Application Note 318

CoreTile Express A15x2 A7x3
Power Management

Document number: ARM DAI 0318

Issued: April 2013

Application Note 318
CoreTile Express A15x2 A7x3 Power Management

Release information

The following changes have been made to this Application Note:

Change history

Date	Issue	Change
10/05/2012	A	First release
21/06/2012	B	Second release
26/06/2012	C	Third release, Debug support board.txt option added
27/09/2012	D	Cluster power up/down sequence updated, Power measurement values updated, SYS_CFG support added
24/04/2013	E	Updates to the Virtual SPC register addresses

Proprietary notice

Confidentiality status

This document is Open Access. This document has no restriction on distribution.

Feedback on this Application Note

If you have any comments on this Application Note, please send email to errata@arm.com giving:

- the document title
- the document number
- the page number(s) to which your comments refer
- an explanation of your comments.

General suggestions for additions and improvements are also welcome.

ARM web address

http://www.arm.com/

Table of Contents

1 Introduction

1.1 Purpose of this application note

This application note discusses the operation of the power management features for a CoreTile Express A15x2 A7x3 (V2P-CA15_A7 HBI-0249). It will cover the power management hardware features provided by the board and software model for interfacing to this hardware.

On reading this Application Note the user should be in a position to develop and debug power management software running on the V2P-CA15_A7 CoreTile. This will include Dynamic Voltage and Frequency Scaling (DVFS) and Power Switching. Both methods are required to support a big.LITTLE power management model.

1.2 Overview of hardware platform

This CoreTile Express A15x2 A7x3 is designed to be fitted on a Motherboard Express uATX (V2M-P1) and uses the ARM Cortex-A Series memory model in Site 1. The following documents should be referred to in conjunction with this document.

[1] Motherboard Express uATX TRM: DUI0447E_v2m_p1_reference_manual.pdf

[2] CoreTile Express A15x2 A7x3 TRM: DDI0503A_v2p_ca15_a7_reference_manual.pdf

1.3 Getting Started

Before using this application note please ensure that the firmware running on your boards has been updated to the following versions or later:

- V2P-CA15_A7 DCC bios: dbb_v110.ebf
- V2M-P1 MCC bios: mbb_v311.ebf
- V2M-P1 Bootmonitor: bm_v519r.axf

Please refer to the TRMs supplied with these boards for the update procedure.

2 System architecture

The CA15_CA7 test-chip provides isolated power domains for the CA15 cluster, CA7 cluster and the SOC. These are supplied by PSU0, PSU1 and PSU2 respectively on the V2P-CA15_A7 board. The DDR and VIO domains are supplied from PSU3 and VIO.

The PSU0, PSU1 and PSU2 power supplies have onboard current sensing resistors with external test-points to allow current measurement and power profiling as shown in *Figure 1*. These supplies also have on-board ADCs for voltage, current and power measurement which is performed through the V2P-CA15_A7 DCC, on board energy meters are also provided for PSU0 and PSU1. In addition PSU0 and PSU1 allow voltage control over a 0.8-1.05V range (0.9V typ.). Process limitations restrict the recommended voltage range of PSU0 and PSU1 to 0.9-1.05V. PSU2 is fixed at 0.9V. All three supplies can be switched off through the Daughterboard Configuration Controller (DCC). The DCC is a Cortex-M3 microcontroller external to the test-chip and is implemented on the V2P-CA15_CA7 board. It directly interfaces with the Serial Power Controller (SPC) implemented within the test-chip. The DCC is responsible for controlling DVFS, cluster power-up/down and cluster wake-up.

The onboard clocking scheme is also shown with the option of clocking the CA15 cluster from OSCCLK0/1/7 and the CA7 cluster from OSCCLK2/3/7. The SOC domain is driven by OSCCLK7. The default OSCCLK values are set by the boards board.txt file stored on the USBMSD. These can also be dynamically changed through the DCC.

There are two methods of interfacing to the DCC, 1. through the legacy motherboard SMB SYS_CFGCTRL registers and 2. through the test-chip SPC registers. The SYS_CFGCTRL interface is slower but allows direct control of all OSCCLKs and PSUs while SPC interface is faster but uses a ‘performance level request’ scheme which is interpreted by the DCC to control OSCCLKs and PSUs. The SPC method is recommended for big.LITTLE power management support which utilises DVFS and power switching.

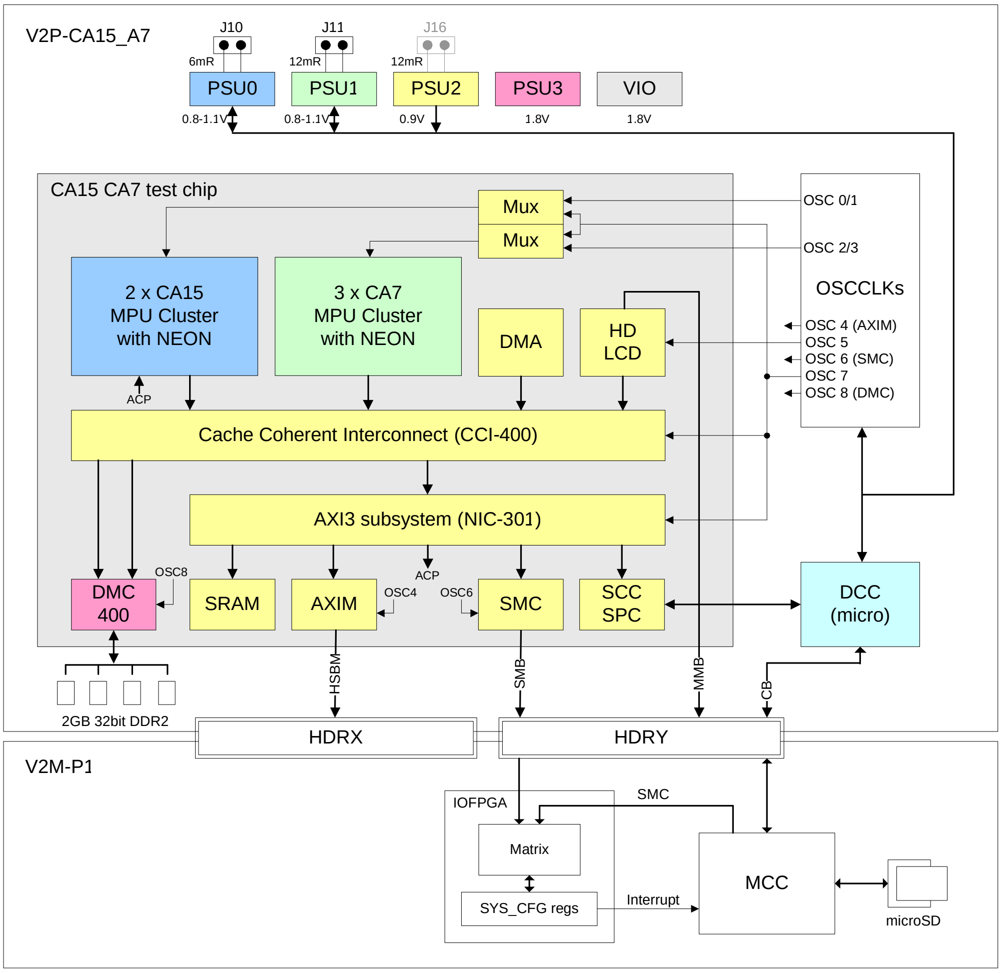

Figure 1, **Power Management system architecture**

3 Power Management

The following sections describe the V2P-CA15_A7 power management support using the test-chip SPC (serial power controller) and SCC (serial configuration controller) interfaces to the V2P-CA15_A7 DCC. The legacy SMB SYS_CFGCTRL interface will not be covered here and for further information on this interface please refer to the Motherboard Express uATX TRM[1] if required.

This V2P-CA15_A7 power management support provides the following three primary functions through the SPC and SCC interfaces to allow software implementation of the big.LITTLE Switcher / MP Model:

- **DVFS (dynamic voltage and frequency scaling)**: a CA15 or CA7 CPU requests a new ‘performance level’ for a particular cluster (CA15 or CA7). This request results in DCC setting a new frequency and/or voltage level for the specified cluster based on a performance table defined in the V2P_CA15_A7 board.txt file.
- **Cluster Power-Down/Up**: a CA15 or CA7 CPU requests the DCC to power-down/power-up the CA15 or CA7 cluster. The DCC manages the cluster resets then disables/enables the appropriate power supply as required.
- **WAKE_UP**: In the situation that both CA15 and CA7 clusters are in power down a method of waking up the clusters through the DCC is provided. This method uses interrupts from the GIC-400 block that are still active in power-down as well as the Generic Timer and Watchdog timer.

