Published in IET Computers & Digital Techniques
Received on 20th September 2012
Revised on 1st February 2013
Accepted on 25th March 2013
doi: 10.1049/iet-cdt.2012.0107

ISSN 1751-8601

Power domain management interface: flexible protocol interface for transaction-level power domain management

Ons Mbarek, Alain Pegatoquet, Michel Auguin

LEAT, University of Nice-Sophia Antipolis, CNRS, BP 145 - 930 Route des Colles, Sophia Antipolis Cedex 06903, France
E-mail: mbarek@unice.fr

Abstract: Defining a power management solution for a system-on-chip imposes the design of a low power architecture composed of multiple power domains and a power management strategy for power domain state control. If these two elements are energy-efficient, an energy-efficient power management solution can be obtained. Transaction-level of modelling allows a rapid exploration of different power management solutions, hence a fast decision-making of the most energy-efficient one. The authors' previous work proposed an abstraction of the unified power format (UPF) standard semantics to enable fast evaluation of different UPF-like power architectures at transaction-level (TL). However, a fast evaluation of different power domain management strategies requires a unique and flexible hardware interface to organise transfers of inter-power-domain transactions according to a well-defined protocol. The proposal of a new hardware protocol interface for power domain state management, called power domain management interface (PDMgIF), and its TL simulation model represents the main contribution of this study. This proposal separates functional and power management communications. It represents a potential extension of existing low power standards (such as UPF) that already miss power domain management semantics. The PDMgIF concepts are demonstrated through an audio application TL platform illustrating a high flexibility and reduced overhead of the TL PDMgIF model.

1 Introduction

Power gating and adaptive voltage scaling are widely used low power methods to minimise power dissipation in today's systems-on-chip (SoCs). These methods rely on partitioning the chip into individually controlled power domains. Each power domain represents a group of blocks that share the same primary supply nets. Therefore appropriately defining the SoC low power architecture (PwARCH) including supply nets, power switches and other power elements (e.g. retention registers and isolation cells) has a direct impact on the possible use of these methods [1]. The recent low power standards, either the common power format (CPF) [2] standard or the IEEE 1801 [3] standard that is commonly known as the unified power format (UPF), define language formats as well as simulation semantics for the specification and simulation of an SoC PwARCH starting from the register transfer level (RTL) design stage.

The control of power domain states is ensured by a power management unit (PMU) hardware block, according to a specific power domain management strategy. In order to change the local state of a power domain, this unit controls in a specific order each power management element involved in this state definition. In order to control the overall system power state, the PMU implements either a static or a dynamic power management strategy to accordingly set states of local power domains. Dynamic power management strategies such as scenario tracking or prediction require enough information about the functional and power state of each component in the design at each point in time. Components have, hence, to communicate this information to the PMU.

These kinds of power domain management depict a strong design dependency between a power-domain-based architecture and the PMU operation. In order to remove this dependency, a generic and common power domain management interface is required. Such an interface has to describe the protocol and data required for inter-power-domain communication while supporting a plug-and-play approach for power domains and PMU. Unfortunately, low power standards such as UPF and CPF only define semantics for a power-domain-based architecture. Defining the PMU block structure and the power management strategy to control such architecture is left to the designer.

In this paper, we propose a new power domain management (PDMgIF) interface dedicated for the control of power domain states. This interface allows the transfer of controls and events between power domains using well-defined concepts and according to specific protocol rules. Advantages of a common interface for inter-power-domain communications transfer while promoting a plug-and-play approach as well as the lack of UPF and CPF semantics for such an interface definition have represented the primary motivations of this current work.

A second motivation of our work comes from the fact that plug-and-play approaches are very useful for fast exploratory studies performed during early stages of an SoC design flow, in particular at the transaction-level of modelling (TLM) [4]. Thus, a common PDMgIF modelled at transaction-level (TL) would enable exploration of different PwARCHs and related power management strategies in order to decide early about the most energy-efficient couple.

Our previous work [5] proposed an approach for building a structured high-level specification of multi-power domain architecture at TLM based on the abstraction of the UPF semantics [3]. Then, we have shown how to integrate such a specification into a TL behavioural SoC model and evaluate different PwARCH alternatives.

However, a rapid exploration of different power domain management strategies requires defining a common and generic TL interface for inter-power-domain communication. The lack of TLM semantics for creating such a communication interface represents a crucial constraint. Nevertheless, by using the TLM-2.0 OSCI standard mechanisms to create protocol-specific TLM-2.0 interfaces [6–8], we provide, in this paper, a TLM 2.0 model of our proposed PDMgIF bus protocol interface that separates functional and power management communications. We show how such an interface model can be efficiently added to a functional TL model and used to construct a complete power-domain-managed TL model. Then, we demonstrate the PDMgIF flexibility and reuse with any power-domain-based PwARCH and management strategy.

The sequel of this paper is organised as follows. In Section 2, we present some related works and we list the contributions of this work. In Section 3, we present our power-domain-based modelling approach. Section 4 presents the PDMgIF protocol features definition and mapping to TLM2.0. In Section 5, we evaluate our PDMgIF protocol interface performance on an audio application TL virtual prototype. Section 6 closes with conclusions and discussions.

2 Related work

2.1 *TL power-domain-based methodology*

In our previous work [5], we proposed a TL power-aware methodology [5] aiming at TL exploration and evaluation of different PwARCH alternatives for a TL functional model. In order to facilitate performing this methodology, we have designed a PwARCH C++ library that includes the different UPF concepts abstracted at TL and their UPF-like power-aware behaviour. This library is used to specify a low PwARCH alternative. Then, a PMU is modeled as an additional SystemC/TLM module. This module controls the specified PwARCH by changing appropriate power elements states. To evaluate the low PwARCH efficiency, energy consumption by power domain is updated when a power element (typically a power switch or a supply net) is in a changed state during simulation. Throughout PwARCH alternatives specification and evaluation phases, a set of properties ensuring the coherence between power and functional features are verified.

Contrary to state-of-the-art power modelling approaches at TLM where a power model per system device is proposed to estimate the system power consumption [9, 10], our methodology assumes that devices in a same power domain share the same power state of their power domain and that the PMU model is a block dedicated for power domain management strategy through handling of power domain local states and interactions between them. In this current work, the same assumptions on the low PwARCH and PMU features are kept and energy efficiency of different power domain management strategies ranging from static to dynamic are additionally evaluated.

2.2 *State-of-the-art power management interfaces*

Power management can be power controller (PC) directed or operating system (OS) directed. On the one side, there are recent protocol interfaces that standardise PC-directed power management communications. In this type of power management, the control of the devices power states is assigned to a specialised hardware unit called PC that manages hardware control signals added for power management purposes. Examples of such power management protocol interfaces are the power management bus (PMBus) open-standard protocol [11] and the system power management interface (SPMI) bus [12] specified by the mobile industry processor interface (MIPI) Alliance System Power Management Working Group. Each defines an enhanced I^2C serial interface. The PMBus focuses on the transport and physical layer as well as on the command language to communicate with power converters. The SPMI bus defines the command set and the protocol for power management and control traffic between PCs of SoC processors and peripheral devices.

Although these both protocol interfaces specifications address the PC-directed power management similarly to our goals, these two busses enable only the control of system devices power states. They offer some semantics that can be adopted in a power domain management context, but new semantics are still required. In this work, our proposed new power domain management hardware interface has been partly inspired by the SPMI bus as explained later.

