DESIGN AND IMPLEMENTATION OF AXI4-LITE TO APB BRIDGE BASED ON ADVANCED MICROCONTROLLER BUS ARCHITECTURE (AMBA) 4.0 USING VERILOG

JUAN PABLO ROMERO GALINDO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO-MECÁNICAS
ESCUELA DE INGENIERIA ELECTRICA, ELECTRONICA Y TELECOMUNICACIONES
BUCARAMANGA
2016

DESIGN AND IMPLEMENTATION OF AXI4-LITE TO APB BRIDGE BASED ON ADVANCED MICROCONTROLLER BUS ARCHITECTURE (AMBA) 4.0 USING VERILOG

JUAN PABLO ROMERO GALINDO

Trabajo de Grado para optar al título de Ingeniero Electrónico

DIRECTOR
ELKIM FELIPE ROA FUENTES
INGENIERO ELECTRICISTA, PH.D

CO-DIRECTOR
CKRISTIAN RICARDO ESTEBAN DURAN BLANCO
INGENIERO ELECTRÓNICO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERIAS FISICO-MECÁNICAS
ESCUELA DE INGENIERIA ELECTRICA, ELECTRONICA Y TELECOMUNICACIONES
BUCARAMANGA
2016

CONTENTS

LIST OF FIGURE

Page.

LIST OF TABLES

Page.

RESUMEN

Título Diseño e impletentación de un puente entre AXI4-lite y APB basado en *Microcontroller Bus Architecture (AMBA*) 4.0 Usando Verilog[*].

Autor Juan Pablo Romero Galindo[**].

Palabras clave Microelectrónica, buses para periféricos**,** APB, AXI-4 Lite, MBus, Wishbone, OPB, STbus, periféricos, protocolo, arquitectura, microcontrolador.

DESCRIPCIÓN

Este trabajo presenta una guía para la selección de buses especializados en el manejo de periféricos de bajo consumo. Da a conocer algunas de las principales características de varios de los principales protocolos que se encuentran disponibles en el mercado, los cuales son Wishbone de Opencores, MBus de la universidad de Michigan, CoreConnect de IBM, STBus de STMicroelectronics y APB de ARM.

El bus APB fue seleccionado para ser implementado dentro de un microcontrolador de 32-bit basado en la arquitectura RISC-V diseñado por el grupo de investigación Onchip. Este bus se seleccionó debido a la practicidad en el manejo de periféricos y a su alta compatibilidad con el sistema en el cual se deseaba implementar, reduciendo el tiempo de adaptación y facilitando la interfaz de conexión. La implementación de este bus se realiza con el fin de manejar de forma más eficiente los periféricos de bajo consumo, disminuir el área y facilitar las interfaces que conectan cada periférico.

Los resultados de síntesis muestran un área de $9\mu m^2$ con una densidad de potencia de 205µW/MHz, siendo implementado en tecnología CMOS 130nm. El bus APB controla un DAC, un ADC, un GPIO y SPI esclavo. Para mitigar posibles problemas de metaestabilidad, se implementan dos configuraciones de sincronizadores de señales para cada señal que pasa a un dominio de reloj diferente.

[*] Trabajo de grado

[**] Facultad de ingenierías Físico-mecánicas. Escuela de Ingeniería Electrica, Electronica y Telecomunicaciones. Director: Elkim Felipe Roa Fuentes. Co-Director: Ckristian Ricardo Esteban Duran Blanco

ABSTRACT

Título Design and Implementation of AXI4-Lite to APB Bridge Based on Advanced Microcontroller Bus Architecture (AMBA) 4.0 Using Verilog[*].

Author Juan Pablo Romero Galindo[**].

Keywords Microelectronics, peripheral buses, APB, AXI-4 Lite, MBus, Wishbone, OPB, STbus, peripherals, protocols, architectures, microcontrollator.

DESCRIPTION

This paper presents a guide for the selection of specialized buses in the management of low power peripherals. It unveils some of the key features of several of the major protocols available in the industry, such as Wishbone from Opencores, MBus from University of Michigan, CoreConnect from IBM, STMus from STMicroelectronics and ARM from APB.

The APB bus was selected to be implemented within a 32-bit RISC-V base microcontroller, designed by the research group Onchip of University Industrial of Santander (UIS). The APB bus was selected due to the practicality in the handling of peripherals and its high compatibility with the system in which it was desired to implement, reducing the time of adaptation and facilitating the connection interface. The implementation of this bus is done in order to more efficiently handle low-power peripherals, decrease the area and facilitate the interfaces that connect each peripheral.

