GALS-SA TEST EXTENSION

Răzvan JIPA[1]

Abstract: *The paper presents a flexible and scalable test solution designed for a new class of hybrid circuits - GALS-SA that combine the structured ASIC advantages with the GALS (Globally Asynchronous Locally Synchronous) architecture. The test solution consists of two parts: intrinsic tests that ensure important functions are always tested at power-up and extension tests that are required only during prototype debug and characterization and are implemented on configurable logic that can be used later for user design for mass production. The proposed solution offer means to implement any user-specific test function without affecting the platform intrinsic architecture with small area penalty.*

Key words: *GALS, synchronous, structured ASIC, test.*

1. Introduction

The advance in microelectronic technology that follows Moor's law offers the digital circuit designer the possibility to build bigger and more complex circuits, but at the same time creates several challenges when building large structures. One of these challenges is to distribute a global clock across the design with minimum skew without implying large design efforts and occupied area.

A possible solution to these problems is the hybrid synchronous-asynchronous circuits or GALS (globally asynchronous locally synchronous) circuits. These circuits were first introduced by Chapiro [1] and they advocate for a synchronous design style for the modules while the interconnections between modules are asynchronous communication channels. This type of design brings the advantage of removing the global clock network and diminishing the clock network power consumption with up to 70% that accounts up to 20% from the total power consumption of a synchronous circuit [3].

Interconnecting synchronous modules with an asynchronous environment is accomplished by the asynchronous wrappers that embed the synchronous module transforming it into an asynchronous one. It contains a local clock generator, and asynchronous finite state machines (AFSM) for synchronous-asynchronous transformation. A wrapper implementation with a modular approach that uses 4-phase bundled data protocol was proposed by Muttersbach in [7] (Figure 1).

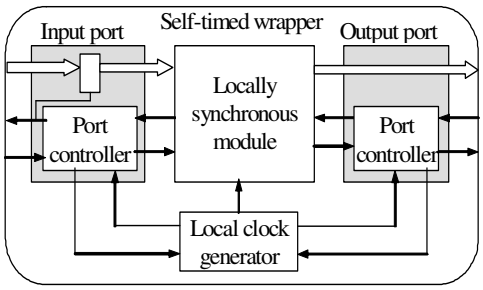

Fig. 1. *Asynchronous wrapper*

[1] Dept. of Electronics and Computers, *Transilvania* University of Braşov.

Another possible solution used to overcome the complexity of large designs is the structured ASIC platforms also called zero mask-change ASIC [8]. These platforms provide pre-manufactured and verified elementary logic structures and require inexpensive metallization process in order interconnect the logic elements to implement the desired function. This technological approach provides cost reduction compared to the traditional standard cell flow and FPGA (for moderate production volumes) as well as power reduction compared with FPGA design offering similar performances.

A combination of those two concepts into new hybrid platform was previously analyzed in [4] and general architectural solutions were provided, but no testing solution was presented. In [2] and there is presented a general purpose test structure for GALS designs based on the JTAG protocol where all modules are tied in a chain. This approach although universal may not offer the best solution in terms of test time and is not compatible with the architectural constraints the GALS-SA platform requires. In [6] is presented a custom test solution that ensures well-known synchronous testing methodologies can be applied to the GALS platforms ensuring asynchronous resources can be tested in order to guarantee their function.

The paper is based on the architectures previously developed by author in [5] and [6] and presents a modified version of the test module that allows test extension modules implementation without using intrinsic circuit area, but utilizing logic array cells that can be reused by user designs if the extension test modules are not required. This proves to be a flexible solution during platform characterization and debug when several additional tests are required while for mass production the test extensions are no longer required leaving more room for the user designs.

2. General Architecture

The architecture proposed by author in [6] ensures that a hybrid GALS-SA platform (GALS architectures implemented on a structured ASIC platform) is able use the same serial link to perform structural and functional tests of the asynchronous channels and to calibrate the local clock generators (specific to a GALS architecture). The solution uses a 6-wire serial protocol to send configuration and calibration information for local clocks as well test commands to intrinsic local test modules attached to each synchronous logic block. In Figure 2 is presented the proposed general architecture of a GALS-SA platform where each synchronous logic block has associated a test module and serial link module.