To the best of our knowledge, Sheets [13] presents the only state-of-the-art PC-directed interface in the form of a session-based domain power interface (DPI) defining the protocol and signals involved in power management communication between power domains and their power manager (PM). However, this interface has only targeted the special case of a wireless sensor network node protocol processor. It remains so close to the power-managed system architecture proposed by Sheets *et al.* [14] and totally independent of existing low power standards semantics for PwARCH specification. Therefore this proposal must be rethought for the power-domain-managed system-on-chip case while using semantics of existing low power standards for PwARCH specification.

On the other side, in an OS-directed power management, the OS implements a global power management strategy to control the devices power states independently of each other. For that, an abstract power-management interface between the OS and the hardware platform is used. This interface defines the global system and devices power states as well as the hardware registers for power management control. It requires that devices expose specific power management capabilities to supply the OS with their activity information through a specific power management hardware interface.

Among components that have been provided with such hardware interfaces are the peripheral component interconnect (PCI) [15] and the peripheral component interconnect express (PCIe) [16] busses. Advanced configuration and power interface (ACPI) is an example of abstract interfaces that enable OS-directed power management [17]. It allows PCI devices control at the OS-level through mapping the PCI device states and registers into those of the ACPI interface.

Note that such software-directed power management interfaces are not well adapted for power domain control. Indeed, they handle the global system state by controlling the system devices independently of each other, while we look for a PMU that handles even the local states of power domains and interactions between them. Moreover, the added power management aspects in these interfaces are only appropriate for an OS-level power state control using specific software-configurable registers. By contrast, power domain state control requires a PMU that changes control signals of this power domain's elements. For instance, this is the case of the Texas Instruments's OMAP3 platform PMU named power reset and clock manager (PRCM) [18]. Nevertheless, some concepts useful for power domain control can be inspired by these OS-directed interfaces concepts such as the power management event (PME) concept in the PCI [15] and PCIe [16] bus specifications as explained later.

2.3 Contributions

The work shown in this paper contributes to:

- Define a new power domain management on-chip protocol for communication between power domains and the PMU using a structured modelling approach;
- Use the TLM2.0 standard extension capabilities to create a protocol-specific simulation model for an SoC power domains management; and
- Demonstrate the reduced simulation overhead and the flexibility of the proposed protocol model by varying power management strategies and PwARCH s.

3 Power-domain-based modelling approach

In this section, our proposed power-managed system structure is presented and the main design requirements to be fulfilled by the PDMgIF protocol interface are extracted.

3.1 Power domains layers

Considering a functional TL system model, we aim at constructing a power-managed system enabling power domain state control. Functional modules belonging to the same power domain share the state and the power control interface of their power domain. Therefore we propose to layer a power domain management structure on top of the functional one. The generic view and components structure of this additional layer is depicted by Fig. 1 and is explained in the following.

Each power domain part wraps the functional modules belonging to a same power domain. This part involves as well the PwARCH specification and the different mechanisms required for the power-aware behaviour simulation of the underlying power domain. Specific PDMgIFs are required at the boundary of each power domain in order to ensure inter-power-domain communications through a dedicated PDMgIF interconnect. As shown in Fig. 1, 'PDMgIF target' and 'initiator modules' are also required in each power domain layer in order to manage state transitions and ordering of each received transaction at the PDMgIF interface.

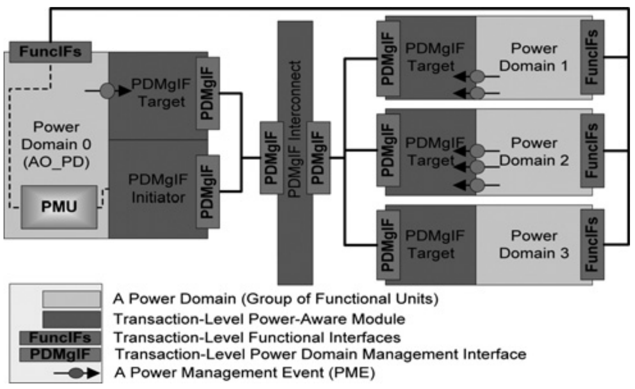


Fig. 1 *Layering the power domain management TL structure on top of the functional TL model*

3.2 Sourced power-aware communications

Modelling a generic PDMgIF requires considering bidirectional communications useful for a power domain management decision. These communications may occur between power domains and the PMU on the the one hand and between the different power domains on the other hand. Thus, two types of transactions are considered in our modelling approach. First, an always-on (i.e. can never be switched-off) power domain (AO_PD) can transfer power control transactions through the PDMgIF interconnect to other power domains or to specific design elements in order to change their power states. A design element represents at least one functional module. Power control transactions are only issued by the AO_PD and may be pipelined depending on the considered power management strategy. The concept of power control transactions have been actually adopted from the control commands that can be transmitted over the MIPI's SPMI [12] power management bus and adapted to the power domain context needs.

The second type consists in PMEs. They are defined as transactions transferred over the PDMgIF interconnect from a power domain to the PMU module as Fig. 1 illustrates. A PME transaction is used either to inform the PMU about a design element functional state, or to request a specific power domain state. Thus, these kinds of transactions are used to handle dependencies between the functional design and the power-aware one.

The concept of a PME that simply informs the PMU about a device state has already been used in the PCIe and PCI bus specifications [15, 16] as well as in the DPI interface [13]. According to our power domain management modelling requirements, we have adapted this concept and added semantics to it. In particular, we have assigned to a PME transaction a high or low priority. Moreover, such a transaction can carry out an event among these three types: a 'power PME' indicates a request to change an active power domain state to another active one. A 'sleep PME' represents a request to switch-off a power domain. It generally corresponds to a module task completion. Finally, a 'wakeup PME' represents a request to switch-on a power domain.

3.3 Identifier-based addressing and PDMgIF compliant components classification

In order to address power domains on the PDMgIF interconnect, identifier numbers are used to identify power domains and design elements. Actually, this addressing method is similar to that of the MIPI's SPMI standard [12], but adapted to a power domain context use. An 'Initiator Identifier (IID)' is given to the PMU unit. A 'Target Identifier (TID)' is given to each design element (i.e. set of functional modules) in a power domain. Each power domain is given a unique 'Power Domain Identifier (PDID)'. As a consequence, a design element of a power domain is identified by a (TID, PDID) pair. Two design elements of different power domains may have the same TID identifier.

The PDMgIF interface supports all power domains as PDMgIF targets, and only the AO_PD power domain as a PDMgIF initiator. PDMgIF targets that can arbitrate for the PDMgIF interconnect to initiate PME transactions are called Request Capable Targets (RCT). Remaining targets are called non-request capable targets (NRCT). Actually, the RCT and NRCT concepts have been inspired by, respectively, the request capable slave (RCS) and the non-request capable slave (NRCS) concepts of the MIPI's SPMI standard [12].

3.4 PDMgIF initiator requirements

Fig. 2 details the structure of the AO_PD (power domain 0) layer of Fig. 1. This power domain corresponds to the always-on SoC power domain and represents the PDMgIF initiator. Our PMU simulation model includes a PM sub-module that coordinates functional blocks activities with their power domain states according to a power management strategy. The PMU module includes as well a domain power controller (DPC) related to each power-gated domain and is responsible for its power-down and power-up sequencing control. Fig. 2 depicts an example of these sequences. For instance, to switch-off a power domain with retention in Fig. 2, control signals are asserted in this order: the CLK_STOP signal to stop the clock, the N_ISOLATE signal to isolate power domain outputs, SAVE signal to save states of retention registers, the N_RESET signal to reset states of non-retained registers and finally the N_PW_REQ signal to power-down the power domain.

At TLM, such a sequence of RTL control signals has to be converted to a single TLM function call and RTL signals will be replaced with a single specialised power socket (tlm_pw_initiator_socket) as Fig. 2 depicts. The TL PMU model will then act as a generator of power control transactions and transmit only abstract data structures. Transactions transmitted through the 'tlm_pw_initiator_socket' PDMgIF port are first received by a generic 'PDMgIF initiator' module (i.e. the PDMgIF initiator module in Fig. 2) that handles their transitions from one phase to another.