The synthesis results show an area of 9µm^2 with a power density of 205µW/MHz, being implemented in 130nm CMOS technology. The APB bus controls a DAC, an ADC, a GPIO, and a SPI slave. To mitigate potential metastability problems, two different configurations of signal synchronizer are implemented for each signal passing to a different clock domain.

[*] Bachelor degree.

[**] Faculty of Physico-Mechanical Engineering. School of Electronics and Electrical engineering and telecommunications. Advisor: Elkim Felipe Roa Fuentes. Co-Advisor: Ckristian Ricardo Esteban Duran Blanco

INTRODUCTION

Nowadays, the reducing power consumption and size on a chip receive a lot of attention within the industry, developing the idea of having a dedicated bus in handling peripherals. Due to the approach of low resources consumption, the handling should be characterized by being a low complexity interface, optimized for low power consumption.

Some of the most popular handling protocols used in the industry are Wishbone from Opencores [1], MBus from Michigan Micro Mote [2], CoreConnect from IBM [3], STBus from STMicroelectronics [4], and APB from ARM [5]. These buses must be analyzed to identify the main features and the corresponding disadvantage in a specific implementation.

This paper offers a survey of these protocols in order to help a reader in the selection of the proper specialized bus. A comparison is performed, providing an overview of what bus may be more convenient according to the application.

An APB bridge has been designed an implemented within a 32-bit microcontroller in 130nm CMOS. Different considerations that must be taken into account for the design and implementation of a bridge between AXI4-Lite and APB buses are studied. The main points in the APB implementations are a proposed Finite State Machine (FSM) to control the read and write transactions, and the study of two different synchronizers topologies needed for the secure connection between the two buses. Finally, the APB bridge is synthesized and integrated into a RISC-V based microcontroller

1. ON-CHIP PERIPHERALS BUSES ARCHITECTURES REVIEW

Using a standard bus is important if you want to increase the probability of success when to integrate a design with another intellectual property (IP) core [6] for the first time, is required.

A standard bus guarantees its performance under diverse operating conditions and facilitates the modular component design of the system on a chip (SoC). When a designers group get to a consensus of a standard bus to implement, the integration of each module is easy and safe.

Some of the main semiconductor companies and major universities around the world have been interested in the design of bus architectures for SoC, expanding the variety of protocols and architectures available for on-chip communication. In the next subsections, some of the most popular and important bus architectures are described and discussed.

A. Wishbone

The Wishbone architecture offers a flexible integration solution that can be easily tailored to a specific application. This was developed with a common interface to facilitate structured design methodologies on large project teams and support variable core interconnection methods. Due to these characteristics, wishbone bus architecture is easy to implement in simple embedded controllers and high-performance systems.

The most outstanding features that have Wishbone, are modular data bus widths, the support of both big endian and little endian data ordering, variable core interconnection methods, and the support of single clock data transfers [1].

B. On-Chip Peripheral Bus (OPB)

The on-chip peripheral bus (OPB) developed by IBM was designed to connect on-chip peripheral devices and provide a common design point for various on-chip peripherals. The OPB operates independently at a separate level of the bus hierarchy. Lower performance peripherals (such as timers, counters, slaves, and other internal peripherals) are attached to the on-chip peripheral bus (OPB). A bridge is provided between the processor local bus (PLB) and OPB to enable data transfer by PLB masters to and from OPB slaves. OPB also provides support to different types of peripherals, which can by 8-bit, 16-bit, 32-bit, and 64-bit. The transfers between slaves and master take at less two clock cycle. Furthermore, the OPB supports multiple OPB bus masters and sequential address protocols [3].

C. STBus Type 1

The STBus interface family contains a number of interface variants with different performances and complexity costs. An STBus has the possibility to choice which type of STBus interface to use in a specific design. For this purpose, the designer should consider the module and system costs, selecting the simplest interface with the appropriate features and data width for that system.

The Type 1 interface is part of the STBus family which is a simple handshake interface supporting a limited set of operations. These operations are mapped into a packet containing one or more cells at the interface. Type 1 interface typically includes standalone modules such as a general purpose input/outputs (GPIOs) and an universal asynchronous receiver/transmitter (UARTS). Furthermore,

modules which require independent control interfaces in addition to their main memory interface, or internal interfaces which do not require pipelines are also included [4].

