Principles of ARM® Memory Maps

White Paper

Document number: ARM DEN 0001C

Principles of ARM Memory Maps
System Software on ARM

Release information

The *Change history* table lists the changes made to this document.

Table 1 Change history

Date	Issue	Confidentiality	Change
12 January 2011	A	Confidential	First release
07 March 2012	B	Confidential	Second release 64-bit ARM update. Addition of 44-bit and 48-bit maps.
19 October 2012	C	Non-Confidential	Migrated to White Paper format. Versatile Express example. Non-Confidential Proprietary Notice

Web Address

http://www.arm.com

Table of Contents

1 Introduction

This document describes the address maps used by ARM for A-class systems, from models and emulators to development boards and complex SoCs.

It explains the choice of address partitioning for memories, peripherals, and expansion spaces.

It describes the issues and constraints when 32bit platform operating systems use a 36-bit or 40-bit address space, and the restrictions imposed by 32-bit bus masters and peripherals.

It extends the memory maps to 48-bits of address space for future 64-bit ARM systems,

1.1 Scope

This document is applicable to ARM systems containing:

- 32-bit ARMv7 A-profile CPUs with *Large Physical Address Extension* (LPAE).
- 64-bit ARMv8 A-profile CPUs

This document replaces the deprecated DEN001 Principles of ARM Memory Maps PDD.

1.2 Additional reading

This section lists publications by ARM and by third parties.

The following documents contain information relevant to this document:

- *ARM Architecture Reference Manual ARMv7-A and ARMv7-R edition* (ARM DDI 0406)
- *ARMv8 documentation has limited availability at this time.*
  Please see ARM info center for latest updates
  .
- *Appendix A: TC2 and Versatile Expressand*
  - o *Motherboard Express µATX (V2M-P1)*
  - o *v2p ca15 a7 reference manual (DDI0503B)*

See Infocenter, http://infocenter.arm.com, for access to ARM documentation.

1.3 Conventions

This paper uses the terms *memory map* and *address map* interchangeably.

2 Overview

ARM creates a variety of development systems to support A-class cortex CPUs, ranging from cycle accurate RTL models, to fast software models, onto FPGAs and full custom SoCs. ARM has been harmonizing the memory maps in these systems to provide internal consistency and software portability, and to address the constraints that come with mixing 32-bit components within larger address spaces.

The introduction of *Large Physical Address Extension* (LPAE) to ARMv7 class CPUs has grown the physical address spaces to 36-bit and 40-bits, providing 64GB or 1024GB (1TB) memory space. Likewise the 64-bit ARMv8 architecture can address 48-bits, providing 256TB.

This paper describes ARM address maps for 32, 36 and 40-bit systems, and proposes extensions for 44 and 48-bit systems. Each larger address map is a superset of the smaller space, to enable the system to boot and interwork between address spaces.

It does define and explain the gross partitioning of the address space for memories, peripherals, and expansion space. It does not specify or require the addresses of individual peripherals.

The principles expressed in this paper help drive some of the design decisions ARM makes when defining system IP such as interconnects and memory controllers.

2.1 CPUs and hardware

2.1.1 32-bit ARMv7

ARM cores such as the Cortex-A9 and Cortex-A5 can only see a 32-bit address space.

The ARMv7 MMU can control memory access down to a 4Kbyte granule.

2.1.2 32-bit ARMv7 with LPAE

The introduction of LPAE enabled access to a 40-bit address space, as found on the ARM Cortex-A15 and Cortex-A7.

A 32bit ARM processor requires an Extended VMSAv7 MMU to access the extended memory addresses. This multistage MMU provides the translations from a 32-bit *virtual address* (VA) to a 40-bit *intermediate physical address* (IPA), and then to a 40-bit *physical address* (PA).

The processor's Memory Model Feature Register, `ID_MMFR3`, provides the size of the actual physical memory supported of 32, 36, or 40 bits.

The ARMv7 MMU with LPAE controls memory access down to a 4Kbyte granule.

2.1.3 64-bit ARMv8

64-bit ARMv8 CPUs can naturally access up to 48bits of physical address space, with a two stage MMU performing VA->IPA->PA translations.

The `ID_A64MMFR0` register describes the size of the actual physical address space, 32, 36,40,42,44 or 48bits.

NOTE: A 42bit address space is not described by this paper.

The ARMv8 MMU translation contexts can be configured to control memory access down to a 4Kbyte or 64Kbyte granule.

2.1.4 SMMU and Bus Masters

Bus mastering peripherals have direct access to the physical memory map and have similar addressing requirements as operating systems.

For example a legacy 32-bit mastering peripheral will be limited to the lowest address space. And will require working data buffers to be within the first 4GB.