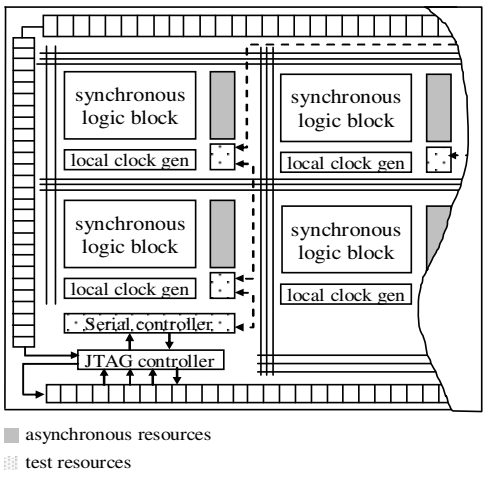

Fig. 2. *GALS-SA general architecture*

The test resources are implemented on the intrinsic structure of the platform and can not be customised later the technological process opposed to the logic blocks that allow the implementation of any synchronous random logic that can be customised on the last stage of the manufacturing process. Due to these particularities the intrinsic test module

implementation has to be a compromise between features and occupied area. The proposed solution in [6] (Figure 3) restricts to a minimum the features to reduce the area overhead. The test modules presented implement only two tests:

- Functional test of the asynchronous channel that exercises the data link and the handshake signals.
- Structural test that measure the asynchronous channel latency.

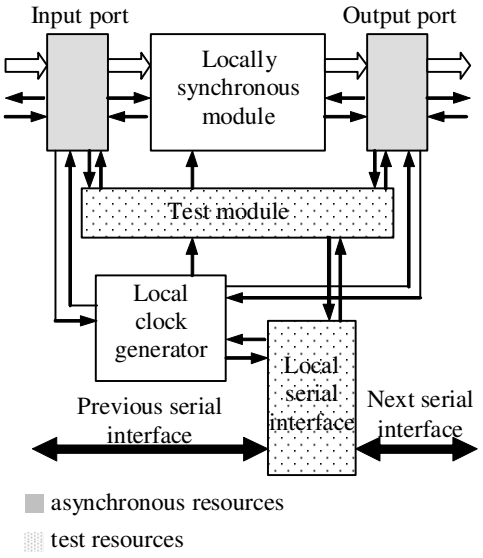

Fig. 3. *Test solution architecture*

Although this architecture offers support to test the most important features of the asynchronous channels, it is not flexible and does not offer any mean to add new asynchronous channel tests.

3. Test Extension Architecture

To ensure that a GALS-SA platform is capable to provide means to implement different tests that targets the asynchronous or the synchronous resources, a test extension needs to be added to the already existent test resources. Since the test resources are intrinsic circuits, then a flexible test extension need to be implemented on the synchronous logic block in order to allow the implementation of any test. This approach requires some modification of the main serial controller and on the local test modules and of the serial transfer protocol. The following sections describe the custom extensible test solution proposed by author for GALS-SA platforms.

3.1. Serial Protocol

The communication is performed using variable length serial frames that contain a set of compulsory fields (serial chain test module address and command) and optional fields whose number, content and length depend on the type of command. For each received frame, a response frame is issued by the target module and contains serial protocol status bits and optional status data. The general format of the frames is described in Figure 4.

The serial protocol status bits are used to specify if the transmitted frame was valid or if it contains the address of a non-existent module or a not-implemented operation was selected. The response frame is be issued by the target module only after the operation is complete and clears the serial bus for another access as described in Figure 5.

Because each serial frame transmitted is accompanied by "valid" signal (*serTxEn/serRxEn*) the frames can have variable length and this provides a flexible enough solution to ensure that new type of frames can be added without any modification of the serial protocol.

