

Re-Usable Low Power DSP IP embedded in an ARM based SoC Architecture

H. H. Hellmich, A. T. Erdogan, and T. Arslan*

Technical University of Berlin
Department of Electrical Engineering
Microelectronics Division
Einsteinufer 17, D-10587 Berlin, Germany.

University of Edinburgh
Department of Electronics and Electrical Engineering
System Level Integration Group
Edinburgh, EH9 3JL, Scotland, UK.

hellmich@mikro.ee.tu-berlin.de

{ahmet.erdogan, tughrul.arslan}@ee.ed.ac.uk

Abstract

The integration of different re-usable IPs (Intellectual Properties) to design SoC (System-on-Chip) devices is widely accepted as the key to achieving higher productivity to meet shorter time-to-market demands. Nevertheless, productivity improvements suffer because the importance of interface definitions and consequently the integration of IPs for the targeted SoC architecture is often treated as a secondary issue. This paper describes a scheme for the integration of a low power DSP IP embedded in an ARM based SoC architecture which is characterised by having two interfaces intended for two different types of on-chip bus configurations. A DSP IP has been implemented using this scheme. The power consumption of the actual FIR filtering algorithm realised within the DSP IP has been compared to a conventional implementation using an Alcatel 0.35 μm CMOS technology.

1. Introduction

The growing gap between the silicon gate capacity and the engineering productivity has led to the advance of complex SoC designs and the need for new forms of design reuse and design methodologies [Bric99]. In order to overcome this gap, design reuse is migrating from source reuse to integration-driven reuse and design methodologies are changing from TDD (Timing-Driven Design) to PBD (Platform-Based Design) [Chang99]. However, PBD gains its productivity by minimising the amount of custom interface design and focuses around a standardised bus architecture.

The AMBA (Advanced Microcontroller Bus Architecture), which defines two types of bus configurations [Amba99], represents an open standardised bus architecture. The system bus, called AHB (Advanced High-performance Bus), is intended for high clock frequency system modules and enables access to high bandwidth memory devices. The peripheral bus, called APB (Advanced Peripheral Bus), is optimised for minimal power consumption and suits low bandwidth peripheral modules (see Figure 1). This hierarchical bus architecture is more suitable for low power consumption and high speed performance in comparison to a single bus architecture, such as standardised on-chip busses like the PI-Bus [Pibus] or the CSI-Bus [Csibus]. The more modules are connected to one bus the slower it operates, and consequently the latency (execution time of a transaction across the bus) of the bus increases. The total bandwidth (product of clock frequency and the byte width of the bus) request of all modules is another reason to form groups of modules around separate busses [Chang99]. Modules, requiring high bandwidth and probably having an intensive interchange of data among each other, should be considered in the same group while modules with much lower communication needs should be bundled together. This provides the facility to powerdown the bus that connects the low bandwidth modules. For complex SoC designs, the increase in bus partitioning will create more than one system and peripheral bus in order to keep bus capacitances at realistic levels and to toggle these only when necessary [Aldw99].

* Also affiliated with the Institute for System Level Integration (ISLI) in Livingston, Scotland, UK.