A System MMU (SMMU) can be placed between a mastering peripheral and the system bus to provide translation and protection services for the peripheral. In this case the SMMU can be programmed to map 32-bit device virtual addresses into 40-bit physical addresses.

2.2 Software Portability

2.2.1 Operating Systems and Hypervisors

In this context hypervisors can be viewed as just another operating system that runs guest operating systems in place of applications.

Operating systems provide applications portability, through defined services which have abstracted the underlying hardware. The portability of the operating system in turn depends upon boot time configuration information, device drivers and system specific adaptation code.

ACPI tables and FDTs are ways of providing platform descriptions at boot time to an operating system. The entire address map can be described using these methods, and fixed addresses are not needed.

Operating systems do have expectations on the address map, such as:

- DRAM comes in large contiguous blocks
- Peripherals can be mapped by MMU page granularity
- Contiguous I/O space is available for dynamically mapped I/O

Pure 32-bit or 64-bit operating systems with flat address mappings have very few constraints on their address maps. Whereas mixed systems of 32-bit software and bus masters in large address spaces need constraints to ensure they can interwork, such as:

- Significant 32-bit addressable DRAM
- 32-bit addressable register mapped I/O
- Backwards compatibility through address map super sets

Note: Experience has shown that general purpose 32-bit operating systems struggle to manage excessive quantities of DRAM gracefully. 32GB of DRAM is often a practical upper limit, for a 32-bit OS kernel.

2.2.2 Firmware and Test Software

Firmware and test software tends to be system specific and less portable than operating systems. Therefore it is advantageous to maintain a strong degree of address space compatibility between iterations and generations of hardware.

Non-operating system software will often execute without using the MMU, and therefore peripherals must exist within its natural address space.

For many systems this software may remain as 32-bit code far longer than the 64-bit operating system in final use.

2.2.3 Trusted OS

The ARM security extensions, known as TrustZone, provide an orthogonal CPU execution state with its own address map that is intended to provide secure services. The execute states are referred to as Secure and Non-Secure.

The ARMv7 Virtualization Extensions are not provided to a CPU in trusted state. A Trusted OS will manage a single stage of VA to PA mappings, and it does not use or see IPAs.

LPAE when present is available to the trusted state, however It is quite possible that a 32-bit Trusted OS will not use LPAE capabilities of the MMU.

Within 64-bit ARM system the Trusted OS can remain as 32-bit software, in which case:

- A 32-bit Trusted OS will natively use the 32-bit address map.
- An updated 32-bit LPAE Trusted OS may access up to the 40-bit address map.

A 32bit trusted OS increases the utilization pressure on the lower 2GB of physical DRAM.

2.3 Address Map overview

The memory maps are defined as a set of growing super sets. As each memory map increases by 4-bits of address space, it contains all of the smaller address maps, at the lower addresses.

Each increment of 4 address bits results in a 16 fold increase in addressable space. The larger address space is partitioned in a repeatable way:

$^{8}/_{16}$	DRAM.
$^{4}/_{16}$	Mapped I/O.
$^{3}/_{16}$	Reserved space.
$^{1}/_{16}$	Previous memory map, that is, without the additional 4 address bits.

For example the 36-bit address map contains the entire 32-bit address map in the lowest 4GB of address space.

The address maps are partitioned into four types or regions:

- **Static I/O and Static Memories**, for register mapped on-chip peripherals, boot ROMs, and scratch RAMs.
- **Mapped I/O**, for dynamically configured, memory mapped buses, such as PCIe.
- **DRAM**, for main system dynamic memory.
- **Reserved space**, for future use.

3 Address map principles

A number of principles are followed by ARM when developing memory maps, to ensure they are suitable for system use cases that software can support.

3.1.1 Uniformity

A uniform SoC address map provides consistent physical addresses to all shared resources. Uniform addressing removes the need for additional inter-component translations and can simplify system integration.

P1. The global SoC address map for application processors and bus mastering peripherals should be consistent with uniform addresses for all shared accesses.

P2. Aliased addresses should not exist for shared resources.

Large address maps remove the need to reuse addresses.

P3. Private local address maps may exist per CPU/Cluster/Device, to provide private peripherals and memories.

For example local GIC interface per CPU.

P4. Private local address maps should not clash with globally visible addresses, for the same master.

Some on-SoC micro-controllers have locally fixed address maps. Address remapping from local-CPU relative maps, into the ARM centric address map may be required, in HW and by SW.

3.1.2 Interworking Super Sets

P5. Each large memory map must contain the complete smaller memory maps within the lower address ranges.

Maintaining smaller sub-set address maps allows for interworking and the forward compatibility of hardware and software designed smaller address spaces.

3.1.3 DRAM

The primary need for extending the SoC address space beyond 32-bits is to provide room for more DRAM.

However the interworking of hardware and software in mixed address space environment will always happen within the lowest common denominator address space. This leads to memory pressure on the DRAM at lower addresses.

