digital Systems.....	17
4.1.4	Modeling GALS	19
4.1.5	Evaluating synchronous systems	20
4.1.6	Evaluating GALS systems	21
4.1.7	Comparing Synchronous and GALS systems	22
4.1.8	GALS granularity effect - from synchronous to asynchronous system	23
4.1.9	A Practical Example	25
4.1.10	Conclusions	29
4.2	LOW-POWER GALS DESIGN - DYNAMIC VOLTAGE AND FREQUENCY SCALING	31
4.2.1	System Architecture and Main Principles.....	33
4.2.1.1	Power Modes Definition.....	34
4.2.1.2	Low Power Manager.....	35
4.2.2	Example of Operation and Proposal for Optimization	35
4.2.3	Unit Power Consumptions Evaluation	36
4.2.4	Conclusion	37



4.3	PROCESS VARIABILITY MITIGATION IN GALS SYSTEMS.....	38
4.3.1	Technique for minimization of the impact of local delay variations	38
4.3.2	Optimization results of variations minimization.....	40
4.3.3	Conclusions	42
5	GALS NETWORKS-ON-CHIP APPROACH ADVANTAGES	43
5.1.1	Tight integration of mesochronous synchronizers	54
5.1.2	Tight integration of Dual-Clock FIFOs.....	56
6	CONCLUSIONS	59

LIST OF FIGURES

Figure 1:	Design flow for GALS.	12
Figure 2:	GalsEmilator – software for modelling EMI in digital systems.	18
Figure 3:	EMI characteristic in the synchronous systems.....	20
Figure 4:	EMI characteristic in GALS systems with different frequencies set and jitter.	21
Figure 5:	Comparing the EMI of the Synchronous and GALS systems.....	22
Figure 6:	Comparing the best results from Synchronous and GALS systems.	23
Figure 7:	Reduction of EMI for different GALS granularity.....	24
Figure 8:	Reduction of EMI in different frequency ranges.....	25
Figure 9:	Architecture of 64-point pipelined GALS FFT processor.....	26
Figure 10:	<i>Amplitude spectrum of current (a) without clock modulation (b) with clock modulation ($f_c=100\text{MHz}$).....</i>	27
Figure 11:	Hierarchical layout of GALS FFT processor.	27
Figure 12:	Peering core VDD ring.....	28
Figure 13:	Amplitude attenuation at clock harmonics.	28
Figure 14:	Amplitude spectrum of core VDD (a) in synchronous mode (b) in GALS mode with clock modulation.....	29
Figure 15:	Throughput versus power for a module in a system.....	32
Figure 16:	<i>GALS DVFS system architecture overview.....</i>	33
Figure 17:	<i>LPM control, sequence example.....</i>	35
Figure 18:	<i>LPM control, sequence example.....</i>	36
Figure 19:	Fastest path evaluator and standard flip-flop.....	39
Figure 20:	Schematic of the reduced overhead FPE flip-flop.	39



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

Figure 21:	Maximum critical path delay distribution for cases without selective redundancy, R=3 and R=3.	41
Figure 22:	<i>Critical path of regular topologies on a 65nm technology.</i>	45
Figure 23:	<i>Speeds achievable with link pipelining.</i>	46
Figure 24:	<i>Tightly coupled mesochronous synchronizer with switch input buffer.</i>	48
Figure 25:	<i>Setup and hold time definitions.</i>	48
Figure 26:	Architecture of the variation detector.	49
Figure 27:	Variation Detector Reset phase.	50
Figure 28:	Timing Margin of the Mesochronous Synchronizer.	52
Figure 29:	Settling time of the incoming data.	53
Figure 30:	Timing Margin of a Mesochronous Synchronizer with a Variation Detector in front of it.	53
Figure 31:	Area of mesochronous switches	55
Figure 32:	<i>Power of mesochronous switches</i>	56
Figure 33:	<i>Switch area comparison.</i>	57
Figure 34:	<i>Switch power comparison.</i>	58

LIST OF TABLES

Table 1:	System partitioning of GALS FFT	26
Table 2:	Frequency modulation parameters	26
Table 3:	Unit Power Modes	34
Table 4:	Relative mean value and standard deviation of the critical path delay.	40
Table 5:	Switch crossing latency	56



1 INTRODUCTION

The CMOS technology miniaturisation will continue, even if increasing difficulties may slow down the pure technological progress. And in fact, the increased complexity, performance requirements, and the need for power and EMI (Electromagnetic Interference) reduction present almost unsolvable challenges to designers of complex embedded systems. The continued technology improvement towards nanoscale dimensions generates additional problems for embedded system design. The combination of complex application requirements and technology imperfections (e.g. process variability and reliability) exacerbate the problems of timing closure and clock tree generation requiring additional design iterations and extended design-to-market time. It is imperative to deal with these issues; one very promising option is the use of a Globally Asynchronous Locally Synchronous (GALS) design methodology.

For the full adaptation of GALS design methodology it is necessary to solve certain issues such as non-mature design-flow, lack of testing methods and no well-established target application. However, despite these issues GALS design methodology may offer significant advantages like much better EMI characteristics, power reduction and better (easier) process variability tolerance.

The advantages (and difficulties in implementation) depend partially on the main application of GALS approach and the way GALSification is performed (what type of GALS interface is implemented). Here, we differ two main applications (which also have different GALS interfaces):

- Digital logic design
- Networks-on-Chip (NoCs)

These two main applications share, of course, many similarities with respect to advantages that GALS approach offers but they deserve to be addressed separately.

With respect to that, this report is organized as follows: in Chapter 3 the evaluation of GALS concept with respect to design efforts for system integration and implementation of digital logic design is presented. The effort is made to evaluate benefits but also drawbacks of GALS design approach. In Chapter 4 GALS advantages in system implementation of digital logic design are investigated with an emphasis to EMI reduction as the most important one. Advantages in possible power savings in low-power GALS applications and process variation tolerance are also explored. In Chapter 5 GALS Networks-on-Chip approach advantages are researched.

This deliverable takes into consideration and thrives to further explain results of previous deliverable such as D11, D12 and D22. It presents advantages and difficulties of GALS implementation as a part of a bigger picture trying to provide some general recommendations about when and why GALS approach should be used. The particular details of GALS approach advantages will be presented in deliverable D25 that reports crossbenchmarking results of fully synchronous vs GALS NoC implementations.



2 REFERENCES

2.1 ACRONYMS

ABB	Adaptive Body Biasing
AFSM	Asynchronous Finite-State Machines
CTS	Clock Tree synthesis
DVS	Dynamic Voltage Scaling
DVFS	Dynamic Voltage-Frequency Scaling
EMI	Electromagnetic Interference
EMR	Electromagnetic Radiation
GALS	Globally Asynchronous Locally Synchronous
IP	Intellectual Property
LCG	Local Clock Generator
LS	Locally Synchronous
NoC	Networks-on-Chip
PE	Processing Element
PWM	Pulse Width Modulation
RTL	Register Transfer Level
STG	Signal Transition Graph
VFI	Voltage-Frequency Island
WID	Within-die

2.2 REFERENCE DOCUMENTS

- [1] J. Muttersbach, T. Villiger, W.Fichtner, "Practical Design of Globally-Asynchronous Locally-Synchronous Systems", Proceedings of the IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC), Eilat, Israel, (2000), pp. 52-59.
- [2] E. Grass, et al., "Enhanced GALS Techniques for Datapath Applications", Proceedings of PATMOS Workshop, LCNS 3728, Springer Verlag, Leuven, Belgium (2005), pp. 581-590.
- [3] S.B. Furber, et al., AMULET2e: "An Asynchronous Embedded Controller", Proceedings of the IEEE, Vol. 87, Num. 2, (1999), pp. 243-256.
- [4] A. Bink, R. York, ARM996HS: "The First Licensable, Clockless 32-Bit Processor Core", IEEE Micro, Vol. 27, Iss. 2, (2007), pp. 58-68.



-
- [5] M. Krstić, E.Grass, C. Stahl, "Request-driven GALS Technique for Wireless Communication System", Proceedings of International Symposium on Asynchronous Circuits and Systems (ASYNC), NY, USA, (2005), pp. 76-85.
 - [6] I. Blunno, C. Passerone, G.A. Narboni, "An automated methodology for low electromagnetic emissions digital circuits design", Proceedings of Euromicro Conference on Digital System Design (DSD), (2004), pp. 540-547.
 - [7] M. Badaroglu, et al., "Clock-skew-optimization methodology for substrate-noise reduction with supply-current folding", IEEE Transactions on CAD/ICAS, Vol. 25, Is. 6, (2006), pp.1146 – 1154.
 - [8] E. Beigne, et al., "Dynamic voltage and frequency scaling architecture for units integration within a GALS NoC", In Proceedings of International Symposium on Networks-on-Chip, (2008), pp. 129-138.
 - [9] M. Badaroglu, et al., "Digital Ground Bounce Reduction by Phase Modulation of the Clock", In Proceedings Design, Automation, and Test in Europe (DATE) conference, vol. 1, (2004), pp. 10088.
 - [10] K. Niyogi, D. Marculescu, "System Level Power and Performance Modeling of GALS Point-to-point Communication Interfaces", Proceedings of the 2005 international symposium on Low power electronics and design (ISLPED), (2005), pp. 381 - 386.
 - [11] Sylvain Miermont, Pascal Vivet and Marc Renaudin, "A Power Supply Selector for Energy- and Area-Efficient Local Dynamic Voltage Scaling", PATMOS'2007, 3-5 September, Göteborg, Sweden, 2007.
 - [12] A. Maxiaguine, S. Chakraborty, and L. Thiele, "DVS for buffer-constrained architectures with predictable qos-energy tradeoffs," in Proc. 3rd IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codes. Syst. Synth. (CODES ISSS), 2005, pp. 111–116.
 - [13] Sylvain Miermont, Pascal Vivet and Marc Renaudin, "A Power Supply Selector for Energy- and Area-Efficient Local Dynamic Voltage Scaling", PATMOS'2007, 3-5 September, Göteborg, Sweden, 2007.
 - [14] J.Hu and R. Marculescu, "Energy-aware communication and task scheduling for network-on-chip architectures under real-time constraints", Proceedings of Design Automation and Test in Europe, DATE'04, 2004, pp. 234-239.
 - [15] M. Es Salhiene, L. Fesquet , M. Renaudin, "Dynamic Voltage Scheduling for Real Time Asynchronous Systems", Twelfth International Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS'2002), September 11-13, 2002, Sevilla, Spain.
 - [16] K. Niyogi and D. Marculescu, "Speed and voltage selection for GALS systems based on voltage/frequency islands," in Proc. ACM/IEEE Asian-South Pac. Des. Autom. Conf. (ASPAC), Jan. 2005, pp. 292–297.
 - [17] S. Miermont, P. Vivet, and M. Renaudin, "A power supply selector for energy- and area-efficient local dynamic voltage scaling," presented at the PATMOS'2007, Göteborg, Sweden, Sep. 3–5, 2007.
 - [18] Advanced Encryption Standard, Federal Information Processing Standards 197 (FIPS 197) (2001)
 - [19] M. Krstic, E. Grass, C. Stahl, M. Piz, System Integration by Request-driven GALS Design, IEEE Proc. Computers & Digital Techniques, Vol. 153, Issue 5, September 2006, pp 362-372.



3 EVALUATING GALS CONCEPT IN RESPECT TO DESIGN EFFORTS FOR SYSTEM INTEGRATION AND IMPLEMENTATION

One of the main motivators for including the GALS concept into design for high complexity nanoscale digital design is of course the need for effective system integration and accelerated design process. It is clear that the advantage of GALS can be visible only for high complexity designs. For other, simpler designs, introducing GALS will only make the design process more complicated. The main indications for introducing GALS are the following:

- The design is of very high complexity, and standard design flow has the problems in timing closure and CTS (clock tree synthesis).
- The design contains the sub-blocks running at different frequencies/phases. Interconnections are needed to integrate mutually asynchronous blocks.
- Implemented design must have low-EMI properties.

If any of those three indicators are valid one should consider involving the GALS design.

Before we evaluate the complexity of introducing the GALS technology, we have to define the GALS design flow and emphasize the differences to the standard synchronous flow.

3.1 GALS DESIGN FLOW

Defining a design flow for Globally Asynchronous Locally Synchronous systems is a difficult task. In general, designs that have asynchronous parts are relatively difficult to implement. A major problem is the lack of support for asynchronous logic designs in commercial EDA tools. The synthesis tools available on the market (apart from Haste offered by Handshake solutions) do not offer any support for asynchronous design. The analysis of critical paths is not possible. Furthermore, synchronous layout tools have special optimization procedures (as, for example, in-place optimization or timing-driven placement) that are not well suited to asynchronous components.

However, several asynchronous EDA tools have been developed and are available to the users. There are two main categories of those tools. First, there are tools for synthesis of hazard-free asynchronous controllers. Some examples are Petrify, MINIMALIST, and 3D. In general, these tools are a good basis for the synthesis of relatively simple asynchronous controllers. They do not offer support for designing large systems, and they do not integrate any simulation tool. On the other hand, those tools are very useful help when some relatively simple asynchronous component has to be generated. The other category of EDA tools offers a complete design framework for system description, simulation and synthesis. Examples are TANGRAM (recently offered commercially to the market with the name HASTE), BALSAs and TAST. The main obstacle for their application in the GALS area is that they are directed towards asynchronous designs and not to mixed synchronous-asynchronous designs. However, in the context of GALAXY project we have considered the extension of BALSAs also to mixed synchronous-asynchronous system that we actually need for GALS design. Therefore, GALAXY BALSAs is one important tool in our design flow.



Therefore, we will define the possible framework for GALS system design. This design flow should be based on existing tools, combined with additional scripts. The proposed design flow for developing our GALS system is a combination of the standard synchronous design-flow with addition of specific asynchronous synthesis tools. However, most of the tools are taken from the pure synchronous world. The reason for that is simple: in general, the asynchronous part of the GALS circuitry is very small in comparison with the synchronous part. Accordingly, simple asynchronous circuits can be generated with the use of asynchronous synthesis tools and then embedded in complex synchronous blocks.

A graphical view to our design flow is shown in Figure 1.

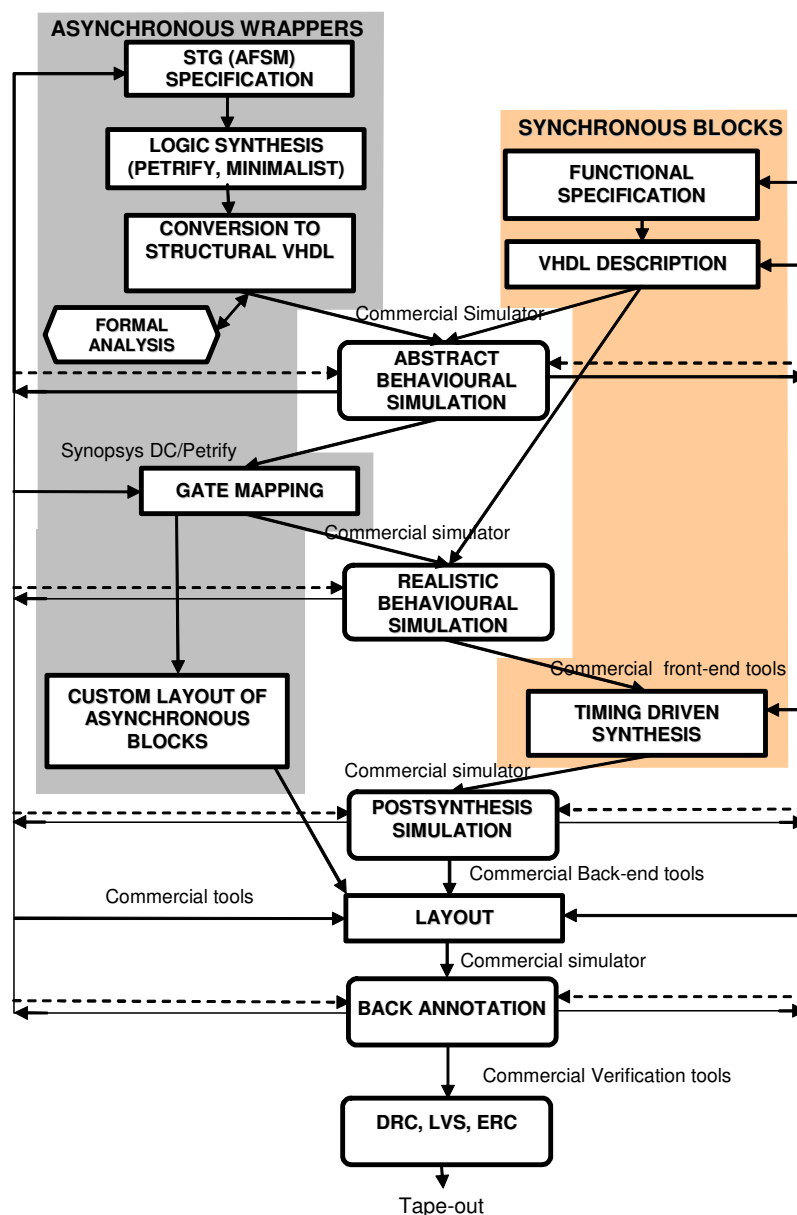


Figure 1: Design flow for GALS.



All asynchronous controllers should be modelled as asynchronous finite-state machines (AFSM) or Signal Transition Graphs (STGs) and subsequently synthesized using the tools such as Petrify, Minimalist, 3D or GALAXY Balsa. After that, a formal analysis of the asynchronous controllers should be performed in order to verify the hazard-free operation of the complete asynchronous wrapper. Asynchronous components has to be then transferred into netlist (manually or using some of the previous mentioned tools). Finally, those small asynchronous components should be if possible custom layouted and verified on the transistor level in order to avoid any possible hazard condition.

In order to generate a complete and reliable flow for GALS design, two types of behavioural simulations are performed. Initially, a functional verification is performed at the level of VHDL descriptions. This simulation run should confirm the conceptual correctness of the system. Accordingly, for this simulation run, the complete synchronous part is modelled in behavioural VHDL code. Asynchronous wrappers are modelled in structural VHDL. Additionally, in this structural code, some artificial delay is assigned to every logic operation. This is done in order to avoid races between the signals in the asynchronous part. On the basis of the synthesized asynchronous parts and VHDL description of the locally synchronous (LS) modules, a realistic behavioural simulation is performed after the abstract simulation. The complete asynchronous wrapper is annotated with a SDF file (based on the typical PVT values). This simulation will, with high probability, prove the correctness of the system function, but also can be performed very fast, because all LS modules are still modelled in behavioural VHDL.

The front-end synthesis is performed using the standard tools by simple linking to already mapped asynchronous components. A complete gate-level simulation is performed using both asynchronous and synchronous parts as synthesized netlists. This simulation is similar to any after-synthesis simulation in the purely synchronous world. Of course, due to the large number of annotated gates, this simulation run is rather slow in comparison with the one previously described. Accordingly, it should be performed only when all possible problems and hazards from the behavioural simulation have been resolved.

Layout is done using conventional synchronous layout tools. For layout we have to use the specific scripts that help the tool not to destroy the timing of the asynchronous paths. After layout, back-annotation is performed. Behavioural, post-synthesis simulation, and back-annotation are performed using a standard VHDL-Verilog simulator. Finally, the verification (LVS, ERC, DRC) is performed using the conventional tools.

Finally, we observe the part of the design flow, that deals with the asynchronous components marked in gray in Figure 1. In addition to the standard approach, we should additionally design, verify, map, layout and characterize the asynchronous components. Since they are relatively simple (couple of gates), this is not very much complex and has to be done only once. When we perform this, the component design can be replicated for each GALS design in the same technology. Additionally, some back-end scripts have to be extended such that the optimization from the synchronous CAD tools does not destroy the timing of asynchronous paths.