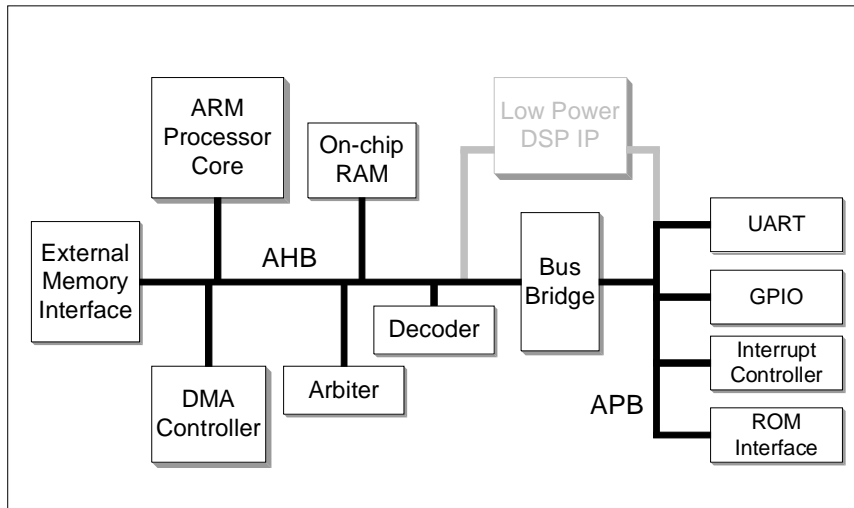


Figure 1: ARM based SoC architecture

Due to the existence of various standardised on-chip bus architectures, IP core based interface standards have been published recently. IPs providing such interfaces are considered to be suitable for connection to any type of on-chip bus architecture using a bus wrapper. The drawback of this approach is that several layers of interface between the core functionality of the IP and the bus arise which may degrade the performance of the IP, and certainly will add gates to the design [Bric99]. The VCI (Virtual Component Interface) standard [Vsia00] and the OCP (Open Core Protocol) [Ocp00] are examples of IP core based interface standards. The VCI standard is viewed to focus only on the data flow aspects of the communication between different IPs [Smith00], such as address, data and command signals. Non-dataflow signals, e.g. interrupts, or test signals are not considered by the VCI standard compared to the OCP. However, this fact could be considered by the AVCI (Advanced Virtual Component Interface), which remains the last of the three VCI standards that has to be defined by the VSIA [Vsia00].

Hence, providing IPs with an IP core based interface facilitates fast integration and results in an increase of productivity which is desperately needed for designing future SoC devices. For low power applications, it is important that such interfaces maintain low power consumption.

2. Proposed Low Power DSP IP

Our proposed low power DSP IP is presented in Figure 2 and its integration into an ARM based SoC architecture is implied in Figure 1. The IP consists in its top level of five modules: the actual low power DSP core, the register block, the VC initiator interface, the VC target interface and a gated clock circuitry. The VC *target* interface is connected via the APB *initiator* bus wrapper to the peripheral bus APB. The protocol between the APB initiator wrapper and the VC target interface is called PPCI (Peripheral Virtual Component Interface), which is one of two VCI standards defined by the VSIA up to now. According to this protocol every valid request by the APB initiator wrapper has to be acknowledged by providing a response by the VC target interface. The VC *initiator* interface connects the DSP IP via the AHB *target* bus wrapper to the system bus AHB. In this case, the protocol between bus wrapper and VC interface is the second defined VCI standard called BVCI (Basic Virtual Component Interface). The BVCI is a superset of the PPCI and provides additional optional protocol signals. The main difference compared to the PPCI is that the BVCI allows independent handshakes for requests and responses. This means that when the VC initiator interface provides a valid request, the AHB target bus wrapper will acknowledge the request. At the time the bus wrapper obtains a valid response it will indicate this to the VC initiator interface which on the other side will acknowledge the receipt of the valid response. This kind of independent handshake for requests and responses is very suitable for modules connected to an on-chip bus where bus arbitration takes place and therefore no guarantee for an immediate response is given. In addition, the DSP IP operates in two different synchronous clock domains indicated by the dashed line in Figure 2. The upper part of the DSP IP will operate at the frequency of the system bus clock (*sclk*) which will usually be the maximum frequency of the SoC design and the lower