D. MBUS

MBus is an ultra-low power system bus. The original design was developed by the Michigan university. However, the goal of MBus is to be a general purpose bus for hyper-constrained systems. MBus supports arbitrary length transfers and features a low-latency priority channel and robust acknowledgments.

Inside Mbus specifications are included the bus configuration, power, address, and protocol design. The purpose of these specifications is to have control of the most important characteristics in the communication and obtain the lowest possible consumption [2].

E. APB

The Advanced Peripheral Bus (APB) is part of the advanced microcontroller bus architecture (AMBA), being a low-cost interface that is optimized for minimal power consumption and reduced interface complexity. The APB protocol is not pipeline and connects low-bandwidth peripherals that do not require the high performance of high-speed buses. Every transfer takes at least two clocks cycles [5].

2. APB ADVANTAGES

Table I compares Wishbone, DCR, STBus Type 1, Mbus, and APB buses. This table shows some of the main aspects to consider when a peripheral bus is selected.

Table 1. Comparative table of peripheral buses

	Nº of signals	Min Nº of clock cycles perread/write transactions	Pipeline	Plus signal
Wishbon	11	2/2	Configurable	Err_I[1]
OPB	22	3/3	No documented	BE
STBus	10	2/2	No	BE[2]
MBus	15	No document	No documented	No documented[3]
APB	11	2/2	No	STRB[4]

1You can configure pipeline or not pipeline mode.

2Indicates an abnormal cycle termination. The source of the error, and the response generated by the MASTER is defined by the IP core supplier.

3The byte-enable signal defines which bytes within a cell are significant.

4This signal indicates which byte lanes to update during a write transfer.

In the particular case of MBus, the bus is still under development and documentation emphasizes the internal composition of the bus and its operation, but lacks of documentation explaining the specifications of the protocol.

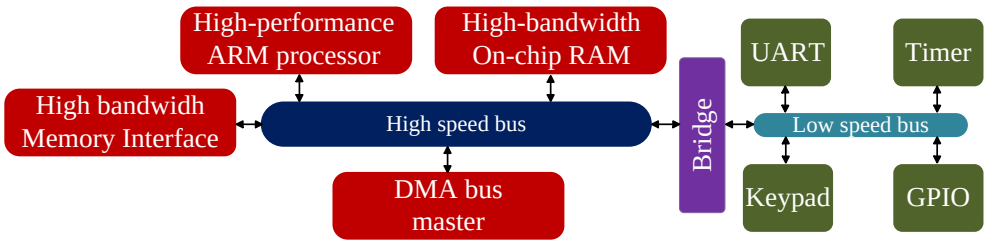
The APB protocol stands out in majority of aspects compared in table I and it is also the most used in the industry. APB protocol is used in ARM processors family since its Cortex- M series is characterized for being the smallest and lowest power cortex processors until Cortes-A family is known for having fast response optimized for high-performance and hard real-time applications. Among the most important

companies processors that use the ARM processors family are Samsung, Atmel, Qualcomm, Silicon Labs, and Texas Instruments.

Because the APB bus has outstanding characteristics of operation, high popularity, and especially the easy of adaptation to AXI4-Lite bus, it was selected for handling the low performance peripherals in the first open hardware microcontroller “A 32-bit 100MHz RISC-V Microcontroller with 10-bit SAR ADC in 130nm CMOS” [7]. The APB bus is designed to work alongside the main bus AXI4-lite, working as a master for ADC, DAC, GPIO and SPI slave peripherals. Emphasis was placed on the study of the list of signals and handshake needed for the APB bridge development [5] [8].

An AMBA-based microcontroller typically consists of a high-performance system backbone bus that is able to sustain the external memory bandwidth. The CPU and other Direct Memory Access (DMA) devices reside on backbone bus, besides a bridge to a narrower APB bus on which the lower bandwidth peripheral devices are located. Fig. 1 shows both AXI4 and APB in a typical AMBA system [8].

Figure 1. A typical AMBA AXI-based system.