3.1 big.LITTLE system

The CA15_A7 test-chip supports both of the big.LITTLE MP models: Heterogeneous Multi-Processing (HMP) and the *Switching Model* (Cluster Migration).

In the big.LITTLE HMP model, all of the big and LITTLE processors are executing a single instance of a big.LITTLE MP model aware kernel. This awareness enables the kernel to schedule tasks to the most appropriate processors, including the ability to power up/down inactive clusters. For example, when running only low intensive tasks, the kernel can chose to power down the big cluster and continue only on the LITTLE.

In the big.LITTLE *Switching Model* (Cluster Migration) only one of the clusters is active at a time. Since the energy efficiency of the CA7 is better than the CA15, high performance applications can be executed on the CA15 cluster and medium and low performance applications can be executed on the CA7.

The big.LITTLE *Switching Model* (Cluster Migration) is a type of HMP where the operating system and applications are executed only on either the CA15 or CA7 cluster, but never on both at the same time. Once the CA7 cluster has reached the highest performance operating point (over drive), or the CA15 cluster has reached the lowest performance operating point (under drive) the execution can switch to the other cluster. The design relies on CCI-400 for the fast switching. The data migration is enabled by access to the outbound side’s cache from the inbound side via snooping, and this prevents expensive access to main memory.

3.2 Switcher software

In the big.LITTLE *Switching Model*, the *switcher* software handles all the mechanisms required to switch between clusters, such as the processor state save-restore, control snooping, migration of interrupts etc. But the switching sequence has some system dependencies.

For example, the inbound cluster has to be powered-up, brought out of reset and isolation has to be removed. Once the inbound cluster is up and running, then the outbound cluster has to be put in reset, isolated, and then powered-down. The *switcher* software uses the test-chip SPC interface to communicate with the external V2P-CA15_A7 DCC which handles all the switching mechanics.

Once execution has switched to the inbound cluster, the L2 of the outbound cluster is kept alive for a certain time period to allow snooping. Instead of a fixed time window, performance counters in CCI can also be used to decide when to power-down the outbound cluster’s L2 cache depending on the snoop rate.

4 DVFS interface

Dynamic Voltage and Frequency Scaling allows the operating system to pick the optimal voltage and frequency for a particular requirement reducing the dynamic power consumption of the system.

It is the responsibility of the operating system to determine the required *performance level* based on the expected performance operating point. This desired performance level is a value between 0 and 7 which is passed to the DCC through the SPC registers PERF_LVL_A15 or PERF_LVL_A7. Writing to these registers results in an interrupt to the DCC which then reads the SPC registers and translates the performance level value to a voltage and frequency value. The translation is user defined through the V2P-CA15_A7 board.txt file, an example is given:

```
[SCC REGISTERS]
TOTALSCCS: xx                   ;Total Number of SCC registers

...
                                ;Power management interface
SCC: 0xC00 0x00000007           ;Control: [0]PMI_EN [1]DBG_EN [2]SPC_SYSCFG
SCC: 0xC04 0x060E0356           ;Latency in uS max: [15:0]DVFS [31:16]PWRUP
SCC: 0xC08 0x00000000           ;Reserved
SCC: 0xC0C 0x00000000           ;Reserved

                                ;CA15 performance values: 0xVVVFFFFF
SCC: 0xC10 0x384061A8           ;CA15 PERFVAL0,  900mV, 20,000*20= 500MHz
SCC: 0xC14 0x38407530           ;CA15 PERFVAL1,  900mV, 25,000*20= 600MHz
SCC: 0xC18 0x384088B8           ;CA15 PERFVAL2,  900mV, 30,000*20= 700MHz
SCC: 0xC1C 0x38409C40           ;CA15 PERFVAL3,  900mV, 35,000*20= 800MHz
SCC: 0xC20 0x3840AFC8           ;CA15 PERFVAL4,  900mV, 40,000*20= 900MHz
SCC: 0xC24 0x3840C350           ;CA15 PERFVAL5,  900mV, 45,000*20=1000MHz
SCC: 0xC28 0x3CF0D6D8           ;CA15 PERFVAL6,  975mV, 50,000*20=1100MHz
SCC: 0xC2C 0x41A0EA60           ;CA15 PERFVAL7, 1050mV, 55,000*20=1200MHz

                                ;CA7 performance values: 0xVVVFFFFF
SCC: 0xC30 0x3840445C           ;CA7 PERFVAL0,  900mV, 10,000*20= 350MHz
SCC: 0xC34 0x38404E20           ;CA7 PERFVAL1,  900mV, 15,000*20= 400MHz
SCC: 0xC38 0x384061A8           ;CA7 PERFVAL2,  900mV, 20,000*20= 500MHz
SCC: 0xC3C 0x38407530           ;CA7 PERFVAL3,  900mV, 25,000*20= 600MHz
SCC: 0xC40 0x384088B8           ;CA7 PERFVAL4,  900mV, 30,000*20= 700MHz
SCC: 0xC44 0x38409C40           ;CA7 PERFVAL5,  900mV, 35,000*20= 800MHz
SCC: 0xC48 0x3CF0AFC8           ;CA7 PERFVAL6,  975mV, 40,000*20= 900MHz
SCC: 0xC4C 0x41A0C350           ;CA7 PERFVAL7, 1050mV, 45,000*20=1000MHz
```

***Figure 2*, Example board.txt file**

There are two sets of eight performance values, one set for each cluster (CA15 or CA7). Each value is 32 bits wide and has the following format:

Bits	**31:28**	**27:24**	**23:20**	**19:16**	**15:12**	**11:8**	**7:4**	**3:0**
Value	V	V	V	F	F	F	F	F
Example	3	8	4	0	C	3	5	0

Bits 19:0 represents the desired frequency value in KHz, the example shows 50,000KHz = 0xC350.

Bits 31:20 represents the desired voltage value in mV, the example shows 900mV = 0x384

Note: the above clock settings assume the CPU PLLs are set to 20:1 in the board.txt file. So with the above example 50,000KHz * 20 = 1GHz

4.1 DVFS sequence

The steps required to change the DVFS operating point of a cluster are as follows. The CA15 cluster is used as an example here:

1. The operating system requests a performance level by writing to the SPC register PERF_LVL_CA15.
2. This asserts PERF_REQ_CA15 register and also generates an interrupt to the DCC.
3. The DCC reads the SPC register PERF_LVL_CA15, and this de-asserts both the PERF_REQ_CA15 register and the corresponding interrupt.
4. The DCC interprets the performance requirement and translates that to an equivalent voltage and clock frequency for the CA15 cluster based on the board.txt file setting.

   When scaling-up the DCC sets the voltage first then increases the frequency.

   When scaling-down the DCC sets the frequency first and then decreases the voltage.
5. The DCC then updates the SPC register PWC_STATUS with the status of the request and whether it was completed successfully.
6. Writing to the SPC register PWC_STATUS asserts the PWC_FLAG and also interrupts the CPU (GIC400 interrupt input 99). The interrupt is cleared when a CA15/CA7 CPU reads the PWC_STATUS register.

4.2 DVFS limits

The following operating limits are recommended for the DVFS values.

	CA15 cluster	**CA7 cluster**
Normal 0.9V	1000MHz	800MHz
Overdrive 1.05V	1332MHz	1065MHz

Note. Although the CA15 and CA7 PSUs are rated over a 0.8 to 1.05V range process limitations in the test-chip limit the minimum voltage range of the test-chip HD RAMS to 0.9V. Also the minimum recommend operating frequency of each cluster is 340MHz. This limitation is due to the PLLs within the test-chip when set at 20:1, lower frequencies can be achieved with different PLL settings or by bypassing the PLLs, in this situation the limit is the OSCCLKs which operate down to 2MHz.

5 Power-down and Power-up modes

The V2P-CA15_A7 test-chip has three power islands relevant to power management, the CA15 cluster, CA7 clusters and SoC (system on chip). A block diagram of the test-chip and power supplies is shown in *Figure 1*. A more detailed diagram of the voltage islands is shown in *Figure 3*.

Note: the power-down and power-up sequences covered in this document provide the generalised sequence of events for the processor revisions implemented in the V2P-CA15_A7 test-chip. For detailed information on the specific implementation you have please refer to the relevant processor TRMs.

