Chapter 3 explains how segmentation converts logical addresses to linear addresses. **Paging** (or linear-address translation) is the process of translating linear addresses so that they can be used to access memory or I/O devices. Paging translates each linear address to a **physical address** and determines, for each translation, what accesses to the linear address are allowed (the address’s **access rights**; and the type of caching used for such accesses (the address’s **memory type**).

Intel-64 processors support three different paging modes. These modes are identified and defined in Section 4.1. Section 4.2 gives an overview of the translation mechanism that is used in all modes. Section 4.3, Section 4.4, and Section 4.5 discuss the three paging modes in detail.

Section 4.6 details how paging determines and uses access rights. Section 4.7 discusses exceptions that may be generated by paging (page-fault exceptions). Section 4.8 considers data which the processor writes in response to linear-address accesses (accessed and dirty flags).

Section 4.9 describes how paging determines the memory types used for accesses to linear addresses. Section 4.10 provides details of how a processor may cache information about linear-address translation. Section 4.11 outlines interactions between paging and certain VMX features. Section 4.12 gives an overview of how paging can be used to implement virtual memory.

### 4.1 PAGING MODES AND CONTROL BITS

Paging behavior is controlled by the following control bits:
- The WP and PG flags in control register CR0 (bit 16 and bit 31, respectively).
- The PSE, PAE, and PGE flags in control register CR4 (bit 4, bit 5, and bit 7, respectively).
- The LME and NXE flags in the IA32_EFER MSR (bit 8 and bit 11, respectively).

Software enables paging by using the MOV to CR0 instruction to set CR0.PG. Before doing so, software should ensure that control register CR3 contains the physical address of the first paging structure that the processor will use for linear-address translation (see Section 4.2) and that structure is initialized as desired. See Table 4-3, Table 4-7, and Table 4-12 for the use of CR3 in the different paging modes.

Section 4.1.1 describes how the values of CR0.PG, CR4.PAE, and IA32_EFER.LME determine whether paging is in use and, if so, which of three paging modes is in use. Section 4.1.2 explains how to manage these bits to establish or make changes in
PAGING

paging modes. Section 4.1.3 discusses how CR0.WP, CR4.PSE, CR4.PGE, and IA32_EFER.NXE modify the operation of the different paging modes.

4.1.1 Three Paging Modes

If CR0.PG = 0, paging is not used. The logical processor treats all linear addresses as if they were physical addresses. CR4.PAE and IA32_EFER.LME are ignored by the processor, as are CR0.WP, CR4.PSE, and CR4.PGE, and IA32_EFER.NXE.

Paging is enabled if CR0.PG = 1. Paging can be enabled only if protection is enabled (CR0.PE = 1). If paging is enabled, one of three paging modes is used. The values of CR4.PAE and IA32_EFER.LME determine which paging mode is used:

- If CR0.PG = 1 and CR4.PAE = 0, 32-bit paging is used. 32-bit paging is detailed in Section 4.3. 32-bit paging uses CR0.WP, CR4.PSE, and CR4.PGE as described in Section 4.1.3.

- If CR0.PG = 1, CR4.PAE = 1, and IA32_EFER.LME = 0, PAE paging is used. PAE paging is detailed in Section 4.4. PAE paging uses CR0.WP, CR4.PGE, and IA32_EFER.NXE as described in Section 4.1.3.

- If CR0.PG = 1, CR4.PAE = 1, and IA32_EFER.LME = 1, IA-32e paging is used. IA-32e paging is detailed in Section 4.5. IA-32e paging uses CR0.WP, CR4.PGE, and IA32_EFER.NXE as described in Section 4.1.3. IA-32e paging is available only on processors that support the Intel 64 architecture.

The three paging modes differ with regard to the following details:

- Linear-address width. The size of the linear addresses that can be translated.
- Physical-address width. The size of the physical addresses produced by paging.
- Page size. The granularity at which linear addresses are translated. Linear addresses on the same page are translated to corresponding physical addresses on the same page.
- Support for execute-disable access rights. In some paging modes, software can be prevented from fetching instructions from pages that are otherwise readable.