In SoC, a hierarchy in handling among its modules is used. Usually, the core is assigned as controller chip (master) and the peripheral is assigned as slaves. In the typical configuration, the AMBA bus handles one or more master controlling the

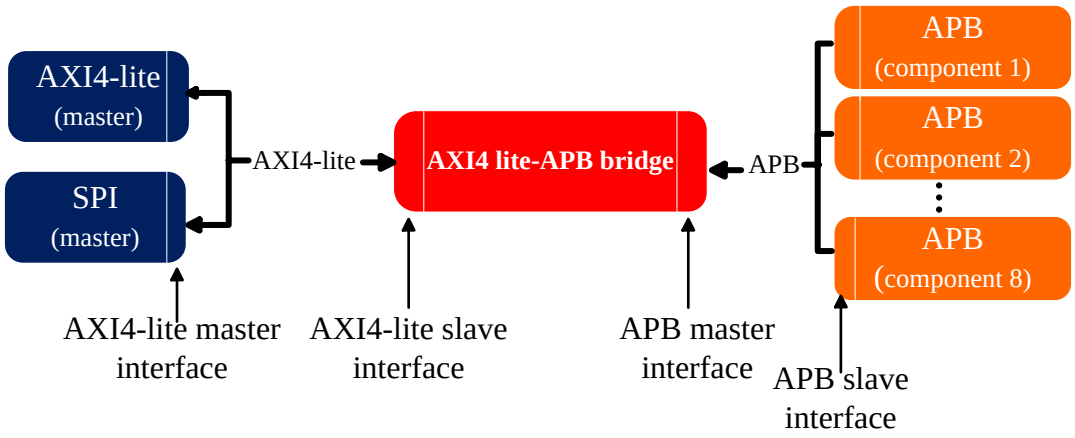
entire chip and an APB module which is seen as a slave in the AXI4 protocol. This last module acts as a master within the APB protocol, which controls the peripherals connected to it.

In Fig. 2 is shown a block diagram of an AXI4-lite to APB bridge used in a simple configuration with two master and eight APB slaves. The bridge provides an interface between the high-speed AXI4-lite domain and the low-power domain. Read and write transfers on AXI4-lite are converted into corresponding transfers on the APB and vice versa. The implementation of the bridge is explained in the following section.

3. IMPLEMENTATION

In the elaboration of the APB bridge, a complete review of the APB and AXI4-lite protocol and a handshake is needed. Taking into account all the considerations that the standard recommends for the properly operation of each transaction.

Figure 2. AXI4-lite to APB bridge block diagram.

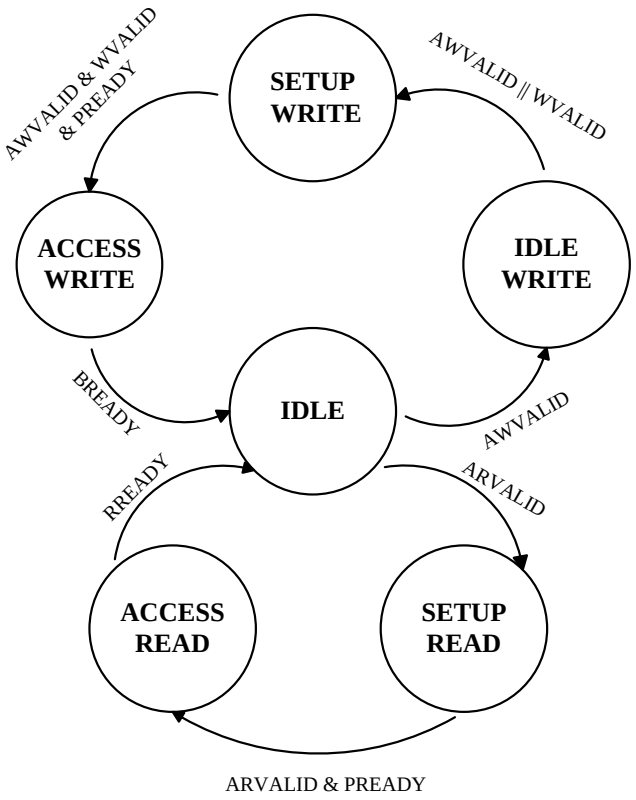


The design of the FSM, their input signals, next state logic, and outputs signals of each state is performed. Behavioral simulations are carried out to fully verify that FSM complies both protocols. Then, the data-path is designed to support all peripherals and perform functions as arbitrator in the selection of each peripheral. This selection is done with tristate buffers for controlling each signal from the peripheral to the APB bridge. These signals are enabled only if the Master initializes a transaction. Finally, the problem of metastability of the circuit is considered.