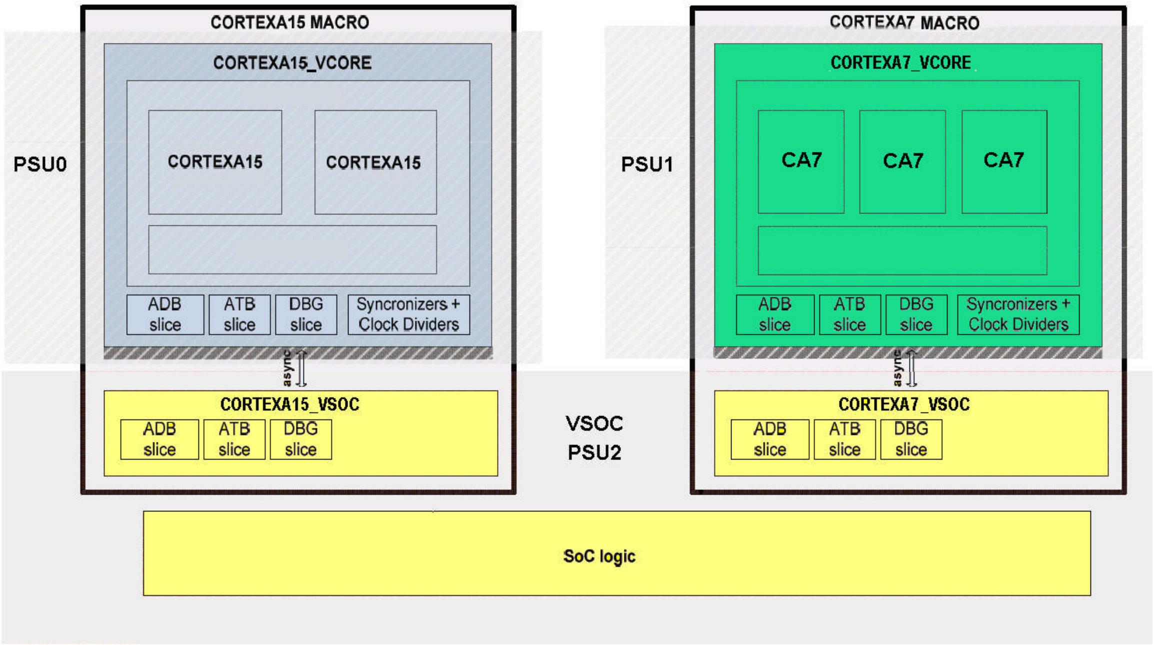

***Figure 3*, Test-chip power islands**

The CA15 and CA7 clusters support several levels of power management. The table below lists all the valid power states for an individual cluster. The CA15 and CA7 clusters could be in any of these states independently depending on the performance required from the system.

CA15/CA7_CPU0	CA15/CA7_CPU1	CA7_CPU2
ON	ON	ON
WFI/RST	ON	ON
ON	WFI/RST	ON
WFI/RST	WFI/RST	ON
WFI/RST	WFI/RST	WFI/RST
OFF	OFF	OFF

***Table 1*, Power down options**

Since the test-chip does not support internal power switches CPU level power down is not supported. Instead any CPU can be put into WFI state where all the clocks are architecturally gated. Once a CPU enters WFI state it can also be put into reset. Once all the CPUs in a cluster are in WFI state then the L2 for that cluster can also be put into WFI mode. Once all the CPUs and L2 of a cluster are in WFI state then the whole cluster can be put in reset or powered down externally at the power module via the DCC.

5.1 Entering WFI mode

The steps required to put a CPU in WFI mode are as follows:

1. Set the SCTLR.C bit to 0.
2. Clean and Invalidate the entire L1 data cache.
3. Set the ACTLR.SMP bit to 0. This prevents any future coherent requests being passed to this CPU from the other processors.
4. Write to the GIC-400 and disable the interrupts to it.
5. Execute a WFI instruction.
6. The CPU will assert the STANDBYWFI output and be in WFI mode.

5.2 Exiting WFI mode

The CPU can program the Generic Timer which is available at 0x2B04_0000 to generate an interrupt after a specified time. The Generic Timer operates based on a Generic Counter which runs at 24MHz clock.

5.3 Reset on WFI

A CPU or the whole cluster can be automatically put in reset once it is in WFI mode by setting the SPC reset hold registers CA15_RESET_HOLD and CA7_RESET_HOLD. The reset will be scheduled for when the CPU or the whole cluster enters the WFI mode.

Figure 4 shows the reset on WFI sequence for a CPU.

- Set the RESET_HOLD register in the SPC - requesting a reset on WFI. The RESET_HOLD register allows the selection of either nCPURESET or nCPUPORESET on entering WFI.
- The CPU enters WFI mode by executing the steps described in Section 5.1.
- STANDBYWFI from the CPU is asserted signaling WFI mode.
- nCPURESET/nCPUPORESET is asserted until the RESET_HOLD register is cleared. The clearing can be done by any other CPU on the test-chip. The reset will be asserted for a minimum of 16 cycles if the RESET_HOLD register is cleared in less than 16 cycles.
- CPU is out of reset.

In a similar way the RESET_HOLD register can be programmed to assert a full cluster reset on seeing STANDBYWFIL2.

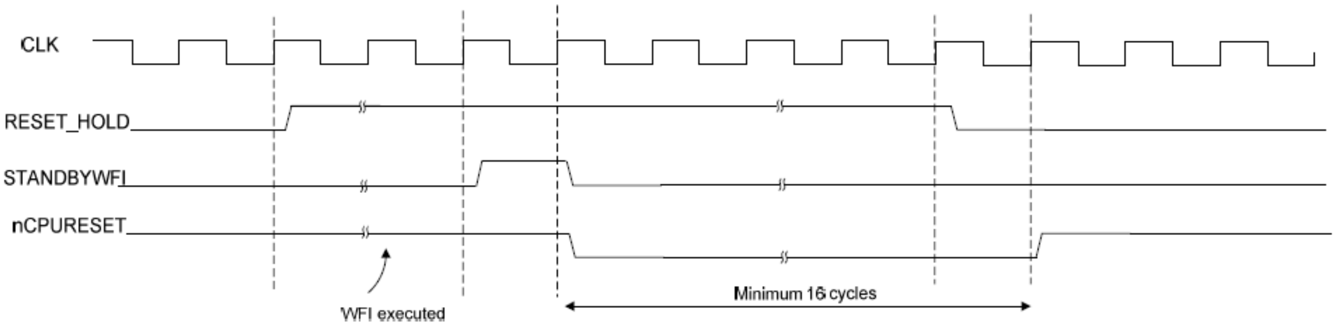

Figure 4, **Reset on WFI**

5.4 Cluster power down

In cluster power down the power supplies to the CA15 or CA7 clusters can be switched off. This is performed by the DCC once it detects the STANDBYWFIL2 signal is active and SPC PWRDN_EN register is enabled. The following steps show the power down sequence using the CA15 cluster as an example.

1. Each CPU writes to the GIC-400 registers to disable the CPU interface and interrupts to this processor.

2. Set the SPC register CA15_PWRDN_EN to 1’b1 to allow power down of the CA15 cluster.

3. Put the non-lead processors in the cluster in WFI mode.

   Set the SCTLR.C bit to 0.

   Clean and invalidate the entire L1 data cache.

   Set the ACTLR.SMP bit to 0. This prevents future coherent requests being passed from the other processors.

   Execute a WFI instruction.

4. Power down the lead processor

   Set the SCTLR.C bit to 0.

   Clean and invalidate the entire L1 data cache.

   Clean and invalidate the entire L2 data cache.

   Write to GIC-400 and disable the interrupts to it.

   Set the ACTLR.SMP bit to 0. This prevents future coherent requests being passed from the other processors.

5. Disable CCI snoop/DVM requests by clearing bits 0/1 in the CCI-400 snoop control register. This register is addressed at 0x2C09_4000 for the CA15 or 0x2C09_5000 for the CA7.

6. Execute a WFI instruction.

7. All STANDBYWFI_CA15 outputs from the test-chip will be asserted.

8. If all STANDBYWFI_CA15 outputs are asserted and the SPC register CA15_PWRDN_EN is set the DCC will start the following sequence to power down the cluster.

   Write to SCC register (CFGREG42) and assert the ACINACTM and AINACTS pins on the CA15 cluster, to idle the ACE and ACP interfaces. This prevents the L2 subsystem from accepting any more requests from the ACE and ACP interfaces. The CA7 cluster does not have an ACP interface, so only the ACINACTM input is asserted HIGH by writing to SCC register CFGREG44.

   Read SPC register CA15_CACTIVE to ensure all CACTIVE signals from the bridges in this cluster are low.