3.5 PDMgIF target requirements

Each AO_PD power domain represents a PDMgIF initiator and a PDMgIF target as Fig. 2 shows. In general, each PDMgIF target wraps a set of functional modules. Power states of these modules' power domain are controlled through power control transactions transmitted by the PMU module over the PDMgIF interconnect. Phase transitions of the received power control transactions at the PDMgIF target interface are handled by a 'PDMgIF target' generic module (i.e. the PDMgIF target module in Fig. 2). Once a power domain changes state, the 'PDMgIF target' module triggers the 'Partial Retention Handling' block shown in

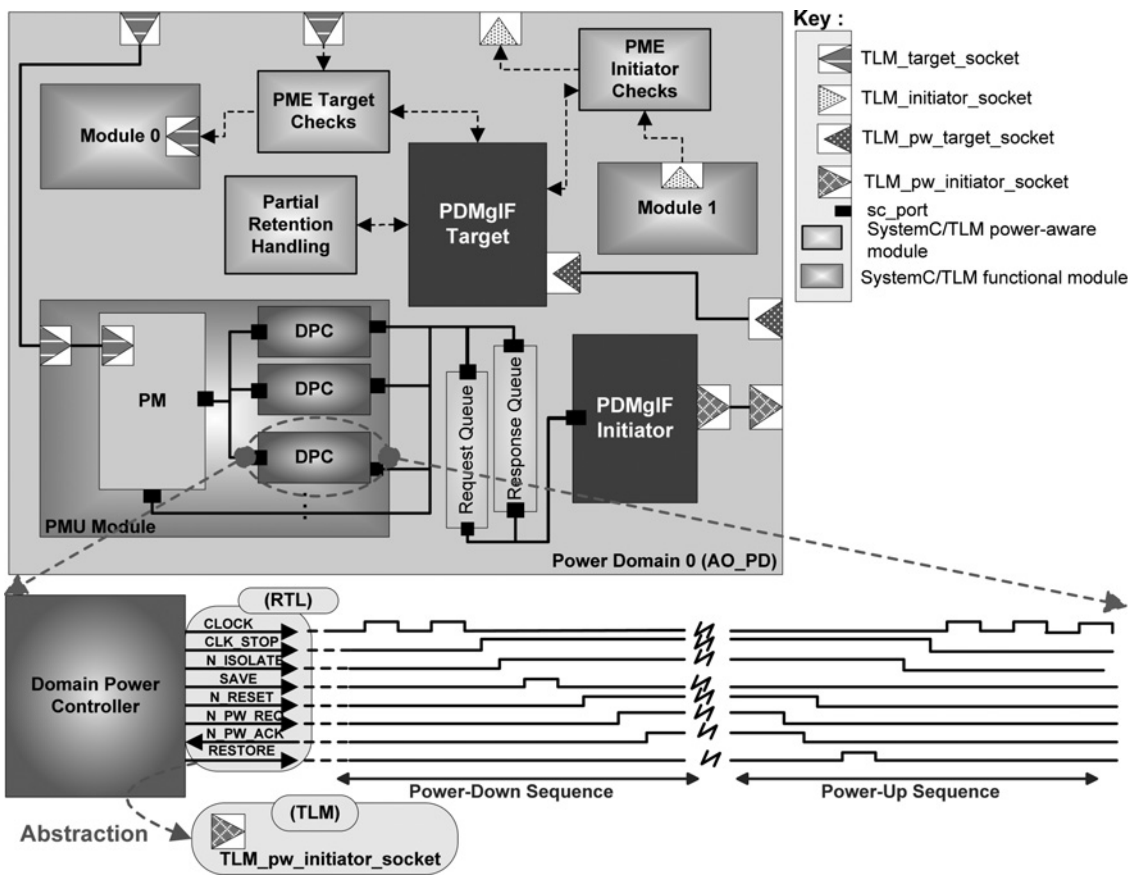


Fig. 2 *Generic example showing the internal structure of the AO_PD power domain*

Fig. 2. This block is in charge of simulating the impact of a power state change on the functional behaviour of a power domain. In particular, when a partial retention strategy is applied to a power domain, this block resets the non-retained registers of this power domain's functional modules on power-down.

In case of an RCT, the 'PDMgIF target' module is also in charge of collecting PMEs from functional modules and transmitting them to the PDMgIF initiator in the form of PME transactions. In general, a PME precedes or succeeds a transaction issued or received at the functional interface module. Therefore intercepting such relevant functional transactions and translating them to PME transactions are required in the power domain layer. This is the responsibility of the 'PME target checks' and 'PME initiator checks' blocks in Fig. 2 at, respectively, the functional target and initiator interfaces.

In the next section, we present our methodology for building a TL model of the PDMgIF protocol interface, which supports all these modelling requirements.

4 PDMgIF protocol features definition and mapping to TLM 2.0

4.1 Issues of modelling the PDMgIF protocol in TLM 2.0

Two issues are encountered when modelling the PDMgIF protocol in TLM 2.0 [6]. The first is that the TLM 2.0 generic payload fields are inappropriate to model the data and controls transmitted over the PDMgIF interconnect. Nevertheless, TLM 2.0 has provided the TLM 2.0 extension mechanism to extend the generic payload with additional user-defined fields. Hence, we have chosen to model the data attributes involved in a power control transaction as a tlm_pwctrl extension and to model those involved in PME as another tlm_pme_handling extension. Although one could put all of the attributes of the two transaction types into a single TLM 2.0 extension, it is rather wise to use two separate generic payload extensions. Indeed, this enables PDMgIF bus pipelined capabilities and extensions can then be processed and routed separately.

The second issue is the modelling of the RCT concept in TLM 2.0. Indeed, modelling a target that initiates transactions would violate the request/response ordering rules of the TLM 2.0 basic protocol. In order to overcome this modelling constraint, a new protocol different from the TLM 2.0 base protocol has been defined. Fortunately allowed by the TLM 2.0 standard, the own request/response rules of this new protocol are also defined independently of the TLM 2.0 basic protocol rules. This new protocol is characterised by a generic payload extended with the two TLM 2.0 extensions (tlm_pwctrl extention and tlm_pme_handling extensions) and a new phase object (named tlm_PDMgIF_phase) gathering all the possible timing points of the two transaction types. The specialised TLM target socket at a PDMgIF target domain interface (named tlm_pw_target_socket in Fig. 2) as well as the specialised TLM initiator socket at a PDMgIF initiator domain interface (named tlm_pw_initiator_socket in Fig. 2) have been customised to this new protocol.

Let us consider a low PwARCH of a SoC platform with' 'n' power domains and a maximum of 'p' design elements in each power domain such as 'n' and 'p' are two generic parameters (positive integers). In this case, the PDMgIF protocol allows defining up to 'n' power domains in an SoC platform and up to 'p' design elements per domain. So, each power domain must be assigned a unique 'n'-bit PDID identifier and each design element in a power domain is assigned a p-bit identifier. n and p parameters choices depend, respectively, on the power domains number in an SoC low PwARCH and on the maximum number of design elements included in this SoC's power domains. In a TL simulation, it is rather recommended to put these parameters generic so as the PDMgIF TL model can be easily reused and adapted to any low PwARCH specification and any TL SoC model.

In the following sections, we set by default these parameters to 7-bit for a PDID and 32-bit for a design element identifier. Table 1 shows the main features of our proposed PDMgIF TLM 2.0 model. They are detailed in the following sections.