3.2 EVALUATING THE EFFORT FOR EXAMPLE DESIGNS

The members of the project consortium have the broad experience in designing the GALS systems. Based on this experience, built in the background and also during the project, we could evaluate the complexity of introducing the GALS technology into the system design. The following analysis is based on the experience from WLAN baseband chip designed in IHP in 2005 using IHP 0.25 μm CMOS process, GALS FFT chip (GALAXY project) designed in 2009 using IHP 0.13 CMOS process, and Moonrake 60 GHz transmitter GALS chip (GALAXY Project) designed in 2010 using 40 nm Infineon library.

3.2.1 WLAN GALS Processor (2005)

The GALS WLAN baseband chip was designed in IHP in 2005 using IHP 0.25 μm CMOS process. It includes the full baseband functionality. This is a relatively complex design for this technology with the area of 45 mm^2 running at 80 MHz. This chip was a feasibility study of a request-driven GALS technique. Since a synchronous version of the same system has been designed, the GALS version could be used to compare the design methodology.

In principle, the GALS design process was faster than the usual synchronous one. Dealing with smaller design blocks was not as difficult as with the synchronous chip. The clocking of the synchronous chip was relatively complex with three different clock domains (80, 40, 20 MHz), with clock divider and clock gates on-chip. With GALS approach, the challenges like global clock tree generation with an enormous number of leaves, clock divider and handling of clock gating has simply disappeared. Clock skew within smaller clock domains was significantly reduced (660 ps \rightarrow 486 ps). However, with more stringent constraints; even better results can be achieved. Since there is no global clock tree, timing closure of the complete design was achieved much more easily.

However, during the design of the GALS, several new design issues appeared. The main difficulties were lack of tool support for asynchronous components, or immaturity of asynchronous tools. For example, due to the limitations of the CAD tool, a direct gate-mapping of the generated logic equations was not possible. Therefore, many operations had to be performed manually. This degrades the performance of the final design and introduces additional delay in the design process. Furthermore, the wrapper evaluation and improvement was performed in parallel to the GALS chip design. These issues caused some iterations of the GALS design process.

Testing of the GALS chip was a problem when using a standard synchronous hardware tester. Special BIST logic was embedded in order to allow the use of a classical hardware tester. We had to perform a special calibration of the ring oscillators during testing in order to match results of the testing and simulation.

The implementation of the GALS system resulted in a hardware overhead of around 3% for the asynchronous wrappers. Power measurements of both chips showed just a marginal improvement of 1% for the GALS case. On the other hand, supply variation noise measurements showed a clear advantage of the GALS solution. The absolute maximum of the power spectrum of the GALS circuit is about 5 dB lower than the absolute maximum for the synchronous circuit. This was achieved without any special setup for asynchronous modules such as adjusting clock phases for different blocks. However, the general conclusion is that GALS led to a certain simplification of the system integration process.



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

3.2.2 GALS FFT Processor (2009)

GALS FFT chip was, in the framework of the GALAXY project, designed in 2009 using IHP 0.13 CMOS process. The chip complexity was around 3 mm², but less than 30% was actually used by the core. The main intent of this chip was to evaluate the possibilities for EMI reduction with GALS, and this was actually achieved, having 13 dB reduction of best GALS mode in comparison to the synchronous mode. However, due to the low complexity of this design, we have to spend comparatively high effort in developing the asynchronous wrappers. Therefore, from the system integration point of view approximately 50% of the time is spent to GALS introduction.

3.2.3 Moonrake OFDM Processor (2009)

Moonrake 60 GHz transmitter GALS chip was designed, in the framework of GALAXY Project, in 2010 using 40 nm Infineon library. The design is highly complex, with the total area ending with 9 mm² in this advanced scaled process clocked effectively with 160 MHz.

Since this is a part of still ongoing design process at this point we could only evaluate the design effort for the front-end part. The basis for this baseband design was FPGA implemented VHDL code that was already ready before starting the ASIC chip. However, the process of migration of this code to the style applicable to the 40 nm library and performing of logical synthesis, and DfT caused the intensive implementation efforts. On the other hand, design of GALS components, since their architecture was already known from the previous projects, was comparatively small. We estimate those additional GALSifying efforts to approximately 15% of the time needed to perform full front-end design.

As a conclusion, we can say that for high complexity designs the additional effort pays it off over the relaxation of the global timing, and ease of the timing closure. Of course, the GALS will bring also certain level of EMI reduction, and potentially power reduction and resistivity to process variations.



4 GALS ADVANTAGES IN SYSTEM IMPLEMENTATION OF DIGITAL LOGIC DESIGN

The biggest advantage of GALS based implementation of digital logic design is certainly EMI (Electromagnetic Interference) reduction and this is the main focus of this Chapter. In addition, advantages in possible power savings in low-power GALS applications and process variation tolerance are also explored.

4.1 EMI REDUCTION USING GALS APPROACH

Since the reduction of EMI was already a subject of the deliverable D11, here we will give just, in a more compacted way, a summary of the information provided already in the respective detailed report. In addition to that we will provide some more details of GALS evaluation in respect to partitioning that was performed after the deliverable release. More information about low-EMI techniques can be found in respective report D11.

4.1.1 Introduction

Continuous development of the process technologies and device miniaturization imposes significant challenges on designers and tools. The classical synchronous paradigm became very difficult to achieve due to the problems in clock tree generation and timing closure. For mixed digital-analog designs, the challenges are even harder. Analog systems are very sensitive to the noise introduced by the digital components. Therefore, additional methods must be used for lowering of EMI introduced by simultaneous switching of the entire clock network.

The GALS methodology has been proposed as a solution for the system integration many years ago and mature solutions are already available [2]. There have been some proposals to use the GALS methodology also for EMI reduction [2]. It has been shown on several examples that asynchronous design can significantly reduce EMI in comparison to the classical synchronous approach. There are several demonstrators of this noise reduction with asynchronous methods, including the designs of ARM processors [3, 4]. There are also some initial studies regarding the GALS approach for EMI reduction [2]. They have shown that the GALS systems can achieve EMI reduction of around 20 dB in comparison with a synchronous design. In time domain, the noise peaks can be lowered up to 40%. However, the real on-chip measurements [5] have shown smaller EMI reduction with GALS approach. Additionally, those activities were not systematic and have been focused only on specific design cases, not taking into consideration GALS as a general methodology for system integration. The technology advance and further device miniaturization increases a demand for deep investigation of EMI because of its detrimental influence on a whole system performance.

Moreover, there are no dedicated tools to model EMI in GALS and synchronous circuits on a high abstract level. It is needed to have the possibility to predict at least approximate values of EMI in designed digital systems. In synchronous systems, which are much more



evaluated, it is easier to estimate EMI. There are many investigations showing the possibility of reduction of EMI in synchronous systems by adding clock skew and phase modulation of the clock [6, 7].

The aim of this work is evaluation of EMI features for GALS/asynchronous and synchronous design style. A first step to make this is to create software able to model and evaluate EMI in synchronous and GALS systems. In this report, we present a software tool able to simulate the EMI behavior caused by clock activity in GALS and synchronous systems on a very high abstract level. Several GALS topologies have been evaluated and compared to their synchronous counterparts. We have also analyzed the effect of partitioning to the EMI-reduction. In this respect we can estimate the EMI-reduction for the system with different partitioning starting from a synchronous systems and ending up with fully asynchronous systems. Finally, we are providing here an example of practical realization of low-EMI GALS system including the verification and measurements of EMI data.

4.1.2 EMI in Digital Circuits

EMI in digital systems is caused by the simultaneous switching of logic cell components. Each active edge of a clock pulse, in a synchronous system, triggers all flip-flops and complete clock buffer tree that generate noise. This triggering is not exactly at a same moment because of the clock tree and its skew that spreads triggering of the flip-flops in time. EMI generated by the digital circuits can be analyzed in the easiest way by analyzing the supply current shape. Reduction of EMI is possible in several ways including improvement of physical elements. However, we have focused here only on a modification of clock behaviour by adding jitter or a fixed phase to each sub block.

Two techniques are mainly applicable to reduce EMI in synchronous systems, when we only concentrate on modifying the clock behavior [2]. First we can add a clock phase shift to each LS block. Phase shift could decrease the global current peaks of a digital circuit, thus reducing EMI. Additionally, we can add jitter to the clock source. Jitter introduces a phase modulation (rapid phase fluctuations) to a clock wave from cycle to cycle influencing EMR (Electromagnetic Radiation) [9]. It modifies slightly the timing event when the rising clock edge appears, while the duty cycle and average clock period remains constant. Hence, jitter can slightly increase or decrease the clock period of a particular cycle, but generally the average base frequency remains the same. The jitter can be introduced in the clock networks in several different ways. Clock oscillators, without PLL are also introducing significant jitter. Additionally, pausable clocks, based on a ring oscillators, as in [1] are significant jitter sources [10]. Alternatively, we could also build a digital jitter generator. For this purposes a Linear Feedback Shift Registers (LFSR) can be utilized as a Pseudo Noise Generator (PNG) [2]. In GALS systems with a pausable clock, we can only add jitter to a system. Integrating a phase shift would be a useless procedure. Variable phase shift between the asynchronous blocks is already present there by the nature of the GALS methodology [1].

4.1.3 Software for Modeling EMI in Digital Systems

In order to evaluate EMI features of the digital systems caused by the digital clock behavior, we have generated a special software tool called "GalsEmilator". GalsEmilator is a program created in Matlab in order to investigate EMI in GALS systems with various structures and topologies of (including synchronous solutions). This software is able model, as precisely as



possible, the different parameters of each GALS/synchronous system, on a high abstract level. Hence, we are able to observe the noise behavior in frequency and time domain.

In the developed tool, we are enabling the modeling of different supply current shapes for different clock cycles. The supply current profile could be modeled as triangular shape or as a superposition of different triangular shapes. In software it is possible to describe up to five different supply current profiles and specify the probability of their appearance in the system. Those profiles are then periodically applied on the clock behavior of the system, depending on their defined probability.

For each block of the synchronous system, we can also set: the local clock frequency, a clock phase shift (in respect to the global clock frequency) and additional jitter. For GALS modules, we can model extra clock jitter and also model pausable clocking [1], as a dominant technique for low-EMI GALS circuits. Our model of GALS with pausable clocking allows us to simulate GALS wrappers that can pause the clock in order to perform a handshake operation. The behavior can be modeled by setting a probability of pause occurrence and a maximum delay of the pause. The delay is variable and, therefore, in our model it is randomized. Additionally, we have the possibility to automatically extract clock behavior directly from RTL simulation. With this approach we can guarantee the equality of the simulation model with the GALS circuit.

All results, in timing and in frequency domain, as shown in Figure 2, can be observed and analyzed both graphically and in a generated table. The complete software has its own user-friendly GUI.

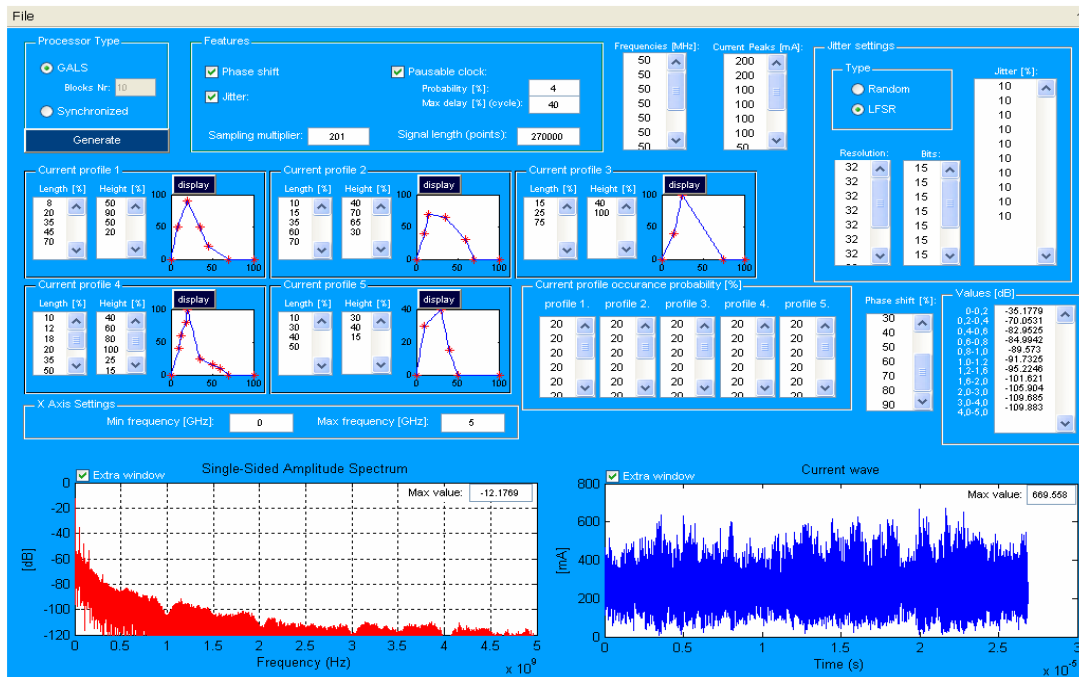


Figure 2: GalsEmilator – software for modelling EMI in digital systems.



4.1.4 Modeling GALS

GALS methodology has been proposed as an effective solution for system integration of complex digital electronics circuits. The basic GALS paradigm is based on a system composed of number of synchronous blocks designed in a traditional way. However, it is assumed that clocks of such synchronous systems are not necessarily correlated, and consequently that those synchronous systems communicate asynchronously using handshake channels. The major differentiator between the different GALS techniques is the strategy used to safely transfer data between the locally synchronous blocks and avoid metastability. In principle, a GALS approach can be implemented in different ways: using pausable clocking, boundary synchronization or FIFO-like interface structure.

We have utilized our software to model and evaluate EMI of different GALS and synchronous systems. For the GALS approach, we have concentrated on the pausable clocking scheme, that is very commonly used in today's GALS NoC systems [1, 8]. In order to exactly model the clock behavior of the GALS system, all evaluated systems have been described in VHDL and simulated. The clock behavior is automatically extracted from the simulation using our software tool. Data is directly fed to the EMI analyzer and evaluated.

The behavior of the clocking changes dramatically with the intensity of the data transfer since the clock pausing appears only during the data transfer process. Therefore, we have modeled three different scenarios of the system behavior:

- Low data transfer, where the data transfer is performed relatively rarely
- Medium-to-high data transfer, where half of the clock cycles are involved in data transfer
- Burst mode, where 80% of the clock cycles are data transfer related.

In our evaluation four different structures of GALS circuits have been analyzed. We have analyzed different system topologies. Here, we have taken into consideration point-to-point, star and mesh topologies. In order to check if a granulation can influence the reduction of EMI, we have also examined the star with a large number of blocks. The goal was to evaluate different interconnect structures and to see their impact on EMI in GALS systems.

For the system modeling, we have defined a base frequency which is a median for all other derived clock frequencies in the system. It was needed to define such frequency to be able to compare a synchronous system that normally operates in a single clock domain with a GALS system that is usually triggered with different clocks. In our model, as an example, this base frequency was 50 MHz.

GALS systems can be implemented very differently. Some systems may use very similar clock frequencies for local blocks (plesiochronous clocking). On the other hand other systems may have totally different LS clock frequencies. Generally, we have used three different frequency sets for modules in our GALS systems. The first set represents plesiochronous operation where the frequencies of each block are almost the same as the base frequency. In the second set, the difference is higher and in the third one, we have frequencies ranging up to a ratio of 1:3.

It can also make a difference which block has the slowest and which block the highest frequency. In particular, for the star topology, if the slowest block is in the centre, the complete



system will be very slow and vice versa. We have tested, therefore, the star topology with both slow and fast setting.

In all cases, we have used 10 GHz sampling frequency. In the model, we have used equal probability of occurrence for each of the five modeled supply current shapes. Also, we have used the same jitter settings for each simulation, with the LFSR length of 15 bits. The sum of the current peaks in all cases was the same in order to correctly compare results. In the 4-module systems we have defined that the peaks were 100 mA, 400 mA, 200 mA, and 300 mA. In 10-module systems the distribution was following: 2 X 200 mA, 4 X 100 mA, 4 X 50 mA.

4.1.5 Evaluating synchronous systems

In Figure 3, we can observe the EMI characteristic and its reduction for different synchronous systems. In particular, we can see the EMI reduction in the synchronous system when jitter is applied. As we can notice, the jitter reduces only higher frequencies starting from 400 MHz. It has no significant influence to the lower spectral range. However, to achieve the reduction, the circuit should be able to be immune to 10% jitter, what is sometimes even not possible in a synchronous design, both for hold and setup time optimization. Moreover, Figure 3 shows the effect of adding a phase shift to a system. We have modeled cases with 0%, 10%, 20%, 30% phase shift of a clock period. By introducing a phase shift to a circuit, we can alleviate EMI in a low frequency range, thus reducing current peaks. The high frequency spectrum remains not significantly changed compared to a basic synchronous approach. The tests were conducted also with a combination of jitter and phase shift. The results are promising because they incorporate advantages of both features. However, it would be a real challenge to guarantee safe data transfer between the blocks in such synchronous system. For many applications, such as processor pipeline, introduction of such low-EMI techniques (jitter, phase shift) is not possible at all due to the difficulties in timing closure.

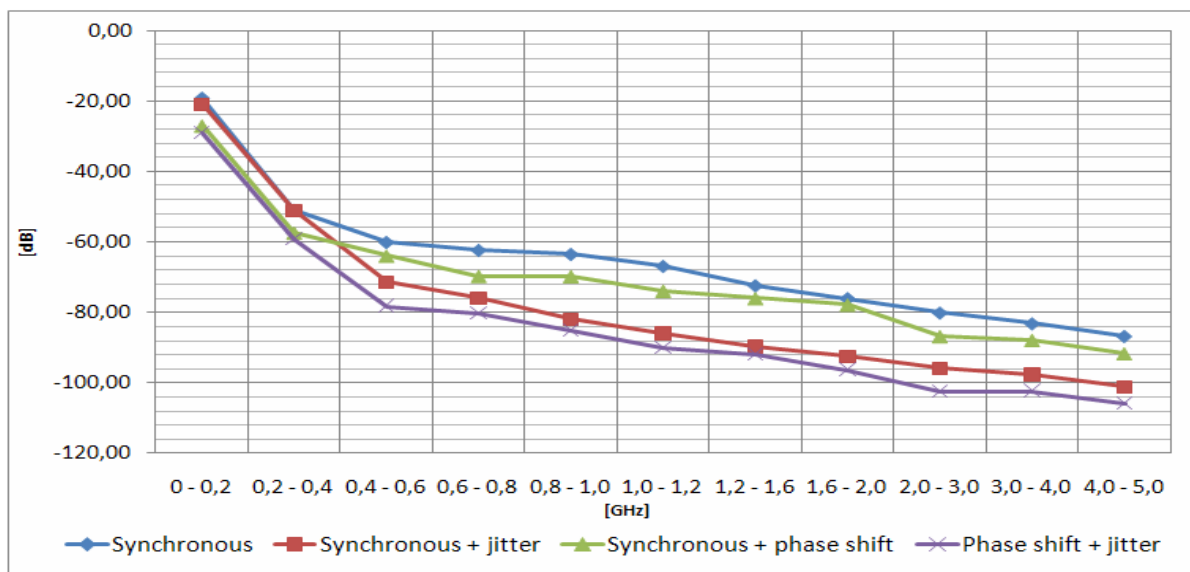


Figure 3: EMI characteristic in the synchronous systems.



4.1.6 Evaluating GALS systems

Figure 4 represents results of comparing two sets of frequencies with medium data transfer and also showing the behavior for low granularity systems (4 GALS blocks) and high granularity systems (10 GALS blocks). We can observe that plesiochronous systems can achieve a significant EMI reduction in the high frequency range. However, it doesn't improve EMI for low frequency operation. On the other hand, the GALS system, with a larger difference of frequencies, reduces much better EMI at low frequencies in a spectrum, still preserving good parameters for higher frequencies. The test performed on various transfer rate with the same frequencies has shown very low differences in EMI reduction. The range of the result variations didn't exceed 5 dB. The clock behavior is automatically extracted from VHDL simulation at RTL level.

Figure 4 shows the EMI reduction in GALS systems with added jitter. We can observe that there is almost no impact of jitter at low frequencies. However, the higher frequency range is more attenuated giving a better reduction of EMI. Every set of frequencies has a similar spectrum starting from 400 MHz and we can observe reductions around 15 dB.