   Write to SPC register CA15_PWRDNREQ to request power down of the async bridges in the CA15 cluster.

   Read SPC register CA15_PWRDNACK to make sure power down acknowledges have been received from the async bridges in the CA15 cluster

   Write to the SPC register CA15_ISOLATE to clamp all the CA15 cluster outputs to appropriate values.

   Write to the SCC register CFGREG6[11:0] to put the whole CA15 cluster in reset.

   The DCC then disables the CA15 cluster power supply, powering down the CA15 cluster.

Notes:

The CA7 cluster can be powered down using a similar sequence.
Sequences shown in blue are performed by the DCC
Actual cluster PSU switch OFF depends on VPCFGREG0 CONTROL[1] described in section 9.3

5.5 Cluster power up

The sequence of steps through which an active CA7 cluster can power up the CA15 cluster is used as an example here. An active CA15 cluster can power up the CA7 cluster in a similar way.

1. Clear the SPC register CA15_PWRDN_EN to 1’b0 to disable power down of the CA15 cluster.
2. The CA7 writes to SPC register PERF_LVL_CA15 with a performance for the CA15 cluster, this can be the same value as was previously set before cluster power down.
3. The PERF_REQ_CA15 output of the test-chip is asserted and this interrupts the DCC.
4. The DCC is aware that CA15 cluster is powered down and goes through the power up sequence instead of just setting the voltage and frequency.

   Read SPC register PERF_LVL_CA15 to get the performance required on CA15 cluster.

   This clears the PERF_REQ_CA15 interrupt.

   Enable the CA15 cluster power supply, powering up the CA15 cluster.

   Remove the CA15 cluster outputs isolation/clamp by writing to SPC register CA15_ISOLATE.

   Write to the SCC register CFGREG6[11:10] to release the CA15 cluster async bridge resets.

   Clear the async bridge power down request by writing SPC register CA15_PWRDNREQ and then wait for acknowledgement from the SPC register CA15_PWRDNACK.

   Write to SCC register (CFGREG40) and clear the ACINACTM and AINACTS pins on the CA15 cluster, to enable the ACE and ACP interfaces. The CA7 cluster does not have an ACP interface, so only the ACINACTM input is cleared by writing to SCC register CFGREG44 on CA7 power up.

   Remove the rest of CA15 cluster resets by writing to SCC register CFGREG6[9:0].

   Set the requested performance level by setting the CA15 cluster voltage and frequency.

   Writes to SPC_PWC_STATUS register acknowledging completion of the power up sequence
5. The CA15 cluster comes out of power down and begins to execute code.
6. Enable CCI snoop/DVM requests by setting bits 0/1 in the CCI-400 snoop control register. This register is addressed at 0x2C09_4000 for the CA15 or 0x2C09_5000 for the CA7.

Notes:

Sequences shown in blue are performed by the DCC

Actual cluster PSU switch OFF depends on VPCFGREG0 CONTROL[1] described in section 9.3

An alternative method to the power up sequence shown above is to use the WAKE_UP interrupt from the test-chip described in section 6 rather than PERF_LVL_CA15 (PERF_LVL_CA7) shown in step 1 above. In this case the DCC firmware reads the WAKE_INT_STAT interrupt status register to determine which cluster and CPU(s) to wake up. With power up through WAKE_UP one or more CPUs can be taken out of reset during the power up sequence compared to a PERF_LVL_CA15 (PERF_LVL_CA7) which will always bring ALL of the CPUs in a cluster out of reset. If a cluster is powered down a WAKE_UP event for that cluster will bring ALL of the CPUs in that cluster out of reset.

The WAKE_UP interrupt has a further function once a cluster is powered up which is to bring CPUs out of reset (WFI), this allows a CPU to bring a CPU in the same cluster or another cluster out of reset. Once a WAKE_UP interrupt occurs for a cluster that is currently powered up the DCC will read the WAKE_INT_STAT register to determine if an interrupt has occurred for a CPU that is currently in reset (WFI). If it has the DCC will bring that CPU out of reset by writing to CFGREG6.

6 CA15 and CA7 cluster Wake up

When both the CA15 and CA7 clusters are powered down the system has to rely on the DCC to power up the clusters in the event of a system interrupt. A single WAKE_UP interrupt exists between the test-chip and the DCC, the DCC also knows when both clusters are in power down. If the DCC detects the WAKE_UP interrupt while both clusters are in power down the DCC reads the SPC register WAKE_INT_STAT to determine the interrupt source and powers up the appropriate cluster.

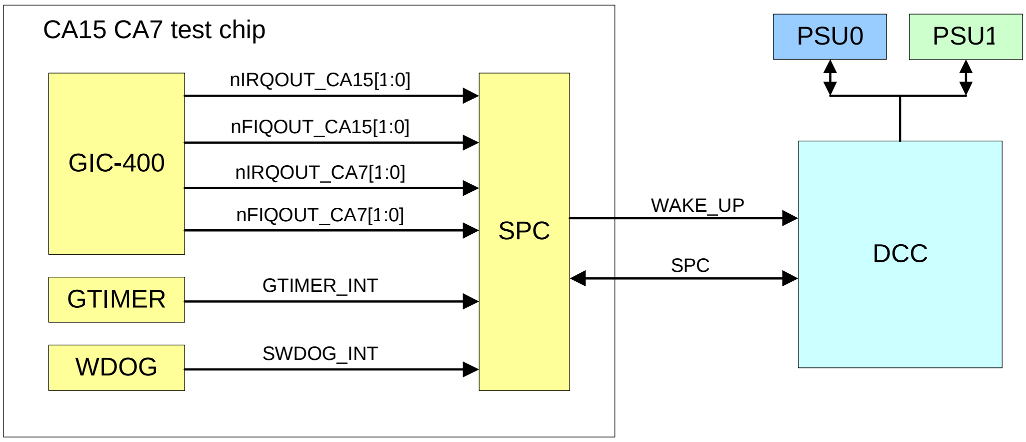

***Figure 5*, WAKE_UP interrupt**

When both CA5 and CA7 clusters are powered down, the GIC-CPU interface (nIRQx/nFIQx signals) to the CA15 and CA7 clusters are all disabled. So the regular GIC-400 outputs cannot be used as wake up sources.

Instead the nIRQOUTx/nFIQOUTx signals from the GIC-400 block are used as wake up sources. These signals mirror the values on nIRQx/nFIQx and are unaffected by disabling of GIC-CPU interface. These interrupts are combined with the Watchdog Timer to implement the wake up source. The scheme is shown in *Figure 5*.

Note: In addition to the GIC400 outputs, interrupts from the Generic Timer and Watchdog Timer can also directly drive the WAKE_UP interrupt however these are currently not supported by the DCC.

7 Power measurement

As mentioned in section 2 the V2P-CA15_A7 board has on board current sensing resistors:

- PSU0 - J10 - 6mR (equivalent resistor + PCB) – CA15 cluster
- PSU1 - J11 - 12mR (equivalent resistor + PCB) – CA7 cluster
- PSU2 - J16 - 12mR (equivalent resistor + PCB) - Internal AXI

The two pin Jumper headers allow an external meter to be used to determine the current consumption on each supply. J16 is not fitted by default as the internal AXI includes many interfaces and a full understand of the measured current consumption is system dependent.

7.1 On board power measurement

The V2P-CA15_A7 board includes on board current, voltage and power measurement of PSU0/1. These values are accessed through the V2M-P1 motherboard SYS_CFGCTRL or test-chip SPC_SYS_CFG interfaces, for more information please refer to section 8. For details on the available V2P-CA15_A7 SYS_CFGCTRL settings please refer to the ‘Serial Configuration Controller’ section in the V2P-CA15_A7 TRM.

7.2 On board energy meter

The V2P-CA15_A7 board includes two on board energy meters for reading PSU0/1 energy consumption. These sample at 10KHz and are also available through the V2M-P1 motherboard SYS_CFGCTRL or test-chip SPC_SYS_CFG interfaces (Device 0-3, Function 13). For more details please refer to the ‘Energy Meter’ section in the V2P-CA15_A7 TRM.

8 SPC based SYS_CFG interface

The motherboard system configuration registers, SYS_CFG allow various operating parameters to be monitored at run time, these include voltage, current, power and temperature. This interface provides access to all boards in the system through a common register interface in the motherboard IOFPGA.

Continuous access to the motherboard SYS_CFG interface, for example when reading power, can have an impact on the overall system performance. This is because of the increase in SMC traffic as well as adding delay caused by the speed of the interface itself.