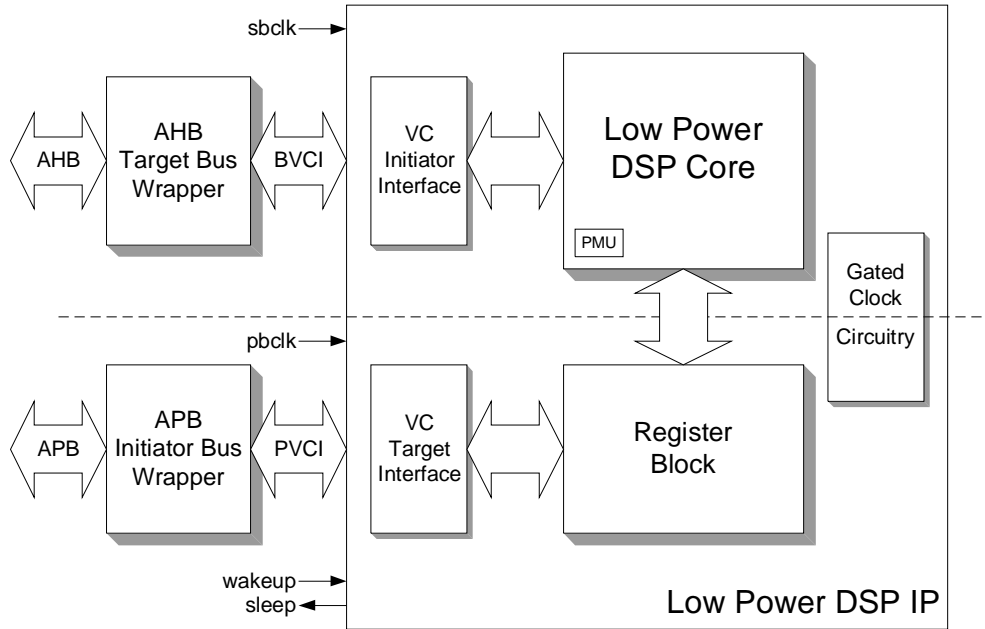


Figure 2: Block Diagram of the Low Power DSP IP

part of the DSP IP will be clocked by the peripheral bus clock ($pbclk$). It is assumed that the following condition for both clock domains is met, in order to avoid additional synchronisation circuitry between the two clock domains within the DSP IP:

$$f_{sbclk} \text{ MOD } f_{pbclk} = 0 \quad , \quad f_{pbclk} \leq 2 \cdot f_{sbclk} \quad (1)$$

In addition to the VCI protocol signals, the DSP IP provides the signals *wakeup* and *sleep* which are intended to be directly connected to the bus bridge of the APB. When the *sleep* signal is asserted the bus bridge knows that the DSP IP is in sleep mode and any access to the register block of the DSP IP could be indicated by the bus bridge with an error transfer response over the AHB (*without transferring any data over the APB*). The *wakeup* signal could be a select signal output of the bus bridge. If the DSP IP is in sleep mode, it can be waken up when a given address on the system bus is provided. This address will be decoded by the bus bridge to generate the select signal which is connected to the *wakeup* input of the DSP IP. These two additional signals communicate with the low power DSP core and allow the integration of the DSP IP into a DPM (Dynamic Power Management). DPM shuts down idle devices and wakes them up when needed [Lu00] and is therefore an efficient method for power savings especially for portable applications.

2.1 Low Power DSP Core

The DSP core is the actual high-performance module of the DSP IP and is shown in Figure 3. It consists of three control units: the PMU (Power Management Unit), the DMU (Data Management Unit) and the FIR core. The PMU controls with its single output the gated clock circuitry of the DSP IP. It will be the only active module during the sleeping state of the DSP IP in order to react to an asserted *wakeup* signal. Two input signals (*sleepmode* and *entersleepmode*) coming from the register block provide information if and when the DSP IP should enter into sleep mode when the DSP core is not active anymore. The information whether the DSP core is still active or not is provided by the DMU. The DMU takes care of reading and writing of data from an external memory device over the AHB. Read data that has to be filtered will be handed to the FIR core, and valid filtered data will be written back to memory. To provide a temporarily storage for these data transfers, a read and a write FIFO (RFIFO and WFIFO) are integrated in the DMU.

