# Low Power Support

## NetSpeed Power Management Architecture

The power management architecture is comprised of 3 layers: the PMU – SoC power control circuitry provided by the user, the NetSpeed Power Supervisor (NSPS), and NoC itself.



Figure 25 NetSpeed Power Management Architecture

### PMU (SoC Power Management Unit)

SoC power management logic, the PMU, is responsible for coordinating power state changes amongst all the elements of the device, including the NoC, but also including the hosts that interact with the NoC and logic beyond. The PMU is provided by the user and generates all power gating controls (isolation, power gate enable, etc.) – NetSpeed logic does not generate these. The PMU is generally expected to exist in an “always on” power domain.

The PMU coordinates power state changes with the NoC via an AMBA Q-channel interface as defined in the ARM Low Power Interface Specification.

### NetSpeed Power Supervisor

The NetSpeed Power Supervisor (NSPS) abstracts away NoC microarchitectural details to present a simple well-defined interface to the PMU for communication of power intent. For each power domain in the NoC, it maintains a Power Domain Finite State Machine (PD FSM) that provides high-level sequencing of the operations required for power removal and power restoration. This FSM drives the Q-channel interface (QREQn/QACCEPTn/QDENY/QACTIVE) to the PMU in conjunction with driving signals to elements in the NoC, needed to coordinate power sequencing activity. It is mapped into a power domain that is always on with respect to the rest of the power domains in the NoC, which could be the same power domain as the PMU, possibly co-located with it.

The NS Power Supervisor also has aggregation logic to combine acknowledgment signals and wake request signals returned from NoC elements. This logic may be distributed in the design to minimize wiring impact.

### NoC Element Power Management Logic:

Logic in each of the NoC elements implements required power management functionality, including supporting coordination with the NSPS (e.g., fencing and draining, idle status and sleep ack, etc.). This logic is located within each NoC element (bridges and routers).

## Power Domains and Relationships to Clock and Voltage Domains

Power domains are the logical construct through which power control is communicated between the PMU and the NoC. They nominally describe the boundaries of regions that share common status regarding clock gating (at the power domain level) and power gating. The sections below describe the interactions between power domains, clock domains and voltage domains.

### Clock Domains

Clock domains may span power domain boundaries, as long as all the power domains have a common voltage domain. Clock domains may not span voltage domain boundaries. However, where clock gating is intended at any level above an individual NoC element (e.g., routers and bridges), one or more power domains must be defined that in aggregate match the boundary of the gated clock domain. For each clock domain, a distinct clock input pin is provided at the NoC interface for each power domain that it spans. A clock gate and corresponding enable may be inserted in front of these pins to create gated clock domains. The SoC PMU would request a power state change for the affected power domain(s) from the NSPS prior to changing the gated status of this clock domain.

In cases where coarse grained clock gating is intended above the NoC element level but at finer granularity than would be power gated (i.e., a power gating domain spans multiple clock gating domains), power domains must be defined for each gated clock domain, and when power gating the aggregating domain, the SoC PMU must request power state change for all the gated clock domains that are part of the aggregate power gating domain. The converse also applies.

### Multiple Voltage Domains

Orion LP supports the definition of multiple voltage domains. Each power domain is assigned to a single voltage domain, with the implication that a power domain may not span multiple voltage domains. However, a voltage domain may be divided into multiple power domains which have independent power state controls. In other words, power domains may not span voltage domains, but a voltage domain may contain multiple power domains. Where signals in the NoC cross voltage domain boundaries, NocStudio will infer appropriate level-shifting structures in the CPF collateral it generates.

Note as stated above, clock domains may not span voltage domains.

## Low Power Signaling Interface Between PMU and NSPS

Orion LP implements the Q-channel low-power signaling protocol according to the ARM AMBA Low Power Interface Specification. One Q-channel interface is provided for each power domain defined for the NoC. Four signals are involved: *QREQn\_<PD>* (driven by PMU), *QACCEPTn\_<PD>*, *QDENY\_<PD>* and *QACTIVE\_<PD>* (these latter 3 driven by NSPS). This interface allows the PMU to issue requests to the NSPS to safely remove and restore power to each power domain. This interface also allows the NSPS to issue requests to wake power domains. The PMU is ultimately responsible for waking domains, either in response to NSPS requests or other criteria.