The V2P-CA15_A7 test-chip provides an alternate solution to the motherboard SYS_CFG interface by re-using the test-chip SPC register interface to the local DCC. Providing the operating parameters required are only from the local DCC this direct interface can be used instead of the motherboard SYS_CFG interface. See *Figure 6*.

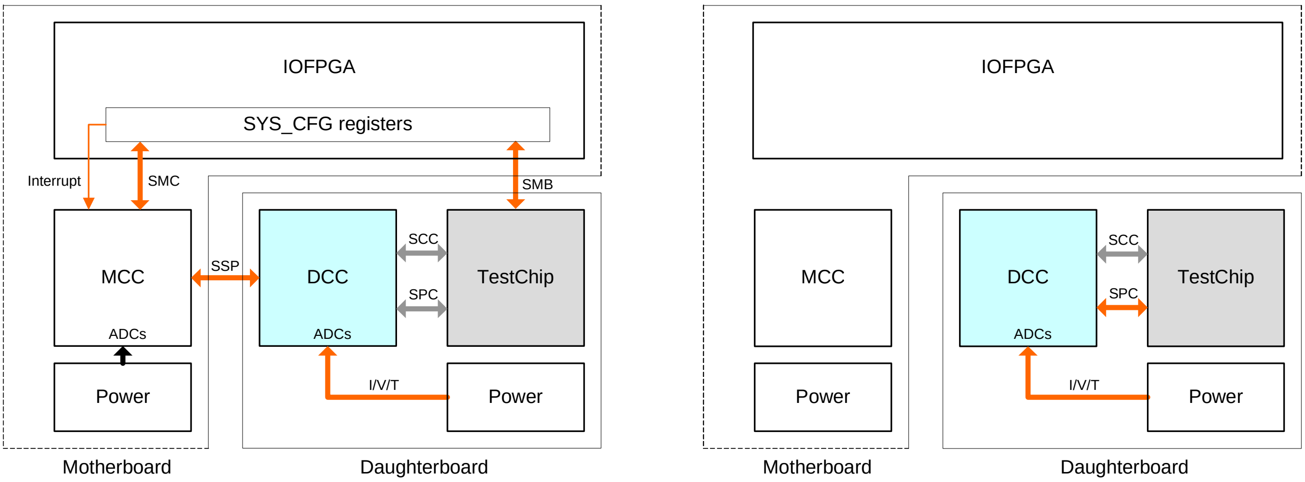

***Figure 6*, Motherboard SYS_CFG (left) and SPC SYS_CFG (right) interfaces**

The SPC based SYS_CFG interface follows the register format of the Motherboard SYS_CFG interface as far as possible using the SPC registers shown in *Table 2*.

Name **Address**	**Width** **Type**	**Reset** **Value**	**Name**	**Description**
PCFGREG4 0xB10	32 R/W	32'd0	SPC_SYS_CFGCTRL	Control register
PCFGREG5 0xB14	1 RO	1'b0	SPC_SYS_CFGREQ	Request in progress: Set when test-chip writes to SPC_SYS_CFGCTRL Cleared when DCC reads SPC_SYS_CFGCTRL
PCFGREG6 0xB18	32 R/W	32'd0	PWC_STATUS	Status (only valid after PWC_FLAG is set): CA15 DVFS request: 0x0001 complete, 0x0002 error CA7 DVFS request: 0x0010 complete, 0x0020 error SPC_SYS_CFG request: 0x0100 complete, 0x0200 error
PCFGREG7 0xB1C	1 RO	1'b0	PWC_FLAG	Request complete: Set when DCC completes DVFS or SPC_SYS_CFG request Cleared when test-chip reads PWC_STATUS This flag also sets GIC400 interrupt 95
PCFGREG28 0xB70	32 R/W	32'd0	SPC_SYS_CFGWDATA	Write data (test-chip -> DCC)
PCFGREG29 0xB74	32 R/W	32'd0	SPC_SYS_CFGRDATA	Read data (DCC -> test-chip)

***Table 2*, SPC_SYS_CFG registers**

Note: to enable the SPC_SYS_CFG interface SCC: 0xC00 bits [0] and [2] must be set in the V2P-CA15_A7 board.txt file.

Bits	31	30	29:26	25:20	19:12	11:0
Function	Start	Write	Reserved	Function	Reserved	Device
Example	1	0	0	2	0	1

***Table 3*, SPC_SYS_CFGCTRL register format**

The SPC_SYS_CFGCTRL register format is shown in Table 3. The *Device* field determines the number of the device to be accessed, the example shows device 1. The *Function* field determines the function to be accessed, the example shows 2=Voltage. When *Write* is set to 1 the access is a write from the test-chip to the DCC (using SPC_SYS_CFGWDATA), When *Write* is set to 0 the access is a read from the DCC to the test-chip (using SPC_SYS_CFGRDATA). *Start* should always be set to 1. The example shows a Read from Voltage Device 1 (CA7 core voltage).

For more information on the Device and Function fields please refer to the SYS_CFG interface in the [1] Motherboard Express uATX TRM and the [2] CoreTile Express A15x2 A7x3 TRM.

Example code to perform a read from the SPC_SYS_CFG interface using the example control value is shown:

```
-----------------------------------------------------------------------------

    // SPC_SYS_CFG registers
    #define SPC_SYS_CFGCTRL  0x7FFF0B10
    #define SPC_SYS_CFGREQ   0x7FFF0B14
    #define PWC_STATUS       0x7FFF0B18
    #define PWC_FLAG         0x7FFF0B1C
    #define SPC_SYS_CFGWDATA 0x7FFF0B70
    #define SPC_SYS_CFGRDATA 0x7FFF0B74


    unsigned int din, stat;

    // Clear any requests in progress or completed requests
    if ((*((volatile unsigned int *)SPC_SYS_CFGREQ) & 0x00000001) ||
        (*((volatile unsigned int *)PWC_FLAG) & 0x00000001))
    {
        // Wait for complete flag (or interrupt 95)
        while (!(*((volatile unsigned int *)PWC_FLAG) & 0x00000001))
            continue;

        // Handle any outstanding requests here

        // Clear the complete flag
        stat = *((volatile unsigned int *)PWC_STATUS);
    }

    // Set the data (only required for writes)
    *((volatile unsigned int *)SPC_SYS_CFGWDATA) = 0x00000000;

    // Set the control value
    *((volatile unsigned int *)SPC_SYS_CFGCTRL) = 0x80200001;

    // Wait for complete flag (or interrupt 95)
    while (!(*((volatile unsigned int *)PWC_FLAG) & 0x00000001))
        continue;

    // Read value back
    din = *((volatile unsigned int *)SPC_SYS_CFGRDATA);

    // Read the status and clear the complete flag
    stat = *((volatile unsigned int *)PWC_STATUS);

    // Display value and status
    printf ("Data:0x%08X, Status:0x%08X\n", din, stat);

-----------------------------------------------------------------------------
```

A comparison of the performance of the motherboard and SPC SYS_CFG interfaces can be found in Table 9.

9 SPC register interface

The SPC register interface uses the same physical interface to the DCC as the SCC register interface. The SPC registers are however mapped to a different address range than the SCC registers. All SCC and SPC registers are directly accessible by the test-chip CPUs at base address 0x00_7FFF_0000 with the exception of the Virtual SPC registers which are loaded from the board.txt file and can only be accessed through motherboard SYS_CFGCTRL register interface.

Interface	Address	Description
SCC	0x000-0x0FC	Static SoC configuration signals
SCC	0x100-0x1FC	Clock control and status signals
SCC	0x200-0x2FC	EMA RAM pin control
SCC	0x300-0x3FC	Process monitor
SCC	0x400-0x4FC	CA15 configuration
SCC	0x500-0x5FC	CA7 configuration
SCC	0x600-0x6FC	CCI400 configuration
SCC	0x700	System Information
SPC	**0xB00-0xB1C**	**DVFS control**
SPC	**0xB20-0xB2C**	**Wake up control**
SPC	**0xB30-0xB44**	**Power down control**
SYS_CFGCTRL	**0xC00-0xC0F**	**Virtual SPC control registers in the DCC**
SYS_CFGCTRL	**0xC10-0xC2C**	**Virtual SPC CA15 performance registers in the DCC**
SYS_CFGCTRL	**0xC30-0xC4C**	**Virtual SPC CA7 performance registers in the DCC**
SCC	0xFF4-0xFFF	System Information

Table 4, **SCC and SPC registers**

9.1 SCC and SPC APB access and control

The SCC and SPC registers can be read at any time through both the serial interface to the DCC and the APB interface to the test-chip CPUs. However write control is only assigned to one interface at a time on register-by-register assignment. Care must therefore be taken when writing to these registers.