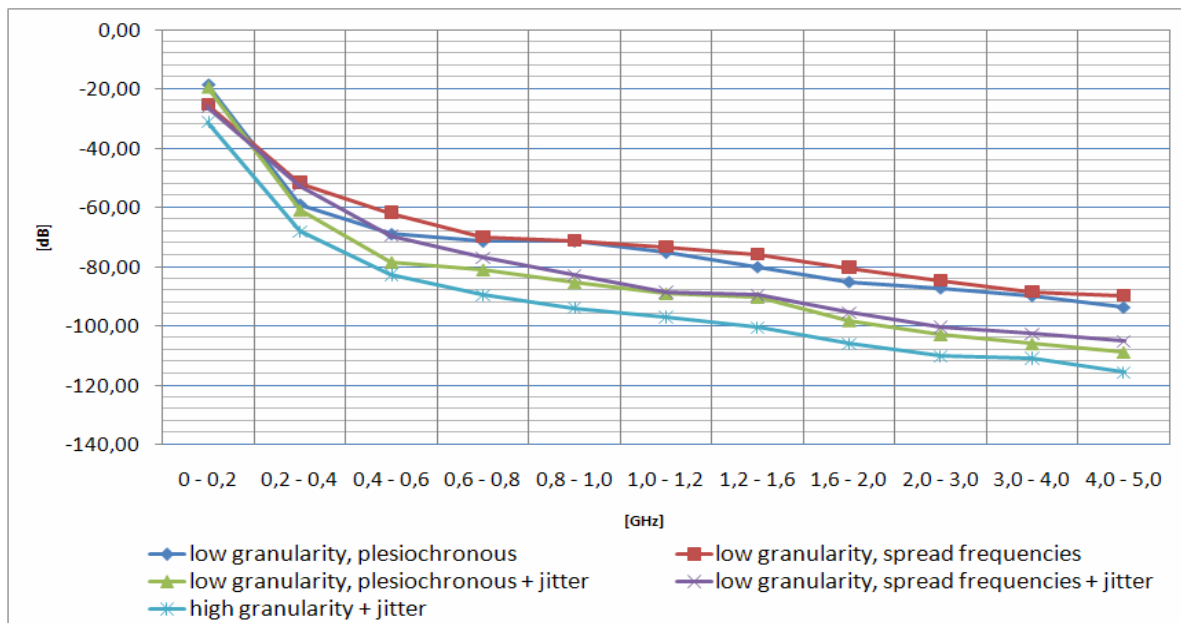


Figure 4: EMI characteristic in GALS systems with different frequencies set and jitter.

Comparing different topologies of 4-modules GALS systems: point-to-point, mesh and star, we have noticed a similar behavior. There is a very little influence of data rate transfer intensity on EMI reduction. The most important parameter is the frequency of LS blocks. The more frequency spread, the better EMI reduction in the lower spectrum. Moreover, in each case jitter has a positive effect on reduction of EMI. It significantly reduces higher frequencies in a spectrum. The average reduction of EMI for all 4-module topologies in comparison to a



totally synchronous system is around 20 dB starting from 400 MHz. We can extract similar results from a star topology with more satellite blocks. However, we notice a greater difference in EMI reduction after adding jitter. The worst results are observable with the frequencies set, where the center block is the slowest one. The best results, in respect to EMI, are achieved when the center block is the fastest one. Hence, we can conclude that the most reasonable architecture from the point of view of the performance, with the fastest center block, is also the most appropriate approach for EMI reduction. We have also compared the effect of block granularity to the final results. Comparing 4-module and 10-module GALS system, we can observe the better results for the more granular design. In Figure 4 this can be clearly observed. The gain reduction is around 5 dB. In general, it means that the finer the granularity of the GALS system the better reduction of EMI.

4.1.7 Comparing Synchronous and GALS systems

Figure 5 shows the comparison between the synchronous and GALS case. We have selected a standard synchronous system, synchronous system with phase shift and jitter, and a GALS system with high block granularity and with jitter. We can notice that the results of the low-EMI synchronous system are around 5 dB worse than the GALS system with the best EMI characteristic. In the low-EMI GALS system EMI is reduced around 25 dB compared to the classical synchronous approach. If we compare the differences in time domain for the classical synchronous and the low-EMI GALS system (10 blocks, star topology with jitter), we can observe 40% current peak reduction.

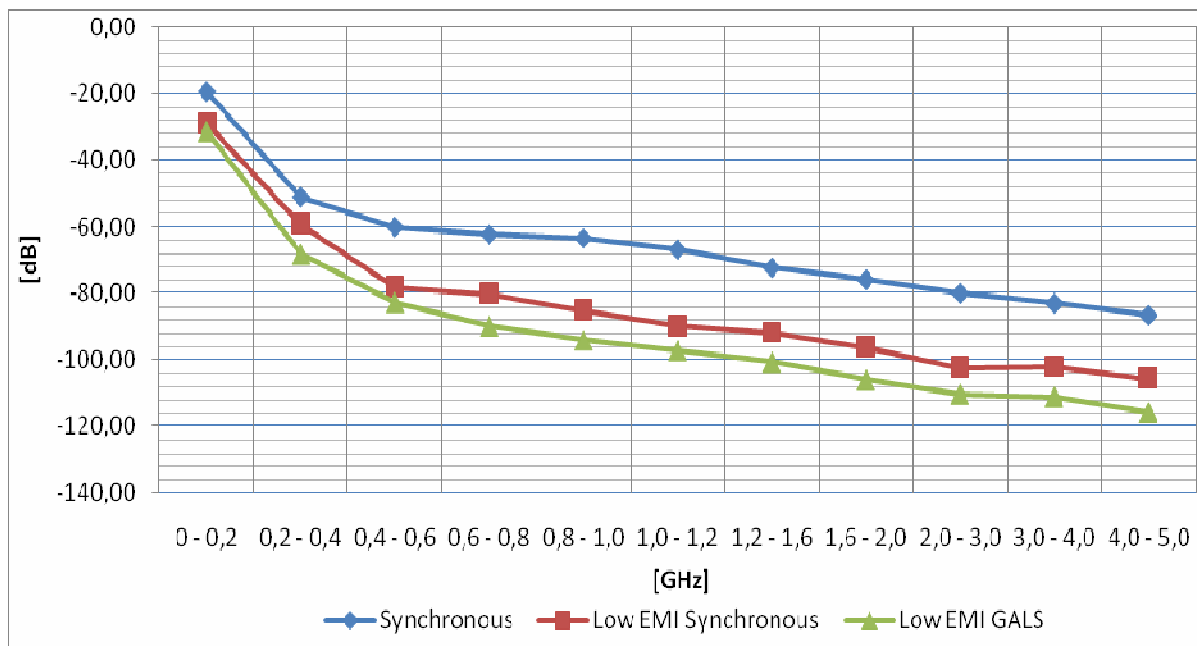


Figure 5: Comparing the EMI of the Synchronous and GALS systems.

We were also comparing the best results of low-EMI synchronous and low-EMI GALS



approach, as shown in Figure 6. Those results could be achieved when the phase between different synchronous blocks is optimally set (i.e. no overlap in energy peaks between the different blocks). For this case we can see that the best low-EMI synchronous and low-EMI GALS profile give very similar results. However, one needs to have in mind that low-EMI synchronous with clock phasing as applied there is based on the same paradigm as GALS system (and it is actually a GALS system without synchronicity between the modules), but without interfaces to enable such operation (as opposed to GALS solution). Therefore, the timing closure of such low-EMI synchronous system is really challenging.

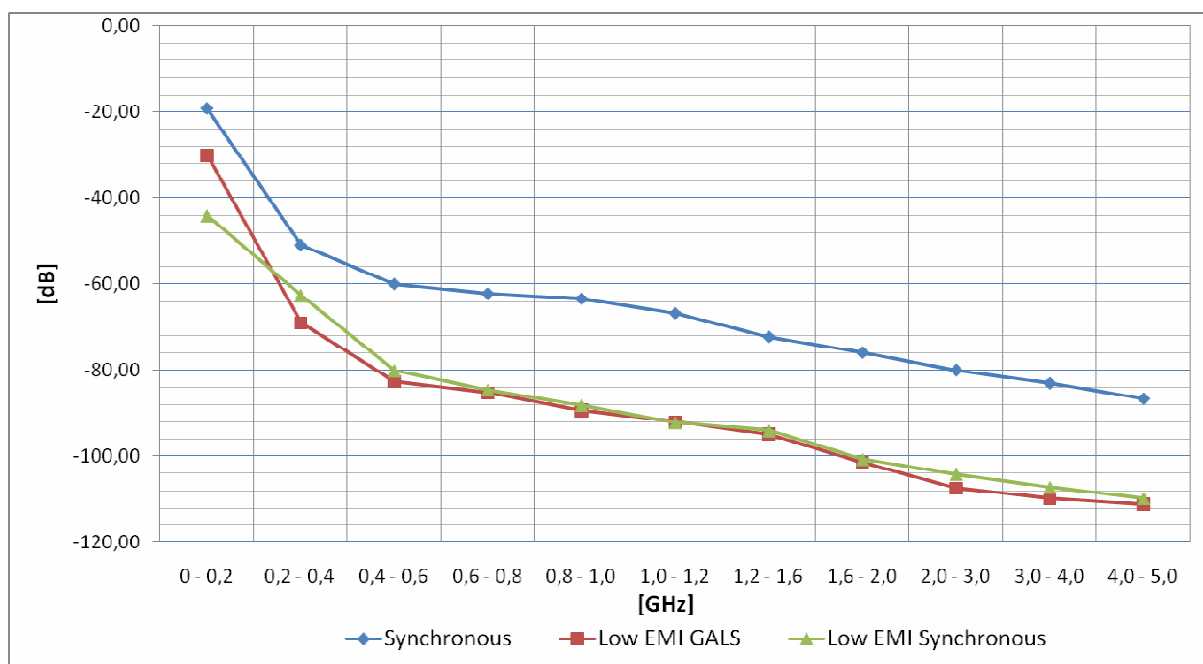


Figure 6: Comparing the best results from Synchronous and GALS systems.

4.1.8 GALS granularity effect - from synchronous to asynchronous system

In the previous section we have shown in the simulations that GALS granularity has a significant effect on EMI profile. However, it is very important to quantify this effect and to observe to which level we could reduce the EMI by deeper GALS partitioning of the system. In order to perform this task we have made the series of simulations based on the same starting system. Since GALSification could have many different parameters (granularity, frequency setup of the local blocks, jitter effects, topology of the resulting system, etc.), we have made the following analysis. The starting case was a pure synchronous system without any sub-blocks. We have defined the base frequency for this system (as in the pervious simulations we have taken 50 MHz as a base frequency), and current profile. Afterwards we were introducing GALS partitioning and increasing the number of GALS blocks, but keeping the same total energy of the system. The partitioned blocks are modeled such that they have plesiochronous behavior around the same median synchronous frequency.



GALAXY

GALS InterfACE for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

The results of those simulations are given in Figures 7 and 8. Both figures show the dependency of EMI in different frequency ranges for different GALS granularity. In Figure 7 we can see how granularity affects the EMI reduction. We can see that after partitioning with 15 different blocks that differences are negligible and that EMI profile saturates. This we can also more clearly see on Figure 8, where we have made the different curves for different frequency ranges. We can see there saturation of EMI in low frequency range after 18 blocks, and in high frequency range already at GALS partitioning with 15 blocks. Achieved maximal reduction in low frequency range was around 25 dB, and in a high frequency range 22 dB.

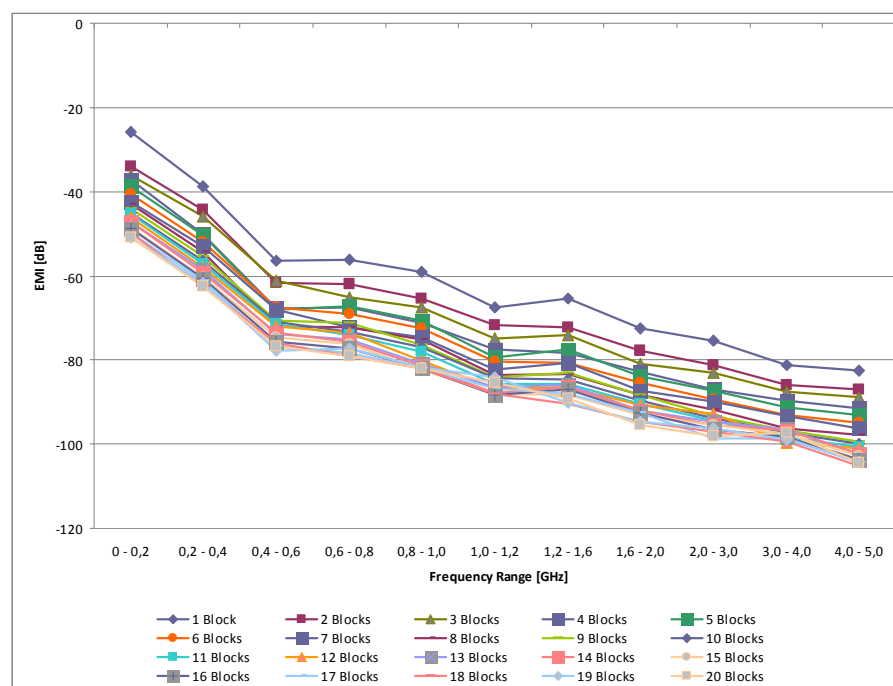


Figure 7: Reduction of EMI for different GALS granularity.

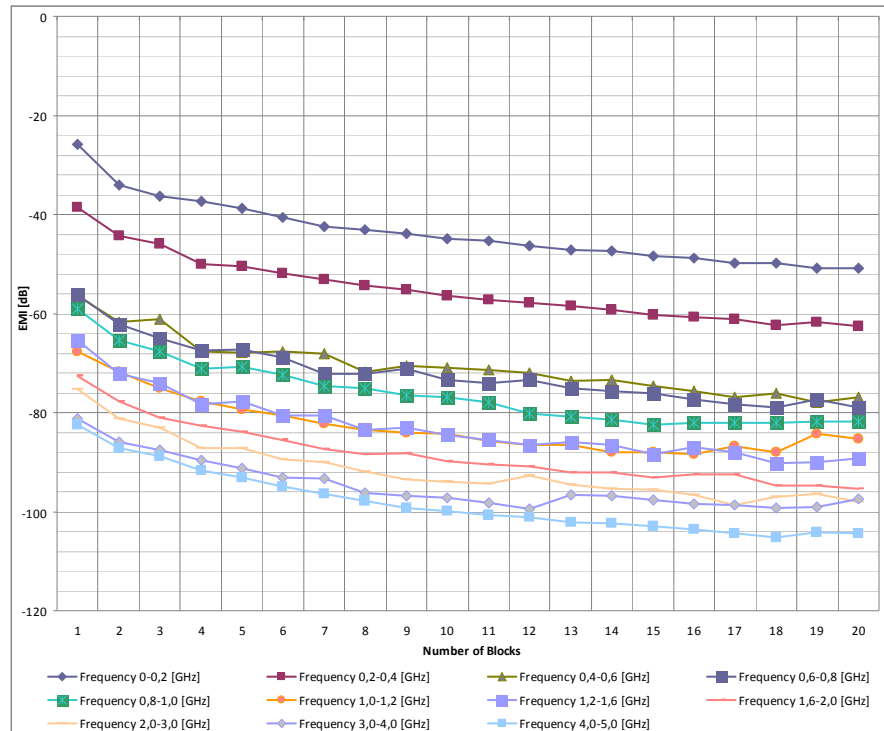


Figure 8: Reduction of EMI in different frequency ranges.

Finally, we can extrapolate those results to evaluate the possibilities for EMI reduction also for the pure asynchronous systems. In general since we see that the system saturates after deeper partitioning we can also estimate the EMI features of the asynchronous systems. Actually, asynchronous system could be presented as GALS system with infinite number of GALS blocks. Indeed we could expect that this system could behave similar as GALS system in saturation.

4.1.9 A Practical Example

As a practical example, a 64-point pipelined GALS FFT processor based on pausable clocking scheme with phase and frequency modulation was implemented and tested. To minimize hardware consumptions, the 64-point FFT was divided and conquered by two cascaded 8-point FFT computation. Low-EMI FFT design is of particular importance for the OFDM communications systems, where large size FFT is required for high data throughput. Hence low-EMI FFT is necessary for substantial mitigation of the unintentional radiated emission from the OFDM systems.

It is obvious that the optimal performance in EMI reduction can be achieved by clock modulation if all the GALS blocks have similar contributions to supply currents. However, to accurately estimate the currents is usually difficult and time-consuming for complex digital systems. As a solution, instead of the current estimation, dynamic power analysis based on the netlist simulation waveforms was performed using Synopsys PrimeTime. To guarantee the analyzing accuracy, quite small timing interval, about 100ps here, was set. Based on the dynamic power analysis, the average power and current of each functional block were estimated, which is accurate enough for system partitioning.



The partitioning of 64-point pipelined FFT design is presented in Table 1. There are totally 4 GALS blocks deployed in the system which are average in terms of supply current. And Figure 9 illustrates the architecture of pausable clocking based GALS FFT processor.

Table 1: System partitioning of GALS FFT

	GALS B1	GALS B2	GALS B3	GALS B4
Functions	BF 1	BF 2/3	CMULT	BF 4/5/6
Area	38556 μm^2	46119 μm^2	47459 μm^2	40547 μm^2
Avg. power	1.2mW	1.5mW	1.2mW	1.7mW
Avg. current	1mA	1.25mA	1mA	1.4mA

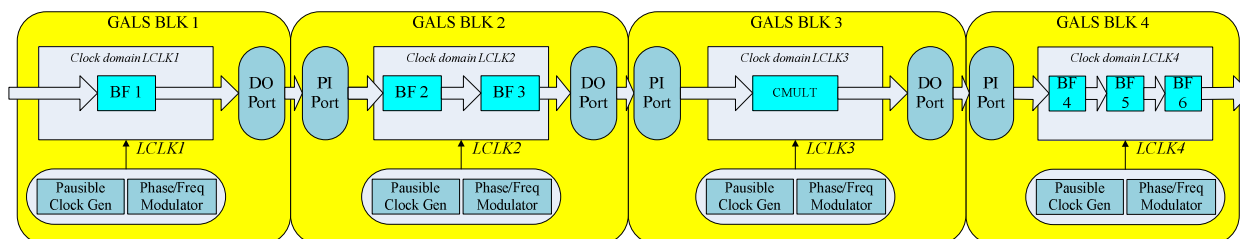


Figure 9: Architecture of 64-point pipelined GALS FFT processor.

As a pipelined design, all the 4 GALS blocks were configured to run at the same frequency. To spread the switching activities of GALS blocks, a skew as long as a quarter of clock period is inserted in between each pair of cascaded GALS blocks. The triangular frequency modulator was applied in the experiment. Table 2 presents the modulation parameters.

Table 2: Frequency modulation parameters

N	$\geq D$	$\geq f$	f_c	f_m	β
8	200ps	10.2Mz	80MHz	2.5MHz	4

Direct simulations on the radiated emission require accurate models of parasitic inductance as well as capacitance of both the packaging structures and the board power traces. However, considering that the far-field radiated strength either increases linearly with the current frequency in common-mode or with square of the current frequency in differential-mode, hence current spectrum can be investigated instead in terms of EMI noise reduction. In previous sections, the MATLAB-based tool GalsEmulator was described to evaluate the current spectrum with clock modulation in SYNC and GALS digital systems. Figure 10 presents the amplitude spectrum of GALS FFT processor with above mentioned phase and frequency modulation by GalsEmulator. Significant reduction is shown over a large spectral region.

Since the performance of local clock generators and asynchronous I/O port controllers are very sensitive to interconnect delays, hierarchical layout was performed for the GALS FFT design. Clock generators and port controllers were separately placed, and the functional



blocks were placed as a multi-clock synchronous core. Manual placement and routing were made on the top level as shown in Figure 11. Note that the port controllers (not highlight in Figure 11) were placed very close to the clock generators to minimize handshake signal delays.