4.2 PDMgIF channels and FSMs definition

The 'tlm_pwctrl channel'(A channel is defined as a group of attributes and timing points. Its definition helps designing the FSM that captures the PDMgIF protocol behaviour.) handles power control transactions initiated by the AO_PD (i.e. the PMU's power domain). Each of these transactions carries either an RESET command to initialise a power domain state, or a SHUTDOWN command to switch-off a power domain without applying retention or a SLEEP_RETAIN command to switch-off a power domain while saving its retention registers and resetting the remaining ones. A WKUP command is used to switch to an active state.

When applying a multi-voltage scaling technique to a power domain, different active power modes are considered. Each corresponds to a voltage value. In this case, the PW_MODE attribute must be appropriately set. The TYPE attribute is set depending on the power control transaction destination. If the transaction intends to control the whole state of a power domain, this attribute is set to FULL. Otherwise, it is set to PARTIAL and the transaction serves to control only the power state of some design elements in a power domain. Such design elements are recognised through the 32-bit TID_MASK attribute of the transaction payload.

The tlm_pwctrl channel attributes are mapped into a 'tlm_pwctrl extension' of the TLM 2.0 generic payload. As Table 1 illustrates, a power control transaction can be split into four timing points that identify the beginning and the end of a power control request and response. Each timing point is mapped into a phase in the custom enumeration phase class, called 'tlm_PDMgIF_phase'.

In order to allow pipelined transactions on the tlm_pwctrl channel, power control transactions are modeled using the non-blocking TLM 2.0 transport interface. Fig. 3*a* depicts the permitted sequence of interactions between an initiator and a target on the TLM 2.0 forward and backward paths [6] during a power control transaction course.

On the other side, the 'tlm_pme_handling channel' transfers PMEs. Attributes and timing points of this channel are listed in Table 1 and are explained in the following. Each PME transaction includes a specific command indicating the type of the PME event. Only RCTs can issue this kind of transactions. Depending on the PME transaction goal, the TYPE attribute is set: PW_STATUS indicates that a transaction simply informs the PMU about a power domain functional status. However, PW_MODE indicates a request to set a specific power state. By setting the PRIORITY attribute, each PME transaction is assigned a high or a low priority value. This field is required for the target arbitration

Table 1 Attributes and timing points of each channel

			Description
tlm_pwctrl Channel	Attributes	CMD	command field enumeration: RESET, SHUTDOWN, WKUP and SLEEP_RETAIN
		RESPONSE	response status enumeration indicating the success or failure of the transaction
		PDID	7-bit power domain identifier
		TYPE	type of the power control transaction (FULL or PARTIAL)
		TID_MASK	32-bit mask to indicate a specific set of design elements in a power domain (only valid if TYPE is PARTIAL)
		PW_MODE	used with the WKUP command and only in case of multi-voltage scaling technique to indicate the required wakeup voltage for a power domain
		TRANS_ID	transaction identifier (needed for handling the transactions state transitions during simulation)
	Timing Points	BEGIN_PW_REQ	initiator has set the power control transaction attributes and made the request
		END_PW_REQ	target has accepted the power control transaction
		REGIN_PW_RSP	target acknowledges that it has correctly handled the power control transaction request
		END_PW_RSP	initiator has accepted the target acknowledge
tlm_pme_handling Channel	Attributes	CMD	command field enumeration; SLEEP_PME, WAKEUP_PME and PW_PME
		RESPONSE	response status enumeration indicating the success or failure of the PME transaction
		TYPE	PME transaction enumeration type (PW_STATUS, PW_MODE_RQST and NO_INFO)
		RETAIN	boolean attribute valid only with the SLEEP_PME command
		PDID	7-bit power domain identifier of the PME transaction's power domain initiator
		TID	32-bit target identifier of the PME transaction's design clement initiator
		SUB_PDID	7-bit power domain identifier of the NRCT power domain, indicated when an RCT requests a power mode setting for another NRCT power domain
		PRIORITY	bus arbitration level for request capable targets (RCTs) (HIGH or LOW)
		LOCK	lock PDMgIF interconnect during a PME transaction
	Timing Points	BEGIN_PME_REQ	a RCT has set PME transaction attributes and initiates the target arbitration process
		END_PME_REQ	the PDMgIF interconnect has accepted the slave arbitration request
		BEGIN_PME_TRANSFER	the PDMgIF interconnect transfers the PME transaction set by the RCT
		END_PME_TRANSFER	the target acknowledges that the PME transfer has been correctly done

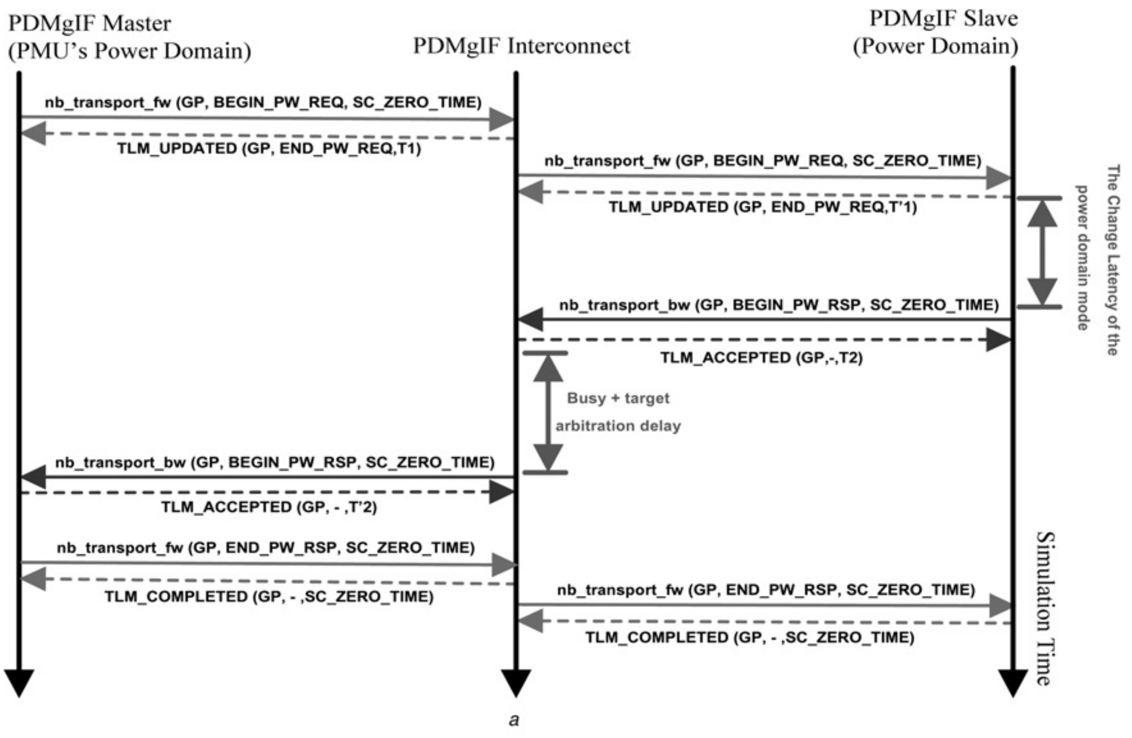
process. Although RCT power domains can issue a series of pipelined PME transactions, the PDMgIF interconnect must be locked once it is granted to an RCT by appropriately setting the LOCK attribute. This will force the PMU to save its current state and power management scheme status and only receive and handle the elected PME transaction. The PMU will then be prevented from exchanging any other data over the PDMgIF interconnect during this period. This is more useful in situations where the transmitted PME is timing-critical or have a direct impact on the power management decisions taken by the PMU.