DRAM requirements:

P6. DRAM must not be aliased, shadowed or decoded to additional physical addresses.

P7. Unpopulated DRAM partitions must not alias any other addresses, accesses should return bus errors.

Coherent systems rely on unique physical addresses to maintain memory consistency, throughout the caches, buses and memories. By forbidding the presence of aliased memories whole classes of errors and bugs can be avoided.

If aliasing can occur explicit software management, in the form or preventative cache flushing, may be required to ensure memory consistency. This flushing will negate the benefits of a coherent system.

Additionally aliased DRAM can have negative security implications.

P8. DRAM holes (when present) must not alias any other address, including DRAM, and accesses should return a bus error.

DRAM holes are optional, see below.

P9. Recommended that DRAM should be populated in large contiguous regions.

Operating systems and hypervisors vary in their ability to manage fragmented physical memory regions. Having fewer memory regions containing contiguous RAM can help increase software compatibility and efficient implementation.

Regions of contiguous physical RAM are often required when handing control over between software environments, such as; boot loaders, hypervisors and operating systems.

P10. Recommend that DRAM is populated into the lowest addresses memory regions first.

Since there is memory pressure on lower addressed regions to provide interworking space.

3.1.4 DRAM holes (optional)

P11. DRAM holes provide a way to partition large DRAM device into multiple address ranges.

Optional DRAM holes are proposed at the start of the higher DRAM address boundaries. These enable a simplified decoding scheme when partitioning a large capacity DRAM device across the lower physically addressed regions.

P12. Aliased DRAM holes must not be used for address space interworking.

The issues with aliasing are described above.

For example a 64GB DRAM part will be sub-divided into three regions. With the address offsets performed by a simple subtraction in the high order address bits.

	Physical Addresses in SoC	Offset	Internal DRAM address

2 GBytes
(32-bit map)
	`0x00 8000 0000 –`
	`0x00 FFFF FFFF`
		`-0x00 8000 0000`	`0x00 0000 0000 –`
			`0x00 7FFF FFFF`

30 GBytes
(36-bit map)
	`0x08 8000 0000 –`
	`0x0F FFFF FFFF`
		`-0x08 0000 0000`	`0x00 8000 0000 –`
			`0x07 FFFF FFFF`

32 GBytes
(40-bit map)
	`0x88 0000 0000 -`
	`0x8F FFFF FFFF`
		`-0x80 0000 0000`	`0x08 0000 0000 –`
			`0x0F FFFF FFFF`

Example of 64GB DRAM partitioning with holes.

In practical systems, with no aliasing, the holes do not “waste” DRAM address space since memories come sized in 2^n quantities. A fully loaded 40bit system, with holes would contain exactly 512GBytes of DRAM. Removal of the holes would only provide an additional 34GBytes of address space.

3.1.5 Empty Regions

Empty inaccessible regions will exist in the global address map.

P13. Empty address space should not alias other addresses.

P14. Any access to an empty address map region should either:

1. *Be ignored*
2. *Result in a recoverable bus error.*

The trapping of incorrectly accessed accesses can aid debugging and development.

3.1.6 32bit Systems Constraints

P15. Boot ROMs (where present) must be in the 32-bit address space

The reset vector for a 32-bit ARM CPU is architected to be address 0x00000000 or 0xFFFF0000 when HiVECs is enabled.

P16. Boot memories, SRAM or DRAM must be in the 32-bit address space

Working RAM within the 32bit address map is required before the MMU can be enabled.

P17. Static Peripheral Registers must be in the 32-bit address space

Static peripheral registers need to be visible to non-MMU software such as boot and test code, and to legacy non-LPAE operating systems.

P18. Significant DRAM must be present in the 32-bit address space

This is required for all non-LPAE software, and interworking with 32-bit bus mastering peripherals.

P19. The 32-bit DRAM region address space should be fully populated, before DRAM regions in higher address spaces.

The pure 32-bit DRAM will be under utilization pressure for interworking between 32-bit peripherals, and use by all non-LPAE software.

P20. Dynamic I/O regions (if present) should exist in the 32-bit address space.

32-bit only operating systems will require access to dynamic I/O, such as PCIe.

3.1.7 36-bit and 40-bit Systems

P21. 36-bit address space must be presented

To support 36-bit x86 PAE compatible operating systems, such as Linux.

3.1.8 Secure and Non-secure address maps

The ARM ARM defines the security extensions and how they interact with system addresses and peripherals. Some address map constraints are:

P22. Normal cacheable memory must be mapped uniquely into the Secure or Non-secure address maps.

Normal memory should not be aliased, or dual mapped, since this will impact the security architecture, and complicate the software view of cache coherency.