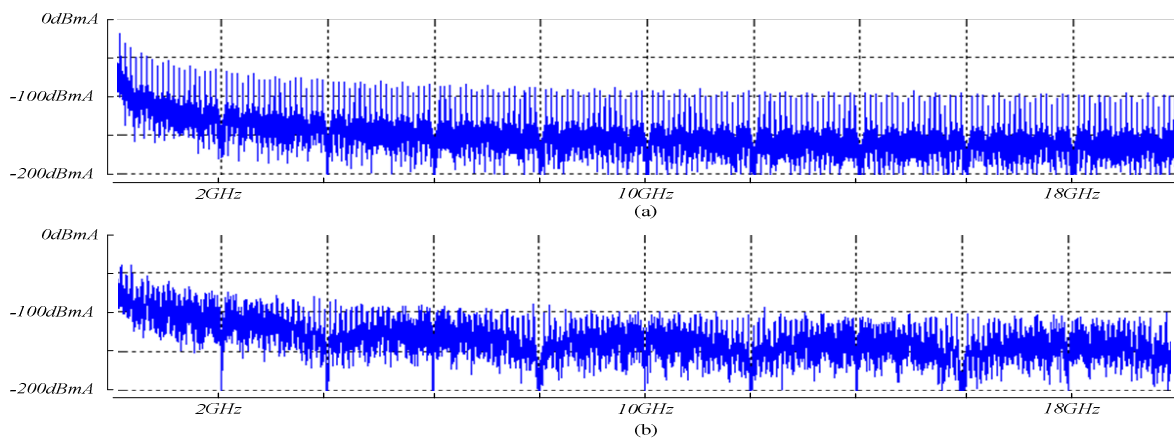


Figure 10: Amplitude spectrum of current (a) without clock modulation (b) with clock modulation ($f_c=100\text{MHz}$).

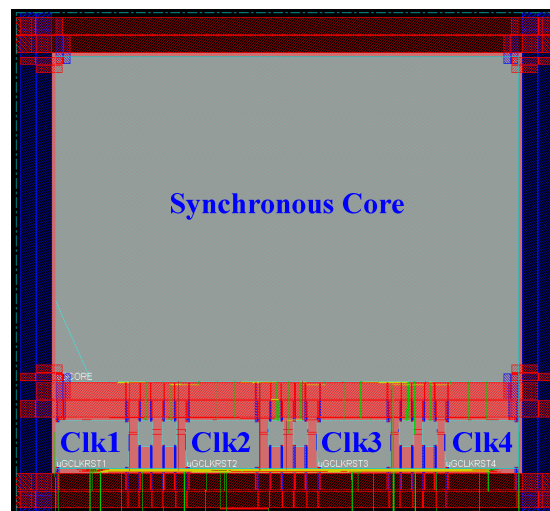


Figure 11: Hierarchical layout of GALS FFT processor.

The pipelined GALS FFT processor was fabricated using IHP 130nm-1.2v standard CMOS process. The die size is $1730\mu\text{m} \times 1730\mu\text{m}$ with 52 pads (including 43 data pads). The chip was packaged in a 64-pin PQFP.

The packaged chip was tested on the CertiMAX test platform. It is difficult to measure current profile using the hardware tester. In this work, a special test strategy was applied. An analog pad was placed on-chip which is dedicated to peering the VDD bounce (Figure 12). This pad was directly connected to internal core VDD ring. By this means we can observe the voltage bounce on core VDD ring using oscilloscope and spectrum analyzer, which also reflect the current fluctuations on flowing power rings.

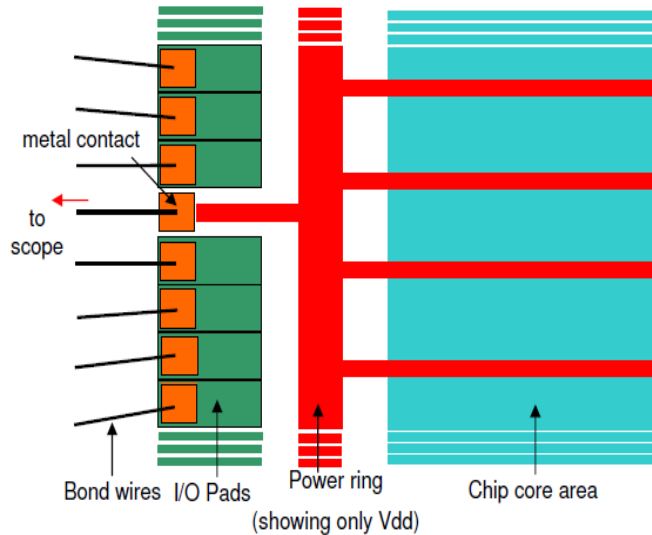


Figure 12: Peering core VDD ring.

For the facility of performance comparison, the chip is designed to be working in both synchronous mode and GALS mode with clock modulation. An external input clock is needed as the global clock signal in the synchronous mode, while the internal local clocks are used in the GALS mode. The amplitude spectrums of core VDD in both modes are demonstrated in Figure 13, and the attenuation at clock harmonics shown in Figure 14. More than 20dB reduction has been measured.

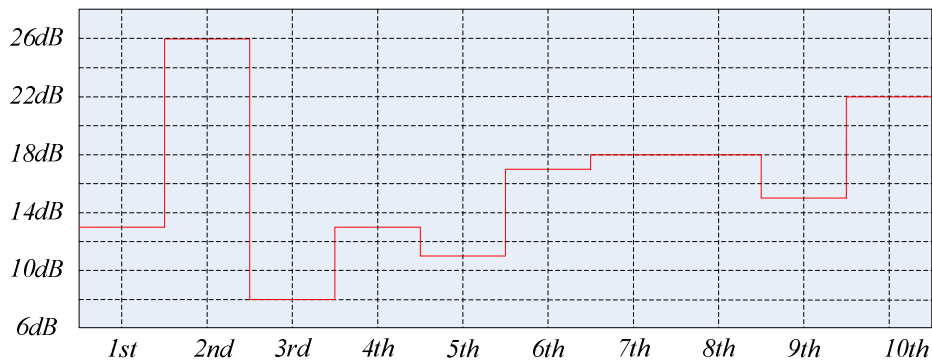


Figure 13: Amplitude attenuation at clock harmonics.

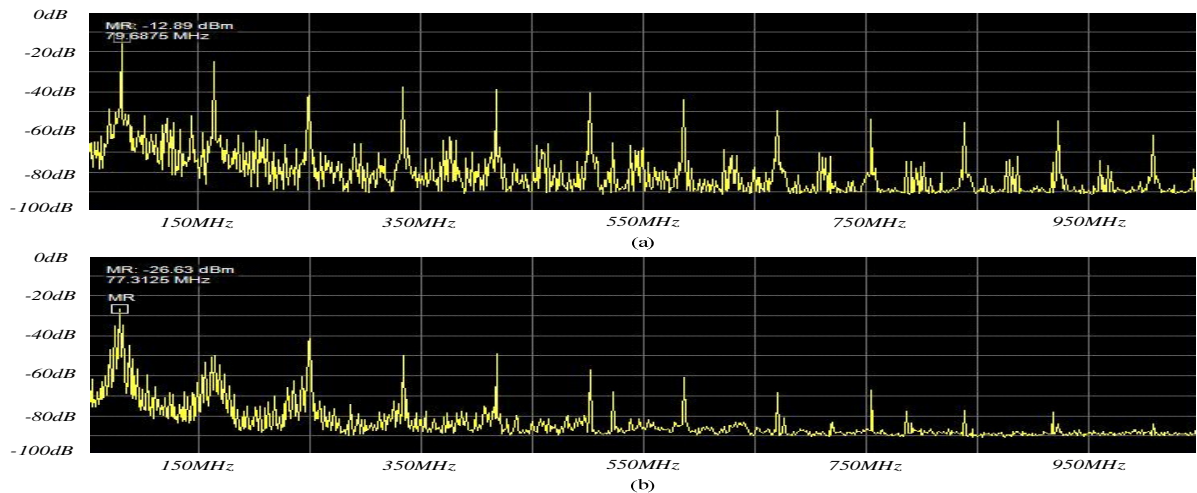


Figure 14: *Amplitude spectrum of core VDD (a) in synchronous mode (b) in GALS mode with clock modulation.*

4.1.10 Conclusions

In this report, the GALS methodology was investigated in order to evaluate the ability for EMI reduction. We have generated a software tool based on Matlab to simulate EMI properties of the digital GALS systems. It supports simulations of GALS/synchronous systems with different granularity, frequencies, current shapes, topology, and other parameters. Using the software, we have modeled GALS and synchronous system in order to evaluate the effect of different topologies, architectures on EMI reduction.

The results show that the reduction of high spectral components can be successfully achieved with jitter introduction. Additionally, for synchronous systems, EMI at low frequencies can be reduced by a phase shift introduction. However combining those two features would be hard in synchronous systems because of problems in the timing closure.

In GALS systems, phase shift is already present by the nature of the GALS methodology. Local clock generators also naturally generate clocks with jitter, but this feature of ring oscillators was not deeply analyzed here. In this work, we have modeled explicit jitter introduction with the special jitter generators based on LFSR structures. By adding jitter in a GALS system, we can achieve a significant reduction over whole spectrum, not affecting the functionality of a system. The reduction of over 20 dB can be achievable, as illustrated in the results. Moreover, the current peaks in time domain can be reduced up to 40% in GALS systems.

We have found that there is almost no correlation between EMI reduction and data transfer intensity (i.e. clock pausing rate) in GALS modules. The greatest impact has the used set of frequencies and the granularity of GALS partitioning. Therefore, we have analyzed the effect of partitioning to EMI profile going from the pure synchronous systems and ending up with the pure asynchronous systems. What we have observed is that with deeper partitioning some measurable results could be seen only up to partition of 15 GALS blocks. Further partition gives no improvement on EMI profile. For the saturated case, that can be also a



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

model of an asynchronous case, the achieved EMI reduction is around 25 dB.

Finally, we have proven the proposed concepts in praxis and implemented low-EMI FFT processor that can operate both in synchronous and GALS mode. The measurements confirmed our theoretical simulations and we have achieved 13 dB reduction of EMI between the best GALS and best synchronous mode.



4.2 LOW-POWER GALS DESIGN - DYNAMIC VOLTAGE AND FREQUENCY SCALING

Shrinking technology nodes combined with the need for higher clock speeds have made it increasingly difficult to distribute a single global clock across a chip while meeting the power requirements of the design. GALS design style can help achieve low power consumption and modularity of a design while greatly reducing the number of global interconnects. Design styles based on GALS methodology alleviate the problem of clock distribution by having multiple clocks, each of which can be distributed to a relatively small portion of the chip.

GALS based design offers a high potential for power reduction:

- Relaxed skew constraints in global clock distribution.
- Possibility to use multiple clock frequencies.
- Possibility to vary supply voltage with clock frequency – DVFS.

In our work DVFS technique have been researched in detail.

In synchronous architecture, if only one global voltage is controlled or scaled down, there is no real optimization at unit level and the whole system is constrained by the most critical functional unit to meet its timing constraints. As the number of IPs integrated in a system is increasing and the power consumption is ever increasing, a fine-grain power management is thus becoming essential.

Multiple clock domain architectures (such as GALS architectures) can benefit from having frequency/voltage values assigned to each domain based on workload requirements. A new hardware-based approach to dynamically change the frequencies and potentially voltages of a local synchronous islands (LSI) or in this case voltage-frequency islands (VFI) driven by a dynamic workload is explored. This technique tries to change the frequency of a synchronous island such that it will have efficient power utilization while satisfying performance constraints. In recent years, there have been major developments, both in industry and academia, in the field of multiprocessor systems. Such multiprocessor systems are very good candidates for VFI design style implementation, where one or more processors can be part of a single VFI.

Most of the designs have irregular workloads when the actual work performed by each block in the system is compared. In general, there are a few modules that are the bottlenecks of the system while most others are idle for large periods of time. As shown in Figure 15, in a system operating at throughput level $Thp1$ and power level $P8$, there is some power wasted since the lower power level $P5$ already meets the performance requirements of the system. Such slack in power of various modules can be exploited by decoupling them into independent VFIs. The finer control of frequency and voltage of these VFIs can enable conversion of slack in performance into power savings without actual loss in performance. Such a distributed approach is necessary as the global scaling of single frequency and voltage may not be able to keep up with the power/energy constraints imposed by cooling and battery technologies.

Assignment of frequency and voltage values to each of the VFIs can be done by using either offline or online methods. Offline methods can be used when the behavior of the application is very predictable for various input conditions and the worst-case behavior is not very different from average-case behavior. However, such an approach is not very suitable for applications that show large variations in their behavior for different input conditions. For such systems, online methods are more suitable. Dynamic voltage and frequency scaling (DVFS)



schemes can be used to adapt the system to meet the performance requirements of a dynamically changing workload while consuming the minimum possible power required to meet the performance targets.

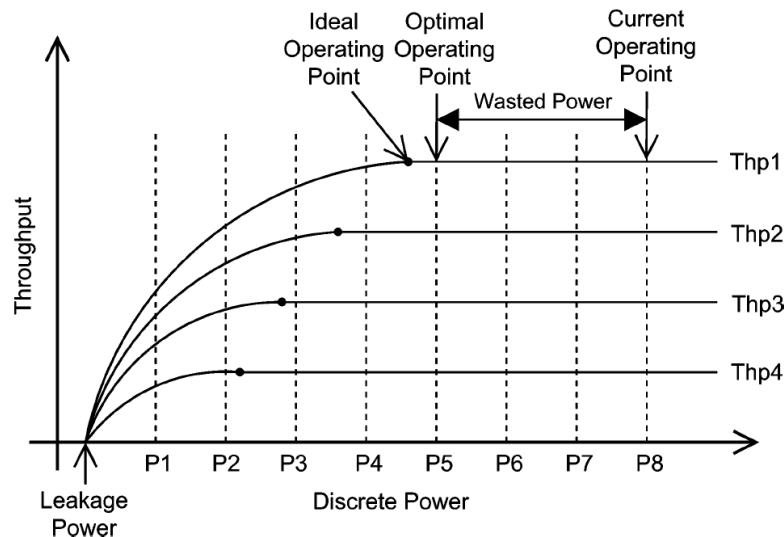


Figure 15: Throughput versus power for a module in a system.

Here we present an online, hardware-based control mechanism for dynamically selecting the operating speed and voltages for individual VFIs in a VFI-based system. The idea behind the hardware-based approach is to have the necessary blocks in the system monitor the application workload at a fine-grain level. The information collected at such a fine-grain level can be used to make local, as well as global decisions about the new frequency and voltage values of various VFIs.

A simple DVFS algorithm can be used along with our proposed hardware approach. Even though our algorithm can be configured for simple applications, it does not consider all the possible workload variations of real applications. Our work concentrates more on the hardware aspects of a DVFS system that can be used as a platform for implementation of various DVFS algorithms [12]. In addition, the hardware platform can also be improved to eliminate the need for significant offline analysis and run realtime applications with random bursts of data, different buffer size requirements, etc. In our proposed approach, the hardware overhead is a small fraction of the overall design and can be controlled during the design process.

The proposed DVFS architecture is based on the association of a local clock generator, its pausable clock system and a local voltage supply unit. The supply unit is based on a VDD hopping technique [13] to control active power modes, combined with a custom designed Power Switch (for details about its implementation and measurement results see Deliverable D12.2) to handle stand-by low leakage mode. Unit voltage and frequency are locally controlled to handle five power consumption modes from high performance to absolutely no leakage. An average voltage and frequency given value is obtained thanks to an automatic hardware mechanism. The main objective is to optimize, at fine grain, the power reduction without any fine control software at system level [14] [15].



4.2.1 System Architecture and Main Principles

The system architecture overview is presented in Figure 16.

Synchronization between the communication router and the units is done thanks to a pausable clock mechanism. A programmable Local Clock Generator (LCG), whose PLL-based low-power realization has been implemented and tested in D12.1, is implemented within each unit to generate a variable frequency in a predefined applicative range. The power unit (Low Power Manager) manages the local unit voltage, sharing a power switch between a VDD hopping technique and an MTCMOS switching circuit. The Power Switch (from D12.2) uses two external voltages: V_{HIGH} and V_{LOW} to be automatically switched during DVS phases.

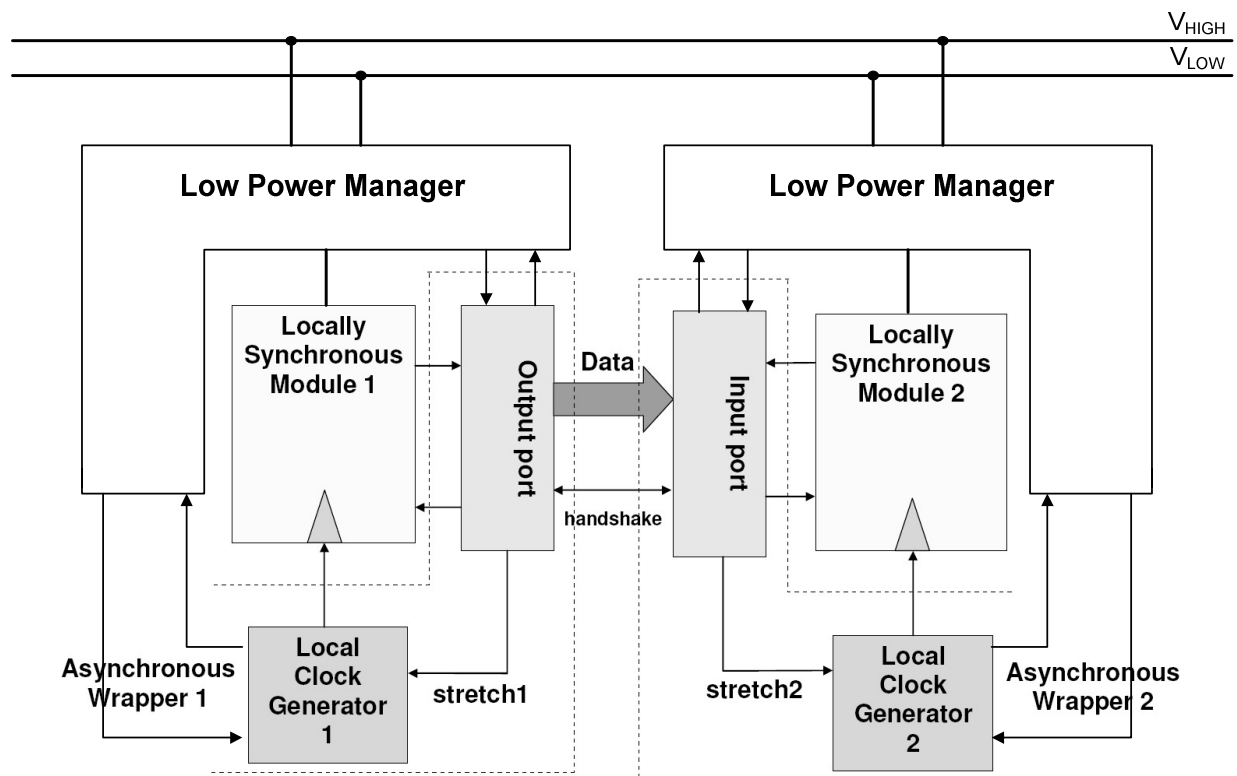


Figure 16: GALS DVFS system architecture overview.

Each synchronous IP unit is defined as an independent power domain (using its dedicated local voltage) and an independent frequency domain (using its dedicated local clock). The unit handles a set of user-defined power modes. In order to perform efficient local DVS, the main objective is to avoid as much as possible low-level software control to ensure a minimal latency cost of DVS during unit's computing. Within the Power Unit, this leads to implement a hardware controller to automatically switch between V_{HIGH} and V_{LOW} . By guarantying smooth DVS transitions, the synchronous IP block can continue its own operation. To obtain an average voltage value between V_{HIGH} and V_{LOW} , the Low Power Manager automatically switches between these two values using a configurable duty-ratio. Since the power domain and frequency domain of the synchronous unit are identical, it is expected that the local frequency scales approximately linearly with the associated voltage scaling. If the frequency/voltage scaling ratio is not exactly linear the delay line must be re-programmed



accordingly. The V_{LOW} delay value is adjusted by a correcting factor with respect to the V_{HIGH} delay value, since the delay line is supplied on the same power domain.