The tlm_pme_handling channel attributes are mapped into a TLM 2.0 'tlm_pme_handling' payload extension. As Table 1 illustrates, four timing points are supported within the lifetime of a PME transaction. Each timing point is mapped into a phase in the 'tlm_PDMgIF_phase' class. According to the TLM 2.0 standard semantics, the same module can act as an initiator and a target when using the non-blocking TLM 2.0 transport interface [6].

This TLM 2.0 modelling feature can be considered to model the RCT concept. Therefore in the context of our work, each PDMgIF target defined as an RCT will use the TLM 2.0 non-blocking transport interface calls on the backward path [i.e. nb_transport_bw() method call] in order to initiate a PME transaction. Fig. 3*b* illustrates this feature and depicts the PME transaction sequencing rules between a 'PDMgIF initiator' and 'target' during a PME transaction transfer.

Given the sequencing between timing points of each channel shown in Figs. 3*a* and *b*, the PDMgIF protocol behaviour on the initiator and target sides can be determined. Figs. 4*a* and *b* show a high-level representation of the state machines for, respectively, the initiator and target sides. States of each state machine correspond either to calling the TLM2.0 'nb_transport' interface methods or to waiting for the arrival of a TLM 2.0 'nb_transport' interface call from targets. More precisely, on the initiator side, states correspond to either sending PDMgIF transactions by calling to the 'nb_transport_fw()' method or to waiting for calls to the 'nb_transport_bw()' method from targets (Fig. 4). On the target side, states correspond to either sending PDMgIF transactions by calling to the 'nb_transport_bw()' method or to waiting for incoming PDMgIF transactions in the form of calls to the 'nb_transport_fw()' method from initiators (Fig. 4). Naturally, the PDMgIF interconnect is considered as both a 'PDMgIF initiator' and 'target'. Transitions between states in Fig. 4 are conditioned by a transaction status or a PME reception.

As it can be observed in Figs. 4*a* and *b*, rules have been defined for the temporal relationship between phases of a power control transaction and that of a PME transaction. For


PDMgIF Master
(PMU's Power Domain)
PDMgIF Interconnect
PDMgIF Slave
(Power Domain)
nb_transport_fw (GP, BEGIN_PW_REQ, SC_ZERO_TIME)
TLM_UPDATED (GP, END_PW_REQ,T1)
nb_transport_fw (GP, BEGIN_PW_REQ, SC_ZERO_TIME)
TLM_UPDATED (GP, END_PW_REQ,T'1)
The Change Latency of the power domain mode
nb_transport_bw (GP, BEGIN_PW_RSP, SC_ZERO_TIME)
TLM_ACCEPTED (GP,-,T2)
Busy + target arbitration delay
nb_transport_bw (GP, BEGIN_PW_RSP, SC_ZERO_TIME)
TLM_ACCEPTED (GP, - ,T'2)
nb_transport_fw (GP, END_PW_RSP, SC_ZERO_TIME)
TLM_COMPLETED (GP, - ,SC_ZERO_TIME)
nb_transport_fw (GP, END_PW_RSP, SC_ZERO_TIME)
TLM_COMPLETED (GP, - ,SC_ZERO_TIME)
Simulation Time
a

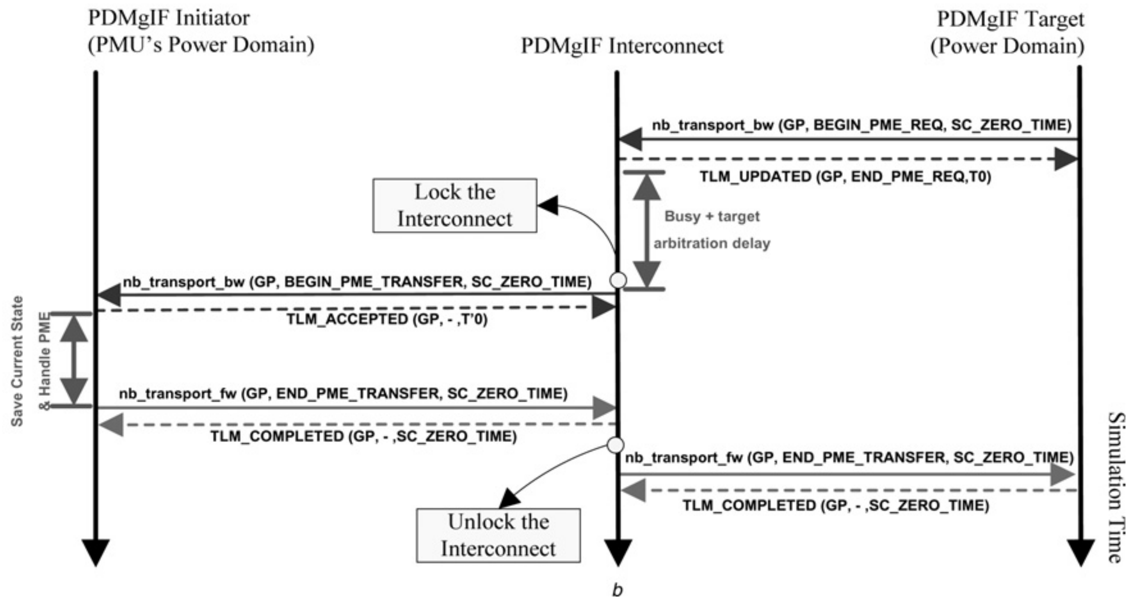

PDMgIF Initiator
(PMU's Power Domain)
PDMgIF Interconnect
PDMgIF Target
(Power Domain)
nb_transport_bw (GP, BEGIN_PME_REQ, SC_ZERO_TIME)
TLM_UPDATED (GP, END_PME_REQ,T0)
Lock the Interconnect
Busy + target arbitration delay
nb_transport_bw (GP, BEGIN_PME_TRANSFER, SC_ZERO_TIME)
TLM_ACCEPTED (GP, - ,T'0)
Save Current State & Handle PME
nb_transport_fw (GP, END_PME_TRANSFER, SC_ZERO_TIME)
TLM_COMPLETED (GP, - ,SC_ZERO_TIME)
nb_transport_fw (GP, END_PME_TRANSFER, SC_ZERO_TIME)
TLM_COMPLETED (GP, - ,SC_ZERO_TIME)
Unlock the Interconnect
Simulation Time
b

Fig. 3 *PDMgIF protocol phase sequences*

a Permitted phase transitions of the tlm_pwctrl channel using the TLM 2.0 standard transport interfaces
b Permitted phase transitions of the tlm_pme_handling channel, using the TLM 2.0 standard transport interfaces

instance, Fig. 4*a* depicts the case when a 'PDMgIF initiator' (i.e. the PMU's power domain) receives a PME transaction while it is waiting for the BEGIN_PW_RSP phase of a power control transaction. Here, the initiator has to urgently treat the PME transaction and perform this PME transaction state transition to the END_PME_TRANSFER phase before handling a potentially received BEGIN_PW_RSP phase (Fig. 4*a*).

4.3 PDMgIF protocol interconnect structure and behaviour definition

Fig. 5 depicts the internal structure and behaviour of a SystemC TLM 2.0 PDMgIF interconnect model. It includes the following modules: 'Identifiers Decoder', 'Target Arbiter', 'PDMgIF Initiator' and 'PDMgIF Target'. The 'Identifiers Decoder' routes each transaction from a power domain to another, for both the forward and backward paths. For that, it uses a map that matches each PDID with its corresponding power sockets. Like each PDMgIF initiator, the PDMgIF interconnect includes a 'PDMgIF Initiator' module that derives from the 'PDMgIF initiator' base module. Moreover, like each PDMgIF target, the PDMgIF interconnect includes a 'PDMgIF Target' module that derives from the 'PDMgIF target' base module. The behaviour of each of the 'PDMgIF initiator' and 'target' base modules is depicted by, respectively, Figs. 4*a* and *b*.