A. Finite State Machine (FSM) and Clock domain

In Fig. 3 is shown the FSM used to implement the APB bridge. The operation of the FSM satisfies the APB and AXI4-lite protocols which can be consulted in the AXI4-lite [8] and the APB manual [5].

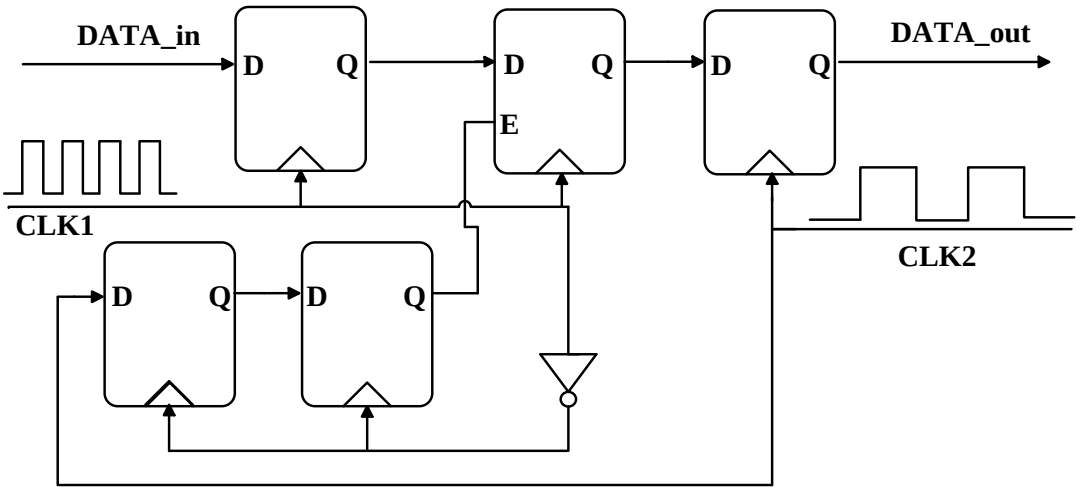
Figure 3. Transactions control FSM.



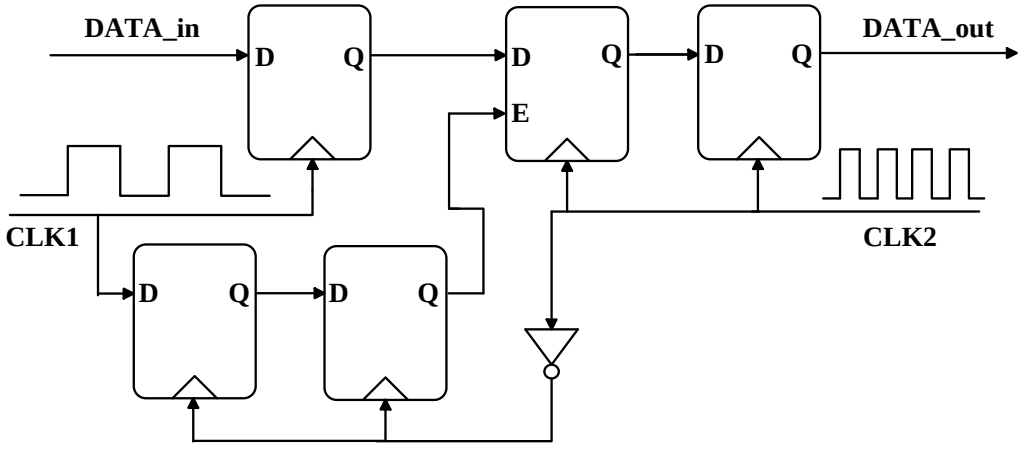
On the other hand, the chosen clock frequency in main bus AXI4-lite of the microcontroller [7]) was 100 [MHz], and 10 [MHz] for APB. Therefore, each signal to be passed from one clock domain to another is passed through a synchronizer circuit to prevent metastability problems. If the signal is a clock domain with frequency f1 to another with frequency f2, where f1 is greater than f2, the configuration shown in Fig. 4(a) is used. Otherwise, if f1 is less than f2, is used the

configuration shown in Fig. 4(b). In this way, the possibility of metastability problems is reduced. [9], [10].

Figure 4. Synchronizer bus: a) fclk1 greater than fclk2 and b) fclk1 less than fclk2.



