1 HOMER

The Hardware Offload Microcode Engine Region (HOMER) is a 4MB region allocated during IPL on a per chip basis for hold the executable and data content for Power Management and other functions.

1.1 Overview

Figure 1.1 depicts the the overall layout of the HOMER memory region.

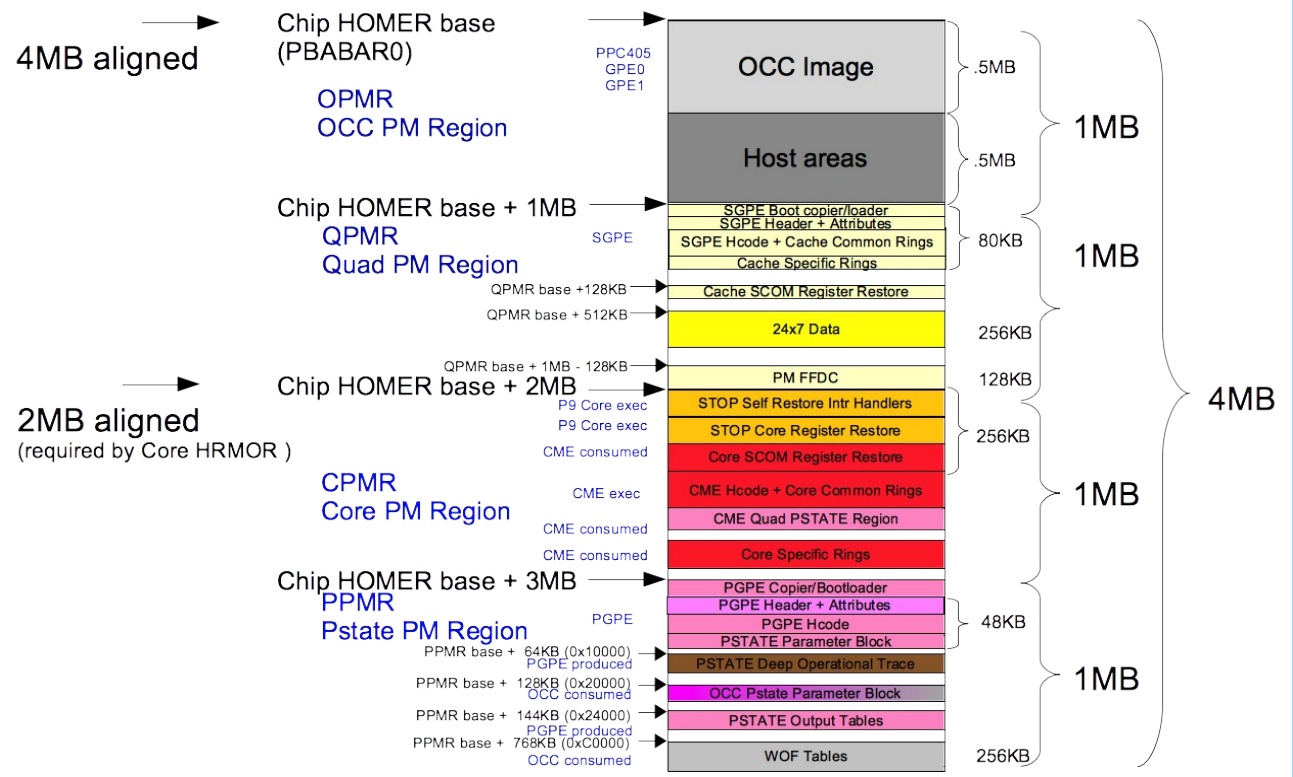

Figure 1.1: HOMER Overview Layout

1.2 OCC PM Region (OPMR)

This regions contains the OCC firmware image which includes the code that runs on PPC405 and GPE0 and GPE1. It is defined by the OCC firmware team and loaded by Hostboot. The Host area is use by the OCC firmware to communicate with Host software (OPAL, Linux) and is also defined by the OCC firmware team.

This document provides no further information on this section.

1.3 Quad PM Region (QPMR)

This region contains the code that runs on the SGPE including STOP and 24x7 functions.