Following is a brief description of how this interface is used to power down and subsequently power up a domain:

### Power Down Sequence



Figure 26 Power Down Waveform Sequence

1. Normal Operation: *QREQn\_<PD>*, *QACCEPTn\_<PD>* == 1, and *QDENY\_<PD>* == 0. *QACTIVE\_<PD>* may be in either state.
2. Power Down Request
   1. PMU drives *QREQn\_<PD>* low
   2. NSPS decides whether it can accept power down request or not:
      1. Will accept: drives *QACCEPTn\_<PD>* low
      2. Will not accept: drives *QDENY\_<PD>* high

Note: PMU must hold *QREQn\_<PD>* low until NSPS responds by asserting either *QACCEPTn\_<PD>* or *QDENY\_<PD>*, and it must return to normal operation by raising *QREQn\_<PD>* high and waiting for *QACCEPTn\_<PD>* to go high or *QDENY\_<PD>* to go low before initiating a new power down request (for this domain).

1. Powered Down: *QREQn\_<PD>*, *QACCEPTn\_<PD>*, *QDENY\_<PD>* and *QACTIVE\_<PD>* all == 0.
   1. PMU removes power
      1. Asserts *iso\_en\_<PD>*
      2. Disables clocks
      3. Asserts *pwr\_shutoff\_<PD>*

### Power Up Sequence



Figure 27 Power Up Sequence Waveforms

1. Wake Request initiated by NSPS: drives *QACTIVE\_<PD>* high.
2. Power Up
   1. PMU restores power
      1. De-asserts *pwr\_shutoff\_<PD>*
      2. Enables clocks and asserts *reset\_n\_<PD>*
      3. De-asserts *iso\_en\_<PD>*
      4. De-asserts *reset\_n\_<PD>*
   2. PMU drives *QREQn\_<PD>* high.
   3. NSPS acks by driving *QACCEPTn\_<PD>* high when logic is safely restored for normal operation.
3. Normal Operation: *QREQn\_<PD>*, *QACCEPTn\_<PD>* == 1, and *QDENY\_<PD>* == 0. *QACTIVE\_<PD>* may be in either state.

## Fencing and Draining

Fencing and draining is a feature of NetSpeed power management that prevents any loss or corruption of transactions due to power state changes. In response to requests from the PMU to shut down one or more power domains (signaled by driving the associated QREQn signals low), the NSPS communicates with all master bridges to ensure 2 things:

1. No new transactions that require the power domain that is going down are allowed to enter the NoC – this is “fencing.”
2. Any transactions in progress at the time the QREQn power down request is received are monitored for completion prior to acknowledging the request – this is “draining.”

To support this feature, all master bridges constantly monitor the requested power status of all relevant power domains via signals driven by the NSPS. When a bridge observes a request to shut down, it initiates fencing and draining for that power domain. Fencing begins immediately, and an acknowledgement signal to the NSPS is asserted only when all outstanding transactions dependent upon that power domain have completed.

The address look-up tables in the master bridges include information about which power domains are required to be in the active state for a given transaction request to be completed successfully. This information, combined with the power status information, is used to make a dynamic decision whether to fence or forward each newly arriving transaction on the host interface.

### Fencing Behavior

When a transaction is fenced, it will be handled in one of two ways depending on how the NoC is configured:

1. DECERR response: completed immediately by the master bridge with a DECERR status.
2. Auto-Wake Request: transaction will be held in local storage in the master bridge while it signals NSPS to request any blocking power domains to be woken by asserting QACTIVE.
   1. Note: the PMU should take action to wake an auto-wake enabled power domain whenever QACTIVE is received for a domain that is powered down. Otherwise the transaction will remain blocked, which in turn blocks all further transactions on that channel (even those that do not have the same power domain dependencies).