(a)



(b)

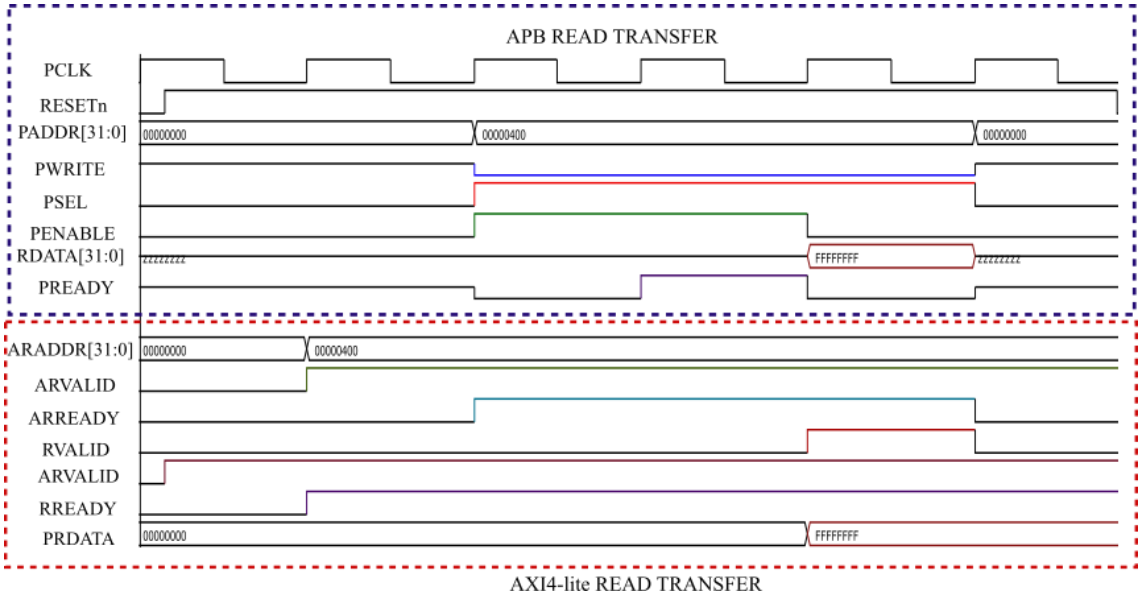
B. Simulations

Table 2. Power, timing and area of apb and axi4-lite bridges

Module	POWER [nW/MHz]	Time Slack [ps @ 100MHz]	Area _m2
Data-path	195,18	1780	1848
FSM	371,85	1521	1621
Synchronizer Bus	8429,52	1367	17049
Total	8996,56	4668	20518

The timing diagrams shown in Fig. 5 and Fig. 6 illustrate the operation of read and write transfers respectively by a peripheral. Characteristics of the APB and AXI4-lite protocols can be observed, such as the initialization of the read transaction by the ARVALID signal and AWVALID in the case of the write transaction, PWRITE signal to indicate the type of transaction that is initialized by the master and the selection of each peripheral by the address signals.

Figure 5. APB / AXI4-lite read transfer



APB bridge is fully synthesized in 130nm CMOS technology. Synthesis results are shown in the table II for FSM, data-path and synchronizer bus. Power consumption and area of the APB and AXI4-lite buses are compared. These results show a remarkable difference between the two buses as shown in table III.