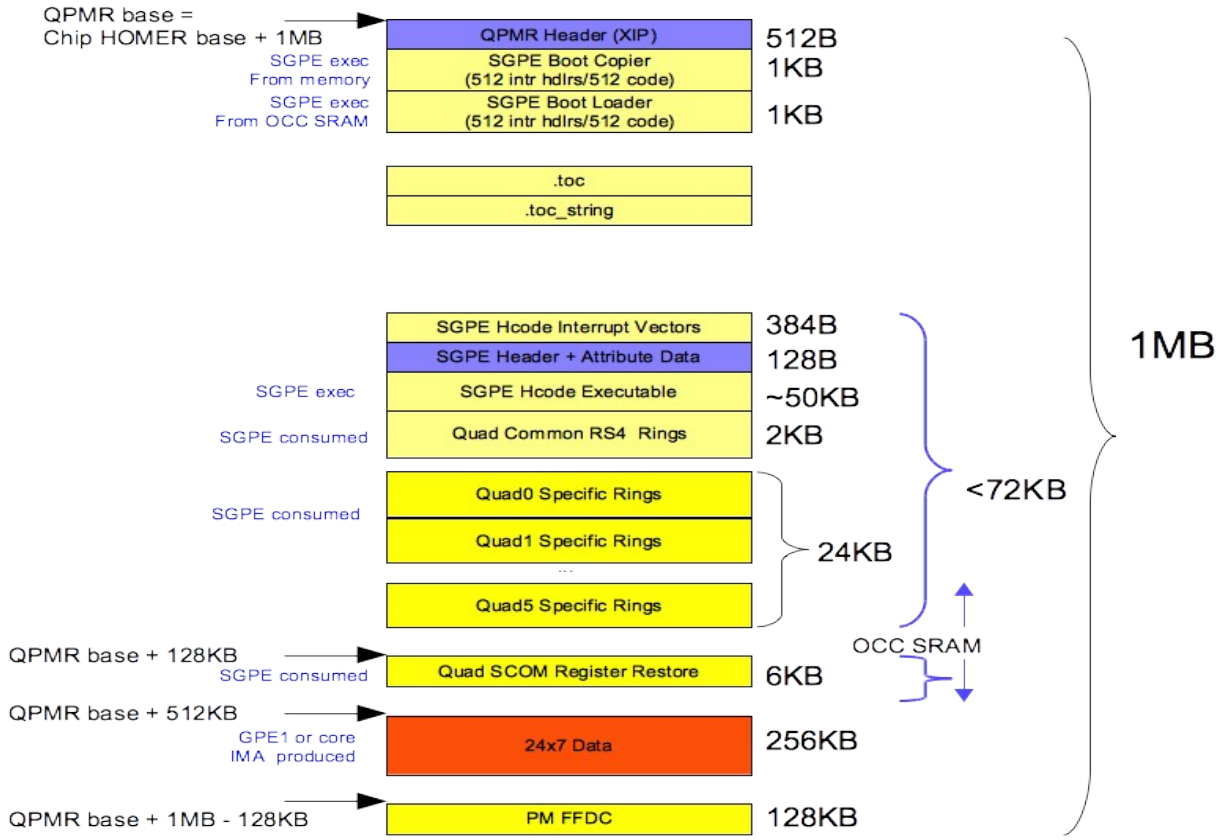

Figure 1.2: QPMR Layout Overview

The SGPE Copier contains code that copies the SGPE Boot Loader from memory to OCC SRAM and start the Boot Loader executing. The SGPE Boot Loader section contains the code that is run from OCC SRAM to copy the remaining necessary contents of this region from main memory (eg QPMR) into OCC SRAM.

1.3.1 QPMR Header

Table 1.1: QPMR Header Layout

Offset	Word 0				Word 1			
0x00	Magic Number							
0x08	Boot Copier Offset				Reserved			
0x10	Boot Loader Offset				Boot Loader Length			
0x18	Build Date				Version			
0x20	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag
0x28	SGPE Hcode Offset				SGPE Hcode Length			
0x30	Quad Common Rings Section Offset				Quad Common Rings Section Length			
0x38	Quad Common Override Rings Section Offset				Quad Common Override Rings Section Length			
0x40	Quad Specific Rings Section Offset				Quad Specific Rings Section Length			
0x48	Quad SCOM Section Offset				Quad SCOM Section Length			

Offset	Word 0	Word 1
0x50	Auxiliary Data Offset	Auxiliary Data Length
0x58	STOP FFDC Offset	STOP FFDC Length
0x60	SGPE Boot Progress Code	SGPE SRAM Image Size
0x68	Maximum Quad SCOM Restore Entries	Enable 24x7 IMA
0x70	PGPE SRAM Region Start	PGPE SRAM Region Size
0x78	SGPE SRAM Region Start	PGPE SRAM Region Size
0x80-0xF8	Reserved (Filled with 00s)	

