Memory Management (4) DXE GCD Mechanism

原创 Pedro Posted on 2016-12-28 14:24:40 Read 4.1k Collection 12 Likes 5

Category Column: UEFI-MemoryManagement Article Tags: gcd

摘要 This article takes a deep dive into the GCD (Global Coherency Domain) mechanism in UEFI, detailing its functionality in managing processor-visible memory and IO resources. The article also outlines the GCD initialization process and its role in DxeCore, and explains several different types of memory spaces.

The summary is generated in C Know , supported by DeepSeek-R1 full version, go to experience>

The previous article briefly introduced the general process of UEFI resource management and reporting. This article mainly introduces the GCD mechanism. The main specs for GCD are PI Vol2_DXE_CIS_1_5 7.2 and Beyond BIOS 2nd chapter8.

GCD:
GCD (Global Coherency Domain) These functions are mainly used to manage memory and IO resources visible to the processor. Memory and IO are separate. Memory is required for UEFI firmware, and IO is optional. So it is divided into the following two categories:

- GCD memory space map
- GCD IO space map

GCD is initialized in Dxe Core, using the information in HOB, so it can be said that GCD is another way to express HOB information. The following is the description of GCD:

1. The address line width that the CPU can access memory
2. The location, size, attributes, etc. of each block of memory address
3. Firmware device mapped to memory address
4. Firmware volume address
5. Dxe previously allocated memory resources (the resources allocated using AllocatePage in the PEI stage mentioned earlier)

The memory described in GCD can be roughly divided into the following categories:

- Nonexistent memory
- System memory
- Memory-mapped I/O
- Reserved memory

Nonexistent memory: refers to the address lines that the CPU can access, but there is no corresponding device (or translated as system components) on this address line.
System memory: This is easy to understand, it corresponds to our DIMM (GCD does not describe Cache)
Memory-mapped I/O: It is a section of address that the CPU accesses, but it will be mapped to the IO device.
Reserved memory: The CPU can access this address space, but the device corresponding to this address space is not a memory controller and IO device.

> GCD describes the start and size of each address space, but does not describe the memory usage. GCD is invisible to the OS and is only used internally by UEFI Firmware.

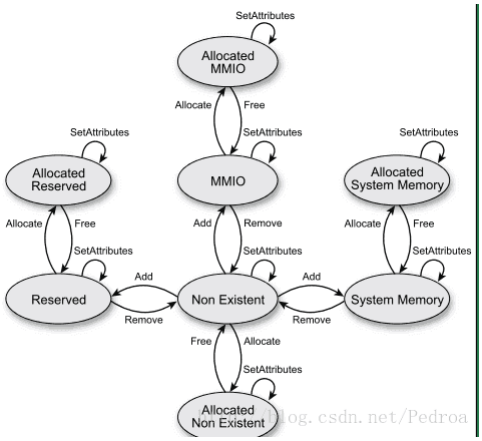
GCD's functions for managing memory resources are:

- AddMemorySpace ( )
- AllocateMemorySpace ( )
- FreeMemorySpace ( )
- RemoveMemorySpace ( )
- SetMemorySpaceAttributes ( )

GCD's functions for querying memory resources are:

- GetMemorySpaceDescriptor ( )
- GetMemorySpaceMap ( )

Below you can see a picture



Look at this picture more and you will find that the more you look at it, the more you feel it.
Regarding this picture, here is a brief introduction to the two parts on the far right of the picture (system memory) that will be introduced next. For example, if we want to allocate memory in the DXE stage, it will first find Non Existent (headquarters) and use Add () to wholesale a memory block of the system memory type. Then you can manage it separately in this block (allocate, free). There is also the block above, which we will introduce in the PCI bus later. This is to allocate mem resources to the PCIe device. First, the PCI bus driver will find Non Existent and call Add () to wholesale a memory block of the MMIO type, and then manage it separately to allocate memory addresses to each PCIe device (this memory actually refers to the memory address of the PCIe domain, which is not the same concept as the CPU domain. Although X86 is a 1:1 map, the address is the same, but the concept is different).

GCD's description of IO is similar to that of memory, so I won't go into detail here. You can refer to the two specs introduced at the beginning.