Figure 6. APB / AXI4-lite write transfer

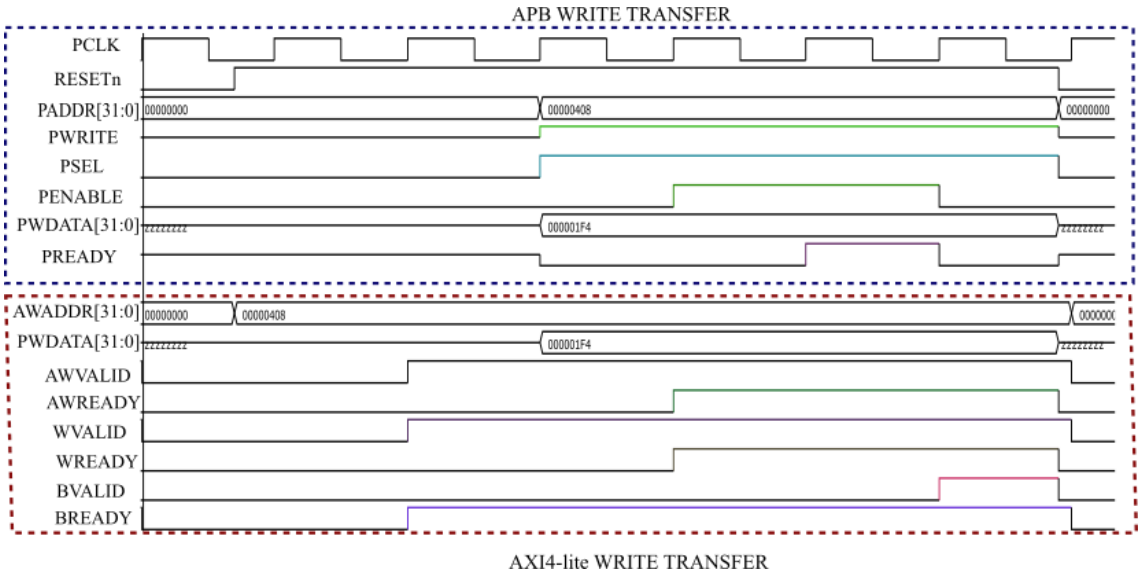
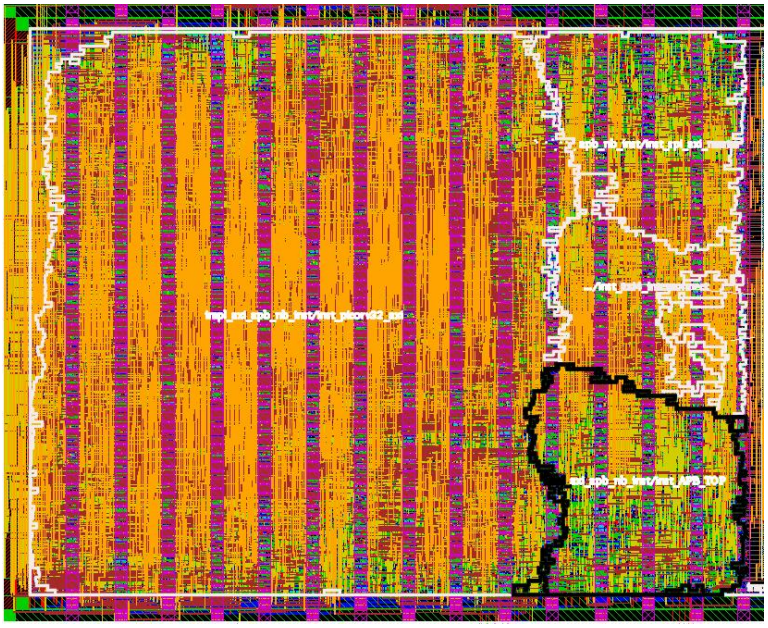


Table 3. Power, timing and area of apb bridge

Modul	POWER [nW/MHz]	Area μm^2
AXI4-lite	3958,29	11281
APB	567,04	3469
AXI4-lite/APB relation	698 %	325%

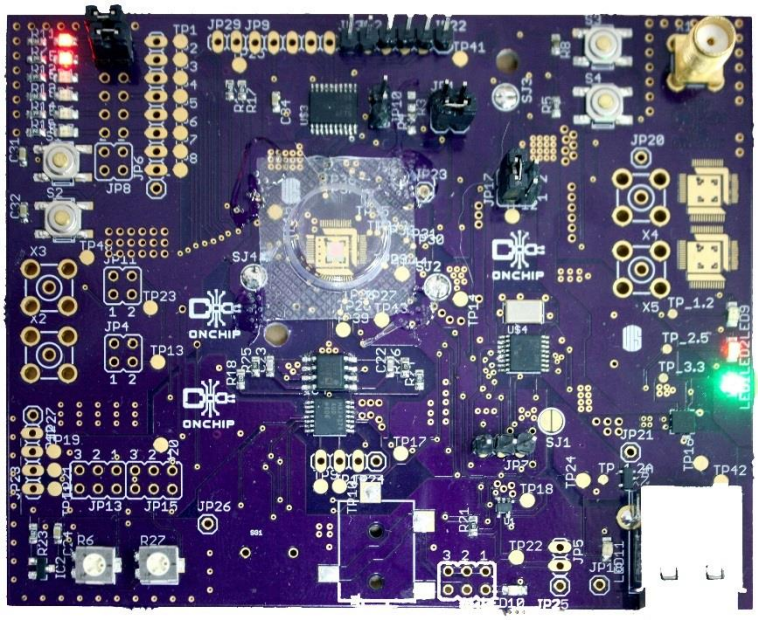
The final layout of the complete Microcontroller is presented in Fig. 7. Each instance is highlighted to exposed the area breakout. The RAM block and core occupy the most of the area in the whole chip, while the APB bridge block occupies a no significant area with a footprint close to 17% of the whole chip. The microcontroller occupies a final area of 349702 μm^2 and an energy consumption of about 161.341 µW/Hz.