While write control of individual registers can be passed between the two interfaces this is not recommended and could result in stale values being read as the serial and APB interfaces have separate write registers. The register on the interface that is currently active is always the one read by both the serial and APB interfaces.

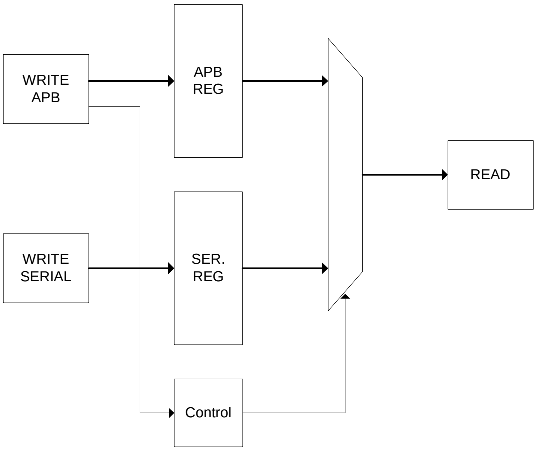

Figure 7, **SCC and SPC access and control**

Before reset the SCC and SPC registers are assigned default values which may be overwritten by the board.txt file settings. After reset access control to the SCC and SPC registers is normally fixed and assigned such that a register is always written by the DCC (serial) or test-chip CPUs (APB) interfaces. *Table 5* shows the default access control assignment of SCC and SPC registers after reset.

Register	Description	Assignment	Register	Description	Assignment
CFGREG0	SMC CS0/1	APB	PCFGREG0	PERF_LVL_CA15	APB
CFGREG1	SMC CS2/3	APB	PCFGREG1	PERF_REQ_CA15	APB
CFGREG2	SMC CS4/5	APB	PCFGREG2	PERF_LVL_CA7	APB
CFGREG3	SMC CS6/7	APB	PCFGREG3	PERF_REQ_CA7	APB
CFGREG4	Misc config 0	APB	PCFGREG4	SPC_SYS_CFGCTRL	APB
CFGREG5	DMA boot address	APB	PCFGREG5	SPC_SYS_CFGREQ	APB
CFGREG6	Reset control	SERIAL	PCFGREG6	PWC_STATUS	SERIAL
CFGREG7	DAP ROM Target ID	APB	PCFGREG7	PWC_FLAG	APB
CFGREG8	DAP ROM Inst. ID	APB	PCFGREG8	COMMS_DATA	SERIAL
CFGREG9	Ring oscillator	APB	PCFGREG9	WAKE_INT_MASK	APB
CFGREG10	Test status	APB	PCFGREG10	WAKE_INT_RAW	APB
CFGREG11	Clock control	SERIAL	PCFGREG11	WAKE_INT_STAT	APB
CFGREG12	Clock status	APB	PCFGREG12	CA15_PWRDN_EN	APB
CFGREG13	SYS PLL control	APB	PCFGREG13	CA7_PWRDN_EN	APB
CFGREG14	SYS PLL value	APB	PCFGREG14	CA7_A15_ISOLATE	SERIAL
CFGREG15	DDR PLL control	APB	PCFGREG15	STANDBYWFI_STAT	APB
CFGREG16	DDR PLL value	APB	PCFGREG16	CA15_CACTIVE	APB
CFGREG17	HDLCD PLL control	APB	PCFGREG17	CA15_PWRDNREQ	SERIAL
CFGREG18	HDLCD PLL value	APB	PCFGREG18	CA15_PWRDNACK	APB
CFGREG19	CA15_0 PLL control	APB	PCFGREG19	CA7_CACTIVE	APB
CFGREG20	CA15_0 PLL value	APB	PCFGREG20	CA7_PWRDNREQ	SERIAL
CFGREG21	CA15_1 PLL control	APB	PCFGREG21	CA7_PWRDNACK	APB
CFGREG22	CA15_1 PLL value	APB	PCFGREG22	CA15_RESET_HOLD	APB
CFGREG23	CA7_0 PLL control	APB	PCFGREG23	CA7_RESET_HOLD	APB
CFGREG24	CA7_0 PLL value	APB	PCFGREG24	CA15_RESET_STAT	APB
CFGREG25	CA7_1 PLL control	APB	PCFGREG25	CA7_RESET_STAT	APB
CFGREG26	CA7_1 PLL value	APB	PCFGREG26	EAG_BX_ADDR0	APB
CFGREG27	EMA control 0	APB	PCFGREG27	EAG_BX_ADDR1	APB
CFGREG28	EMA control 1	APB	PCFGREG28	SPC_SYS_CFGWDATA	APB
CFGREG29	EMA control 2	APB	PCFGREG29	SPC_SYS_CFGRDATA	SERIAL
CFGREG30	EMA control 3	APB	PCFGREG30	KF_BX_ADDR0	APB
CFGREG31	EMA control 4	APB	PCFGREG31	KF_BX_ADDR1	APB
CFGREG32	EMA control 5	APB	PCFGREG32	KF_BX_ADDR2	APB
CFGREG33	EMA control 6	APB	PCFGREG33	-	APB
CFGREG34	Process Monitor 0	APB	PCFGREG34	SPC_WARM_RESET_REG	APB
CFGREG35	Process Monitor 1	APB			
CFGREG36	Process Monitor 2	APB			
CFGREG37	Process Monitor 3	APB			
CFGREG38	Process Monitor 4	APB			
CFGREG39	Process Monitor 5	APB			
CFGREG40	Process Monitor 6	APB			
CFGREG41	CA15 config 0	APB			
CFGREG42	CA15 config 1	APB			
CFGREG43	CA7 config 0	APB			
CFGREG44	CA7 config 1	APB			
CFGREG45	CCI400 config 0	APB			
CFGREG46	CCI400 config 1	APB			
CFGREG47	CCI400 config 2	APB			
CFGREG48	System info	APB			

***Table 5*, SCC and SPC register control assignement**

It is not recommended for the test-chip CPUs writes to registers under DCC control as this will prevent writes from DCC updating the active register value. However a mechanism does exist to pass control back to the DCC which involves writing to the SCC APB Control Clear register at offset 0xFF4. The value written is 0xA50F5RRR where RRR is the address of the register to be reverted back to the DCC. Care must be taken to avoid stale values become active when the access control is changed.

9.2 SPC registers

Name **Address**	**Width** **Type**	**Reset Value**	**Name**	**Description**
PCFGREG0 0xB00	32 R/W	0x000000FF Set to 0x00000005 in board.txt	PERF_LVL_CA15	Performance level requested for the CA15 cluster. This register is updated by the CPU when a new performance level is required. Once this register is updated, the PERF_REQ_CA15 output is asserted until the register is read via the Serial Interface. 0x000 - 0x007 = min to max
PCFGREG1 0xB04	1 RO	1’b0	PERF_REQ_CA15	This register mirrors the behaviour of the PERF_REQ_CA15 output. Set when PERF_LVL_CA15 register is written by the CPU and cleared when the PERF_LVL_CA15 register is read by the DCC.
PCFGREG2 0xB08	32 R/W	0x000000FF Set to 0x00000005 in board.txt	PERF_LVL_CA7	Performance level requested for the CA7 cluster. This register is updated by the CPU when a new performance level is required. Once this register is updated, the PERF_REQ_CA7 output is asserted until the register is read via the Serial Interface. 0x000 - 0x007 = min to max
PCFGREG3 0xB0C	1 RO	1’b0	PERF_REQ_CA7	This register mirrors the behaviour of the PERF_REQ_CA7 output. Set when PERF_LVL_CA7 register is written by the CPU and cleared when the PERF_LVL_CA7 register is read by the DCC.
PCFGREG4 0xB10	32 R/W	0x00000000	SPC_SYS_CFGCTRL	This register is used by the CPU to send SYS_CFG_CTRL requests to the DCC.
PCFGREG5 0xB14	1 RO	1’b0	SPC_SYS_CFGREQ	This register mirrors the SPC_SYS_CFGCTRL output behaviour. Set when SPC_SYS_CFGCTRL is written by the CPU. Cleared when the SPC_SYS_CFGCTRL is read by the DCC.
PCFGREG6 0xB18	32 R/W	0x00000000	PWC_STATUS	This register is used by the DCC to communicate status/faults to the test-chip. CA15 DVFS request: 0x0001 complete, 0x0002 error CA7 DVFS request: 0x0010 complete, 0x0020 error SPC_SYS_CFG request: 0x0100 complete, 0x0200 error
PCFGREG7 0xB1C	1 RO	1b0	PWC_FLAG	Set when PWC_STATUS register is updated through the SPC. Cleared when PWC_STATUS register is read by the CPU. PWC_FLAG is also connected to GIC400 interrupt input 95, and will interrupt the CPU when PWC_STATUS register is written by the DCC. The interrupt is cleared when the CPU reads the PWC_STATUS register.
PCFGREG8 0xB20	32 R/W	32’d0	-	Reserved
PCFGREG9 0xB24	12 R/W	12’d0	WAKE_INT_MASK	Deep shutdown wake-up mask register [11] GTIMER_INTR_MSK [10] SWDOG_INTR_MSK [9] FIQ_CA7_MSK[2] [8] FIQ_CA7_MSK[1] [7] FIQ_CA7_MSK[0] [6] IRQ_CA7_MSK[2] [5] IRQ_CA7_MSK[1] [4] IRQ_CA7_MSK[0] [3] FIQ_CA15_MSK[1] [2] FIQ_CA15_MSK[0] [1] IRQ_CA15_MSK[1] [0] IRQ_CA15_MSK[0]
PCFGREG10 0xB28	12 RO	1’b0, and then depends on interrupt status	WAKE_INT_RAW	Deep shutdown wake up sources including the GIC400 outputs. All bits are active HIGH. [11] GTIMER_INTR [10] SWDOG_INTR [9] FIQ_CA7[2] [8] FIQ_CA7[1] [7] FIQ_CA7[0] [6] IRQ_CA7[2] [5] IRQ_CA7[1] [4] IRQ_CA7[0] [3] FIQ_CA15[1] [2] FIQ_CA15[0] [1] IRQ_CA15[1] [0] IRQ_CA15[0]