Dual mapping is allowed for a limited number of Secure OS use cases, such as limiting access to DRM'd media buffers. Additional care has to be taken to ensure cache coherency is maintained inside and outside of these use cases.

P23. Non-cached Device addresses can be dual mapped as Secure and Non-secure.

Allowing for devices to be claimed by Secure or Non-secure transactions. Policing of the NS bit is dependent upon the SoC interconnect fabric and device interface.

P24. NS access to a Secure register in a dual mapped peripheral should be WI/RAZ, this should not generate a bus error.

A dual mapped peripheral must enforce security at a register level.

3.1.9 I/O partitions

P25. Static I/O region should be subdivided into 64KB large pages.

In preparation for 64-bit systems with 64Kbyte pages..

3.1.10 Boot ROM

ARMv7 32bit CPUs are architected to boot from address `0x00000000` or `0xFFFF0000.`

The boot address of ARMv8 64bit CPUs is implementation defined.

P26. A secure SoC using TrustZone™ technology must boot from on-chip ROM.

The secure root of trust starts with the CPU booting from immutable trusted code.

P27. Boot ROM must not be aliased or dynamically switchable for RAM.

To ensure security boot ROMs must not be remapped to RAM after boot.

Note: This is a change from earlier ARM development systems, where boot ROMs and RAMs where overlaid.

P28. The boot ROM area must consume at least 1×64KB of address space.

The boot ROM is not required to populate the entire 64KB of address space. Within the boot ROM address region, repeated aliases of the boot ROM are acceptable.

3.1.11 SoC peripherals

P29. Each unique peripheral should occupy one or more 64KB pages.

To enable page granularity access controls by ARMv8 MMUs when using 64KB pages.

3.1.12 On-chip RAM

P30. Regions of on-chip RAM should be contained within 64KB pages.

The on-chip RAM is not required to populate the entire 64KB region.

P31. On chip RAM should not be aliased.

Unpopulated space should return a recoverable bus error.

4 32-bit, 36-bit and 40-bit ARM Address Maps

Address map in use in ARM development systems today

```
          - 32-bit -     - 36-bit -     - 40-bit -
1024GB +              +              +--------------+   <- 40-bit
       |                             | DRAM         |
       ~              ~              ~              ~
       |                             |              |
       |                             |              |
       |                             |              |
       |                             |              |
 544GB +              +              +--------------+
       |                             | Hole or DRAM |
       |                             |              |
 512GB +              +              +--------------+
       |                             | Mapped       |
       |                             | I/O          |
       ~              ~              ~              ~
       |                             |              |
 256GB +              +              +--------------+
       |                             | Reserved     |
       ~              ~              ~              ~
       |                             |              |
  64GB +              +--------------+--------------+   <- 36-bit
       |              | DRAM                        |
       ~              ~              ~              ~
       |              |                             |
       |              |                             |
  34GB +              +--------------+--------------+
       |              | Hole or DRAM                |
  32GB +              +--------------+--------------+
       |              | Mapped I/O                  |
       ~              ~              ~              ~
       |              |                             |
  16GB +              +--------------+--------------+
       |              | Reserved                    |
       ~              ~              ~              ~
   4GB +--------------+--------------+--------------+   <- 32-bit
       | 2GB of DRAM                                |
       |                                            |
   2GB +--------------+--------------+--------------+
       | Mapped I/O                                 |
   1GB +--------------+--------------+--------------+
       | ROM & RAM & I/O                            |
   0GB +--------------+--------------+--------------+   0
          - 32-bit -     - 36-bit -     - 40-bit -
```

Figure 1 32-bit, 36-bit and 40-bit Address Map

4.1 32-bit Memory Map

This section describes the ARM 32-bit memory map.

```
4GB +-----------------+   <- 32-bit
    | DRAM            |
    |                 |
2GB +-----------------+
    | Mapped I/O      |
1GB +-----------------+
    | ROM & RAM & I/O |
0GB +-----------------+   0
```

Figure 2 32-bit memory map

The 32-bit address map provides the working space for existing non-LPAE 32-bit OSes and 32-bit peripherals. DRAM and I/O must exist in the 32-bit address space to enable test code and system booting, before enabling the MMU.

Greater than 32-bit bus masters must access the 32-bit DRAM to interwork with 32-bit systems.

4.1.1 `0x0000 0000` – `0x0000 FFFF`. Boot ROM

ARMv7 32bit CPUs are architected to boot from address `0x00000000` or `0xFFFF0000.`

The boot address of ARMv8 64bit CPUs is implementation defined.

A secure on chip boot ROM is required to provide a secure root of trust.

4.1.2 `0x0001 0000` – `0x3FFF FFFF`. ROM, RAM, SoC I/O

This I/O address region is divided into 64KB pages, to align with 64KB page support in ARMv8 MMUs.