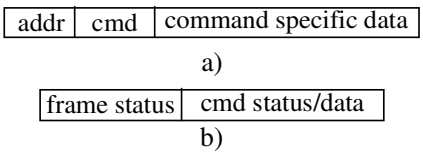

Fig. 4. *Serial frames structure for transmit* (*a*) *and receive* (*b*)

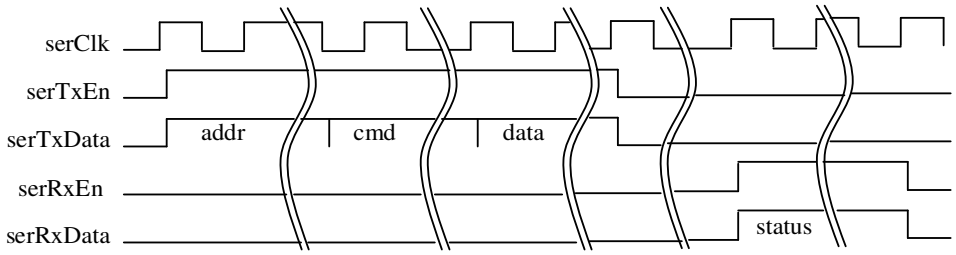

Fig. 5. *Serial connection with TX and RX frames*

4. Test Extension

The test extension implementation needs to follow several constraints in order to obtain a cost-effective, flexible solution:

- Keep to a minimum the hardware structures added to the already existent intrinsic modules.
- Do not affect the timing of already existent modules.
- Provide a simple extension that ensures ease of design and implementation of the additional test using the platform logic blocks.

The proposed solution, whose architecture is described in Figure 6, was devised such way to respect all the above enumerated constraints.

From the transmitter point of view, the main serial controllers (Figure 2), there are minimum changes required in order to support the new test extension:

- Add a 1-bit via programmable input that enable/disable the test extension.
- Add a new input that specified the extended test frame length based on the extended test command type; this value is generated by a logic module implemented on the synchronous logic blocks.

As described in the implementation section this increases the area with a negligible amount.

The local serial controller needs to implement several changes, because it is responsible for translating the received serial frames into test action and to compose the return frames based on the test results and status. Local serial controller uses a shift register to convert from serial to parallel the incoming frames and passes through to test extension interface the command and its specific data if there are any. This approach solves data receiving part of the test extension with zero area overhead.

On the transmit side, when return frames are generated, in order to avoid increasing the length of the serialise circuit, already kept to a minimum by the short length of the intrinsic implemented test commands, the solution is to let the external test extension to implement its own serialize circuit using logic blocks and implement only the circuits that perform the selection between the intrinsic logic and external test logic (Figure 7).

5. Implementation and Results

To compare the results in terms of occupied between the solution proposes in [6] and the improved solution in this paper the same technology and EDA was selected for implementation. The target technology was a 90 nm standard cell and Magma BlastFusion was used tool to perform all the stages from RTL synthesis to layout.

Also a test extension that offers two new tests was included. The new tests logic were implemented using as target technology the 90-nm eASIC Nextreme logic fabric as synchronous logic blocs

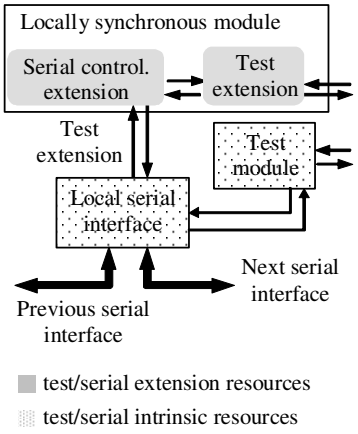

Fig. 6. *Test solution architecture*

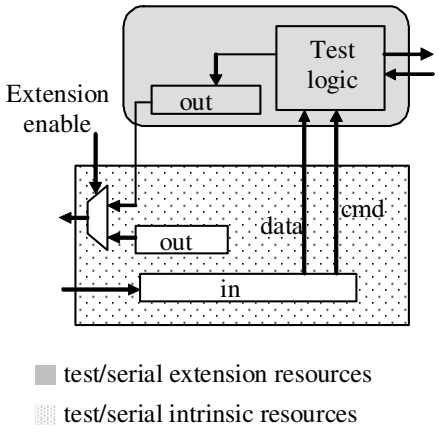

Fig. 7. *Test solution architecture*

together with Magma BlastCrate SA for a complete implementation.