The type of response is controlled for each master bridge via the bridge property *axi4m\_autowake\_enable* in NocStudio. When regbus is enabled, this setting appears in a register that may be dynamically configured.

In addition to the master bridge setting, NocStudio maintains a property for each power domain which indicates whether or not it is wakeable. This is set via an optional argument to the *add\_power\_domain* command, or it may also be updated via the *set\_power\_domain\_auto\_wakeup* command.

Auto-wake requests will be signaled by asserting QACTIVE\_<PD> for each power domain that is blocking a transaction only when both the following conditions exist:

1. Bridge property *axi4m\_autowake\_enable* is set true.
2. All power domains that are currently blocking the transaction are auto-wake enabled.

Correspondingly, if any power domain blocking the transaction is not wakeable, the transaction response is forced to become a DECERR, and QACTIVE\_<PD> is not asserted to wake any power domains.

### Fencing and QDENY

When a power down request (*QREQn\_<PD>* 1->0) is received for an auto-wake capable power domain, if there are any pending transactions in the NoC that depend on that domain, the NSPS will reject the power down request by asserting *QDENY\_<PD>* instead of initiating fencing and draining. The PMU must return Q-channel to the Q\_RUN state by raising *QREQn\_<PD>*, and it must wait for all activity that depends on the target power domain to complete before a power down request will be accepted (though it may retry repeatedly while waiting). This behavior holds regardless of the fencing response type configuration at the master bridges (*axi4m\_autowake\_enable*).

Note that *QACTIVE\_<PD>* is driven high whenever there is pending activity dependent upon a power domain. This may be used by the PMU during the Q\_RUN state to determine whether or not a power down request is likely to be accepted.



Figure 28 QDENY Waveform Sequence

## Low Power Signals

### NetSpeed Power Supervisor Interface to PMU

The interface between the NetSpeed Power Supervisor and the PMU follows the AMBA Low Power Interface Specification, specifically the Q-Channel defined there. It is defined to be asynchronous and capturing logic on both sides must properly synchronize the signals to a local clock.

One set of the signals is implemented per power domain. The interface will not change through NoC design iterations as long as the power domains are not changed.

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Source | Destination | Purpose |
| QREQn\_<PD> | PMU | NSPS | Inform NS logic of intent to remove/restore power from/to power domain <PD>. When transitioning 1->0, requests power-down, when transitioning 0->1, requests wake-up. |
| QACCEPTn\_<PD> | NSPS | PMU | Response to *QREQn\_<PD>*, acknowledges request for power down or power up. |
| QDENY\_<PD> | NSPS | PMU | Asserted instead of *QACCEPTn\_<PD>* to reject a power down request. |
| QACTIVE\_<PD> | NSPS | PMU | When powered down (Q\_STOPPED state - *QREQn\_<PD>* and *QACCEPTn\_<PD>* are both low, NSPS in SLEEP\_ACK state), *QACTIVE\_<PD>* is asserted to signal a wake-up request. In other states, it serves to indicate the idle status of NoC logic within the power domain. |

Table 1 NSPS to PMU Interface

### PMU to Power Domain Interface

The PMU (or related logic in the SoC) drives the following signals related to power management, one set of these signals per power domain.

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Source | Destination | Purpose |
| reset\_n\_<PD> | PMU | NoC elements | Per power domain reset. This is asserted to establish clean state following the first power-on event and subsequent power cycles for the power domain. Point to multipoint, subject to clock/voltage domain crossing treatment. |
| iso\_en\_<PD> | PMU | isolation clamps | Defined in ns\_soc\_ip.cpf, this controls isolation clamps that are inferred where outputs of the power domain connect to inputs that exist in other power domains. Clamping function is enabled when this signal is driven high. |
| pwr\_shutoff\_<PD> | PMU | power gates | Defined in ns\_soc\_ip.cpf, this controls power gates for the power domain. Power is removed when this signal is asserted high. |

Table 2 PMU to Power Domain Interface

## Bridge Logic in Host Power Domain

There are a couple of situations where a portion of the bridge logic must exist in the host’s power domain (as specified by the *power\_domain\_host* property of the bridge).