Table 1.2: QPMR Header Fields

Field	Description
Magic Number	Set to ASCII “QPMR_1.0”
Boot Load Copier Offset	Offset from the beginning of the QPMR where the SGPE Copier (interrupt vector + executable) begins. This is debug information as to where this is placed.
Boot Load Loader Offset	Offset from the beginning of the QPMR where the SGPE Boot Loader (interrupt vector + executable) begins. Used by the SGPE Boot Copier to copy the SGPE Boot Loader from memory to OCC SRAM.
Boot Load Loader Length	Length, in bytes, of the SGPE Boot Loader executable section. Used by SGPE Boot Copier to determine how much content to copy from HOMER into OCC SRAM.
Boot Load Branch Instruction	Relative branch to boot copier. Executed upon starting the SGPE as the IAR will point to this location (QPMR + 16B).
Build Date	Date which the image was built. Hex in the form YYYYMMDD where YYYY is the hex value of the year (eg 2016 = 0x07E0)
Version	Build controlled version number (eg hour:minute:second) to differentiate image versions
SGPE Hcode Offset	Offset from the beginning of the QPMR where the SGPE Hcode header and executable begins. Used by SGPE Bootloader to determine where in QPMR to beginning copying.
SGPE Hcode Length	Length, in bytes, of the CME Hcode header and executable sections. The header is considered part of the executable. Used by SGPE Bootloader to determine how much content to copy from HOMER into OCC SRAM.
Quad Common Rings Section Offset	Offset from the beginning of the QPMR where the Quad Common Rings section begins. Used by SGPE Bootloader to determine where in QPMR to beginning copying.
Quad Common Rings Section Length	Length, in bytes, of the Quad Common Rings section. Used by SGPE Bootloader to determine how much content to copy from HOMER into OCC SRAM as all of this section is copied..
Quad Specific Rings Section Offset	Offset from the beginning of the CPMR where the Quad Common Rings section begins. Used by SGPE Bootloader to determine where in QPMR to beginning copying.
Quad Specific Rings Section Length	Length, in bytes, of the Quad Specific Rings section. Used by SGPE Bootloader to determine how much content to copy from HOMER into OCC SRAM as all of this section is copied..
Quad Specific SCOM Section Offset	Offset from the beginning of the QPMR where the Quad Specific SCOM section begins. Used by SGPE Bootloader to determine where in QPMR to beginning copying.
Quad Specific SCOM Section Length	Length, in bytes, of the Quad Specific SCOM section. Used by SGPE Bootloader to determine how much content to copy from HOMER into OCC SRAM as all of this section is copied..
Auxiliary Data Offset	Offset from the beginning of the QPMR where the Auxilary content begins. This data will be written by the 24x7 component of the OCC Firmware and consumed by performance harvesting tools.
Auxiliary Data Length	Length, in bytes, of the Auxiliary data section. Used by OCC Firmware and performance harvesting tools to know the limits for this data region.
STOP FFDC Offset	Offset from the beginning of the QPMR where the Power Management FFDC collected during a PM Complex reset is stored.
STOP FFDC Length	Length, in bytes, of the Power Management FFDC region.
SGPE Boot Progress Code	Progress code of the SGPE Boot. See Error: Reference source not found.

Field	Description
SGPE SRAM Image Size	SGPE Image size to be down loaded upon boot.
PGPE SRAM Region Start	Starting OCC SRAM address of the allocated space for PGPE Set by pgpe_img_edit during build.
PGPE SRAM Region Size	Amount of OCC SRAM space allocated for PGPE and all of its supporting data Set by pgpe_img_edit during build.
SGPE SRAM Region Start	Starting OCC SRAM address of the allocated space for SGPE Set by sgpe_img_edit during build.
SGPE SRAM Region Size	Amount of OCC SRAM space allocated for SGPE and all of its supporting data Set by sgpe_img_edit during build.

1.3.2 QPMR Size Allocation

Table 1.3: QPMR Size Summary

Section	Allocation
SGPE Header (including Global Data)	512B (72B used)
SGPE + 24x7 Hcode Section	~30KB (aligned to 512B for Interrupt vectors)
Quad Common Scan Rings	2KB (512 RS4 container x 3 (Func, GPTR, Time) → rounded to 2KB
PADDING	Added by ***p9_hcode_image_build*** to fill to 32B bound
Quad Specific Scan Rings	24KB (1KB RS4 container per EX (Repr) x 12 + RS4 container per EQ (Repr) x 6) → round to 24KB
Quad Specific SCOM Register Restore	6KB (63 registers per cache x 16B/entry → ~1008B x 6) → round to 6KB
Total	< 64KB

[1] see Error: Reference source not found

1.4 Core PM Region (CPMR)

The Core PM Regions (CPMR) contains the executable and related data for the CMEs to manage the core STOP function plus the Quad Management functions pertaining to the Pstate protocol.