Figure 7. Layout of the core and microcontroller, highlighting the APB bridges and peripheral interfaces.



The research group Onchip has designed a development board. This is composed by a 32-bit 100MHz Microcontroller with a RISC-V instruction set architecture (ISA), 10-bit SAR ADC, 12-bit DAC, RAM 4 MB, 8 GPIO pin SPI slave and master, which is shown in Fig. 8. In the next version of the development board, Onchip group plans to implement the APB bus for the management of the peripherals ADC, DAC, SPI, and GPIO.

Figure 8. Test setup board



4. SUMMARY

This paper presents an APB bridge in a 130nm CMOS technology. The APB bridge proposed has a more efficient management of peripheral to low performance, using a second protocol (APB) with a frequency ten times less than the main bus (AXI4-lite). The FSM and data-path were designed for having a low power consumption, low complexity, and reduced area. The correct operation was fully verified covering whole transactions in the DAC, ADC, and GPIO peripherals. Finally, the APB bridge was proved through an SPI interface.

The APB bridge was developed to be integrated into a platform which is currently being developed. This new platform is planned to have a RISC-V microcontroller with RAM, ROM, CNN using APB interconnector of the peripheral such as DAC, ADC, GPIO and SPI peripherals.

REFERENCES

[1] Opencores, “WISHBONE System-on-Chip (SoC)Interconnection Architecture for Portable IP Cores.” OpenCores, 2010.

[2] Pat Pannuto, Yoonmyung Lee et. al, “MBus Specification,” June 14, 2015.

[3] IBM, “On-Chip Peripheral Bus Architecture Specifications.” IBM, April 2001.

[4] STmicroelectronic, “STBus Communication System Concepts and Definitions.” STmicroelectronic, October 2012.

[5] ARM, “AMBA APB Specification (Rev 2.0).” ARM, 13 April 2010.

[6] Sudeep Pasricha, Nikil Dutt, On-Chip Communication Architectures: System on Chip Interconnect. Morgan Kaufmann Publishers, 2008.

[7] C. Duran, E. Roa et. al, “A 32-bit risc-v axi4-lite bus-based microcontroller with 10-bit sar adc,” in 2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS), Feb 2016, pp. 315–318.

[8] ARM, “AMBA AXI and ACE Protocol Specification.” ARM Limited, 2004.

[9] C. E. Cummings, “Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog.” SNUG Boston, 2008.

[10] R. Ginosar, “Fourteen ways to fool your synchronizer,” in Asynchronous Circuits and Systems, 2003. Proceedings. Ninth International Symposium on, May 2003, pp. 89–96.

BIBLIOGRAPHY

ARM, “AMBA APB Specification (Rev 2.0).” ARM, 13 April 2010.

ARM, “AMBA AXI and ACE Protocol Specification.” ARM Limited, 2004.

C. Duran, E. Roa et. al, “A 32-bit risc-v axi4-lite bus-based microcontroller with 10-bit sar adc,” in 2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS), Feb 2016, pp. 315–318.

C. E. Cummings, “Clock Domain Crossing (CDC) Design & Verification Techniques Using SystemVerilog.” SNUG Boston, 2008.

IBM, “On-Chip Peripheral Bus Architecture Specifications.” IBM, April 2001.

Opencores, “WISHBONE System-on-Chip (SoC)Interconnection Architecture for Portable IP Cores.” OpenCores, 2010.

Pat Pannuto, Yoonmyung Lee et. al, “MBus Specification,” June 14, 2015.

R. Ginosar, “Fourteen ways to fool your synchronizer,” in Asynchronous Circuits and Systems, 2003. Proceedings. Ninth International Symposium on, May 2003, pp. 89–96.

STmicroelectronic, “STBus Communication System Concepts and Definitions.” STmicroelectronic, October 2012.

Sudeep Pasricha, Nikil Dutt, On-Chip Communication Architectures: System on Chip Interconnect. Morgan Kaufmann Publishers, 2008.