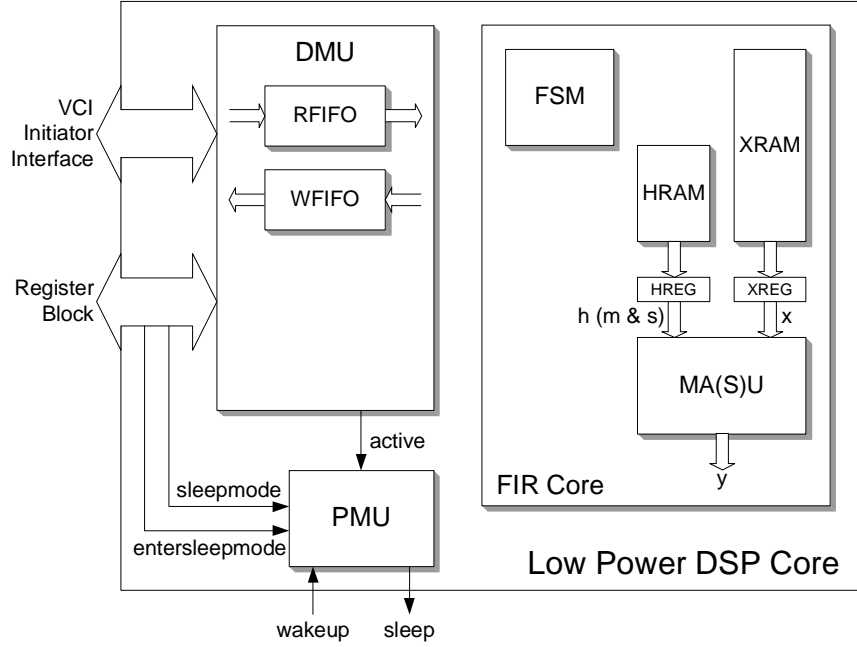


Figure 3: Block Diagram of the Low Power DSP Core

2.2 FIR Core

The FIR core within the DSP core (see Figure 3) contains two static RAM blocks to store coefficients (HRAM) and input data (XRAM), an FSM and a component that performs the actual arithmetic operations that are necessary for direct form FIR filtering (MA(S)U). For the implementation of the conventional FIR core this component is called MAU (Multiply-Add-Unit) which performs a multiplication of a coefficient h and an input data x and adds the product with the previous value stored in the register output y . The MAU is therefore identical to a conventional MAC (Multiplier Accumulator). For the low power implementation of the FIR core a FIR filtering algorithm based on coefficient segmentation [Erdo99] is used. According to this algorithm the coefficient h is segmented into two numbers m and s . For the utilised two's complement representation of data the following condition is valid:

$$h_k = s_k + m_k, \quad m_k \geq 0 \quad (2)$$

The input data x and the number m are applied to the multiplier while a shift operation of x will be performed according to the shift value of s . The sum of the multiplication and the shift operation is added to the previous value stored in the register output y . This process of dividing each coefficient into two parts can reduce power in the multiplier section by more than 50 % for 16-bit wide input data [Erdo99]. Since a multiplication and a shift operation is performed for this low power FIR filtering algorithm, the component performing the arithmetic operations is called MASU (Multiply-Add-Shift-Unit).

The size of XRAM is assumed to be twice as large as the size of HRAM so that the FIR core is able to calculate one valid output data without the need of requesting new input data during a folded FIR filtering. The size of the static RAM blocks depend on the maximum number of coefficients that have to be stored in the HRAM. In case the size of the RAM blocks exceed 256 bits (data width \times address depth) a RAM macro block of the targeted CMOS technology should be used to ensure a power efficient implementation of the FIR core.