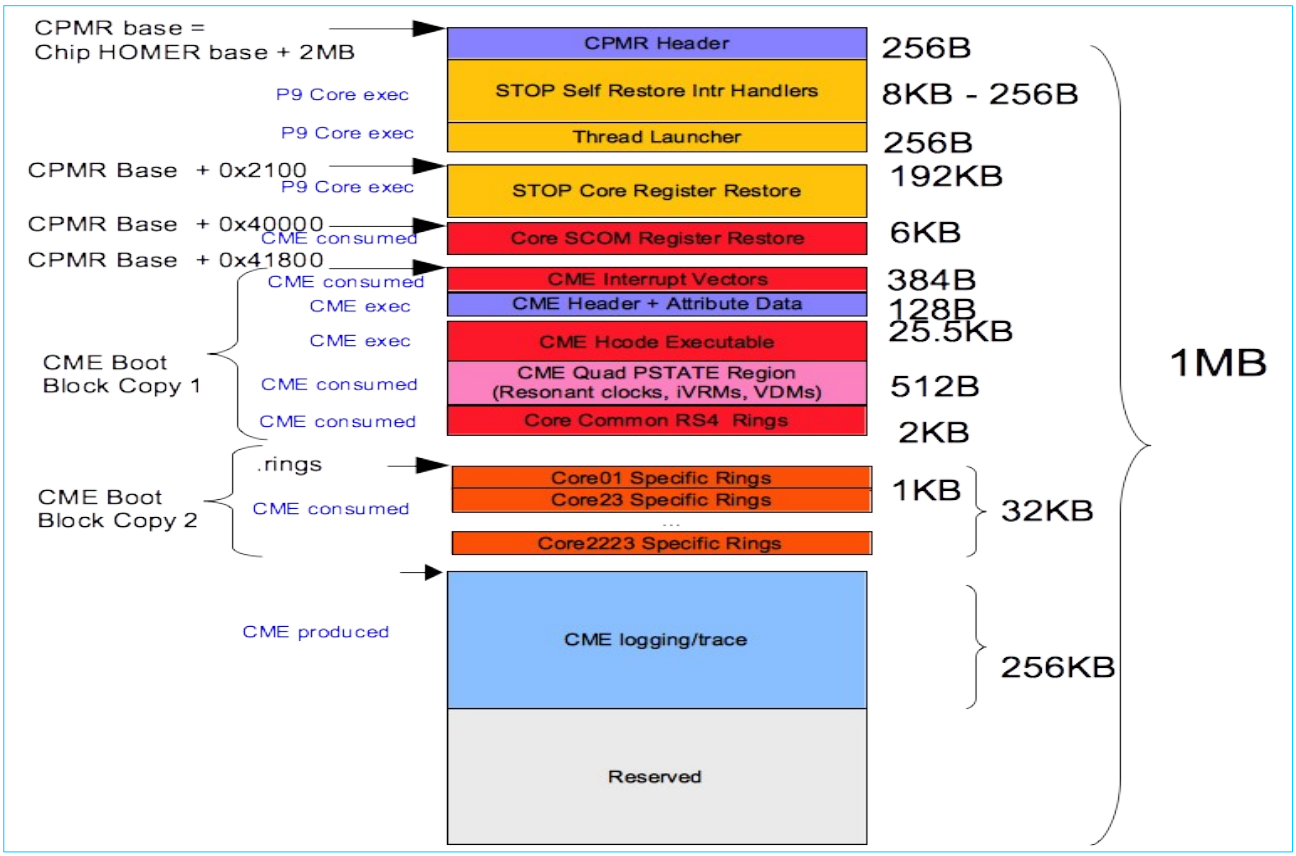

Figure 1.3: CPMR Layout Overview

1.4.1 CPMR Header

Table 1.4: CPMR Header Layout

Offset	Word 0				Word 1			
0x00	ATTN instruction				ATTN instruction			
0x08	Magic Number							
0x10	Build Date				Version			
0x18	Resvd Flag	Resvd Flag	Resvd Flag	Resvd Flag	SR Version	STOP API version	URMOR Fix	Fused Flag
0x20	CME Hcode Offset				CME Hcode Length			
0x28	CME Common Rings Section Offset				CME Common Rings Section Length			
0x30	CME Local Pstate Region Offset				CME Local Pstate Region Length			
0x38	Core Specific Rings Section Offset				Core Specific Rings Length			
0x40	Core SCOM Restore Offset				Core SCOM Restore Length			
0x48	Self Restore Offset				Self Restore Length			
0x50	Max Quad SCOM Restore Entries				Quad0 Pstate Offset			
0x58	Quad1 Pstate Offset				Quad2 Pstate Offset			
0x60	Quad3 Pstate Offset				Quad4 Pstate Offset			