Pages can exclusively contain internal ROMs, RAMs, static memory, SoC peripheral registers, dynamically mapped I/O or empty space.

4.1.3 `0x4000 0000` – `0x7FFF FFFF`. Mapped I/O and additional SoC I/O

This I/O address region is primarily intended for mapped I/O such as PCIe.

Also used as overflow space for SoC register mapped peripherals.

Notionally divided into 64KB pages to align with ARMv8 MMUs.

4.1.4 `0x8000 0000` – `0xFFFF FFFF`. DRAM

2GB of contiguous address space for DRAM.

In a LPAE or mixed 32/64bit system, it is strongly recommended that the 32bit DRAM region is fully populated before regions at higher addresses.

4.2 36-bit Memory Map

```
64GB +-----------------+   <- 36-bit
     | DRAM            |
     ~                 ~
     |                 |
     |                 |
34GB +-----------------+
     | Hole or DRAM    |
32GB +-----------------+
     | Mapped I/O      |
     ~                 ~
     |                 |
16GB +-----------------+
     | Reserved        |
     ~                 ~
 4GB +-----------------+   <- 32-bit
     ~                 ~
     ~                 ~
     ~                 ~
 0GB +-----------------+   0
```

Figure 3 36-bit memory map

This memory map is a superset of the 32-bit address map, with the additional space being split as 50% DRAM with a optional hole in it, 25% mapped I/O space and reserved space.

4.2.1 `0x1 0000 0000` – `0x3 FFFF FFFF`. Reserved

12GB of address space reserved for future use.

4.2.2 `0x4 0000 0000` – `0x7 FFFF FFFF`. Mapped I/O

16GB of address space is available for dynamically mapped I/O.

4.2.3 `0x8 0000 0000` – `0x8 7FFF FFFF FFFF`. Hole or DRAM

2GB of address space that can contain either:

- Hole to enable DRAM device partitioning (as described in Section 3.1.4)
- DRAM.

4.2.4 `0x8 8000 0000` – `0xF FFFF FFFF`. DRAM

30GB of address space for DRAM.

4.3 40-bit Memory Map

```
1024GB +-----------------+   <- 40-bit
       | DRAM            |
       ~                 ~
       |                 |
       |                 |
       |                 |
 544GB +-----------------+
       | Hole or DRAM    |
       |                 |
 512GB +-----------------+
       | Mapped          |
       | I/O             |
       ~                 ~
       |                 |
 256GB +-----------------+
       | Reserved        |
       ~                 ~
       |                 |
  64GB +-----------------+   <- 36-bit
       ~                 ~
       ~                 ~
       ~                 ~
   0GB +-----------------+   0
```

Figure 4 40-bit memory map

The 40-bit address map is a superset of the 36-bit address map, and follows the same pattern of 50% DRAM with a optional hole in it, 25% mapped I/O space and reserved space.

4.3.1 `0x10 0000 0000` – `0x3F FFFF FFFF`. Reserved

192GB of address space reserved for future use.

4.3.2 `0x40 0000 0000` – `0x7F FFFF FFFF`. Mapped I/O

256GB of address space is available for dynamically mapped I/O.

4.3.3 `0x80 0000 0000` – `0x87 FFFF FFFF`. Hole or DRAM

32GB of address space that can contain either:

- Hole to enable DRAM device partitioning (as described in Section 3.1.4)
- DRAM

4.3.4 `0x88 0000 0000` – `0xFF FFFF FFFF`. DRAM

480GB of address space for DRAM.