### AHBLM Compound Bridge

For the AHB Lite master bridge, protocol conversion logic translates the host interface protocol to AXI4 which is driven to an AXI4 master bridge. In this case, due to protocol restrictions described further in section 4.7, the conversion logic must live in the host’s power domain. When the bridge’s *power\_domain* property is set to a different power domain than the *power\_domain\_host* property, the boundary between these domains appears between the protocol converter and the AXI bridge.

### Voltage Domain Boundaries between Host and Bridge

Where a voltage domain boundary appears, NocStudio inserts a set of asynchronous clock crossing structures built from specially designed FIFOs. These FIFOs separate the read mux and associated logic into one voltage domain and the storage array and associated write strobe logic into the other voltage domain. Level shifters are properly inferred to allow safe signal crossing between the two voltage domains.

When a voltage boundary is created between host and bridge, one side of the voltage domain crossing structures must live in the host power domain. In the case of compound bridges, the voltage domain crossing structures are inserted between the protocol conversion logic and the AXI4 bridge. In all other cases, the voltage domain crossing structures are inserted between the host and bridge at the host interface to the bridge.

## AHB Lite Master Bridge (AHBLM)

The AHB protocol, due to its pipelined nature, requires the host interface of the AHBLM bridge to hold HREADY high whenever the interface is idle. This means the bridge must always accept any newly initiated transactions, so there is no protocol compliant way for AHBLM to hold off the host interface at a clean transaction boundary. Because of this limitation, the following restriction applies:

The host driving the AHBLM interface must guarantee that no new transactions are initiated any time the host power domain (*power\_domain\_host)* is not in the Q\_RUN state.

## Regbus Master and Tunnel

The regbus master bridge must be configured so that its power domain (*power\_domain*) and its host’s power domain (*power\_domain\_host*) must be the same. When regbus tunnel is deployed, the rbm/s and rbm/m instances must be configured such that their *power\_domain* and *power\_domain\_host* properties are all the same.

NocStudio automatically enforces these restrictions by updating all properties that must remain the same to take the new value whenever an update is made to any of the properties.

## Shared Interface Bridge

A shared interface bridge instance inherits its *power\_domain* and *power\_domain\_host* settings from the *power\_domain* setting of all the master bridges that feed into it, and that setting must be the same across all these bridges. NocStudio verifies this when mapping is executed, and it will flag an error if the *power\_domain* settings are not consistent.

## Always On Power Domains

Always on power domains are those for which the *power\_domain\_always\_on* property is set to “yes.” The fundamental expectation is that logic in such power domains is powered before or coincident with other power domains in the NoC (i.e., at initial start-up or boot time), and that it remains powered continuously during operation of the device. They are defined in CPF power intent files so that signal transitions between them and other power domains can be recognized and properly handled. However, these power domains do not have power gating (*pwr\_shutoff\_<PD>*) or isolation (*iso\_en\_<PD>*) logic or signaling inferred in the CPF files as such logic is unnecessary. Q-channel interfaces and the associated NSPS power control logic is not present for always on power domains.

### Clock Gated Only Power Domains

Clock gated only power domains are always on power domains that also have the *power\_domain\_force\_q\_channel* property set to “yes.” These domains are treated like other always on power domains in CPF files, so they do not have power shutoff or isolation circuitry, but a Q-channel interface and associated NSPS power control logic is provided which allows the PMU to switch these domains between active (Q\_RUN) and quiet (Q\_STOPPED) states. Fencing and draining is implemented for these power domains, allowing clocks to be safely gated when they are in the Q\_STOPPED state.

## Restrictions

* User regbus bridges are not currently fully supported in low power mode
* Multi-voltage: AHBLM master bridge, APB & AHB slave bridges do not implement multi-voltage support.
* Support has not been implemented for:
  + DVFS.
  + UPF power intent format
    - Impacts LP simulation and synthesis with Synopsys tools.