Offset	Word 0	Word 1
0x68	Quad5 Pstate Offset	Reserved
0x70-0xF8	Reserved (Filled with ATTN instructions)	

Table 1.5: CPMR Header Fields

Field	Description
ATTN instruction	Instructions purposely added in the event errant software attempts to execute from the beginning of the CMPR
Magic Number	Set to ASCII “CPMR_1.0”
Build Date	Date which the image was built. Hex in the form YYYYMMDD where YYYY is the hex value of the year (eg 2016 = 0x07E0)
Version	Build controlled version number (eg hour:minute:second) to differentiate image versions
SR Version	Self-Restore Version Defines the version layout of the self-restore region. Used the STOP API to determine what format to populate the region with. 0: SPR list was not ordered. 1: SPR list was ordered and supports self-save. SMF registers added. Others: reserved
STOP API Version	STOP API Version Written by STOP API code to indicate the version supported to the Self-Restore code. 0: SR 0 only supported 1: SR 0 and 1 formats supported Others: reserved
URMOR Fix	Indicates the hardware will add the mtspr urmor, Rx instruction to the value before writing. Thus, URMOR value written to the Self-Restore region must subtract the instruction from the actual value to overcome this hardware issue (HW403111).
Fused Flag	If IPL is indicates fused mode, set to 0xBB; else set to 0x00.
CME Hcode Offset	Offset from the beginning of the CPMR where the CME Hcode header and executable begins. This offset is in terms of 32B blocks (for use by the CME block copy engine); this, this value is the real memory offset / 32.
CME Hcode Length	Length, in bytes, of the CME Hcode header and executable sections. The header is considered part of the executable.
CME Common Rings Section Offset	Offset from the beginning of the CPMR where the CME Common Rings section begins
CME Common Rings Section Length	Length, in bytes, of the CME Common Rings section
CME Local Pstate Section Offset	Offset from the beginning of the CPMR where the CME Local Pstate section begins
CME Local Pstate Section Length	Length, in bytes, of the CME Local Pstate section
Core Specific Rings Section Offset	Offset from the beginning of the CPMR where the Core Specific Rings section begins. Used by SGPE to setup a CME Block Copy Engine BAR to allow the CME to load the necessary rings for the specific cores that the CME supports.
Core Specific Rings Length	Length, in bytes, of the maximum Core Specific Rings section. Used by CME STOP initialization to determine how much content to copy from HOMER into CME SRAM. Upon building the CPMR, the p9_hcode_image_build will place the specific rings elements for the maximally sized ring container set present.
Core SCOM Restore Offset	Offset from the beginning of the CPMR where the Core SCOM Restore section begins
Core SCOM Restore Length	Length, in bytes, of the Core SCOM Restore section
Self Restore Offset	Offset from the beginning of the CPMR where the Core Self Restore section begins
Self Restore Length	Length, in bytes, of the Core Self Restore section
Max Quad SCOM Restore Entries	Maximum number of Quad SCOM Restore Entries allocated

Field	Description
Quad0 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 0
Quad1 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 1
Quad2 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 2
Quad3 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 3
Quad4 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 4
Quad5 Pstate Offset	Offset from the beginning of PPMR to the Local Pstate Parameter Block specific to Quad 5

1.4.2 CPMR Size Allocation

Table 1.6: CPMR Size Summary

Section	Allocation	CME Block Copy[1]
CME Common Header	72B	
CME Hcode Section	25.5KB	
CME Common Scan Rings	2KB (512 RS4 container x 4 (Func, Mode. GPTR, Time)	
CME Quad P-State Area	512B	
PADDING	Added by p9_hcode_image_build to fill to 32B bound required by CME BCE	
Core Specific Scan Rings	32KB (1KB RS4 container per core (Repr) x 24 → rounded to 32KB	2
Total	< 64KB	

[1] see Error: Reference source not found

1.4.3 STOP Self-Restore Interrupt Handler

This section is 8KB in size and implements the architected interrupt handlers needed for self-restoration. Only the SRESET (0x100) handler is present and is approximately 28B in size. The unutilized words are filled with ATTN opcode (0x00000200) during the section's build (eg by ***p9_hcode_image_build***)

1.4.4 Thread Launcher

The thread launcher section is 256B and contains code which determines which thread on which core is executing and then branches to the core and thread specific restoration location. Actual size is approximately 156B. Unutilized words to be filled with ATTN opcode (0x00000200).