5 Proposed 44-bit and 48bit Address Maps

Extending the 1/16th partitioning scheme leads to larger address maps.

```
          - 32 bit -     - 36 bit -     - 40 bit -     - 44 bit -     - 48 bit -
 256TB +              +              +              +              +--------------+  < 48 bit
       |                                                           | DRAM         |
       ~              ~              ~              ~              ~              ~
       |                                                           |              |
 136TB +              +              +              +              +--------------+
       |                                                           | Hole or DRAM |
 128TB +              +              +              +              +--------------+
       |                                                           | Mapped       |
       |                                                           | I/O          |
       ~              ~              ~              ~              ~              ~
  64TB +              +              +              +              +--------------+
       |                                                           | Reserved     |
       ~              ~              ~              ~              ~              ~
       |                                                           |              |
  16TB +                             +              +--------------+--------------+  < 44 Bit
       |                                            | DRAM                        |
       ~              ~              ~              ~              ~              ~
       |                                            |                             |
8604GB +              +              +              +--------------+--------------+
       |                                            | Hole or DRAM                |
   8TB +              +              +              +--------------+--------------+
       |                                            | Mapped                      |
       |                                            | I/O                         |
       ~              ~              ~              ~              ~              ~
   4TB +              +              +              +--------------+--------------+
       |                                            | Reserved                    |
       ~              ~              ~              ~              ~              ~
       |                                            |                             |
   1TB +                             +--------------+--------------+--------------+  < 40 Bit
       |                             | DRAM                                       |
       ~              ~              ~              ~              ~              ~
       |                             |                                            |
 544GB +              +              +--------------+--------------+--------------+
       |                             | Hole or DRAM                               |
 512GB +              +              +--------------+--------------+--------------+
       |                             | Mapped                                     |
       |                             | I/O                                        |
       ~              ~              ~              ~              ~              ~
 256GB +              +              +--------------+--------------+--------------+
       |                             | Reserved                                   |
       ~              ~              ~              ~              ~              ~
       |                             |                                            |
  64GB +              +--------------+--------------+--------------+--------------+  < 36 Bit
       |              | DRAM                                                      |
       ~              ~              ~              ~              ~              ~
       |              |                                                           |
  34GB +              +--------------+--------------+--------------+--------------+
       |              | Hole or DRAM                                              |
  32GB +              +--------------+--------------+--------------+--------------+
       |              | Mapped I/O                                                |
       ~              ~              ~              ~              ~              ~
  16GB +              +--------------+--------------+--------------+--------------+
       |              | Reserved                                                  |
       ~              ~              ~              ~              ~              ~
   4GB +--------------+--------------+--------------+--------------+--------------+  < 32 Bit
       | 2GB DRAM                                                                 |
       |                                                                          |
   2GB +--------------+--------------+--------------+--------------+--------------+
       | Mapped I/O                                                               |
   1GB +--------------+--------------+--------------+--------------+--------------+
       | ROM & RAM & I/O                                                          |
   0GB +--------------+--------------+--------------+--------------+--------------+  0
          - 32 bit -     - 36 bit -     - 40 bit -     - 44 bit -     - 48 bit -
```

Figure 5 Proposed 44-bit and 48-bit Address Maps

5.1 44-bit Memory Map

```
16384GB +----------------+   <- 44-bit
         | DRAM           |
         ~                ~
         |                |
         |                |
         |                |
         |                |
 8604GB +----------------+
         | Hole or DRAM   |
         |                |
 8192GB +----------------+
         | Mapped         |
         | I/O            |
         ~                ~
         |                |
 4096GB +----------------+
         | Reserved       |
         ~                ~
         |                |
 1024GB +----------------+   <- 40-bit
         ~                ~
         ~                ~
         ~                ~
    0GB +----------------+   0
```

Figure 6 44-bit memory map

The 44-bit address map is a superset of the 40-bit address map, and follows the same pattern of 50% DRAM with a optional hole in it, 25% mapped I/O space and reserved space.

5.1.1 `0x100 0000 0000` – `0x3FF FFFF FFFF`. Reserved

3TB of address space reserved for future use.

5.1.2 `0x400 0000 0000` – `0x7FF FFFF FFFF`. Mapped I/O

4TB of address space is available for dynamically mapped I/O.

5.1.3 `0x800 0000 0000` – `0x87F FFFF FFFF`. Hole

512GB of address space that can contain either:

- Hole to enable DRAM device partitioning (as described in Section 3.1.4)
- DRAM

5.1.4 `0x880 0000 0000` – `0xFFF FFFF FFFF`. DRAM

7680GB (7.5TB) of address space for DRAM.

5.2 48-bit Memory Map

```
256TB +-----------------+   <- 48-bit
      | DRAM            |
      ~                 ~
      |                 |
      |                 |
      |                 |
      |                 |
136TB +-----------------+
      | Hole or DRAM    |
      |                 |
128TB +-----------------+
      | Mapped          |
      | I/O             |
      ~                 ~
      |                 |
 64TB +-----------------+
      | Reserved        |
      ~                 ~
      |                 |
 16TB +-----------------+   <- 44-bit
      ~                 ~
      ~                 ~
      ~                 ~
  0GB +-----------------+   0
```

Figure 7 48-bit memory map

The 48-bit address map is a superset of the 44-bit address map, and follows the same pattern of 50% DRAM with a optional hole in it, 25% mapped I/O space and reserved space.

5.2.1 `0x1000 0000 0000` – `0x3FFF FFFF FFFF`. Reserved

48TB of address space reserved for future use.

5.2.2 `0x4000 0000 0000` – `0x7FFF FFFF FFFF`. Mapped I/O

64TB of address space is available for dynamically mapped I/O.

5.2.3 `0x8000 0000 0000` – `0x87FF FFFF FFFF`. Hole

8TB of address space that can contain either:

- Hole to enable DRAM device partitioning (as described in Section 3.1.4)
- DRAM

5.2.4 `0x8800 0000 0000` – `0xFFFF FFFF FFFF`. DRAM

120TB of address space for DRAM.