Finally, using Pausable Clocking technique, the synchronous IP unit is paused during synchronization period which additionally saves the power.

4.2.1.1 Power Modes Definition

In the proposed architecture, each unit can be set in one of the six available power modes (Table 3):

- in INIT mode, supply voltage is V_{HIGH} , and no clock is sent to the core. This is the “post-reset” mode.
- in HIGH mode, supply voltage is V_{HIGH} and core clock is on. This is the “nominal” working high performance mode.
- in LOW mode, core clock is still on, but supply is switched to V_{LOW} . Clock frequency is lower than nominal, and energy per cycle decreases. This is the “low power” mode.
- in HOPPING mode, core clock is on and supply voltage automatically hops between V_{HIGH} and V_{LOW} . Frequency and duty-ratio of hopping transitions is configurable. The obtained performance is an average value between and modes, depending on the given duty-ratio. This is the “DVFS” mode.
- in IDLE mode, core clock is off and leakage power is reduced due to the V_{LOW} supply voltage. This is the “lowpower dormant” mode.
- in OFF mode, the unit is switched off by an MTCMOS classical approach to further reduce the leakage power. Since LSI is inactive in this mode, the OFF mode is enabled/disabled through an external “cut_off” signal must be controlled by an external unit. This is the “low-leakage” mode.

All power modes, beside the OFF mode, can be selected per each unit through programming of the unit Low Power Manager.

Table 3: Unit Power Modes

INIT	At reset, the unit is at V_{high} with no clock
HIGH	The unit is supplied by V_{high} voltage
LOW	The unit is supplied by V_{low} voltage
HOPPING	The unit is automatically switched between V_{high} and V_{low} voltages, <i>for DVFS</i>
IDLE	The unit is idle, with maintain of its current state at V_{low} voltage, <i>for reduced leakage power</i>
OFF	The unit is switched OFF, with no maintain of its current state, <i>for minimal leakage power</i>



4.2.1.2 Low Power Manager

The Low Power Manager integrated into the Asynchronous wrapper is in charge of handling the unit's power modes. This manager contains a set of programmable registers, to define the unit power mode, to configure LCG, and to configure and control the Power Switch (described in D12.2). The Low Power Manager contains a “mode” register to define the unit power mode from Table 3.

In INIT and IDLE mode, the Low Power Manager set an idle signal in order to gate the clock of the core unit. The Power Switch is then required to be in V_{HIGH} supply in INIT mode while in V_{LOW} supply in IDLE mode.

In order to control the HOPPING mode, a Pulse Width Modulation (PWM) is used. Two registers are dedicated for the “PWM frequency” and the “PWM duty-ratio”. As an illustration, Figure 17 shows a sequence between the HIGH and LOW states using the PWM.

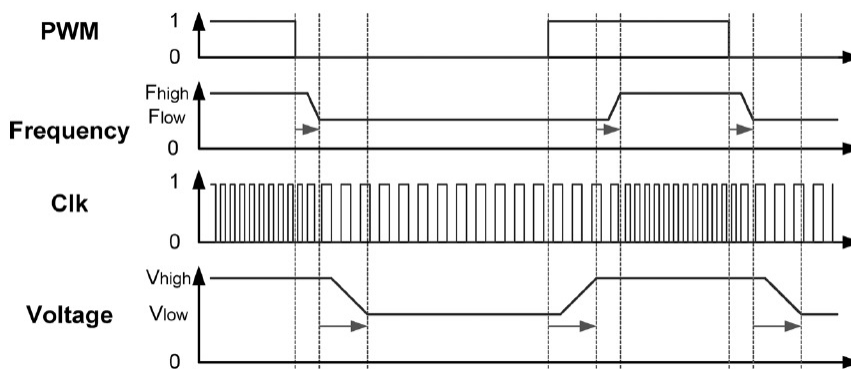


Figure 17: LPM control, sequence example.

Finally, the LPM also contains a set of registers to precisely control the necessary Hopping-unit signals: the hopping-unit clock control (through a dedicated programmable delay-line) and transition slopes control between V_{HIGH} and V_{LOW} .

4.2.2 Example of Operation and Proposal for Optimization

DVFS is implemented in the form of V_{DD} -Hopping with dithering [17] which uses two voltages V_{HIGH} and V_{LOW} provided by external DC-DC converters to control the local voltage of the functional core. Figure 18 illustrates the difference between a traditional DVFS and V_{DD} -Hopping. Figure 18(a) illustrates a classical continuous DVFS approach obtained using an integrated local DC-DC converter. Figure 18(b) represents a discrete DVS approach using three voltage supply levels but without voltage dithering. As a comparison, Figure 18(c) illustrates the same system with voltage dithering. By using a dithering method between two power modes (voltage/frequency pairs such as $[V_{HIGH}, F_{HIGH}]$ and $[V_{LOW}, F_{LOW}]$) V_{DD} -Hopping allows to control the core average frequency and to reduce the power nearly as efficiently as a continuous voltage converter. The obtained F_{AVG} depends on the duty ration between, T_{HIGH} time spent in HIGH mode and T_{LOW} , time spent in LOW mode:

$$F_{AVG} = \frac{(F_{LOW} \cdot T_{LOW}) + (F_{HIGH} \cdot T_{HIGH})}{T_{LOW} + T_{HIGH}}$$

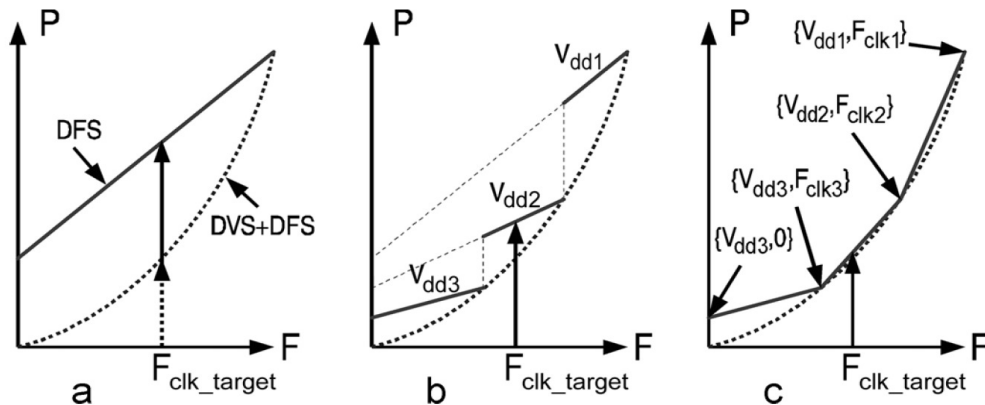


Figure 18: LPM control, sequence example.

Given the proposed power modes (INIT, HIGH, LOW, HOPPING, IDLE, OFF), two execution cases are foreseen at system level. According to power application requirements, the software programmer can either use an explicit way to program each IP unit using the HIGH, LOW, IDLE, OFF modes; or use an implicit way to program each IP for a pre-determined average V_{CORE} value using the HOPPING mode. In the explicit way, the software must configure at real time the appropriate mode according to unit's data flow (workload), while in the implicit way, the pre-programmed V_{CORE} value is fully managed by hardware.

At a first glance, we may consider that in the explicit HIGH and OFF cases, the programmer will drive the system according to strong applicative latency constraints, while in the HOPPING, LOW and IDLE cases, the system is driven according to stronger power budget constraints. For the OFF mode targeting leakage reduction, special care must be taken in software since the IP unit execution must be completed or saved before switch-off. For switch-on, the unit must be restarted by a dedicated signal and reset.

A precise and optimized usage of the proposed power modes obviously strongly depend on the relative power values of the IP blocks in each mode (see further). The global power management strategy is out of scope of this research. For instance, the units' average performances can be determined thanks to an off-line algorithm, while reducing the global energy and respecting a global latency constraint [14].

However, we propose a realization for local power optimization using V_{DD} -Hopping. In order to achieve this, a workload monitoring circuit has to be implemented in GALS wrapper, monitoring the activity of I/O ports/latches, for example by counting synchronization events. Then some form of dependency (based on statistical data) between frequency of I/O events and F_{AVG} should be derived in off-load mode and implemented using simple look-up table. The big advantage of this realization in GALS design is relative ease of implementation by simply counting synchronization events. A digital logic circuit necessary for this would consume practically negligible amount of power compared to the main IP core.

4.2.3 Unit Power Consumptions Evaluation

Lowering the supply voltage reduces both dynamic and static power consumption. Synthesis results and spice level simulations shows that for $V_{HIGH}=1.2V$ and $V_{LOW}=0.8V$, clock speed is commonly reduced by 75%, leakage power by 45%, and dynamic power by 88%. For a typical



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

synchronous unit where leakage would represent 20% of the total power, the energy per cycle is reduced for that unit of 35% between LOW and HIGH modes.

In a multi-configuration application, when operating in a low performance mode, the unit can be set in one of the available power modes (IDLE, LOW or HOPPING) chosen according to the performance constraints. For example, an OFDM IP unit of about 300 kgates consumes 80mW@220MHz in HIGH mode, while consumes only 10mW@66MHz in LOW mode. If a reduced performance is only required, HOPPING will save the corresponding ratio: if 145Mhz fits the application requirements, a 50% hopping ratio will save half the power.

4.2.4 Conclusion

In order to automatically control both dynamic and leakage power we have proposed a complete Dynamic Voltage and Frequency Scaling architecture for IP unit integration within a GALS design. In the proposed architecture, each synchronous unit is encapsulated with its own local clock generator and local supply power unit and can be programmed through a set of dedicated power modes. No fine control software is required during voltage and frequency programming and thus minimal latency cost is observed. A potentially easy implementation of V_{DD} -Hopping technique together with dynamic workload (synchronization activity) monitoring offers a huge benefit to GALS realization compared to fully synchronous design. This DVFS architecture can be fully integrated in CMOS techniques with an area cost affordable for coarse grain IP units.



4.3 PROCESS VARIABILITY MITIGATION IN GALS SYSTEMS

Since the process variability mitigation was already a subject of the deliverable D22, here we will give just, in a more compacted way, a summary of the information provided already in the respective detailed report. In addition to that we provide some more details of GALS evaluation in comparison with fully synchronous design that was performed after the deliverable release. More information about process variability can be found in respective report D22.

Traditional synchronous design methodologies, that require the system to adhere to strict timing constraints over the entire chip area, suffer badly from wide ranging parameter variations. The need to satisfy worst-case timing constraints will limit the performance of future designs severely. GALS systems may offer significant advantages in this field, as they essentially divide a complex global system into small, independent modules.

Partitioning of the design into LSIs effectively reduces the length of the critical paths but unfortunately increases the effects of local process variability since the variability of the critical paths is affected by the length of the critical path in an inverse proportion as demonstrated in D22.

The proposed technique for variability minimization can be universally applied in digital logic on synchronous as well as on GALS design. However, as it has been demonstrated in D22, the inherent property of the technique is that the larger the variations are, the greater is the reduction in the maximum critical path delay, and therefore, the higher is the improvement in the maximum frequency the circuit can run. Having reduced lengths of critical paths in GALS design increases the impact of local parameter variations and also increases the effectiveness of the proposed variability delay minimization technique.

4.3.1 *Technique for minimization of the impact of local delay variations*

The general principle of the technique consists in replicating R times a critical path, and evaluating critical path outputs using a function that senses the fastest out of R replicated paths. In this way, not only the standard deviation but also the mean value of the critical path delay is reduced as demonstrated further in Table 4.

The proposed technique can be also used to support recovering correct operation disrupted by other sources of variations such as unexpected voltage drops in the power supply network, local temperature fluctuations, crosscoupling noise, etc.

The circuit that performs the evaluation of the fastest input path (fastest path evaluator - FPE) is shown in Figure 19. It senses the path where the first signal change occurs. Depending on the previous signal state input signals are ANDed or ORed sensing the first falling or raising edge respectively. Since the information about the previous signal state is necessary in the fastest path evaluation, it is a natural idea to integrate this circuit into existing flip-flops. A key requirement for the integrated fastest path evaluator (FPE) flip-flop is that the delay overhead compared to the standard flip-flop should be minimal.

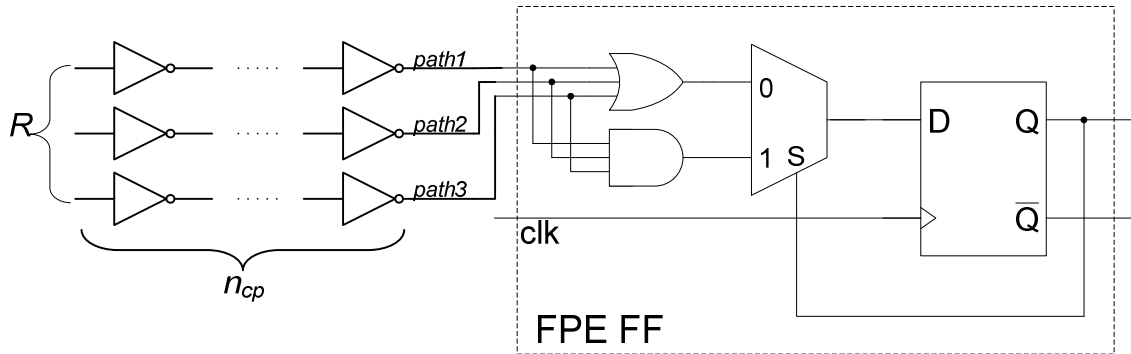


Figure 19: Fastest path evaluator and standard flip-flop.

Several methods are applied to reduce the delay overhead of the FPE flip-flop. The master latch is duplicated and its input inverter is replaced with AND and OR gates. The multiplexer at the input of the flip-flop causes a significant delay overhead, and is therefore moved to replace the output inverter of the master latch, as shown in Figure 20. The FPE flip-flop is designed in the state-of-the-art 65nm technology and the delay overhead compared to the equivalent type of standard library flip-flop is less than 7%.

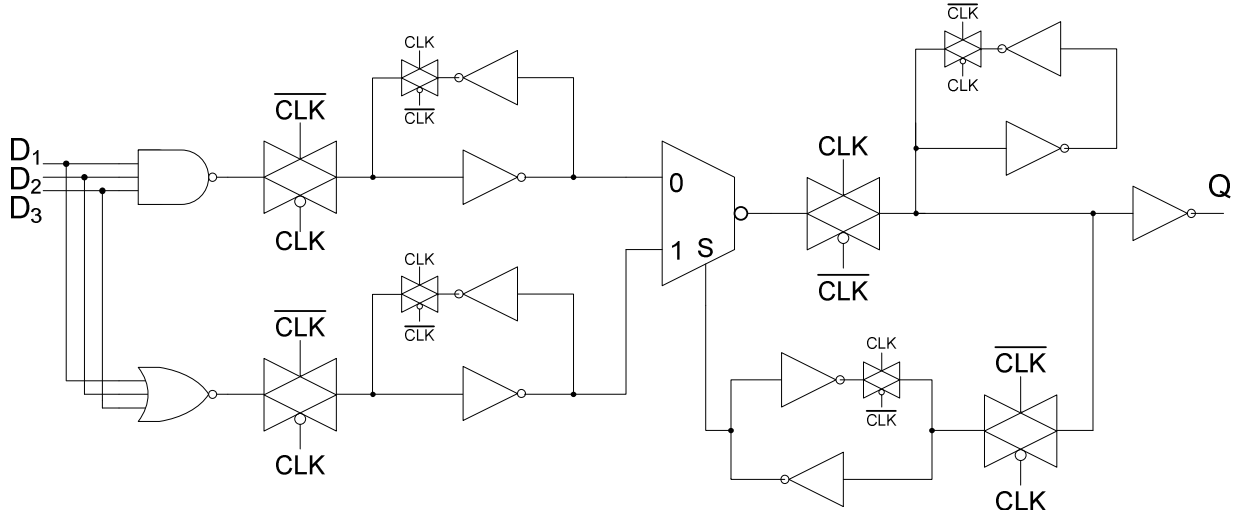


Figure 20: Schematic of the reduced overhead FPE flip-flop.

The improvement of the relative mean value $\frac{\mu_{FPE}}{\mu_{T_{cp}}}$ and relative standard deviation $\frac{\sigma_{FPE}}{\mu_{T_{cp}}}$

of the critical path delay is shown in Table 4 for different critical path lengths n_{cp} and different redundancy factors (R). The values are acquired from Monte Carlo SPICE simulations. The additional delay introduced with the FPE flip-flop is also accounted. The reduction in the mean value of delay ranges from 0.85%-2.38% and for standard deviation ranges from 17%-25% for $R=2$ and $R=3$, respectively. The variance reduction from $R=2$ to $R=3$ is not as significant as the reduction between no redundancy and $R=2$.



Table 4: Relative mean value and standard deviation of the critical path delay.

%	<i>R=2</i>			<i>R=3</i>	
	$\frac{\sigma_{T_{cp}}}{\mu_{T_{cp}}}$	$1 - \frac{\mu_{FPE}}{\mu_{T_{cp}}}$	$\frac{\sigma_{FPE}}{\mu_{T_{cp}}}$	$1 - \frac{\mu_{FPE}}{\mu_{T_{cp}}}$	$\frac{\sigma_{FPE}}{\mu_{T_{cp}}}$
	$n_{cp} = 6$	5.44	0.85	4.50	2.38
$n_{cp} = 8$	4.78	0.88	3.96	2.23	3.59
$n_{cp} = 10$	4.44	0.97	3.67	2.22	3.33
$n_{cp} = 12$	4.08	0.97	3.37	2.12	3.06
$n_{cp} = 16$	3.62	0.99	2.99	2.01	2.71
$n_{cp} = 20$	3.32	1.00	2.74	1.94	2.49
$n_{cp} = 24$	2.98	0.94	2.46	1.78	2.23

The important result is that for shorter critical paths (as the case in GALS design compared to fully synchronous) the effectiveness of the technique is higher.

4.3.2 Optimization results of variations minimization

The effect of reducing the mean delay and standard deviation of critical paths by replicating a critical path and inserting an evaluation block is exploitable only when the delay reduction is large enough to compensate for the area/power overhead introduced by the critical path replication.

An architecture that is optimized in the following in terms of delay variations is an AES encryption core [18]. The design is partitioned in three local synchronous islands: encryption module, key generation module and control module. Each of these three islands has been synthesized for the maximum operating frequency. Encryption module as the critical one in terms of speed has been additionally optimized for variations minimization using the presented technique of fastest path evaluation.

The encryption module has $N_{cp}=200$ independent critical paths (with slack smaller than 1% of the maximum operating frequency), with typical critical path length of $n_{cp}=16$. The distribution for a normalized maximum critical path delay is shown in Figure 21, considering cases with and without optimization. The optimization is performed with redundancy factors of 2 and 3.

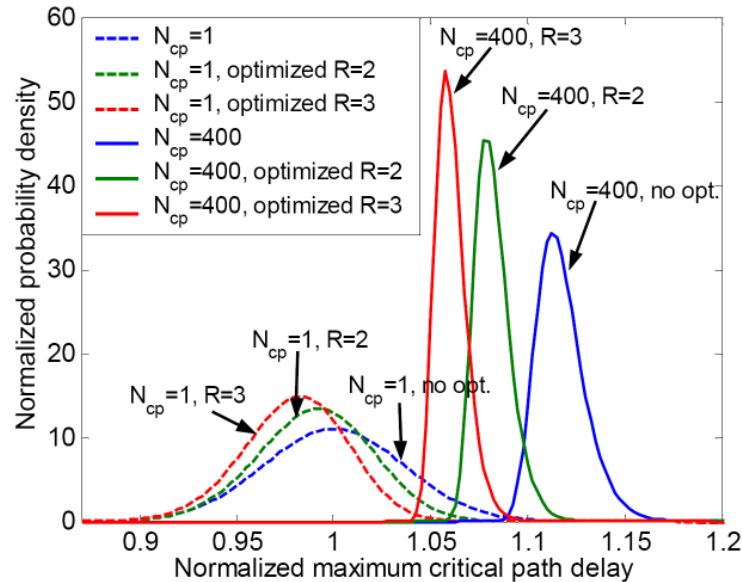


Figure 21: Maximum critical path delay distribution for cases without selective redundancy, $R=2$ and $R=3$.