1.4.5 Core Register Restore

1.5 Pstate PM Region (PPMR)

This region contains the code that runs on the PGPE as well as the parameters for the Pstate functions (PGPE) and Workload Optimized Frequency (WOF) functions that run on the OCC.

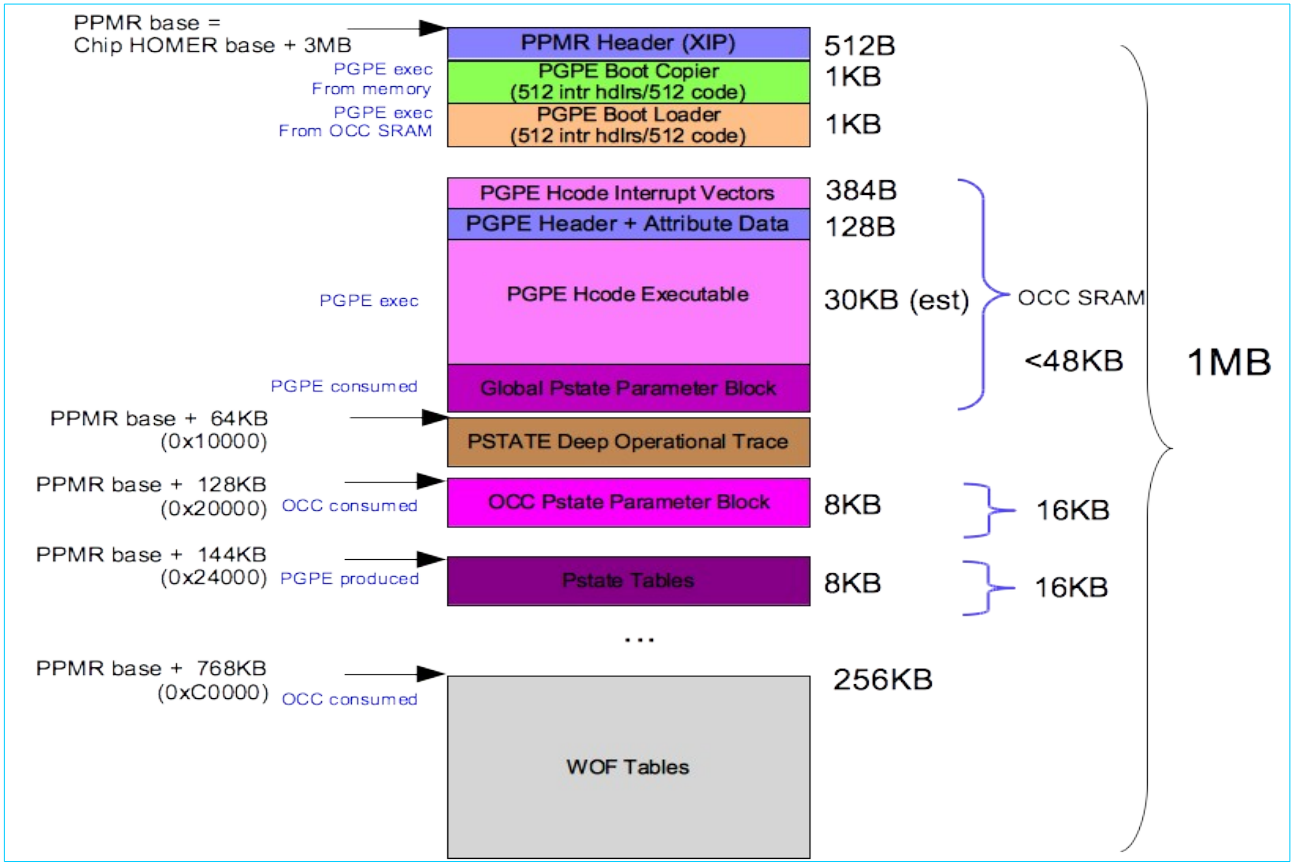

Figure 1.4: PPMR Layout Overview

The PGPE Copier contains code that copies the PGPE Boot Loader from memory to OCC SRAM and starts the Boot Loader executing. The PGPE Boot Loader section contains the code that is run from OCC SRAM to copy the remaining necessary contents of this region from main memory (eg PPMR) into OCC SRAM.