9.2.1 SPC registers continued

Name Address	Width Type	Reset Value	Name	Description
PCFGREG11 0xB2C	12 RO	1'b0, and then depends on interrupt status	WAKE_INT_STAT	Deep shutdown masked status register [11] GTIMER_INTR [10] SWDOG_INTR [9] FIQ_CA7[2] [8] FIQ_CA7[1] [7] FIQ_CA7[0] [6] IRQ_CA7[2] [5] IRQ_CA7[1] [4] IRQ_CA7[0] [3] FIQ_CA15[1] [2] FIQ_CA15[0] [1] IRQ_CA15[1] [0] IRQ_CA15[0] These masked interrupt registers are OR-ed together to generate the WAKE_UP output
PCFGREG12 0xB30	1 R/W	1'b0	CA15_PWRDN_EN	Allows the CA15 cluster to go into shutdown when STANDBYWFI[1:0] and STANDBYWFIL2 from the CA15 cluster are all asserted. Fine grain power down of CPU is not supported since the test-chip does not support internal power switches.
PCFGREG13 0xB34	1 R/W	1'b0	CA7_PWRDN_EN	Allows CA7 cluster to go into shutdown when STANDBYWFI[1:0] and STANDBYWFIL2 from the CA7 cluster are all asserted. Fine grain power down of CPU is not supported since the test-chip does not support internal power switches.
PCFGREG14 0xB38	2 R/W	1'b0 1'b0	CA7_ISOLATE CA15_ISOLATE	[1] Clamp CA7 cluster outputs [0] Clamp CA15 cluster outputs
PCFGREG15 0xB3C	7 RO	7'd0	STANDBYWFI_STAT	[6] CA7 STANDBYWFIL2 [5] CA7 STANDBYWFI[2] [4] CA7 STANDBYWFI[1] [3] CA7 STANDBYWFI[0] [2] CA15 STANDBYWFIL2 [1] CA15 STANDBYWFI[1] [0] CA15 STANDBYWFI[0]
PCFGREG16 0xB40	4 RO	4'd0	CA15_CACTIVE	[3] cactive from ADB slave slice on CA15 ACP [2] cactive from ADB master slice on CA15 ACP [1] cactive from ADB slave slice on CA15 master [0] cactive from ADB master slice on CA15 master
PCFGREG17 0xB44	4 RW	4'd0	CA15_PWRDNREQ	Set this register to request power down of the ADB-400 bridges on CA15 ACP, master interfaces and bridges on the ATB interfaces. [4] Debug disable APB access to CA15 [3] Power down req to CA15 ATB1 bridge [2] Power down req to CA15 ATB0 bridge [1] Power down req to CA15 ACP ADB400 [0] Power down req to CA15 master ADB400
PCFGREG18 0xB48	4 RO	4'd0	CA15_PWRDNACK	Power-down acknowledge from the ADB400 bridges on CA15 ACP and master interfaces. [3] Power down ack from CA15 ATB1 bridge [2] Power down ack from CA15 ATB0 bridge [1] Power down ack from CA15 ACP ADB400 [0] Power down ack from CA15 Master ADB400
PCFGREG19 0xB4C	2 RO	2'd0	CA7_CACTIVE	[1] cactive from ADB slave slice on CA7 master [0] cactive from ADB master slice on CA7 master

9.2.2 SPC registers continued

Name Address	Width Type	Reset Value	Name	Description
PCFGREG20 0xB50	4 RW	4’d0	CA7_PWRDNREQ	Set this register to request power down of the ADB-400 bridge on CA7 master interface and bridges on ATB interface [4] Debug disable APB access to CA7 [3] Power down req to CA7 ATB bridge 2 [2] Power down req to CA7 ATB bridge 1 [1] Power down req to CA7 ATB bridge 0 [0] Power down req to CA7 ADB400
PCFGREG21 0xB54	4 RO	4’d0	CA7_PWRDNACK	[3] Power down ack to CA7 ATB bridge 2 [2] Power down ack to CA7 ATB bridge 1 [1] Power down ack to CA7 ATB bridge 0 [0] Power down ack to CA7 ADB400
PCFGREG22 0xB58	5 RW	5’d0	CA15_RESET_HOLD	Reset hold register allows the individual CPUs or the whole cluster to be put in reset state, once it is in WFI state. [8] Assert CA15 cluster reset on STANDBYWFIL2 [5:4] Assert CA15 nCPUPORESET[1:0] on STANDBYWFI[1:0] [1:0] Assert CA15 nCPURESET[1:0] on STANDBYWFI[1:0]
PCFGREG23 0xB5C	7 RW	7’d0	CA7_RESET_HOLD	Reset hold register allows the individual CPUs or the whole cluster to be put in reset state, once it is in WFI state. [8] Assert CA7 cluster reset on STANDBYWFIL2 [6:4] Assert CA7 nCPUPORESET[2:0] on STANDBYWFI[2:0] [2:0] Assert CA7 nCPURESET[2:0] on STANDBYWFI[2:0]
PCFGREG24 0xB60	5 RO	5’d0	CA15_RESET_STAT	Returns the reset status of CA15 cluster. [8] CA15 cluster is held in reset [5:4] CA15 nCPUPORESET[1:0] is asserted [1:0] CA15 nCPURESET[1:0] is asserted
PCFGREG25 0xB64	7 RO	7’d0	CA7_RESET_STAT	Returns the reset status of CA7 cluster [8] CA7 cluster is held in reset [6:4] CA7 nCPUPORESET[2:0] is asserted [2:0] CA7 nCPURESET[2:0] is asserted
PCFGREG26 0xB68	32 R/W	32’d0	EAG_BX_ADDR0	Branch address for CA15 CPU0 after warm reset
PCFGREG27 0xB6C	32 R/W	32’d0	EAG_BX_ADDR1	Branch address for CA15 CPU1 after warm reset
PCFGREG28 0xB70	32 R/W	32’d0	SPC_SYS_CFGWDATA	Write data to the SPC based SYS_CFG interface, test-chip to DCC.
PCFGREG29 0xB74	32 R/W	32’d0	SPC_SYS_CFGRDATA	Read data from the SPC based SYS_CFG interface, DCC to test-chip.
PCFGREG30 0xB78	32 R/W	32’d0	KF_BX_ADDR0	Branch address for CA7 CPU0 after warm reset
PCFGREG31 0xB7C	32 R/W	32’d0	KF_BX_ADDR1	Branch address for CA7 CPU1 after warm reset
PCFGREG32 0xB80	32 R/W	32’d0	KF_BX_ADDR2	Branch address for CA7 CPU2 after warm reset
PCFGREG33 0xB84	32 R/W	32’d0	-	Reserved
PCFGREG34 0xB88	32 R/W	32’d0	SPC_WARM_RESET_REG	Used by application software to determine between warm and cold boot