Figure 21 early shows reduction in the mean value of the maximum critical path delay which mostly benefits from the reduced mean value of each critical path. Moreover, a reduced standard deviation of each critical path also reduces the mean value of the maximum critical path delay. The improvement in the mean value of the maximum critical path delay is:

- 3.3% and 5.4% compared to the default case (without optimization) for $R=2$ and $R=3$, respectively.

There is a noticeable reduction in the standard deviation of the maximum critical path delay:

- 25.5% and 35.8% compared to the default case (without optimization) for $R=2$ and $R=3$, respectively.

However, for comparison in fully synchronous design there are no partitions and the typical length of the critical path is $n_{cp}=24$. The improvement of the mean value of the maximum critical path delay in this case is:

- 1.9% and 3.6% compared to the default case (without optimization) for $R=2$ and $R=3$, respectively.

The reduction in the standard deviation of the maximum critical path delay is:

- 15.1% and 26.3% compared to the default case (without optimization) for $R=2$ and $R=3$, respectively.

As can be clearly seen, the results of implementing the proposed delay variations minimization technique are noticeably better in GALS design compared to fully synchronous.



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

4.3.3 Conclusions

The benefit of the selective redundancy-based technique to limit the impact of local delay variations has been demonstrated. Only critical segments (critical paths) are replicated and the viable improvement is achieved with the redundancy factor as low as $R=2$. This way, the proposed technique can optimally be used in the design of large synchronous digital systems. The inherent property of the technique is that the larger the variations are, the greater is the reduction in the maximum critical path delay, and therefore, the higher is the improvement in the maximum frequency the circuit can run. Thanks to this fact, GALS design with shorter critical paths (higher impact of local variations) benefit more from the technique than fully synchronous design as demonstrated.

The studies in D22 have shown that the technique can already be successfully applied for 65nm CMOS technology process where an optimal point has been shown to exist in the speed vs. area/power tradeoff. Even more advantage is expected for future nanoscale CMOS technologies such as 45nm and 32nm nodes. The technique can be especially beneficial in the low-power domain where reduced supply voltages lead to significantly increased local variation.



5 GALS NETWORKS-ON-CHIP APPROACH ADVANTAGES

There is no doubt on the fact that process variations instil a probabilistic flavour to circuit timing analysis. With this respect, traditional synchronous design methodologies, that require the system to adhere to strict timing constraints over the entire chip area, suffer badly from wide ranging parameter variations.

GALS systems may offer significant advantages with this respect, as they essentially divide a complex system into small, loosely coupled and often even independent modules. This way, the implementation of a fault-containment mechanism becomes easier to implement and a networked multi-core system can be designed to survive to the failure of one or more IP cores and/or network links. This architecture design paradigm is today seen as a key enabler to sustain yield. In fact, the intricacies of nanoscale physics give rise to a physical gap that can be bridged only by building reliable components out of an inherently unreliable silicon technology where the characterization of physical parameters can be often only given in a statistical way.

Thanks to the GALS paradigm, it is possible to implement a network with support for unexpected and unpredictable topology irregularity: failing nodes just have to be isolated while preserving correct behaviour of the remaining portion of the system. Deliverable D22 has illustrated how routing design is affected by this requirement. On one hand, among all fault-tolerant routing algorithms that can still provide complete post-silicon connectivity, the designer has to select the most performance-efficient one with respect to application traffic patterns. On the other hand, he is faced by the choice of the routing implementation, which has deep implications on the final architecture and on overall performance and complexity figures. In fact, every fault-tolerant routing algorithm lends itself to a table-based implementation. Unfortunately, routing tables are expensive in terms of access time and resources, and feature poor scalability properties. Therefore, D22 has pushed the development of table-less routing implementations to minimise the impact of variability on operating frequency and manufacturing yield. In practice, the challenge to make a logic-based distributed routing flexible enough to be reusable for different irregular topologies and routing algorithm has been addressed. The advent of GALS technology will definitively foster this kind of research and development efforts as an effective means of sustaining yield in the presence of manufacturing defects and process variations.

Another advantage of the system partitioning and decoupling that GALS technology enables is associated with the performance maximization of regular topologies for general purpose multi-core systems. The 2D mesh is the reference solution for designers of such systems, in that it provides a good match with the 2D silicon surface, it has good predictability of electrical parameters and facilitates application mapping. In fact, the most communication demanding task of an application can be placed in the middle of the network and the other tasks can be placed around in a step-by-step mapping process. This approach is at the core of many mapping strategies reported in the open literature.

Unfortunately the 2D mesh features poor scalability properties of its diameter, average minimal hop count and bisection bandwidth with the increase of the number of nodes connected to the network. Other regular topologies might be mutated from the off-chip interconnection network domain and used as a workaround for this problem. Fat-trees for NoCs date back to the infancy of this interconnection discipline, however they have been



rapidly discarded due to a clear mismatch between their connectivity pattern and the 2D silicon surface. Topologies with more than 2 dimensions could also be used, such as the k-ary n-mesh topologies (with n greater than 2). The use of more dimensions enables to cut down on the diameter and to increase the bisection bandwidth. Again, when these topologies are mapped on silicon, a number of layout issues arise: wires of uneven length, wire crossings, high switch radix. The high switch radix occurs because the switch needs to allocate more input/output ports to connect to the different dimensions of the topology. However, there is also another scenario where high switch radix is an issue, namely when the number of dimensions is reduced and more cores are connected to the same switch, thus minimizing the number of hops in the topology.

When going through the physical implementation, the issues above result into some form of performance degradation. Wire crossings determine a larger use of metal levels, which are a scarce resource particularly in embedded computing platforms. A more important implication stems from the use of long wires. In fact, the reverse scaling of interconnects as technology scales down implies a rapid increase of wire delay and the shift of the network critical path from switching logic to switch-to-switch links. In practice, the maximum operating speed of a network topology depends on the longest link of the topology itself, and for multidimensional topologies the performance reduction effect might be extremely apparent. To the limit link pipelining might have to be used to change the propagation delay across the links into latency, thus preserving the operating speed of the network overall. Finally, a high switch radix determines a dramatic reduction of the switch operating speed. In fact, the internal arbiters and the crossbar become slower due an increased amount of inputs and outputs. Moreover, for a switch radix beyond 14x14, it has been demonstrated in the open literature that the layout might not even be feasible. The physical synthesis toolflow fails to meet the timing constraint, to avoid crosstalk and to solve the routing congestion issue. The only workaround for this problem would be to reduce the area utilization parameter during place and route or to reduce the target operating speed. The suggestion is therefore to never exceed a 14x14 switch or conservatively even a 10x10.

One might argue that these problems are not strictly related to GALS systems, but hold in general. This is true, however, in the absence of a GALS synchronization paradigm, these effects become even more severe. In fact, if the IP cores and the network HAVE TO run at the same operating speed (due to the absence of a decoupling synchronizer inbetween, which is the typical scenario of a fully synchronous system), then the physical effects degrading network operating speed would end up degrading also the maximum speed of the IP cores, irregarding of their actual critical path. Only by decoupling the speed of the IP cores from each other and from the network, each block could be operated at a speed depending only on its own critical path without destructive interdependencies.

As an example, let us consider regular topologies for a 64-tile system. After physical synthesis (placement-aware logic synthesis, floorplanning, place&route) on a 65nm technology library, the critical path of the topologies is illustrated in the plot below. The distance between IP cores is assumed to be 2mm.

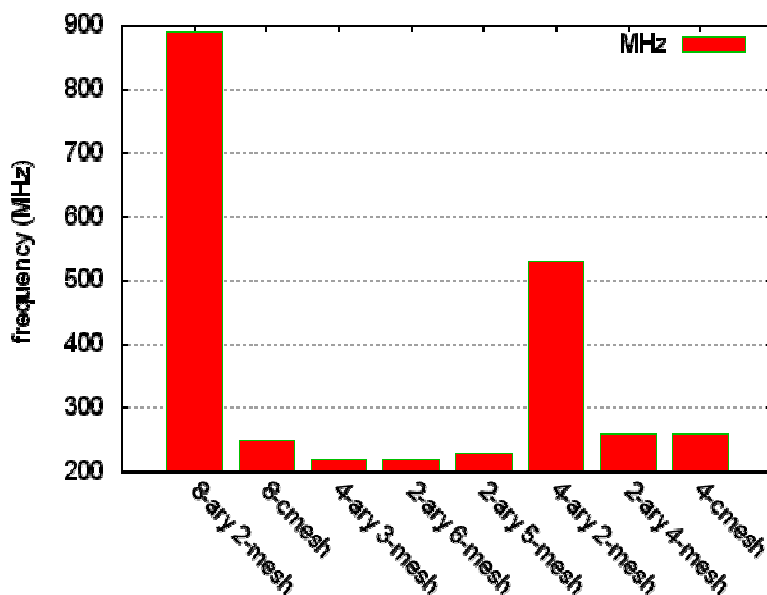


Figure 22: Critical path of regular topologies on a 65nm technology.

The results (depicted Figure 22) are impressive. Most of the topologies under test are not able to operate even at 300 MHz, with the exception of the 8-ary 2-mesh (the 2D mesh) and of the 4-ary 2-mesh (a 2D mesh with 4 cores per switch). These results materialize the intuition above about the physical degradation effects that come into play during physical synthesis. Now the question is: what are the implications on the IP core speed? If there is no synchronization whatsoever between IP cores and network, then the IP cores are forced to run at the same speed of the network (in general below 300 MHz) even though such cores might potentially run at 500 MHz or 1 GHz. The simplest way to decouple these speeds is to use frequency-ratioed synchronization: the network interface might be designed in such a way that the network speed is an integer multiplier of the IP core speed or viceversa. This would be the first step on the way to a fully GALS approach. However, this approach has been used so far to decouple a slow IP core from a fast NoC, such as in the xpipesLite architecture. With the onset of physical degradation effects during topology physical synthesis, the architecture would have to be changed to reflect a network slower than the IP cores. However, this would question the applicability and effectiveness of the network-based approach to system interconnection. The picture above can be somehow changed by implementing link pipelining. The speed-up achieved in this way by the different topologies is showed in the Figure 23 below:

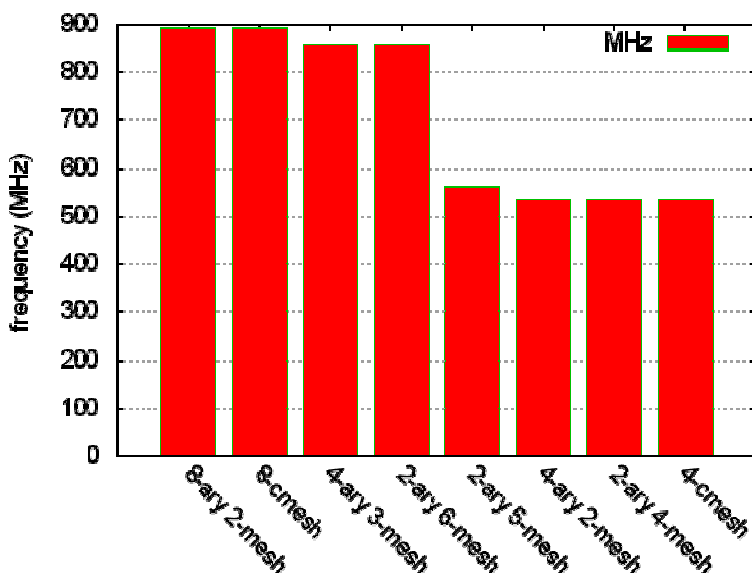


Figure 23: Speeds achievable with link pipelining.

Clearly, most topologies have taken advantage of link pipelining, however in many cases there is a saturation effect due to the inherent speed limitation associated with high radix switches. Moreover, it should be considered that link pipelining can substantially contribute to overall network area. Even in this case, it is evident that a decoupling between IP core speed and network speed is at least desirable, in the direction of a full orthogonalization of concerns. The decoupling that a GALS design philosophy materializes can in the end make the use of such multi-dimension topologies feasible: in fact, their inherent better abstract theoretical properties might be only partially offset by the physical degradation effects, and the IP cores might still take advantage of them if not speed-limited by them.

The benefits of GALS NoCs presented so far come with their own risks. In particular, designers now have to devote special care to the design of process-variation tolerant synchronization interfaces. In fact, these latter become the true weak-point of the system that needs to be safeguarded against timing failures.

As an example, it might take too much time to generate the strobe signal out of the source synchronous clock at the receiving end of the GALS link, and current data might be lost and the next one might be erroneously sampled. This scenario is by itself very sensitive to unforeseen link delays arising as an effect of process variations. Still, source synchronous links are generally designed under the assumption that there will no or very limited routing skew between data lines and the source synchronous clock: this is an assumption that a process-variation sensitive manufacturing process might easily impair. Other examples could be reported to make the following point: GALS links and synchronization interfaces should be designed with specialized design techniques in order to make them less vulnerable to process variation effects. This issue adds up to the historical concerns of reducing performance penalties inside local islands of synchronicity.

Deliverable D22 has illustrated how this challenge can be addressed by relying on a cross-layer design and optimization philosophy.



In particular, a new physical routing approach for robust bundled signalling on NoC links has been proposed. Tightly matched signal propagation and strong crosstalk protection are key requirements for next-generation NoC links featuring GALS synchronization and low-swing signaling. In D22, we presented a new methodology for NoC global link routing which addresses these challenges. Our approach creates bundled link routes with geometrically matched wires, thus leading to much reduced intra-link variations. Moreover, our link router supports high-regularity wire spacing and shielding strategies. Delay variation among different wires of a link is 25% to 70% lower than what can be achieved using a state-of-the-art timing-driven global routing flow. Additionally, crosstalk effects are reduced by more than 30%. This physical routing solution effectively deals with the systematic component of process variations, in that it forces all wires of a GALS link to undergo the same routing and layout conditions.

D22 has also presented circuit-level compensation techniques for full swing vs low-swing on-chip interconnects affected by process variations. Adaptive body bias (ABB) and adaptive supply voltage (ASV) are effective methods for post-silicon tuning to reduce variability on generic combinational circuits or microprocessor circuit sub-blocks. In D22 we focus on global point-to-point interconnects, which are evolving into complex communication channels with drivers and receivers, in an attempt to mitigate the effects of reverse scaling and reduce power. This work is at lowest level of the design hierarchy, and deals with the reliable design of the electrical wires building up a GALS link, although it is not restricted by the specific synchronization paradigm. The characterization of the performance spread of these links and the exploration of effective and power-aware compensation techniques for them is becoming a key design issue. D22 compares the effectiveness of ABB vs. ASV when put at work on two on-chip point-to-point link architectures: a traditional full-swing and a low-swing signaling scheme for low-power communication. D22 provides guidelines for the post-silicon variability compensation of these communication channels, while considering realistic layout effects. In particular, the implications of cross-coupling capacitance on the effectiveness of variability compensation are analysed.

The above physical design techniques aim at making GALS links bridging two different clock domains reliable in the presence of process variations. However, synchronization interfaces should not be seen only as a source of timing concerns, since a thorough analysis of their implementation may reveal interesting opportunities to enforce timing margins against non-idealities of the final layout. In order to understand this, let us introduce a few basic timing metrics that help assess timing margins across a mesochronous synchronization interface. It is worth recalling that mesochronous synchronizers are used for switch-to-switch communication in the GALS architecture paradigm chosen in the context of the Galaxy project (and in Workpackage 6 in particular).

Let us refer to the architecture illustrated in the Figure 24. It illustrates the input buffer of a GALS NoC switch where the mesochronous synchronizer has been tightly coupled with the switch buffering resources. Input data arrive with the transmitter clock along the source synchronous link and is sampled in a rotating way in the front-end latch banks. Such banks are driven by a token ring counter triggered by the transmitter clock. When data at the output of the latches are safely settled, they are selected by a multiplexer and fed to the switch internal logic (e.g., arbiters and crossbar). Since the transmitter clock is available at this switch for its function of data strobe, it is reused also for directly synchronizing the backward flow control signal (denoted as “stall” in Figure 24). This way, the stall signal does not need any further synchronization at the upstream switch, and can be directly sampled in that domain.

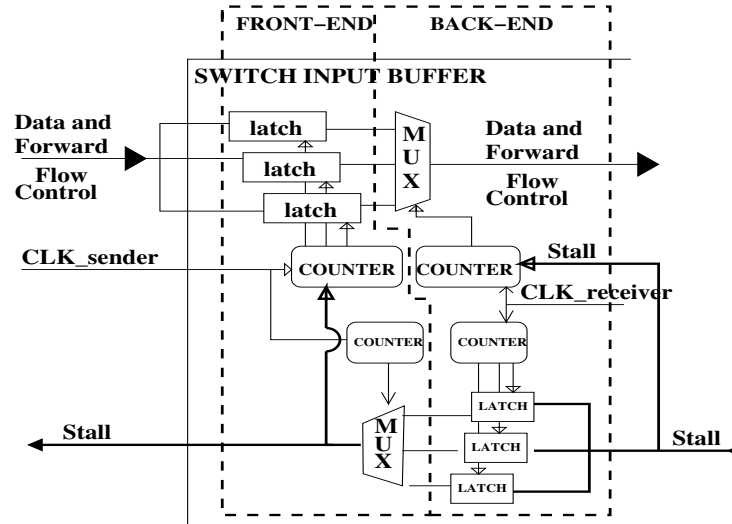


Figure 24: *Tightly coupled mesochronous synchronizer with switch input buffer.*

Going back to the datapath, it is obvious that sampled input data will remain stable at the output of the latch banks for around three clock cycles (since there are three latch banks), and that a good choice of the multiplexer selection window consists of selecting such input data in the middle of its “stability” window.

During the mux window, data at latch outputs is selected for forwarding to the switch internal logic and finally to the sampling flip flop in the switch output port. Its duration closely follows that of the clock period. Sampling occurs on the next rising edge of the receiver clock inside the mux window. We denote the time between the starting point of the mux window and such sampling instant as the Setup time. Conversely, after an Hold time since the rising edge of the clock the mux window terminates. This is the time required by the back-end counter to switch the multiplexer selection signals. These two definitions are pictorially illustrated below.

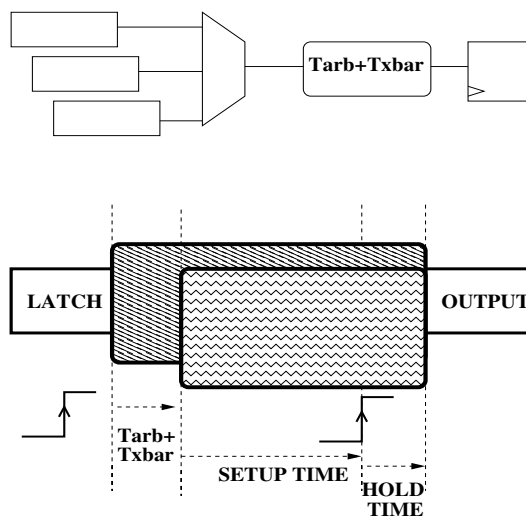


Figure 25: *Setup and hold time definitions.*



Skew tolerance of the architecture depends on the relative alignment of data arrival time at latch outputs, multiplexer selection window and sampling edge in the receiver clock domain. In fact, due to a negative skew, the mux window moves to the left and selects latch output data early in its stability window. How early this occurs depends on the amount of negative skew, and safe operation is guaranteed until the mux window selects a data for sampling which is not stable yet. In other words, the amount of left-shift that the mux window can afford before incurring a sampling failure determines the skew tolerance budget of the architecture. However, another scenario exists: the mux window stays in a constant position, while the lefthand edge of the latch stability window moves to the right. The ultimate result is the same (consumption of the skew tolerance budget), but the source is different. This might be due to the link delay, which delays the assertion of the latch stability window since the sampling clock is travelling in phase with the input data. Such synchronization interfaces are typically designed to support from -100% to +100% relative skew between transmitter clock and receiver clock. However, in practice such high values of the skew are not realistic. During hierarchical clock tree synthesis, a relaxed skew constraint of 50 or 60% is generally believed to be sufficient to infer a low power top clock tree. Therefore, a large percentage of the timing margin generally enforced to absorb the skew is available to absorb other circuit non-idealities. Process variations and consequent delay variability are an example thereof. In fact, data wires and strobe signal in the source synchronous link are generally assumed to have no routing skew between each other, while in practice this is not true. Random process variations determine a different arrival time of data propagating along the data wires and a misalignment of data with the transmitter clock. The simple insertion of a delay variability detector allows to absorb such delay variability by consuming the unused margin against clock skew. An example circuit that can do that is illustrated below.