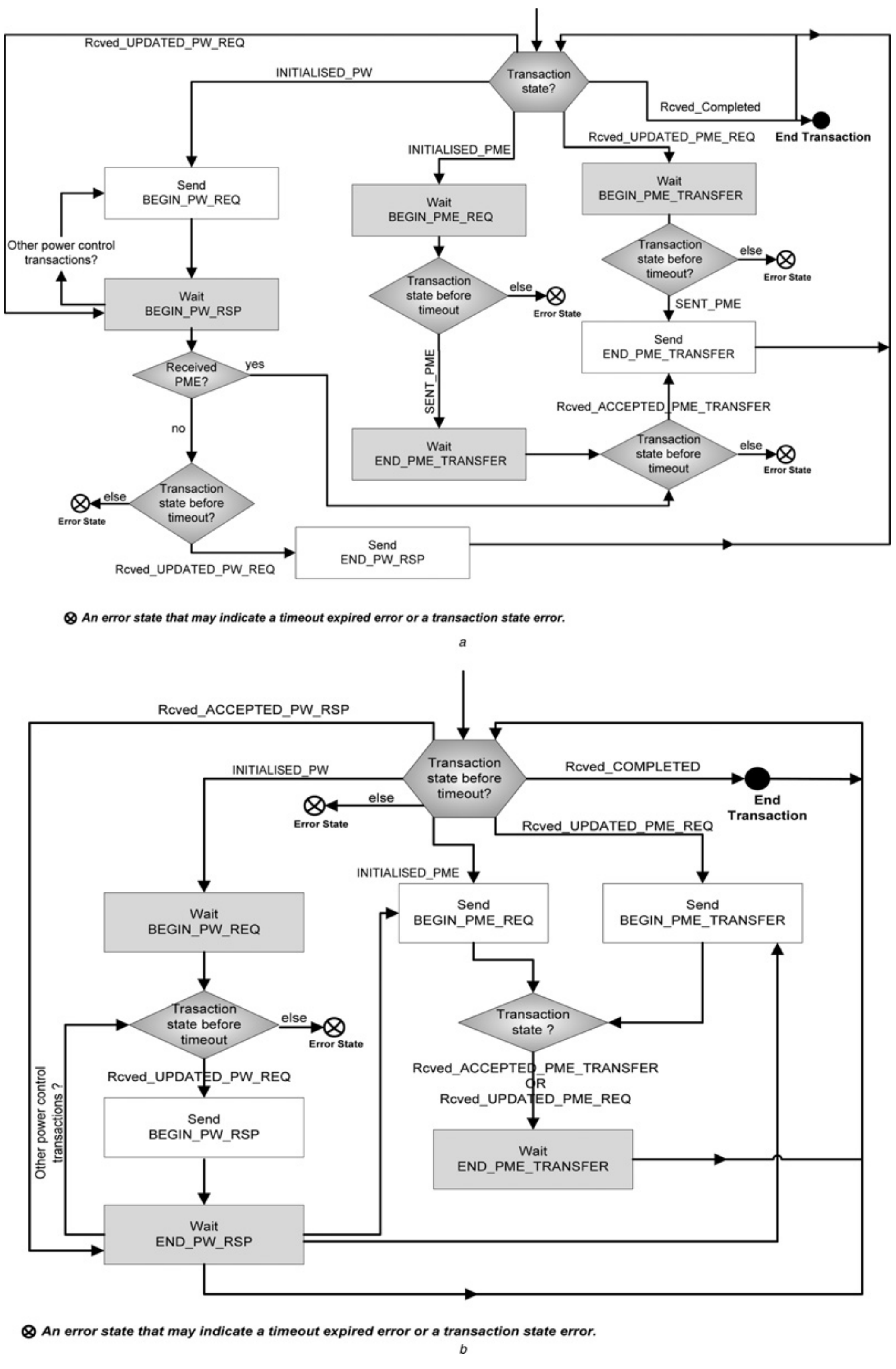


Fig. 4 *Mapping channels FSMs to initiator and target state machines*
a Finite state machine for the initiator side of the PDMgIF protocol
b Finite state machine for the target side of the PDMgIF protocol

Both initiator and target base modules include an active part that contains the protocol channels state machines for initiating the outgoing transactions. While the active part of the ‘PDMgIF initiator’ derives the forward path of a transaction, the active part of the ‘PDMgIF target’ derives the backward path. The initiator and target base modules also contain a reactive part that processes the incoming transactions by implementing the related ‘nb_transport’ transport interface. Depending on the received phase, this method notifies the adequate FSM in the active part. Therefore a synchronisation layer (events and arrays of ongoing transactions status) is required between the two

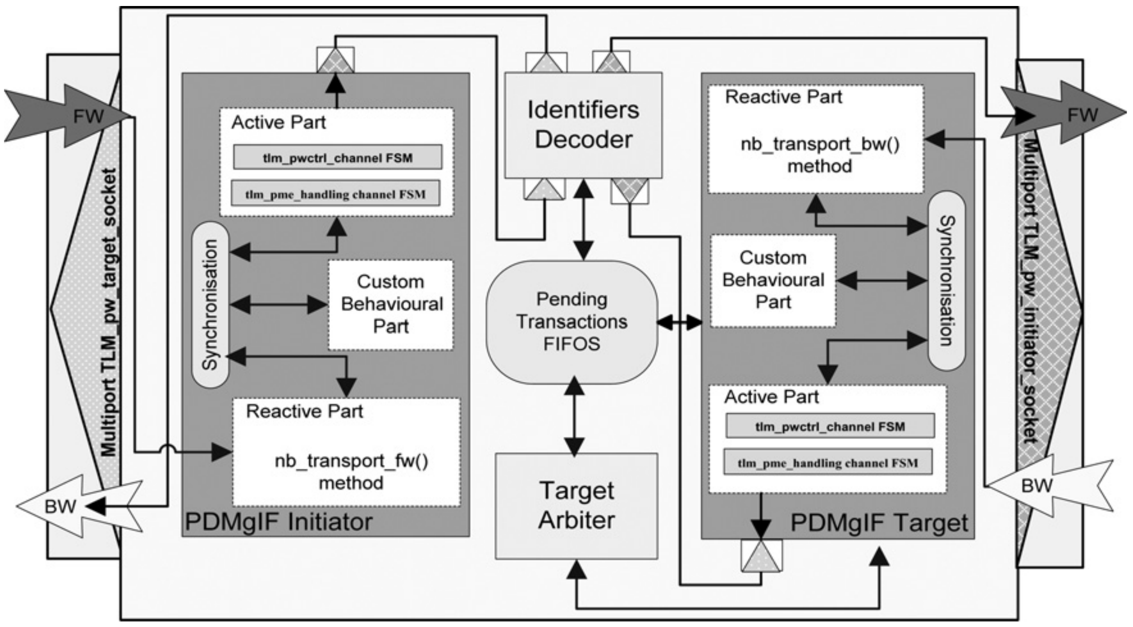


Fig. 5 *Internal structure and behaviour modelling of the PDMgIF interconnect using the TLM 2.0 standard transport interfaces*

parts of each base module. As shown in Fig. 5, a custom behavioural part is added to customise the phase transitions sequencing defined in the base modules active parts.