Table 4-1 illustrates the key differences between the three paging modes.

Because they are used only if IA32_EFER.LME = 0, 32-bit paging and PAE paging is used only in legacy protected mode. Because legacy protected mode cannot produce

1. The LMA flag in the IA32_EFER MSR (bit 10) is a status bit that indicates whether the logical processor is in IA-32e mode (and thus using IA-32e paging). The processor always sets IA32_EFER.LMA to CR0.PG & IA32_EFER.LME. Software cannot directly modify IA32_EFER.LMA; an execution of WRMSR to the IA32_EFER MSR ignores bit 10 of its source operand.
linear addresses larger than 32 bits, 32-bit paging and PAE paging translate 32-bit linear addresses.

Because it is used only if IA32_EFER.LME = 1, IA-32e paging is used only in IA-32e mode. (In fact, it is the use of IA-32e paging that defines IA-32e mode.) IA-32e mode has two sub-modes:

- Compatibility mode. This mode uses only 32-bit linear addresses. IA-32e paging treats bits 47:32 of such an address as all 0.
- 64-bit mode. While this mode produces 64-bit linear addresses, the processor ensures that bits 63:47 of such an address are identical. IA-32e paging does not use bits 63:48 of such addresses.

NOTES:
1. The physical-address width is always bounded by MAXPHYADDR; see Section 4.1.4.
2. The processor ensures that IA32_EFER.LME must be 0 if CR0.PG = 1 and CR4.PAE = 0.
3. 32-bit paging supports physical-address widths of more than 32 bits only for 4-MByte pages and only if the PSE-36 mechanism is supported; see Section 4.1.4 and Section 4.3.
4. 4-MByte pages are used with 32-bit paging only if CR4.PSE = 1; see Section 4.3.
5. Execute-disable access rights are applied only if IA32_EFER.NXE = 1; see Section 4.6.
6. Not all processors that support IA-32e paging support 1-GByte pages; see Section 4.1.4.

Table 4-1. Properties of Different Paging Modes

<table>
<thead>
<tr>
<th>Paging Mode</th>
<th>CR0.PG</th>
<th>CR4.PAE</th>
<th>LME in IA32_EFER</th>
<th>Linear-Address Width</th>
<th>Physical-Address Width¹</th>
<th>Page Size(s)</th>
<th>Supports Execute-Disable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>0</td>
<td>N/A</td>
<td>N/A</td>
<td>32</td>
<td>32</td>
<td>N/A</td>
<td>No</td>
</tr>
<tr>
<td>32-bit</td>
<td>1</td>
<td>0</td>
<td>0²</td>
<td>32</td>
<td>Up to 40³</td>
<td>4-KByte</td>
<td>No</td>
</tr>
<tr>
<td>PAE</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>32</td>
<td>Up to 52</td>
<td>4-KByte</td>
<td>Yes⁵</td>
</tr>
<tr>
<td>IA-32e</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>48</td>
<td>Up to 52</td>
<td>4-KByte 2-MByte</td>
<td>Yes⁵</td>
</tr>
</tbody>
</table>

NOTES:
1. The physical-address width is always bounded by MAXPHYADDR; see Section 4.1.4.
2. The processor ensures that IA32_EFER.LME must be 0 if CR0.PG = 1 and CR4.PAE = 0.
3. 32-bit paging supports physical-address widths of more than 32 bits only for 4-MByte pages and only if the PSE-36 mechanism is supported; see Section 4.1.4 and Section 4.3.
4. 4-MByte pages are used with 32-bit paging only if CR4.PSE = 1; see Section 4.3.
5. Execute-disable access rights are applied only if IA32_EFER.NXE = 1; see Section 4.6.
6. Not all processors that support IA-32e paging support 1-GByte pages; see Section 4.1.4.