The OCC Pstate Parameter Block is built from attribute and VPD content an placed into HOMER. It is downloaded into OCC SRAM by OCC firmware into OCC owned SRAM region. The HOMER allocation is 16KB even though the projected size of the table is less than 8KB.

The Pstate Tables are created by PGPE Hcode upon booting to represent the key data that it will be making Pstate and WOF decision on. It is meant as debug information but can be used during run-time if needed. It is ONLY updated upon PGPE boot during a PM Reset action as is not consumed by Hcode. The HOMER allocation is 16KB even though the projected size of the table is less than 8KB.

The WOF Tables content is loaded from PNOR as selected by the sort and power mode of the machine. It is put in the last 256KB of the PPMR.

The PGPE Trace Area is a reserved 64KB space for PGPE Hcode to produce trace data for debug. At present, this region is NOT represented in the PPMR Header.

Note that there are blank regions in

1.5.1 PPMR Header

Table 1.7: PPMR Header Layout

Offset	Word 0	Word 1
0x00	Magic Number (magic)	
0x08	Boot Copier Offset (bc_offset)	Reserved
0x10	Boot Loader Offset (bl_offset)	Boot Loader Length (bl_length)
0x18	Build Date (build_date)	Version (version)
0x20	Resvd Flag | Resvd Flag | Resvd Flag | Resvd Flag	Resvd Flag | Resvd Flag | Resvd Flag | Resvd Flag
0x28	PGPE Hcode Offset	PGPE Hcode Length
0x30	Global Pstate Parameter Block Offset (gppb_offset)	Global Pstate Parameter Block Length (gppb_length)
0x38	Local Pstate Parameter Block Offset (lppb_offset)	Local Pstate Parameter Block Length (lppb_length)
0x40	OCC Pstate Parameter Block Offset (oppb_offset)	OCC Pstate Parameter Block Offset (oppb_length)
0x48	Pstate Tables Offset (pstables_offset)	Pstate Tables Offset (pstables_length)
0x50	PGPE SRAM Image Size	PGPE Boot Program Code
0x58	WOF Tables Offset (woftables_offset)	WOF Tables Length (woftables_length)
0x60	Auxiliary Task Offset (aux_task_offset)	Auxiliary Task Length (aux_task_length)
0x68	PGPE Deep Operational Trace Offset (ppge_doptrace_offset)	PGPE Deep Operational Trace Length (ppge_doptrace_length)
0x70	PGPE SRAM Region Start	PGPE SRAM Region Size
0x78	SGPE SRAM Region Start	PGPE SRAM Region Size
0x80-0x1F8	Reserved (Filled with 00s)	

Table 1.8: PPMR Header Fields

Field	Description
Magic Number	Set to ASCII “PPMR_1.0”
Build Date	Date which the image was built. Hex in the form YYYYMMDD where YYYY is the hex value of the year (eg 2016 = 0x07E0)
Version	Build controlled version number (eg hour:minute:second) to differentiate image versions
PGPE Hcode Offset	Offset from the beginning of the PPMR where the PGPE Hcode executable begins. The PGPE Header is contained within the executable at offset 0x180.
PGPE Hcode Length	Length, in bytes, of the PGPE Hcode executable section. The header is part of the executable.
Global Pstate Parameter Block Offset	Offset from the beginning of the PPMR where the Global Pstate Parameter Block (GPPB) begins.
Global Pstate Parameter Block Length	Length, in bytes, of the Global Pstate Parameter Block (GPPB) section.
Local Pstate Parameter Block Offset	Offset from the beginning of HOMER where the Local Pstate Parameter Block (LPPB) begins. (Note: this is provided in the event that PGPE Hcode needs to update fields in the LPPB. This is not currently planned but may in the future).
Local Pstate Parameter Block Length	Length, in bytes, of the Local Pstate Parameter Block (LPPB) section.
OCC Pstate Parameter Block Offset	Offset from the beginning of HOMER where the OCC Pstate Pstate Parameter Block (OPPB) begins. This block of data is consumed by the OCC firmware but is resident in the PPMR so that the build flow only deals with PPMR memory (and not OPMR memory).