The 'Target Arbiter' module handles target arbitration requests. These requests consist in PME transactions initiated by an RCT PDMgIF target via the TLM 2.0 backward path. The 'Target Arbiter' module decides which RCT power domain gets the bus based on the PRIORITY attribute (see Table 1). A PME transaction with a high priority level transmits timing-critical information and must be granted the PDMgIF interconnect once received. In order to guarantee that all RCT power domains can access the PDMgIF interconnect, each of them shall obey the following rule: a power domain that has transmitted a high priority PME transaction can only transmit, henceforth, a low priority PME transaction until another power domain issues a high priority PME.

5 Evaluation on an audio application virtual prototype

Performance and flexibility of the TL PDMgIF interface have been tested with a TL virtual prototype for an ADPCM-based audio application shown in Fig. 6*b* [19]. The test consists in recording and playing a 5-s voice message. To record a voice message, linear audio samples are first stored in the static random access memory (SRAM) memory. Then, a block of ten samples is transferred from the SRAM memory to the G711 encoder in order to encode them using the G711 voice-compression algorithm. The resultant encoded samples are transferred back to the SRAM once their G711 compression is completed. This step is repeated until the end of linear samples. At this point, the G726 encoding process is performed in the same way as the G711 encoding one. Blocks, each including ten G711 encoded samples, are successively copied from the memory to the G726 encoder to obtain compressed using the G726 voice-compression algorithm. Each encoded block is then stored in the SRAM memory. To listen to a recorded message, the reverse procedure of recording is executed starting by the G726 decoding and ending with the G711 decoding.

In addition to this sequential execution form, we have also tested the pipelined execution form, in which a previously G711 encoded block will be processed by the G726 encoder while a new block is being encoded using the G711 encoder. The same pipelined execution principle is applied to the decoding part.

As shown in Fig. 6, three different PwARCH alternatives have been considered. Each alternative has been first defined using our PwARCH library [5]. PDMgIF interfaces and power domains layers are then added according to our modelling approach in order to control the specified PwARCH. Fig. 6*b* shows an example of a power-domain-managed structure (corresponding to the first PwARCH alternative illustrated by Fig. 6*a*) layered on top of the initial functional platform. In alternative 2 (Fig. 6*c*), each audio codec submodule is put in a single-power domain and the SRAMC belongs to the AO_PD. Alternative 3 (Fig. 6*d*) is the same as alternative 2, except the SRAMC module is put on a single power-gated domain that can be switched-off while encoding or decoding samples.

Each TL functional platform execution version (sequential or pipelined) has been simulated with the three PwARCH alternatives, while considering three different power management strategies for each alternative. The considered power management strategies are: 'scenario-based', reactive and 'scenario-tracking' strategies.

A 'scenario-based strategy' relies on the specification of a static power state table (PST), which summarises possible system power modes. Each system power mode represents a combination of power domain states and corresponds to power requirements of a specific software scenario. This PST-based strategy is originally adopted by the UPF standard [3]. An example of a PST for the PwARCH alternative in Fig. 6*a* is given by Table 2. Here, the 'Record' system power mode corresponds to the record voice scenario, where both the G.711 and G.726 encoding are performed. Therefore the Audio_enc_PD power domain (including the G711 and G726 encoders) must be powered-on before this scenario execution. In general, when

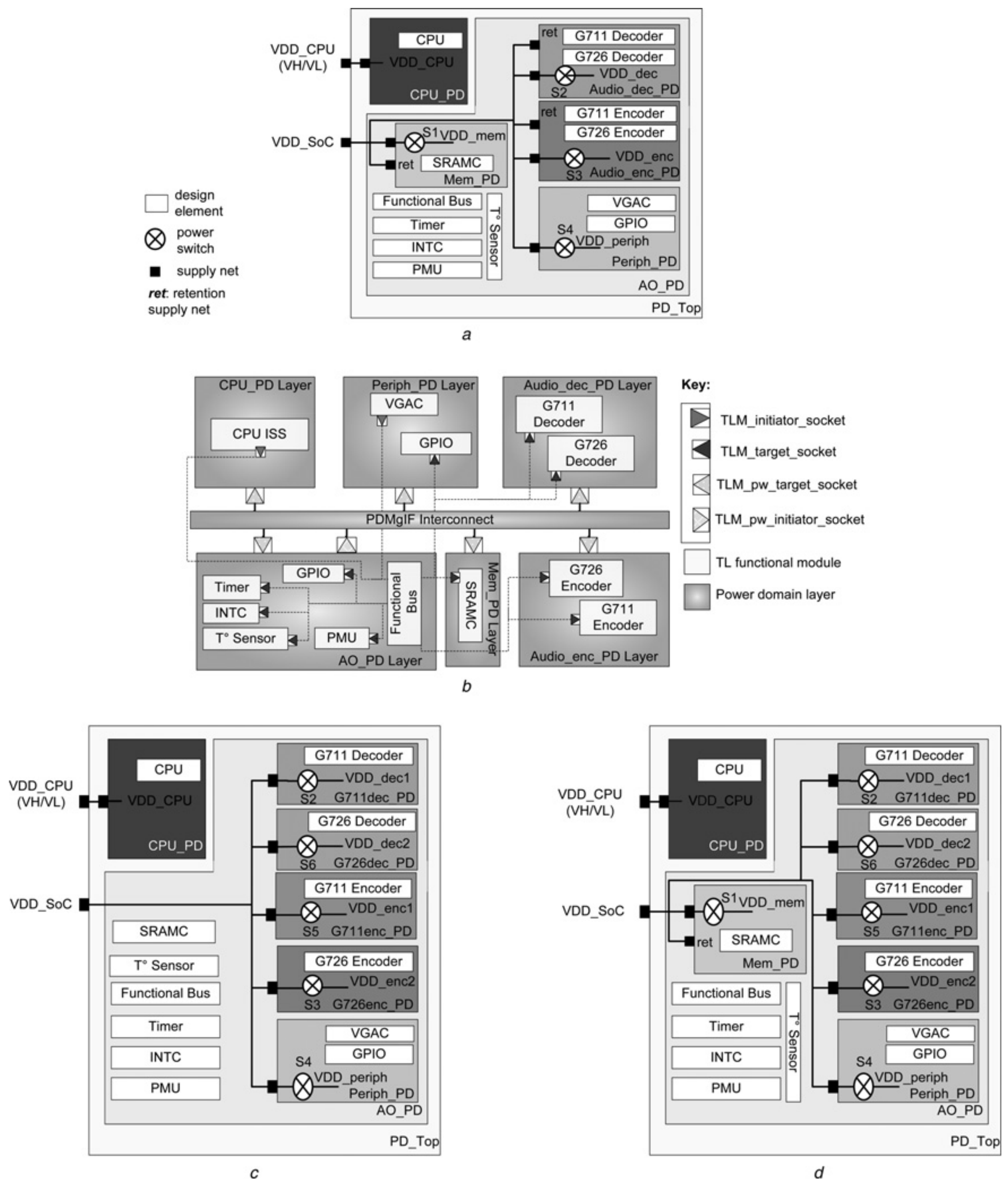


Fig. 6 *Considered PwARCH alternatives*

a Alternative 1
b Building power domain layers and PDMgIF interfaces
c Alternative 2
d Alternative 3

Table 2 An example of a PST for the power architecture alternative

	PD state						
Pw mode	PD_Top	AO_PD	CPU_PD	Mem_PD	Audio_enc_PD	Audio_dec_PD	Periph_PD
allon	ON	ON	VH	ON	ON	ON	ON
alloff	OFF	OFF	OFF	OFF	OFF	OFF	OFF
record	ON	ON	VL	OFF	ON	OFF	OFF
play	ON	ON	VL	OFF	OFF	ON	OFF
transfer_mem	ON	ON	VH	ON	–	–	OFF

– : do not care.