1. Such an address is called canonical. Use of a non-canonical linear address in 64-bit mode produces a general-protection exception (#GP(0)); the processor does not attempt to translate non-canonical linear addresses using IA-32e paging.
4.1.2 Paging-Mode Enabling

If CR0.PG = 1, a logical processor is in one of three paging modes, depending on the values of CR4.PAE and IA32_EFER.LME. Figure 4-1 illustrates how software can enable these modes and make transitions between them. The following items identify certain limitations and other details:

- IA32_EFER.LME cannot be modified while paging is enabled (CR0.PG = 1). Attempts to do so using WRMSR cause a general-protection exception (#GP(0)).
- Paging cannot be enabled (by setting CR0.PG to 1) while CR4.PAE = 0 and IA32_EFER.LME = 1. Attempts to do so using MOV to CR0 cause a general-protection exception (#GP(0)).

Figure 4-1. Enabling and Changing Paging Modes
• CR4.PAE cannot be cleared while IA-32e paging is active (CR0.PG = 1 and IA32_EFER.LME = 1). Attempts to do so using MOV to CR4 cause a general-protection exception (#GP(0)).
• Software can always disable paging by clearing CR0.PG with MOV to CR0.
• Software can make transitions between 32-bit paging and PAE paging by changing the value of CR4.PAE with MOV to CR4.
• Software cannot make transitions directly between IA-32e paging and either of the other two paging modes. It must first disable paging (by clearing CR0.PG with MOV to CR0), then set CR4.PAE and IA32_EFER.LME to the desired values (with MOV to CR4 and WRMSR), and then re-enable paging (by setting CR0.PG with MOV to CR0). As noted earlier, an attempt to clear either CR4.PAE or IA32_EFER.LME cause a general-protection exception (#GP(0)).
• VMX transitions allow transitions between paging modes that are not possible using MOV to CR or WRMSR. This is because VMX transitions can load CR0, CR4, and IA32_EFER in one operation. See Section 4.11.1.

4.1.3 Paging-Mode Modifiers
Details of how each paging mode operates are determined by the following control bits:
• The WP flag in CR0 (bit 16).
• The PSE and PGE flags in CR4 (bit 4 and bit 7, respectively).
• The NXE flag in the IA32_EFER MSR (bit 11).

CR0.WP allows pages to be protected from supervisor-mode writes. If CR0.WP = 0, software operating with CPL < 3 (supervisor mode) can write to linear addresses with read-only access rights; if CR0.WP = 1, it cannot. (Software operating with CPL = 3 — user mode — cannot write to linear addresses with read-only access rights, regardless of the value of CR0.WP.) Section 4.6 explains how access rights are determined.

CR4.PSE enables 4-MByte pages for 32-bit paging. If CR4.PSE = 0, 32-bit paging can use only 4-KByte pages; if CR4.PSE = 1, 32-bit paging can use both 4-KByte pages and 4-MByte pages. See Section 4.3 for more information. (PAE paging and IA-32e paging can use multiple page sizes regardless of the value of CR4.PSE.)

CR4.PGE enables global pages. If CR4.PGE = 0, no translations are shared across address spaces; if CR4.PGE = 1, specified translations may be shared across address spaces. See Section 4.10.1.4 for more information.

IA32_EFER.NXE enables execute-disable access rights for PAE paging and IA-32e paging. If IA32_EFER.NXE = 0, software may fetch instructions from any linear address that paging allows the software to read; if IA32_EFER.NXE = 1, instructions fetches can be prevented from specified linear addresses (even if data reads from the addresses are allowed). Section 4.6 explains how access rights are determined. (32-bit paging always allows software to fetch instructions from any linear address that...
may be read; IA32_EFER.NXE has no effect with 32-bit paging. Software that wants to limit instruction fetches from readable pages must use either PAE paging or IA-32e paging.)