9.3 Virtual SPC register

Name Address	Width Type	Reset Value	Name	Description
VPCFGREG0 0xC00	32 R/W	0x00000007	CONTROL	[0] Enable Power Management interface [1] Enable debugger support (cluster PSUs always ON) [2] Enable SPC SYSCFG interface [3] Reserved
VPCFGREG1 0xC04	32 R/W	0x060E0356	LATENCY	[15:0] Maximum DVFS interface latency in uS [31:16] Maximum Power Up latency in uS
VPCFGREG2 0xC08	32 R/W	0x00000000	-	RESERVED
VPCFGREG3 0xC0C	32 R/W	0x00000000	-	RESERVED
VPCFGREG4 0xC10	32 R/W	0x00000000	CA15_PERFVAL0	CA15 performance value 0
VPCFGREG5 0xC14	32 R/W	0x00000000	CA15_PERFVAL1	CA15 performance value 1
VPCFGREG6 0xC18	32 R/W	0x00000000	CA15_PERFVAL2	CA15 performance value 2
VPCFGREG7 0xC1C	32 R/W	0x00000000	CA15_PERFVAL3	CA15 performance value 3
VPCFGREG8 0xC20	32 R/W	0x00000000	CA15_PERFVAL4	CA15 performance value 4
VPCFGREG9 0xC24	32 R/W	0x00000000	CA15_PERFVAL5	CA15 performance value 5
VPCFGREG10 0xC28	32 R/W	0x00000000	CA15_PERFVAL6	CA15 performance value 6
VPCFGREG11 0xC2C	32 R/W	0x00000000	CA15_PERFVAL7	CA15 performance value 7
VPCFGREG12 0xC30	32 R/W	0x00000000	CA7_PERFVAL0	CA7 performance value 0
VPCFGREG13 0xC34	32 R/W	0x00000000	CA7_PERFVAL1	CA7 performance value 1
VPCFGREG14 0xC38	32 R/W	0x00000000	CA7_PERFVAL2	CA7 performance value 2
VPCFGREG15 0xC3C	32 R/W	0x00000000	CA7_PERFVAL3	CA7 performance value 3
VPCFGREG16 0xC40	32 R/W	0x00000000	CA7_PERFVAL4	CA7 performance value 4
VPCFGREG17 0xC44	32 R/W	0x00000000	CA7_PERFVAL5	CA7 performance value 5
VPCFGREG18 0xC48	32 R/W	0x00000000	CA7_PERFVAL6	CA7 performance value 6
VPCFGREG19 0xC4C	32 R/W	0x00000000	CA7_PERFVAL7	CA7 performance value 7

***Table 6*, Virtual SPC registers**

The Virtual SPC registers are stored within the DCC and loaded from values in the board.txt file, they are not physical registers within the test-chip. These registers can be read at run time through the motherboard SYS_CFGCTRL interface.

VPCFGREG0 CONTROL[0] enables the V2P-CA15_A7 power management interface (DVFS and Power Switching). When disabled (1'b0) the Virtual SPC registers, Power Management and SPC SYSCFG interfaces are disabled. In this mode the test-chip SPC registers are ignored by the DCC and the CA15 and CA7 clocks are controlled by the OSCCLK[3:0] settings in the board.txt file, PSU[1:0] default to 0.9V.

VPCFGREG0 CONTROL[1] enables debug support during cluster power down. When enabled the cluster PSUs (PSU0 and PSU1) do not actually switch OFF when cluster power down is requested allowing debug-through-power-down. However for accurate power measurement this bit should be cleared (disabling debugger support).

VPCFGREG0 CONTROL[2] enables the SPC based SYS_CFGCTRL interface see section.

The LATENCY value (VPCFGREG1) is used to indicate to the maximum DVFS and Power Up latency values. The default setting relates to the values shown in *Table 8*. The values are not used by the DCC firmware.

10 Electrical specifications

Symbol	Parameter	Min.	Max.	Tolerance	Unit
V_{PSU0}	CA15 power supply voltage see section 4.2	0.8	1.05	+/-2%	V
V_{PSU1}	CA7 power supply voltage see section 4.2	0.8	1.05	+/-2%	V
V_{PSU2}	SoC power supply voltage	0.9	0.9	+/-2%	V
I_{PSU0}	CA15 power supply current	0	10	-	A
I_{PSU1}	CA7 power supply current	0	3	-	A
I_{PSU2}	SoC power supply current	0	3	-	A
I_{mon0}	CA15 power supply current monitor - 12bits	0	10	+/-2%	A
I_{mon1}	CA7 power supply current monitor - 12bits	0	3	+/-2%	A
I_{mon2}	SoC power supply current monitor - 12bits	0	3	+/-2%	A
I_{energ0}	CA15 on board energy meter - 10KS/sec	0	64 bits	+/-3%	J
I_{energ1}	CA7 on board energy meter – 10KS/sec	0	64 bits	+/-3%	J
F_{CA15}	CA15 clock frequency see section 4.2	2	1332	-	MHz
F_{CA7}	CA7 clock frequency see section 4.2	2	1065	-	MHz
F_{SoC}	SoC clock frequency see section 4.2	2	666	-	MHz
T_{onn15}	CA15 PSU on time (<=0.9V)	-	160	-	uS
T_{onn7}	CA7 PSU on time (<=0.9V)	-	480	-	uS
T_{ono15}	CA15 PSU on time (>0.9V)	-	440	-	uS
T_{ono7}	CA7 PSU on time (>0.9V)	-	760	-	uS
T_{off15}	CA15 PSU off time (<0.05V)	-	800	-	uS
T_{off7}	CA7 PSU off time (<0.05V)	-	2,000	-	uS

Table 7**, Electrical specifications**

10.1 Timing specifications

The DVFS and Power Switching firmware in the DCC is optimised to achieve minimum latency for all operating conditions, this is shown in *Table 8*. Because the test-chip has two OSCCLKs per cluster (and 4-way mux) these are programmed to the frequencies defined by the last two performance levels. So if only two performance levels are currently in use switching between the two levels (using 4-way mux) is relatively efficient.

For DVFS the optimum conditions are achieved when the two OSCCLKS (per cluster) have been set and the DVFS software is only switching between these two operating points – $t_{ds.}$

For Power Switching the optimum conditions are achieved when powering Down and then Up to Normal Voltage (<=0.9V) and to one of the currently performance points. – t_{pd} and $t_{pu.}$

Symbol	Parameter	Conditions	CA15	CA7	Unit
t_{ds}	DVFS switching	No OSCCLK or Voltage change	150	144	uS
t_{dc}	DVFS OSCCLK change	No Voltage change	800	800	uS
t_{dvu}	DVFS Voltage change up	No OSCCLK change	450	432	uS
t_{dvd}	DVFS Voltage change down	No OSCCLK change	160	150	uS
t_{dcv}	DVFS OSCCLK and Voltage change	Voltage up or down	854	846	uS
t_{pd}	Power down	Any Voltage or OSCCLK	135	135	uS
t_{pu}	Power up	No OSCCLK or Voltage change (<=0.9V)	500	767	uS
t_{pup}	Power up to different performance level	No OSCCLK or Voltage change (<=0.9V)	550	815	uS
t_{puc}	Power up and OSCCLK change	No Voltage change (<=0.9V)	1,260	1,510	uS
t_{puv}	Power up and Voltage change	No OSCCLK change (<=0.9V)	850	1,150	uS
t_{pucv}	Power up, OSCCLK and Voltage change	(<=0.9V)	1,310	1,550	uS
t_{puo}	Power up to Overdrive	No OSCCLK change (>0.9V)	810	1,090	uS
t_{puco}	Power up, OSCCLK and Overdrive	OSCCLK (>0.9V)	1,510	1,760	uS

***Table 8,* Timing specifications**

10.2 Timing specifications SYS_CFG interfaces

A comparison of the motherboard and SPC SYS_CFG interfaces is shown in Table 9

		Motherboard only		Motherboard to DCC		SPC to DCC		
Ref	**Function**	**Read**	**Write**	**Read**	**Write**	**Read**	**Write**	**Unit**
1	Transfer only	20	20	233	233	100	100	uS
2	Frequency	233	3,135	238	820	100	700	uS
3	Voltage	1,090		247	239	123	100	uS
4	Current			245		125		uS
5	Temperature	78		298		198		uS
6	SCC			232	237	114	100	uS
7	MUXFPGA	28	28					uS
8	DVIMODE	10,639	10,676					uS
9	Power			297		200		uS
10	Energy			233	233	100	100	uS

***Table 9*, motherboard and SPC SYS_CFG interface timing**