One additional test was used to increase the coverage of the asynchronous channel functional test by allowing user selected patterns to be sent over the channel opposed to the intrinsic test that exercise the channel with a series of pseudorandom generated patterns. The second test allow the user to read back the signature computed after receiving each patter on the same asynchronous channel function test. In case of communication failure the user has a mean to observe closely the failing values localize the source of errors during debug session.

Comparing the results (Table 1) of the solution implemented in [6] with the proposed solution, one can notice the area increase that might seems moderate is rather negligible when compared with the area of a single synchronous block. Even the total area occupied by the test solution including external test extension is small compared with the logic block area.

The test extensions occupy a small portion of the platform total available logic blocks providing valuable testing options that may ease the possible debug process. The small footprint is achieved because a lot of resources are shared with the intrinsic test and serial modules and in case of necessity the area can be used by user design if the prototype debug process is complete.

Implementation results Table 1

Module	**Area [um^2]**	**Overhead [%]**
Architecture without test extension		
Main serial controller	2666	-
Local serial controller	4472	-
Test controller	8679	-
Architecture with test extension		
Main serial controller	2829	5.66
Local serial controller	5369	16.7
Test controller	10257	15.4
Main serial extension	95	-
Local serial extension	973	-
Total area	3706	23.4
Overhead compared to logic block area		
Total area	19523	4.64
Area increase	3706	0.88
Logic block	420000	-

6. Conclusions

The paper presents a custom test extension developed by the author to address the testing needs of GALS-SA hybrid platform. The proposed test solution relies on the intrinsic test structure of the

platform that offer limited testing capabilities, in order to gain access to the logic module that needs to be verified and adds resources to allow the designer to implement as many additional test as required using the programmable logic blocks. In this way a is obtained a flexible and scalable test solution that address the test requirement and the post manufacturing debug features of the GALS-SA hybrid platforms.

The area penalty for implementing the intrinsic portion of the test resources such is rather negligible - 4.64% overhead compared with the total synchronous resources area and is permanent for all designs. The area occupied by the logic implementing the test module depends on the complexity and number of test, but this is not considered a permanent overhead since the modules are implemented on programmable logic that can be reused n other purpose when a different design is implemented and the test extensions modules are not required.

References

1. Chapiro, D.: *Globally-Asynchronous Locally-Synchronous Systems*. In: Ph.D. Thesis, Stanford University, STAN-CS-84-1026, Oct. 1984.
2. Gurkaynak, F.K., et al.: *A Functional Test Methodology for Globally-Asynchronous Locally-Synchronous Systems.* In: Proceedings of Eight International Asynchronous Symposium, ASYNCH 2002, Manchester, UK, 2002, p. 181-189.
3. Hemani, A., Meincke, T., et al.: *Lowering Power Consumption in Clock by Using Globally Asynchronous Locally Synchronous Design Style.* In: Proceedings of 36th Design Automation Conference, New Orleans, USA, 1999, p. 873-878.
4. Jipa, R., Tulbure, T.: *Gals Designs Implementation on "Structured ASIC" Platforms.* In: Bulletin of the *Transilvania* University of Braşov, Vol. 1 (50), Series I, 2008, p. 351-356.
5. Jipa, R.: *Dedicated Solution for Local Clock Programming in GALS Designs.* In: Proceedings of International Semiconductor Conference CAS 2008, Sinaia, Romania, 2008, p. 393-396.
6. Jipa, R.: *Custom Test Solution for GALS Modules Implemented on Structured ASIC Platforms,* In: Proceeding of the Third International Electronics, Computers and Artificial Intelligence Conference, ECAI 2009, Piteşti, Romania, 2009, p. 7-13.
7. Muttersbach, J., Villiger, T., Fichtner, W.: *Practical Design of Globally-Asynchronous Locally-Synchronous Systems.* In: Proceedings of Sixth International Asynchronous Symposium, ASYNCH 2000, Eilat, Israel, 2000, p. 52-59.
8. http://www.easic.com.