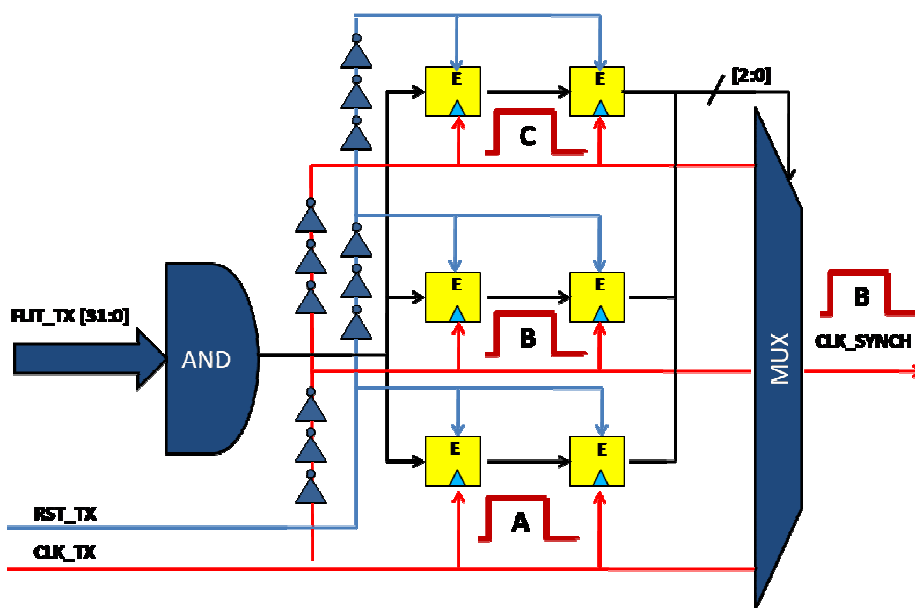


Figure 26: Architecture of the variation detector.

The variation detector architecture senses the offset between the source clock signal and the data signal. Therefore, it provides the receiving synchronizer with a clock signal version able to guarantee a safe data sampling. The clock signal is a delayed replica of the transmitted



source synchronous clock. The detector is composed of a parametric number of brute force synchronizers sampling the output of an AND block. The AND gate receives the incoming data as input generating a high value as soon as all the data input bits are high. The source synchronous clock signal (routed along with the data) crosses a set of delay chains introducing an incremental amount of delay with respect to the nominal source clock.

In order to preserve the synchronism property between reset and clock signal, the reset is bundled with the data as well and it crosses the delay chains together with the clock. The number of delay chains depends on the number of brute force synchronizers in order to guarantee to every synchronizer block a clock/reset with a different offset. Moreover, it depends on the amount of expected routing skew between data and clock and between data lines with themselves.

Finally, the outputs of the brute force synchronizers select through the multiplexer the clock signal offset required to feed the receiving mesochronous synchronizer with a safe strobe signal. The selection of the safe clock signal takes place during the NoC reset phase and it represents a key step in the proposed architecture. However, it is worth recalling that once the strobe signal is selected during the reset process (for instance, at system bootstrap), then the selection stays the same throughout the entire use cycle of the system, thus not generating an overhead for correct system behaviour. Used over time, the proposed architecture allows the network to deal also with wear-out effects.

The architecture of the variation detector provides the safe clock signal before the synchronizer starts its operation. To achieve this result, the proposed architecture comes into play during the NoC reset phase. Furthermore, to guarantee the synchronization between the incoming data and the source synchronous clock signal, the circuit must detect the exact arrival time of every data bit in the GALS link. This is possible by sensing the incoming bits when they are switching from a high logic value to a low logic value or vice versa (a stable bit does not provide any arrival time information). As a result, the proposed architecture can properly sense routing skews when it receives two consecutive data patterns so that every data bits of the first transaction is negated with respect to the data bits of the other one. These two transactions can be purposely generated from the out_buffer of the upstream switch during the reset process.

The Figure 27 reports the post-layout reset phase waveforms of a variation detector composed of 5 delay lines and 5 brute force synchronizers.

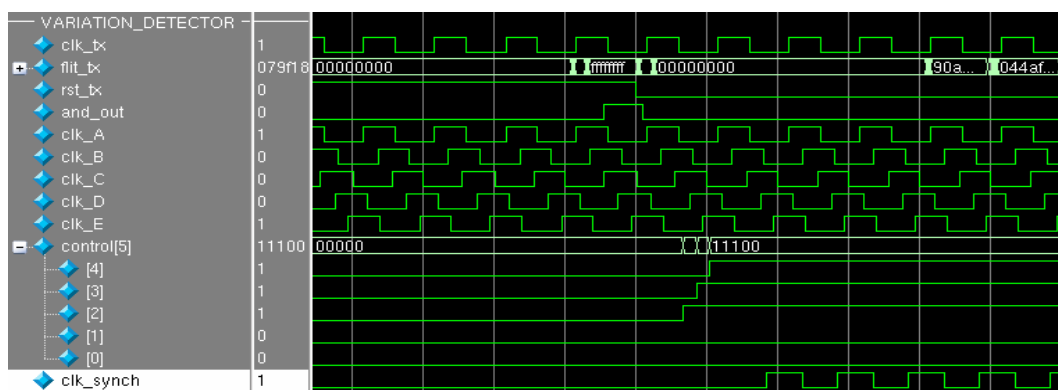


Figure 27: Variation Detector Reset phase.



As represented in Figure 27, once the first sequence of zeros is driven at the variation detector input port, the output of the AND module is set to a low logic value (“and_out”). In the meantime, the source synchronous clock signal (“clk_tx”) is propagated through the delay lines and it generates 5 replicated clock signals with a different offset (“clk_A”, “clk_B”, “clk_C”, “clk_D”, “clk_E”). Until the incoming sequence has low value, the brute force synchronizers sample the low AND output and they set to zero the control signal driving the multiplexer (see “control[5]” in Figure 27). In this configuration the multiplexer output is permanently at a low logic value and the receiver mesochronous synchronizer does not receive any clock signal.

Then, the transmitter starts to drive the sequence of ones and the data bits (“flit_tx”) start to switch at the variation detector input port. Finally, when the incoming data is stable and every data bit has switched to the high logic value, the AND output assumes a high logic value.

Although the AND output feeds all the brute force synchronizers, only some of them are able to sample the high value at the AND output. In fact, the brute force synchronizers driven by clock signal with an early positive edge will not reveal the high value; on the contrary, the brute force synchronizers driven by the clock signal with a late positive edge will sample it and will set the multiplexer control signal. As result, the control signal collects all the information about the result of the AND output sampling and this latter information is exploited to distinguish between the safe clock signal and the potentially unsafe clock signals (early clock signals). In Figure 27, the control[3] and the control[4] signals are set to the high logic value and this means that the clk_D and clk_E are safe: the positive edges of these latter clock signals occur after each data bit have switched to the high logic value (i.e. the data is stable).

Since it is the first of the safe clock signals to be selected to cross the multiplexer, in our example, the clk_D is the clock signal driving the receiver synchronizer (“clk_synch”). It should be noted that a source synchronous synchronizer in a similar scenario could not properly work if driven by the nominal source clock signal (clk_tx). In fact, as showed in Figure 27, the positive edge of the source clock signal occurs when the incoming data are still switching.

The sequence of 1s is driven by the transmitter during the last clock cycle of the reset phase. As a result, once the control signal in the variation detector is correctly set, the reset switch to the low value and the variation detector circuit interrupts its operation by freezing the safe clock signal at the multiplexer output. Since the detector circuit works only during the reset phase, the power consumption overhead is minimized.

To validate the proposed architecture a set of experiments was performed. Since the goal of the architecture is to support a source synchronous communication in a high variability environment, we instantiated a platform composed of a transmitter sending data along with the clock to a mesochronous synchronizer (the receiver) and we injected process variations in the link wires. The mesochronous synchronizer is tightly coupled into the switch architecture.

In particular, in order to study the robustness of the communication when the ideal alignment between the data and the source synchronous clock is not respected, we inject an increasing delay into the clock signal wire. Therefore, we compared the performance of a baseline source synchronous architecture with the one achieved by a source synchronous architecture augmented with the proposed variation detector.

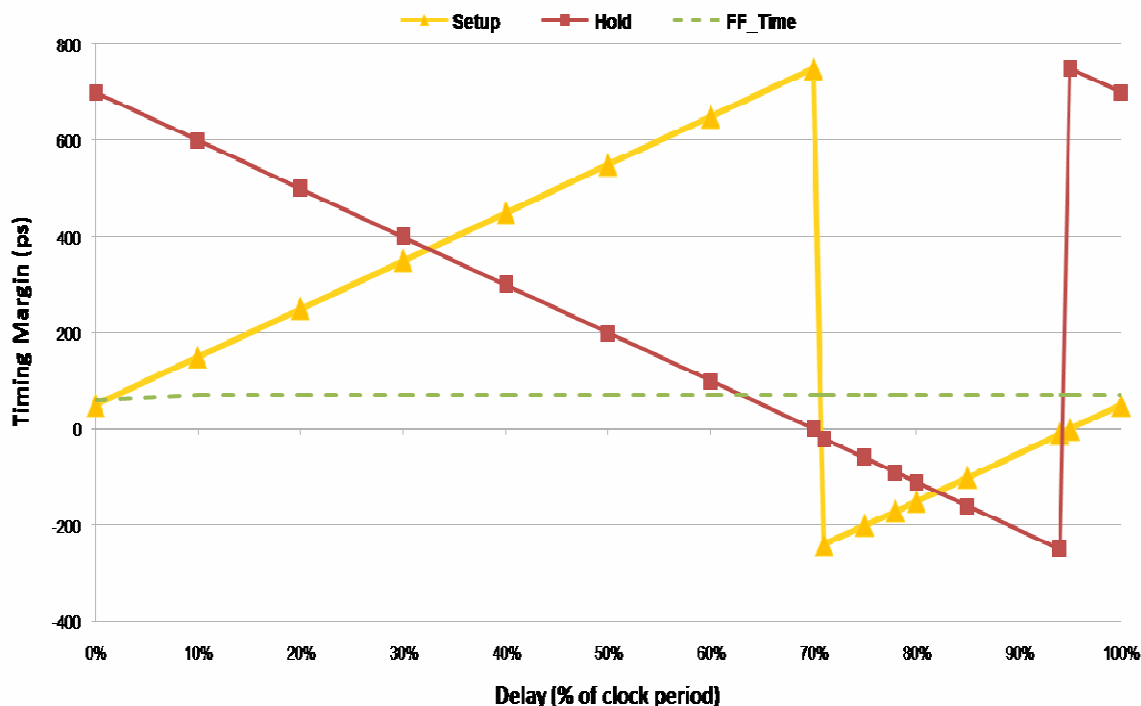


Figure 28: Timing Margin of the Mesochronous Synchronizer.

The Figure 28 reports the timing margin derived from a post-layout analysis of a source synchronous mesochronous synchronizer synthesized at 1GHz WITHOUT the variation detector. Setup and hold times have been experimentally measured by driving the mesochronous synchronizer under test with a source synchronous clock affected by an increasing delay and by monitoring the relative waveforms. X-axis reports the amount of delay (expressed as percentage of the clock period) injected into the clock line with respect to an ideal clock with null routing skew. Figure 28 also compares setup and hold times with the minimum values required by the technology library for correct sampling (denoted FF_Time).

First of all, we observe that the setup time increases and the hold time decreases linearly with the increase of the clock delay. When the clock delay reaches 65% of the clock period, the hold time violates the minimum timing margin and a failure is detected in the mesochronous synchronizer. Injecting a clock delay between 70% and 100% of the clock period, the source synchronous clock starts to sample the next incoming data and the behaviour of the system is unpredictable. As a conclusion, the baseline mesochronous synchronizer was able to support a clock skew with data between 0% and 65% of the clock period. 35% of the clock period is unsafe and a positive clock edge, loosely synchronous with the data, can only occur during the early 65% of the clock period.

This experiment was carried out considering an incoming data stable for 75% of the clock period at the mesochronous synchronizer input port. This accounts for the delay variability between data lines and for the non-null settling time usually found in post-layout link switchings.

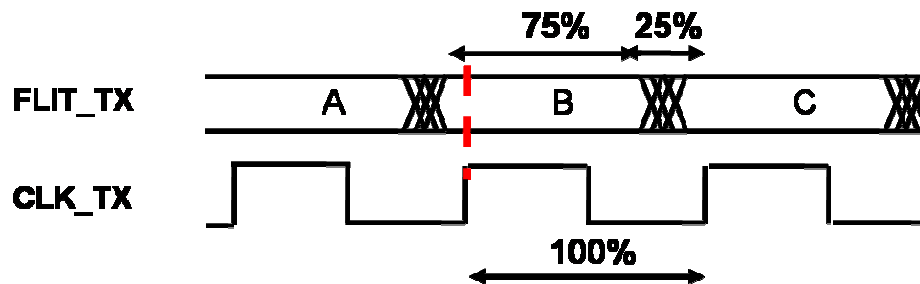


Figure 29: Settling time of the incoming data.

In order to perform a fair comparison, a similar experiment was performed for a mesochronous synchronizer WITH the variation detector connected in front of it. In this scenario, the transmitter clock crosses the variation detector before to reach the mesochronous synchronizer. The timing margin of this second system is showed in Figure 30.

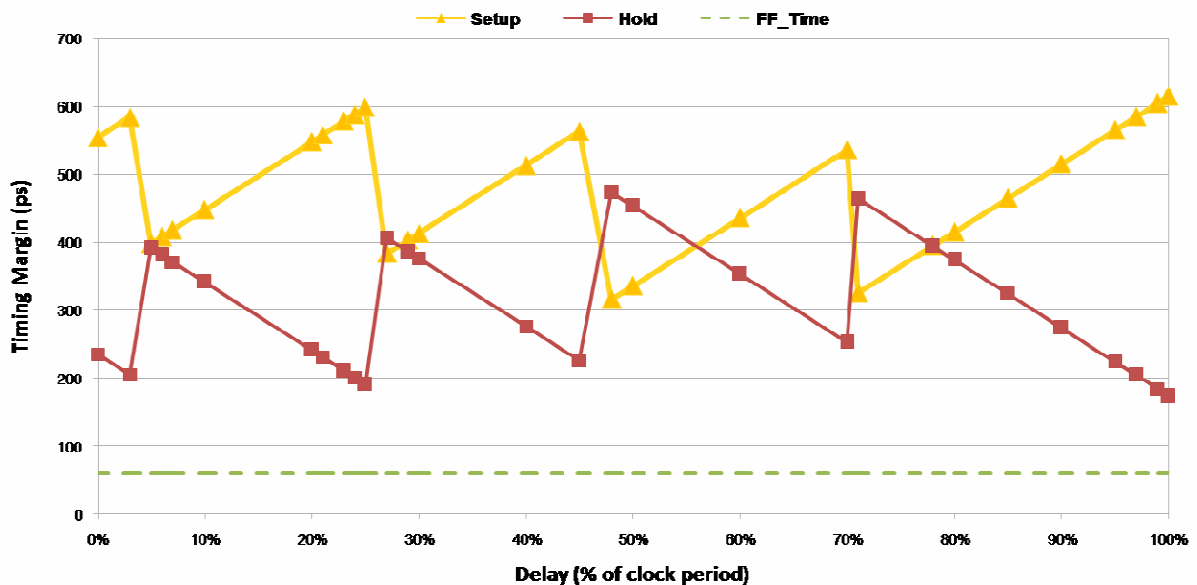


Figure 30: Timing Margin of a Mesochronous Synchronizer with a Variation Detector in front of it.

We can notice a significant difference of results due to the variation detector integration. The setup margin is strictly symmetric with respect to the hold margin. Moreover, the timing margin is composed of the periodic repetitions of the same triangular-shaped trend.

As before, at the increase of the clock delay corresponds a decrease of the hold margin although, in this case, as soon as the hold time degradation becomes significant and dangerous for correct circuit operation the detector re-establishes safe margins. This effect is achieved with the help of the variation detector selecting a delayed clock with a different offset when the timing margin becomes low. As result, we have 5 periodic repetitions of a triangular-shaped trend corresponding to the output of the 5 delay lines sequentially selected.



In conclusion, the proposed architecture guarantees a safe communication in a mesochronous link in every clock delay scenario as opposed to a baseline source mesochronous architecture. Further experimental results on the use of the variability detector can be found in deliverable D22.

We have reported here this long insight on the architecture of the variability detector and on its validation to make one point: unlike fully synchronous systems, the synchronization interface in a GALS system can be seen as a point in the architecture where layout non-idealities can be absorbed. A similar intuition has been used in the past to make also synchronous systems more flexible with respect to process variations. For instance, the Razor flip flop consists of sampling data with the nominal clock edge in the receiver block but also with a delayed clock edge to account for delay variability. Apart from the need to invalidate operations in the receiver block when a timing fault is detected (i.e., nominal data and delayed data differ) and to preserve the circuit from hold time violations, this solution goes in the direction of making circuit timing less strict. In other words, it is an early step in the direction of a GALSification process that the Galaxy project has thoroughly detailed.

Above all, how much does it cost to get the high level of decoupling and non-ideality absorption that a GALS system can provide? Traditionally, the general feeling is that implementing synchronization implies a significant cost. And this is indeed true. However, the Galaxy project has demonstrated that an investment in architecture development and refinement can make GALS technology accessible at no or marginal area, power and latency cost with respect to a fully synchronous architecture. The guiding principle for this optimization was to integrate synchronizers with the switch building blocks, thus sharing expensive buffering resources. We denote this approach as *tight coupling* as opposed to the *loose coupling*, where synchronizers are external blocks placed in front of the switches. Moreover, by merging synchronizer and switch, the former one does not represent a source of additional latency in switch-to-switch links, and synchronization latency can be reduced to the minimum. Design techniques for tight coupling can be found in Deliverable D6.

For the experimental results that follow and demonstrating the point above, the chosen switch radix is 5, to reflect switches commonly found in 2D mesh topologies. Post-layout results are obtained with a 65nm industrial technology library.

5.1.1 Tight integration of mesochronous synchronizers

As regards latency, while the fully synchronous switch takes 1 cycle in the upstream link and 1 cycle in the switch itself, the novel switch with tightly integrated synchronizers takes from 1 to 3 clock cycles to cross the same path, depending on the negative or positive skew ranging from -100% to +100%, respectively.

In order to estimate the area savings with the tight integration design strategy, we went through a commercial synthesis flow and refined RTL description of the mesochronous switches (tightly and loosely coupled) up to the physical layout. All the systems were synthesized, placed and routed at the same target frequency of 1GHz.

In Figure 31, the area footprint of switches is reported along with a breakdown pointing out the contribution of synchronizers and/or input buffers.

The tightly and loosely coupled solutions are compared against a vanilla (i.e., fully synchronous) switch; input_buffer area for this switch only refers to the area occupancy of a normal 2 slot input buffer.