Field	Description
OCC Pstate Parameter Block Length	Length, in bytes, of the OCC Pstate Parameter Block (OPPB) section.
Pstate Tables Offset	Offset from the beginning of the PPMR where the allocated section for the PGPE Hcode generated Pstate Tables section is located.
Pstate Tables Length	Length, in bytes, of the PGPE Hcode generated Pstate Tables section.
PGPE SRAM Image Size	Total size, in bytes, that the PGPE boot loader will copy into OCC SRAM
PGPE Boot Program Code	Error code if PGPE booting fails
WOF Tables Offset	Offset from the beginning of the PPMR where the allocated section for the WOF Tables is located.
WOF Tables Length	Length, in bytes, of the WOF Tables section.
Auxiliary Task Offset	Offset from the beginning of the PPMR where the allocated section for any PGPE Auxiliary Thread instructions/data are stored.
Auxiliary Task Length	Length, in bytes, of the Auxiliary Task section
PGPE Deep Operational Trace Offset	OffsetOffset from the beginning of the PPMR where the PGPE Deep Operational Trace section is allocated (set by p9_hcode_image_build) – default: 64KB
PGPE Deep Operational Trace Length	Length, in bytes, of the Auxiliary Task section (set by p9_hcode_image_build)
PGPE SRAM Region Start	Starting OCC SRAM address of the allocated space for PGPE Set by pgpe_img_edit during build.
PGPE SRAM Region Size	Amount of OCC SRAM space allocated for PGPE and all of its supporting data Set by pgpe_img_edit during build.
SGPE SRAM Region Start	Starting OCC SRAM address of the allocated space for SGPE Set by sgpe_img_edit during build.
SGPE SRAM Region Size	Amount of OCC SRAM space allocated for SGPE and all of its supporting data Set by sgpe_img_edit during build.

1.5.2 PPMR Size Allocation

Table 1.9: PPMR Size Summary

Section	Allocation
PGPE Header (including Global Data)	512B (72B used)
PGPE Hcode Section	~30KB (aligned to 512B for Interrupt vectors); padded to 8B
Global Pstate Parameter Block (PPB)	6KB
OCC Pstate Parameter Block (PPB)	6KB
Pstate Tables	8KB
PADDING	Added by ***p9_hcode_image_build*** to fill to 32B bound
Total	< 64KB

[1] see Error: Reference source not found

1.5.3 Global Pstate Parameter Block

```
/// Global Pstate Parameter Block
///
/// The GlobalPstateParameterBlock is an abstraction of a set of voltage/frequency
/// operating points along with hardware limits.  Besides the hardware global
/// Pstate table, the abstract table contains enough extra information to make
/// it the self-contained source for setting up and managing voltage and
/// frequency in either Hardware or Firmware Pstate mode.
///
/// When installed in PMC, Global Pstate table indices are adjusted such that
/// the defined Pstates begin with table entry 0. The table need not be full -
/// the \a pmin and \a entries fields define the minimum and maximum Pstates
/// represented in the table.  However at least 1 entry must be defined to
/// create a legal table.
///
/// Note that Global Pstate table structures to be mapped into PMC hardware
/// must be 1KB-aligned.  This requirement is fullfilled by ensuring that
/// instances of this structure are 1KB-aligned.


typedef struct {

    /// Pstate options
    ///
    /// The options are included as part of the GlobalPstateTable so that they
    /// are available to all procedures after gpsm_initialize().
    PstateOptions options;

    /// The frequency step in KHz
    uint32_t frequency_step_khz;

    /// Operating points
    ///
    /// VPD operating points are stored without load-line correction. Frequencies
    /// are in MHz, voltages are specified in units of 1mV, and currents are
    /// in units of 100mA.
    VpdOperatingPoint operating_points[VPD_PV_POINTS];

    /// Loadlines and Distribution values for the VDD rail
    SysPowerDistParms vdd_sysparm;

    /// Loadlines and Distribution values for the VCS rail
    SysPowerDistParms vcs_sysparm;

    /// Loadlines and Distribution values for the VDN rail
    SysPowerDistParms vdn_sysparm;

    /// The "Safe" Voltage
    ///
    /// A voltage to be used when safe-mode is activated
    uint32_t safe_voltage_mv;

    /// The "Safe" Frequency
```

```
    ///
    /// A frequency to be used when safe-mode is activated
    uint32_t safe_frequency_khz;

    /// Pstate stepping delay (in nanoseconds)
    uint8_t vrm_stepdelay_ns;

    /// The minimum core frequency in the system, defined by MRW
    uint32_t floor_freq_khz;

} GlobalPstateParmBlock;
```