4.1.4 Enumeration of Paging Features by CPUID

Software can discover support for different paging features using the CPUID instruction:

- **PSE**: page-size extensions for 32-bit paging.
  If CPUID.01H:EDX.PSE [bit 3] = 1, CR4.PSE may be set to 1, enabling support for 4-MByte pages with 32-bit paging (see Section 4.3).

- **PAE**: physical-address extension.
  If CPUID.01H:EDX.PAE [bit 6] = 1, CR4.PAE may be set to 1, enabling PAE paging (this setting is also required for IA-32e paging).

- **PGE**: global-page support.
  If CPUID.01H:EDX.PGE [bit 13] = 1, CR4.PGE may be set to 1, enabling the global-page feature (see Section 4.10.1.4).

- **PAT**: page-attribute table.
  If CPUID.01H:EDX.PAT [bit 16] = 1, the 8-entry page-attribute table (PAT) is supported. When the PAT is supported, three bits in certain paging-structure entries select a memory type (used to determine type of caching used) from the PAT (see Section 4.9).

- **PSE-36**: 36-Bit page size extension.
  If CPUID.01H:EDX.PSE-36 [bit 17] = 1, the PSE-36 mechanism is supported, indicating that translations using 4-MByte pages with 32-bit paging may produce physical addresses with more than 32 bits (see Section 4.3).

- **NX**: execute disable.
  If CPUID.80000001H:EDX.NX [bit 20] = 1, IA32_EFER.NXE may be set to 1, allowing PAE paging and IA-32e paging to disable execute access to selected pages (see Section 4.6). (Processors that do not support CPUID function 80000001H do not allow IA32_EFER.NXE to be set to 1.)

- **Page1GB**: 1-GByte pages.
  If CPUID.80000001H:EDX.Page1GB [bit 26] = 1, 1-GByte pages are supported with IA-32e paging (see Section 4.5).

- **LM**: IA-32e mode support.
  If CPUID.80000001H:EDX.LM [bit 29] = 1, IA32_EFER.LME may be set to 1, enabling IA-32e paging. (Processors that do not support CPUID function 80000001H do not allow IA32_EFER.LME to be set to 1.)

- **CPUID.80000008H:EAX[7:0] reports the physical-address width supported by the processor.** (For processors that do not support CPUID function 80000008H, the width is generally 36 if CPUID.01H:EDX.PAE [bit 6] = 1 and 32 otherwise.) This width is referred to as MAXPHYADDR. MAXPHYADDR is at most 52.
CPUID.80000008H:EAX[15:8] reports the linear-address width supported by the processor. Generally, this value is 48 if CPUID.80000001H:EDX.LM [bit 29] = 1 and 32 otherwise. (Processors that do not support CPUID function 80000008H, support a linear-address width of 32.)

4.2 HIERARCHICAL PAGING STRUCTURES: AN OVERVIEW

All three paging modes translate linear addresses using hierarchical paging structures. This section provides an overview of their operation. Section 4.3, Section 4.4, and Section 4.5 provide details for the three paging modes.

Every paging structure is 4096 Bytes in size and comprises a number of individual entries. With 32-bit paging, each entry is 32 bits (4 bytes); there are thus 1024 entries in each structure. With PAE paging and IA-32e paging, each entry is 64 bits (8 bytes); there are thus 512 entries in each structure. (PAE paging includes one exception, a paging structure that is 32 bytes in size, containing 4 64-bit entries.)

The processor uses the upper portion of a linear address to identify a series of paging-structure entries. The last of these entries identifies the physical address of the region to which the linear address translates (called the page frame). The lower portion of the linear address (called the page offset) identifies the specific address within that region to which the linear address translates.

Each paging-structure entry contains a physical address, which is either the address of another paging structure or the address of a page frame. In the first case, the entry is said to reference the other paging structure; in the latter, the entry is said to map a page.

The first paging structure used for any translation is located at the physical address in CR3. A linear address is translated using the following iterative procedure. A portion of the linear address (initially the uppermost bits) select an entry in a paging structure (initially the one located using CR3). If that entry references another paging structure, the process continues with that paging structure and with the portion of the linear address immediately below that just used. If instead the entry maps a page, the process completes: the physical address in the entry is that of the page frame and the remaining lower portion of the linear address is the page offset.