3. Power Based Simulation Results

The comparison of power consumption between a conventional FIR core implementation and an implementation using coefficient segmentation has been performed. For logic synthesis and the estimation of power consumption the Synopsys suite of synthesis tools have been used. Both FIR cores are described

in VHDL and characteristics like data width and the maximum number of coefficients are parameterisable. The MAU and MASU are described in MCL (Module Compiler Language), which syntax is similar to Verilog, in order to be synthesized by the Synopsys' data path synthesis tool Module Compiler. The utilised ASIC library was an Alcatel 0.35 μm CMOS technology.

For the power based gate-level simulation two different low pass filters were realised by the FIR cores and which filter characteristics are illustrated in Table 1.

Filter Type	Passband (kHz)	Stopband (kHz)	Passband ripple (dB)	Stopband attenuation (dB)	Filter Length
LPF1 / LPF2	0 - 3.375 / 0 - 1	5.625 - 10 / 1.5 - 5	0.002 / 0.0135	90 / 56	42 / 61

Table 1: Low Pass Filter Characteristics

For each simulation an unfolded filter structure and 512 randomly generated 16-bit-wide input data were used. During the simulation it was assumed that at every stage the FIR core requested new input data, they could be directly provided by the DMU. A reduction of the cell internal power (*without considering the RAM blocks*) for the FIR core utilising a MASU was observed. For the first low pass filter (LPF1) a power reduction of 6.3 % and for the second low pass filter (LPF2) a power reduction of 7.2 % for the FIR utilising a MASU was obtained compared to the FIR core implemented with a MAU.

4. Conclusion

In this paper the implementation scheme for a low power DSP IP has been presented. The DSP IP is characterised by having two interfaces intended for two different types of on-chip bus configurations. The initiator interface is intended for a high-performance, high-bandwidth bus while the target interface is intended for a low-performance, low-bandwidth bus. The power reduction of 6.3 % and 7.2 % of the FIR core using coefficient segmentation for the realisation of two different low pass filters encourages the further investigation and optimisation of an FIR core utilising this kind of FIR filtering algorithm.

5. References

- [Aldw99] P. J. Aldworth, "System-on-a-Chip Bus Architecture for Embedded Applications", *IEEE International Conference on Computer Design*, Oct. 10-13, 1999, Austin, Texas, USA.
- [Amba99] AMBA™ Specification, Revision 2.0, May 1999, © ARM Ltd., www.arm.com/Pro+Peripherals/AMBA.
- [Lu00] Y.-H. Lu, E.-Y. Chung, T. Simunic, L. Benini and G. De Micheli, "Quantitative Comparison of Power Management Algorithms", *Design, Automation and Test in Europe (DATE)*, March 27-30, 2000, Paris, France.
- [Bric99] M. Keating and P. Bricaud, "Reuse Methodology Manual", Kluwer Academic Publishers, 2nd Edition, 1999.
- [Chang99] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNeilly and L. Todd, "Surviving the SOC Revolution – A Guide to Platform-Based Design", Kluwer Academic Publishers, 1999.
- [Csibus] CSI (Configurable System Interconnect) Bus, © Triscend Corp., www.triscend.com/products/IndexTechLit.html.
- [Erdo99] A. T. Erdogan and T. Arslan, "A Coefficient Segmentation Algorithm for Low Power Implementation of FIR Filters", *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 30 – June 2, 1999, Orlando, Florida, USA.
- [Ocp00] Open Core Protocol™ Specification 1.0, Document Version 1.1, Jan. 2000, © Sonics Inc., www.sonicsinc.com/mnet/ocp_descr.html.
- [Pibus] PI (Peripheral Interconnect) Bus, © Open Microprocessor Initiative (OMI), www.sussex.ac.uk/engg/research/vlsi/projects/pibus.
- [Smith00] E. Smith, "Bus protocols limit design reuse of IP", *EEdesign article*, May 2000, www.eedesign.com/story/OEG20000515S0026.
- [Vsia00] VSI Alliance™ Virtual Component Interface Standard (OCB 2 1.0), On-Chip Bus Development Working Group, March 2000, © Virtual Socket Interface Alliance, www.vsi.org/library/specs/summary.htm.