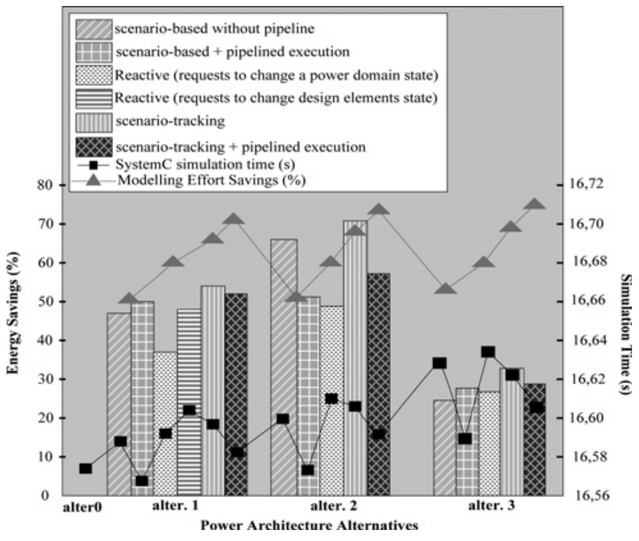


Fig. 7 *Energy savings, modelling effort savings and simulation time for the various power management strategies and PwARCH alternatives*

using the scenario-based strategy, transactions on the PDMgIF interconnect consist only in power control transactions.

In a 'reactive strategy', the PMU only responds to each RCT power domain requesting a power state change through a PME transaction. A 'scenario-tracking' strategy is similar to the scenario-based one, since the PMU still uses a PST. However, PME transactions that simply inform the PMU about a system functional state are allowed. This information helps the PMU to decide about the right PST power mode to set. As a simple example, consider that while the system in Fig. 6 is recording a message, the temperature sensor issues a PME transaction to request a state change of the Audio_enc_PD power domain because of the detection of an excessive heating. Here, the PMU has to stop recording and just play the encoded samples. For that, it has to switch-off the Audio_enc_PD power domain and switch-on the Audio_enc_PD power domain instead.

Fig. 7 shows the obtained energy savings for each alternative compared with the initial non-partitioned functional platform (alternative 0 in Fig. 7). These results highlight the ability of our PDMgIF protocol model to handle various power management solutions. In our case-study example, the scenario-tracking strategy and the PwARCH alternative 2 together represent the most energy-efficient power management solution for this platform as it saves energy by up to 70%.

Fig. 7 shows also the modelling effort savings achieved by using the common PDMgIF protocol interface instead of SystemC signals. Modelling effort refers to the source code lines and ports number added for power domain management. These results show that our PDMgIF interface achieves flexibility to consider all types of power domain management strategies with a reduced modelling effort. As Fig. 7 shows, modelling effort is saved by up to 70% with the PDMgIF use compared with the signal-based management use for the three PwARCH alternatives when applying a pipelined execution and scenario-tracking strategy. This is because of the PDMgIF high flexibility and fast reuse whatever the applied PwARCH and management strategy.

Fig. 7 gives also the simulation time required by each strategy to record and play the same voice message according to each alternative. Differences between elapsed simulation times are because of the PDMgIF latencies and time penalties required for power state transitions. Note that only a small simulation time overhead, lower than 6%, is incurred by our PDMgIF-based modelling approach. This enforces the ability of our PDMgIF-based approach to rapidly explore different PwARCH and domain management alternatives at the TL.

6 Conclusions and discussions

We have presented a TLM 2.0 simulation model of a new and flexible inter-power-domain protocol interface. A great benefit of this interface is the easy reuse and the platform-independency. It allows an easy integration of a power-domain-managed architecture into a functional SoC model and enables power domains reuse in different platforms. Separation of functional and power concerns promotes this easy integration. The PDMgIF proposed features represent a potential extension of the UPF and CPF standards that miss PwARCH control semantics.

Nevertheless, a more formal study of our proposed protocol properties is strongly needed in the future. This study would allow checking this protocol completeness and correctness and solving its potential ordering and deadlock freedom issues. In addition, our PDMgIF protocol actually shows a limited scalability. For instance, a large power domains number in an SoC and a frequent change of power domain states would cause a high-power management overhead. This requires finding the best trade-off between power domains number and system energy efficiency. As a potential solution for the PDMgIF scalability improvement, studies on hierarchical organisation of PMs in relation to hierarchical power domains partitioning are currently carried out.

7 Acknowledgments

The authors gratefully acknowledge the financial support from the French National Research Agency (ANR) under the HOPE project bearing reference ANR 12 INSE 0003.

8 References

1 Keating, M., Flynn, D., Aitken, R., Gibbons, A., Shi, K.: 'Low power methodology manual: For system-on-chip design (integrated circuits and systems)' (Springer, 2007)
2 S. I. Initiative, Common Power Format (CPF) 1.1 Specification, September, 2008
3 Unified Power Format (UPF 2.0) Standard, IEEE1801 TM: 'IEEE standard for design and verification of low power integrated circuits', 27 March, 2009
4 Veller, Y., Matalon, S.: 'Why you should optimize power at the electronic system level'. Mentor Graphics white paper, 2010, www.mentor.com
5 Mbarek, O., Pegatoquet, A., Auguin, M.: 'Using unified power format standard concepts for power-aware design and verification of systems-on-chip at transaction level', *IET Circuits Devices Syst.*, 2012, **6**, (5), pp. 287–296
6 Open SystemC initiative, SystemC Transaction Level Modeling Library 2.1.0, 2009, www.systemc.org
7 Van Molland, H.W.M., Corporaal, H., Reye, V., Boonen, M.: 'Fast and accurate protocol specific bus modeling using TLM 2.0'. Proc. Int. Conf. Design, Automation and Test in Europe (DATE '09), Nice, France, 2009, pp. 316–319
8 Damm, M., Moreno, J., Haase, J., Grimm, C.: 'Using transaction level modeling techniques for wireless sensor network simulation'. Proc. Int. Conf. Design, Automation and Test in Europe (DATE'10), Dresden, Germany, March 2010, pp. 1047–1052
9 Lee, I., Kim, H., Yang, P., *et al.*: 'PowerViP: Soc power estimation framework at transaction level'. Proc. ASP-DAC'06, 2006, pp. 551–558

10 Le Lebreton, H., Vivet, P.: 'Power modeling in systemC at transaction level, application to a DVFS architecture'. Proc. ISVLSI, Montpellier, France, 2008, pp. 463–466

11 Power System Management Protocol Specifications: http://pmbus.org/specs.html

12 System Power Management Interface Specification: http://www.mipi.org/specifications/system-power-management-interface

13 Sheets, M.: 'Standby power management architecture for deep-submicron systems'. PhD thesis, University of California Berkeley, May 2006

14 Sheets, M., Burghardt, F., Karalar, T., Ammer, J., Chee, Y.H., Rabaey, J.: 'A power-managed protocol processor for wireless sensor networks'. Proc. Int. Symp. VLSI Circuits Digest of Technical Papers, 2006

15 PCI Bus Power Management Interface Specification. Revision 1.2. March, 2004: http://www.pcisig.com/specifications/conventional/pcipm1.2.pdf

16 PCI Express Base Specification, Revision 1.1, PCI-SIG: http://www.pcisig.com/specifications/pciexpress/base/

17 Advanced Configuration & Power Interface (ACPI) Specification, Revision 4.0a, April, 2010: http://www.acpi.info/

18 OMAP35xx Applications Processor, Power, Reset, and Clock Management, Texas Instruments OMAP™ Family of Products, February 2008, http://maemo.jacekowski.org/docs/power_managment_sprufa5.pdf

19 ITU-T website: http://www.itu.int/itu-t/recommendations/rec.aspx?id=10651