6 Glossary

Table 2 describes some of the terms used in this document.

Table 2 Glossary terms

Term	Description
Large Physical Address Extension (LPAE)	Optional extension to VMSAv7 that provides an address translation system supporting physical addresses of up to 40 bits at a 4KB granularity of translation.
Memory Management Unit (MMU)	Address translation unit that can transform addresses and check access permissions. In one stage from a VA to a PA, or in two stages from a VA to IPA to PA. Translation descriptions are provided by translation tables held in memory.
Intermediate Physical Address (IPA)	In an implementation of virtualization, the address to which a Guest OS maps a physical address.
Physical Address (PA)	Identifies a memory location within the hardware system.
System Memory Management Unit (SMMU)	System MMU controls peripheral access into the system through address translation, access permissions, and memory attribute determination.
TrustZone	The security technology from ARM that enables the construction of a Normal world and a Secure world.
Virtual Address (VA)	Is an address used by the CPU and software when the MMU enabled.
Virtual Memory System Architecture (VMSA)	A memory system architecture that supports virtual addresses.

7 Appendix A: Memory map example, ARM Versatile Express

Below is an example ARM memory map from Versatile Express development system with a V2P-CA15 core tile, also known as TC2.

The Region/Subsystem column maps to the regions described in this paper.

Additional information on these products can be found in ARM infocenter.

- o *Motherboard Express µATX (V2M-P1)*
- o *v2p ca15 a7 reference manual (DDI0503B)*

Start Addr	End Addr	Region	V2P-CA15 (TC2)	VE Motherboard RS1/2
Start	**End**	**Subsystem**	**Peripheral**	**Peripheral**
0x0000_0000	0x03FF_FFFF	Peripheral Space	SMC CS 0 (NOR)	SMC CS 0 (NOR) Aliased
0x0400_0000	0x07FF_FFFF		Secure RAM	
0x0800_0000	0x0BFF_FFFF		SMC CS 0 (NOR)	SMC CS 0 (NOR)
0x0C00_0000	0x0FFF_FFFF		SMC CS 4 (NOR)	SMC CS 4 (NOR)
0x1000_0000	0x13FF_FFFF		SMC CS 5 (Reserved)	SMC CS 5 (Reserved)
0x1400_0000	0x17FF_FFFF		SMC CS 1 (PSRAM)	SMC CS 1 (PSRAM)
0x1800_0000	0x19FF_FFFF		SMC CS 2 (Periphs)	Video RAM
0x1A00_0000	0x1AFF_FFFF			Ethernet
0x1B00_0000	0x1BFF_0000			USB
0x1C00_0000	0x1C00_FFFF		SMC CS 3 (Periphs)	Local DAP ROM
0x1C01_0000	0x1C01_FFFF			System Registers
0x1C02_0000	0x1C02_FFFF			SP810 (Sysctrl)
0x1C03_0000	0x1C03_FFFF			2Wire (PCIe)
0x1C04_0000	0x1C04_FFFF			PL041 (Aaci)
0x1C05_0000	0x1C05_FFFF			PL180 (Mci/Mmci)
0x1C06_0000	0x1C06_FFFF			KMI0 (PL050)
0x1C07_0000	0x1C07_FFFF			KMI1
0x1C08_0000	0x1C08_FFFF			Reserved
0x1C09_0000	0x1C09_FFFF			Uart0 (PL011)
0x1C0A_0000	0x1C0A_FFFF			Uart1
0x1C0B_0000	0x1C0B_FFFF			Uart2
0x1C0C_0000	0x1C0C_FFFF			Uart3
0x1C0D_0000	**0x1C0E_FFFF**			Reserved
0x1C0F_0000	0x1C0F_FFFF			Wdog (SP805)
0x1C10_0000	0x1C10_FFFF			Reserved
0x1C11_0000	0x1C11_FFFF			Timer01 (SP804)
0x1C12_0000	0x1C12_FFFF			Timer23 (SP804)
0x1C13_0000	**0x1C15_FFFF**			Reserved
0x1C16_0000	0x1C16_FFFF			2Wire (DVI)
0x1C17_0000	0x1C17_FFFF			RTC (PL031)
0x1C18_0000	**0x1C19_FFFF**			Reserved
0x1C1A_0000	0x1C1A_FFFF			CF Card