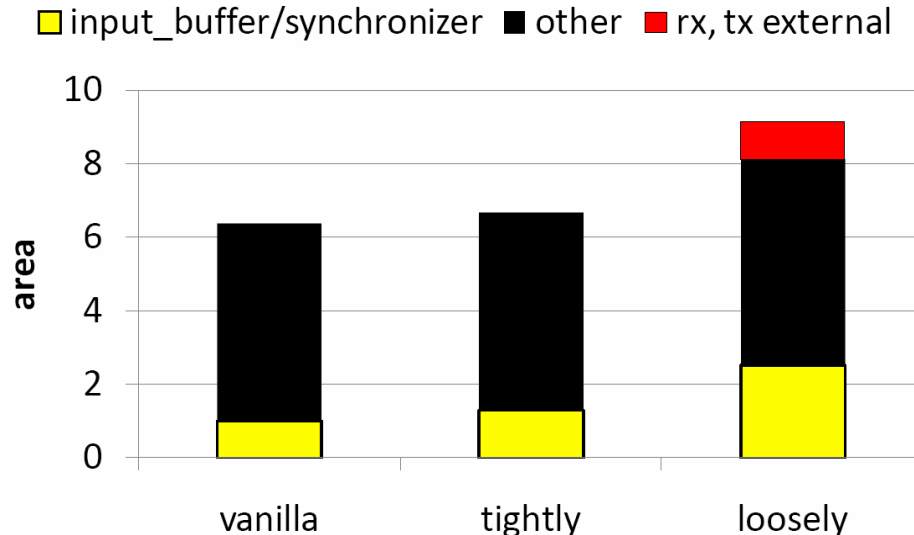


Figure 31: Area of mesochronous switches

For the loosely coupled solution, a 4 slot buffer is needed to cover the round trip latency, and this is most of the overhead for this solution. As clearly pointed out by the area breakdown in Figure 32, the sum of the transmitter and of the receiver synchronizers is almost equal to that of a 2 slot buffer, i.e., of the input buffer in the vanilla switch.

For the tightly coupled solution, input_buffer /synchronizer area refers to the multi-purpose switch input buffer (which is also the synchronizer). Clearly, there is almost no area overhead when moving from a fully synchronous to a tightly integrated mesochronous switch as they employ similar buffering resources.

From the performance viewpoint, our post-layout synthesis results confirm that the critical path of the switch is not impacted by the replacement of the vanilla input buffer with the tightly integrated mesochronous synchronizer. By experimenting with different switch radix, the critical path deviates only marginally in the two cases, therefore no performance penalty should be expected for the mesochronous switch.

A further step of our exploration was to contrast power consumption of the proposed mesochronous schemes. Our target design is a 5x5 switch in three different variants: the first has a tightly integrated mesochronous synchronizer per input port; the second has a pair of loosely coupled rx- and tx-synchronizers per input port; the last one is a vanilla switch with 5 fully synchronous input ports. Three different traffic patterns have been experimented to carry out the power analysis: idle, request for a random output port and parallel communication flows.

Post-layout simulation frequency was 700MHz for all the designs. As showed in Figure 32, in all the cases the highest power consumption is drained by the most buffer demanding solution, i.e., the loosely coupled design. Power consumption of the vanilla and tightly coupled designs are similar as expected; this is mainly due to the equivalent buffering resources deployed in both switches.

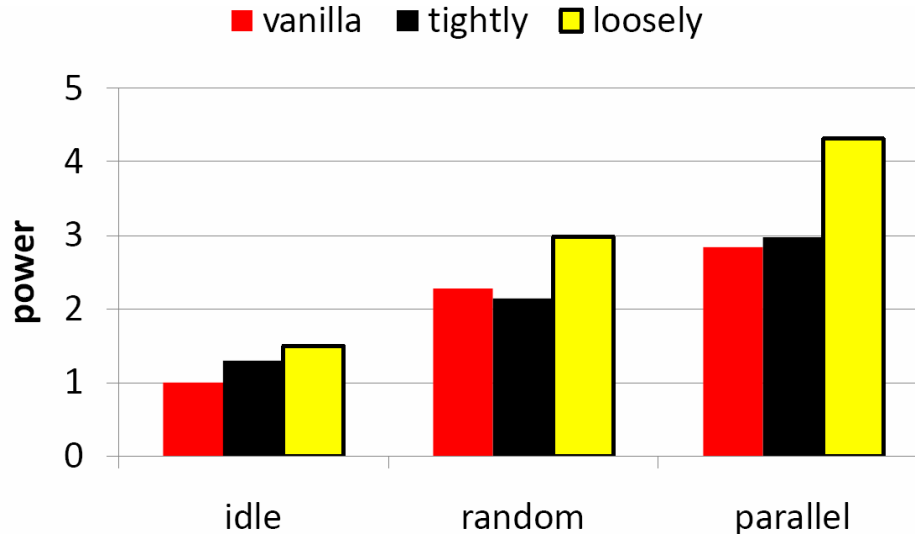


Figure 32: Power of mesochronous switches

5.1.2 Tight integration of Dual-Clock FIFOs

In the GALS architecture paradigm explored in the Galaxy project, dual-clock FIFOs are used to decouple IP core speed from network speed, a network which then undergoes inference of mesochronous clocking. Also dual-clock FIFOs can be tightly integrated into switch building blocks, with design techniques reported in deliverable D13.

As the sender and the receiver have different clocks, the latency of the dual-clock FIFO depends on the relation between these two signals. The aforementioned metric can be decomposed into two parameters: the first is a ΔT_{rx} that is the time between the falling edge of the clock sender and the rising edge of the clock receiver. ΔT_{rx} can vary between 0 and 1 clock cycle depending on the offset between the clock signals. The second parameter is the number of clock cycles required by the read pointer to reach the location pointed by the writer. As reported in the Table 5, three different scenarios have been analyzed in order to characterize the crossing latency of the switch with the integrated dual-clock FIFO.

Table 5: Switch crossing latency

I°	minimum latency	$\Delta T_{rx} + 2Clock_{rx}$
II°	empty deassertion	$\Delta T_{rx} + 3Clock_{rx}$
III°	maximum latency	$\Delta T_{rx} + Clock_{rx} \times (BufferDepth - 1)$

In the first, when the read and write pointers are adjacent and the writer is preceding the reader, a minimum latency occurs. In this case, the read pointer opens the mux window after ΔT_{rx} for the data it is pointing to and the next data (which is the one being currently written) will be read after a further clock cycle. Therefore, the minimum latency to traverse the FIFO synchronizer in this case is $\Delta T_{rx} + 1 Clock_{rx}$.

In the second case, when the buffer is empty and a write operation occurs, a $\Delta T_{rx} + 1 Clock_{rx}$ is needed to clear the emptiness condition and a further clock cycle is required to enable the data at the multiplexer output.



Finally, when the distance between the pointers is maximum (fullness condition), the required time for the reader to point the current writer position is given by a ΔT_{rx} (offset between the clock signals) plus a contribution which depends on the number of buffer slots preceding the one currently pointed by the writer (which accounts to $BufferDepth-2$).

Please note that in all latency results 1 Clock_{rx} cycle has been added to account for the time from the FIFO output to the input of the switch output buffer. In fact, the Table reports the overall switch crossing latency.

In order to characterize the integration of a dual-clock FIFO into a NoC switch architecture from the area and power viewpoint, three different systems have been considered. The first is the conventional 5x5 vanilla switch with a 2-slot input buffer and a 6-slot output buffer per port. The second version is a switch where a dual-clock FIFO with 6 buffer slots has been tightly integrated into each input port. In order to carry out a fair comparison with the vanilla system, total buffering resources have been kept equal, i.e., the output buffer size in the switch with the FIFO synchronizer has been reduced from 6 to 2 slots. The last configuration is a vanilla switch (2-slot input, 6-slot output) with an external dual-clock FIFO (6 buffer slots) per input port.

To assess area occupancy, all the above switch configurations have been synthesized, placed and routed at the same target frequency of 1GHz. Total area of the tightly coupled system exhibits almost the same area footprint of the vanilla switch. This is a direct consequence of the fact that exactly the same buffering resources have been deployed in a specular fashion (between input and output) in these systems as explained above.

As showed in Figure 33, the main difference is in the distribution of the cell area devoted to synchronization. Since the classical 2-slot input buffer has been replaced with a larger 6-slot one in the tightly coupled system, the switch presents a higher input_buffer/synchronization area with respect to the vanilla system.

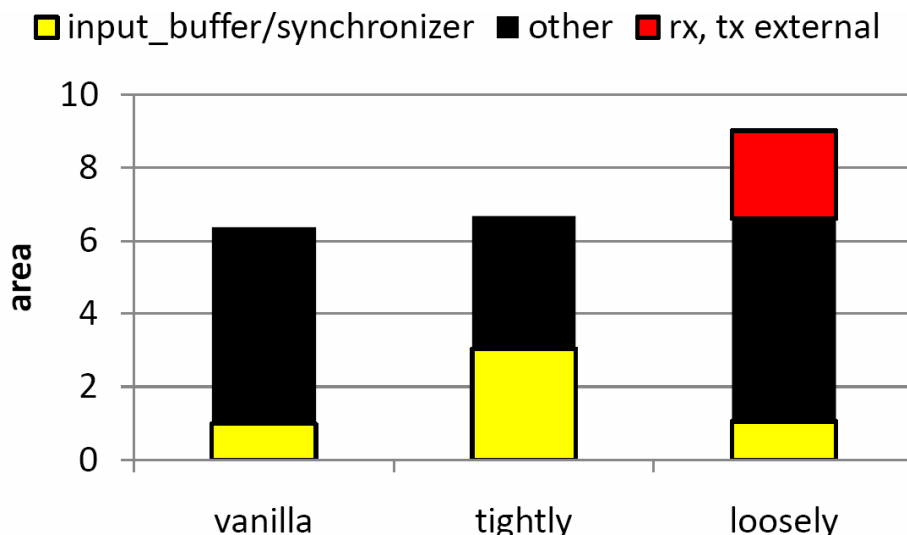


Figure 33: Switch area comparison

The loosely coupled system features the same area overhead (with the same distribution of input_buffer and “other” cell area) of the fully synchronous switch plus a further synchronization area due to the external block implementing the dual-clock FIFO. Again, area



overhead of the tightly coupled switch variant is negligible with respect to a fully synchronous switch.

As for the mesochronous synchronizer, a number of experiments has been carried out to assess the power consumption of a switch integrating dual-clock FIFOs on each input port. The vanilla, tightly and loosely coupled systems have been tested under different traffic patterns: idle, random and parallel. Post-layout simulations have been carried out at 800MHz. All the results were grouped per traffic pattern. As we found out in previous experiments with the switch connected with an external mesochronous synchronizer, the area overhead comes along with a power penalty. In fact, the switch with the external dual-clock FIFO is the most power greedy under all possible traffic patterns, as showed in the Figure 34. This is due to a larger amount of buffering resources. From the power viewpoint, there is a substantial benefit when integrating the dual-clock FIFO in the switch architecture, in fact, the tightly coupled design is the most power saving among those under test.

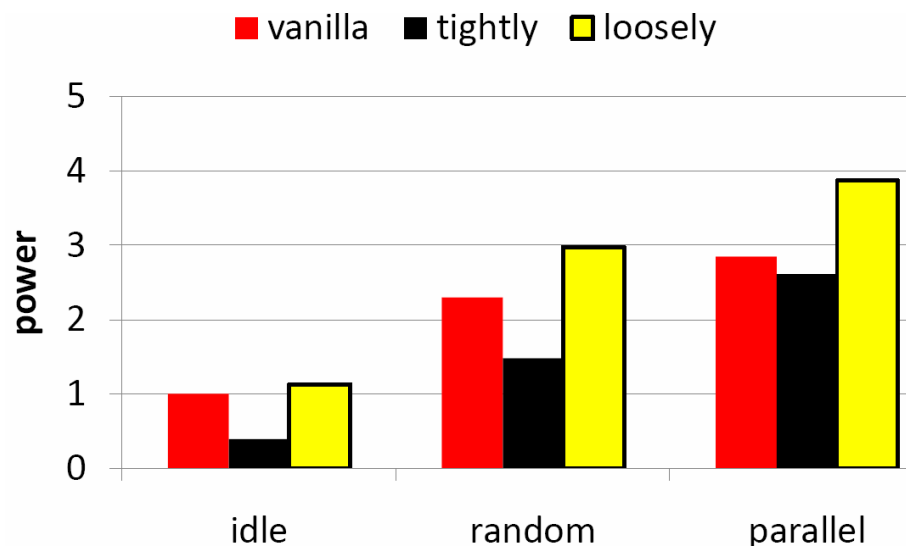


Figure 34: Switch power comparison

The motivation lies in the inherent clock gating which is implemented by our dual-clock FIFO, which clocks only one bank of flip-flops at a time out of the total input buffer. If the incoming data is not valid, then the token ring circuit does not even switch thereby gating the entire input buffer. Obviously a similar clock gating technique can be applied to the vanilla switch as well, and in fact the key take-away here should be that the dual-clock FIFO integration into the switch does not imply any major power overhead, as long as buffer depths of at least 6 flits are used in all switch variants for performance optimization.

The experiments above prove one important point: it is possible to evolve a fully synchronous switch into a GALS counterpart with marginal area and power overhead while preserving the operating speed. This approach holds promise of resulting in cost-effective implementations of GALS NoCs where the source synchronous synchronization paradigm is used for clock domain crossing and the switching and synchronization architecture are co-designed.



6 CONCLUSIONS

GALS systems despite having well-known drawbacks such as non-mature design-flow, lack of testing methods and no well-established target application, may offer significant advantages like better EMI characteristics, power reduction and process variability mitigation. These advantages have been explored in detail with theoretical research and chip fabrication and measurements. In addition, a systematic evaluation of GALS methodology benefits in NoC design process is performed.

Power saving was always considered an inherent property of asynchronous design. Therefore, the expectations from GALS designs were always very high. On the other hand, most of the practical GALS demonstrators have showed only marginal improvement in this direction. For example, the GALS baseband processor for WLAN [19] showed an improvement of only 1% in comparison to the respective synchronous chip.

In retrospect, it is easy to understand this result. The inherent power reduction obtained by using a GALS-based system is essentially based on the same paradigm as the low-power techniques for synchronous circuits: trigger the locally synchronous block only when it is needed and lower the switching activity to minimum. Therefore, the achieved results are very similar.

We can expect more if the GALS approach is applied in conjunction with voltage and frequency scaling as demonstrated in this work. GALS architecture is very well suited for such technique since the mutual communication between blocks is asynchronous. The boundaries of the GALS blocks are defined and the partitioning naturally leads to a hierarchical layout process. Therefore, the introduction of different power rings in the layout and insertion of DC-DC converters are easier. In the proposed architecture, each synchronous unit is encapsulated with its own local clock generator and local supply power unit and can be programmed through a set of dedicated power modes. No fine control software is required during voltage and frequency programming and thus minimal latency cost is observed. A potentially easy implementation of V_{DD} -Hopping technique together with dynamic workload (synchronization activity) monitoring offers a huge benefit to GALS realization compared to fully synchronous design. This DVFS architecture can be fully integrated in CMOS techniques with an area cost affordable for coarse grain IP units.

Another very important property of GALS designs is the reduction of electro-magnetic interference (EMI). In this report, the GALS methodology was investigated in order to evaluate the ability for EMI reduction. We have generated a software tool based on Matlab to simulate EMI properties of the digital GALS systems. It supports simulations of GALS/synchronous systems with different granularity, frequencies, current shapes, topology, and other parameters. Using the software, we have modeled GALS and synchronous system in order to evaluate the effect of different topologies, architectures on EMI reduction.

The results show that the reduction of high spectral components can be successfully achieved with jitter introduction. Additionally, for synchronous systems, EMI at low frequencies can be reduced by a phase shift introduction. However combining those two features would be hard in synchronous systems because of problems in the timing closure.

In GALS systems, phase shift is already present by the nature of the GALS methodology. Local clock generators also naturally generate clocks with jitter, but this feature of ring



oscillators was not deeply analyzed here. In this work, we have modeled explicit jitter introduction with the special jitter generators based on LFSR structures. By adding jitter in a GALS system, we can achieve a significant reduction over whole spectrum, not affecting the functionality of a system. The reduction of over 20 dB can be achievable, as illustrated in the results. Moreover, the current peaks in time domain can be reduced up to 40% in GALS systems.

We have found that there is almost no correlation between EMI reduction and data transfer intensity (i.e. clock pausing rate) in GALS modules. The greatest impact has the used set of frequencies and the granularity of GALS partitioning. Therefore, we have analyzed the effect of partitioning to EMI profile going from the pure synchronous systems and ending up with the pure asynchronous systems. What we have observed is that with deeper partitioning some measurable results could be seen only up to partition of 15 GALS blocks. Further partition gives no improvement on EMI profile. For the saturated case, that can be also a model of an asynchronous case, the achieved EMI reduction is around 25 dB.

Finally, we have proven the proposed concepts in praxis and implemented low-EMI FFT processor that can operate both in synchronous and GALS mode. The measurements confirmed our theoretical simulations and we have achieved 13 dB reduction of EMI between the best GALS and best synchronous mode.

GALS systems may offer significant advantages with respect variability mitigation, as they essentially divide a complex system into small, loosely coupled and often even independent modules. This way, the implementation of a fault-containment mechanism becomes easier to implement and a networked multi-core system can be designed to survive to the failure of one or more IP cores and/or network links. However, partitioning the design into LSIs effectively reduces the length of the critical paths and which has the effect of increasing local process variability impact since the variability of the critical path is affected by the length of the critical path in an inverse proportion. Therefore, an effective technique that limits the impact of local delay variations is necessary.

The benefit of the selective redundancy-based technique to limit the impact of local delay variations is demonstrated. Only critical segments (critical paths) are replicated and the viable improvement is achieved with the redundancy factor as low as $R=2$. This way, the proposed technique can optimally be used in the design of large synchronous digital systems. The inherent property of the technique is that the larger the variations are, the greater is the reduction in the maximum critical path delay, and therefore, the higher is the improvement in the maximum frequency the circuit can run. This is specifically beneficial for GALS design because the effectiveness of the technique is higher on local synchronous partitions than on the whole synchronous circuit.

The studies have shown that the technique can already be successfully applied for 65nm CMOS technology process where an optimal point has been shown to exist in the speed vs. area/power tradeoff. Even more advantage is expected for future nanoscale CMOS technologies such as 45nm and 32nm nodes. The technique can be especially beneficial in the low-power domain where reduced supply voltages lead to significantly increased local variation.

The potential advantages of NoCs in variability mitigation thanks to the GALS paradigm is that it is possible to implement a network with support for unexpected and unpredictable topology irregularity: failing nodes just have to be isolated while preserving correct behaviour



GALAXY

GALS InterfAce for CompleX Digital
SYstem Integration

Confid. Level: Public
Date : 30/07/2010
Issue: 1

of the remaining portion of the system. In particular, a new physical routing approach for robust bundled signalling on NoC links has been proposed. Tightly matched signal propagation and strong crosstalk protection are key requirements for next-generation NoC links featuring GALS synchronization and low-swing signaling. In our work, we presented a new methodology for NoC global link routing which addresses these challenges. Our approach creates bundled link routes with geometrically matched wires, thus leading to much reduced intra-link variations. Moreover, our link router supports high-regularity wire spacing and shielding strategies. Delay variation among different wires of a link is 25% to 70% lower than what can be achieved using a state-of-the-art timing-driven global routing flow. Additionally, crosstalk effects are reduced by more than 30%. This physical routing solution effectively deals with the systematic component of process variations, in that it forces all wires of a GALS link to undergo the same routing and layout conditions.

We have also presented circuit-level compensation techniques for full swing vs low-swing on-chip interconnects affected by process variations. Adaptive body bias (ABB) and adaptive supply voltage (ASV) are effective methods for post-silicon tuning to reduce variability on generic combinational circuits or microprocessor circuit sub-blocks. The focus was on global point-to-point interconnects, which are evolving into complex communication channels with drivers and receivers, in an attempt to mitigate the effects of reverse scaling and reduce power. This work is at lowest level of the design hierarchy, and deals with the reliable design of the electrical wires building up a GALS link, although it is not restricted by the specific synchronization paradigm. The characterization of the performance spread of these links and the exploration of effective and power-aware compensation techniques for them is becoming a key design issue.