0x1C1B_0000	0x1C1B_FFFF			Uart4 (not used)
0x1C1C_0000	**0x1C1E_FFFF**			Reserved
0x1C1F_0000	0x1C1F_FFFF			CLCD (PL111)
0x1C20_0000	**0x1FFF_FFFF**			Reserved
0x2000_0000	0x2000_0FFF	CoreSight Space	DAP ROM	
0x2000_1000	0x2000_1FFF		ETB	
0x2000_2000	0x2000_2FFF		CTI	
0x2000_3000	0x2000_3FFF		TPIU	
0x2000_4000	0x2000_4FFF		Funnel	
0x2000_5000	0x2000_5FFF		ITM	
0x2000_6000	0x2000_6FFF		SWO	
0x2000_7000	**0x2000_FFFF**			
0x2001_0000	0x2001_FFFF		Reserved	
0x2002_0000	0x2002_0FFF		A15 ROM Table	
0x2002_1000	0x2002_FFFF		Reserved	
0x2003_0000	0x2003_0FFF		CPU0 DBG	
0x2003_1000	0x2003_1FFF		CPU0 PMU	
0x2003_2000	0x2003_2FFF		CPU1 DBG	
0x2003_3000	0x2003_3FFF		CPU1 PMU	
0x2003_4000	0x2003_4FFF		CPU2 DBG	
0x2003_5000	0x2003_5FFF		CPU2 PMU	
0x2003_6000	0x2003_6FFF		CPU3 DBG	
0x2003_7000	0x2003_7FFF		CPU3 PMU	
0x2003_8000	0x2003_8FFF		CPU0 CTI	
0x2003_9000	0x2003_9FFF		CPU1 CTI	
0x2003_A000	0x2003_AFFF		CPU2 CTI	
0x2003_B000	0x2003_BFFF		CPU3 CTI	
0x2003_C000	0x2003_CFFF		CPU0 PTM	
0x2003_D000	0x2003_DFFF		CPU1 PTM	
0x2003_E000	0x2003_EFFF		CPU2 PTM	
0x2003_F000	0x2003_FFFF		CPU3 PTM	
0x2004_0000	**0x29FF_FFFF**		Reserved	
0x2A00_0000	0x2A0F_FFFF	System Peripheral Space	NIC301 GPV	
0x2A10_0000	**0x2A41_FFFF**			
0x2A42_0000	0x2A42_FFFF		SCC (Alias)	
0x2A43_0000	0x2A43_FFFF		Gcounter	
0x2A44_0000	**0x2AFF_FFFF**			
0x2B00_0000	0x2B00_FFFF		HDLCD	
0x2B01_0000	0x2B01_FFFF			
0x2B02_0000	0x2B02_FFFF		UART(0) cfg	
0x2B03_0000	**0x2B05_FFFF**			
0x2B06_0000	0x2B06_FFFF		System Watchdog	
0x2B07_0000	0x2B07_FFFF		BLM Cntrl	
0x2B08_0000	**0x2B09_FFFF**			

0x2B0A_0000	0x2B0A_FFFF		DMC cfg
0x2B0B_0000	**0x2BFF_FFFF**		
0x2C00_0000	**0x2C00_7FFF**	CPU Space	GIC-400
0x2C01_0000	**0x2C08_FFFF**		
0x2C09_0000	0x2C09_FFFF		CCI
0x2C0A_0000	0x2C0B_FFFF		
0x2C0C_0000	0x2C0C_00FF		A15/A7 Periph Base
0x2C0C_0100	**0x2CFF_FFFF**		
0x2D00_0000	0x2D00_FFFF	Graphics Space	Reserved
0x2D01_0000	0x2DFF_FFFF		Reserved
0x2E00_0000	0x2EFF_FFFF	Embedded RAM	Internal SRAM 64kB
0x2F00_0000	0x2FFF_FFFF	128-bit User expansion	External AXI
0x3000_0000	0x3FFF_FFFF		A15 ACP
0x4000_0000	0x5FFF_FFFF		External AXI
0x6000_0000	**0x7FEE_FFFF**	64-bit User expansion	Reserved
0x7FEF_0000	0x7FEF_FFFF		DMC phy cfg
0x7FF0_0000	0x7FF1_FFFF		DMA
0x7FF2_0000	**0x7FFC_FFFF**		Reserved
0x7FFD_0000	0x7FFD_FFFF		PL354 cfg
0x7FFE_0000	0x7FFE_FFFF		Reserved
0x7FFF_0000	0x7FFF_FFFF		SCC
0x8000_0000	0xFFFF_FFFF	DRAM	DRAM
0x01_0000_0000	0x03_FFFF_FFFF	Reserved	
0x04_0000_0000	0x07_FFFF_FFFF	128-bit User expansion	
0x08_0000_0000	0x08_7FFF_FFFF	Reserved	Aliased
0x08_8000_0000	0x0F_FFFF_FFFF	DRAM	DRAM
0x10_0000_0000	0x3F_FFFF_FFFF	Reserved	
0x40_0000_0000	0x7F_FFFF_FFFF	128-bit User expansion	
0x80_8000_0000	0x87_FFFF_FFFF	Reserved	Aliased
0x88_0000_0000	0xFF_FFFF_FFFF	DRAM	DRAM