The following items give an example for each of the three paging modes (each example locates a 4-KByte page frame):

- With 32-bit paging, each paging structure comprises \(2^{10}\) entries. For this reason, the translation process uses 10 bits at a time from a 32-bit linear address. Bits 31:22 identify the first paging-structure entry and bits 21:12 identify a second. The latter identifies the page frame. Bits 11:0 of the linear address are the page offset within the 4-kByte page frame. (See Figure 4-2 for an illustration.)

- With PAE paging, the first paging structure comprises only \(2^2\) entries. Translation thus begins by using bits 31:30 from a 32-bit linear address to identify the first paging-structure entry. Other paging structures comprise
512 \(=2^9\) entries, so the process continues by using 9 bits at a time. Bits 29:21 identify a second paging-structure entry and bits 20:12 identify a third. This last identifies the page frame. (See Figure 4-5 for an illustration.)

- With IA-32e paging, each paging structure comprises 512 = 2\(^9\) entries and translation uses 9 bits at a time from a 48-bit linear address. Bits 47:39 identify the first paging-structure entry, bits 38:30 identify a second, bits 29:21 a third, and bits 20:12 identify a fourth. Again, the last identifies the page frame. (See Figure 4-8 for an illustration.)

The translation process in each of the examples above completes by identifying a page frame. However, the paging structures may be configured so that translation terminates before doing so. This occurs if process encounters a paging-structure entry that is marked “not present” (because its P flag — bit 0 — is clear) or in which a reserved bit is set. In this case, there is no translation for the linear address; an access to that address causes a page-fault exception (see Section 4.7).

In the examples above, a paging-structure entry maps a page with 4-KByte page frame when only 12 bits remain in the linear address; entries identified earlier always reference other paging structures. That may not apply in other cases. The following items identify when an entry maps a page and when it references another paging structure:

- If more than 12 bits remain in the linear address, bit 7 (PS — page size) of the current paging-structure entry is consulted. If the bit is 0, the entry references another paging structure; if the bit is 1, the entry maps a page.
- If only 12 bits remain in the linear address, the current paging-structure entry always maps a page (bit 7 is used for other purposes).

If a paging-structure entry maps a page when more than 12 bits remain in the linear address, the entry identifies a page frame larger than 4 KBytes. For example, 32-bit paging uses the upper 10 bits of a linear address to locate the first paging-structure entry; 22 bits remain. If that entry maps a page, the page frame is \(2^{22}\) Bytes = 4 MBytes. 32-bit paging supports 4-MByte pages if CR4.PSE = 1. PAE paging and IA-32e paging support 2-MByte pages (regardless of the value of CR4.PSE). IA-32e paging may support 1-GByte pages (see Section 4.1.4).

Paging structures are given different names based their uses in the translation process. Table 4-2 gives the names of the different paging structures. It also provides, for each structure, the source of the physical address used to locate it (CR3 or a different paging-structure entry); the bits in the linear address used to select an entry from the structure; and details of about whether and how such an entry can map a page.

### 4.3 32-BIT PAGING

A logical processor uses 32-bit paging if CR0.PG = 1 and CR4.PAE = 0. 32-bit paging translates 32-bit linear addresses to 40-bit physical addresses.\(^1\) Although 40 bits
Vol. 3 4-9

PAGING

corresponds to 1 TByte, linear addresses are limited to 32 bits; at most 4 GBytes of linear-address space may be accessed at any given time.

32-bit paging uses a hierarchy of paging structures to produce a translation for a linear address. CR3 is used to locate the first paging-structure, the page directory. Table 4-3 illustrates how CR3 is used with 32-bit paging.

32-bit paging may map linear addresses to either 4-KByte pages or 4-MByte pages. Figure 4-2 illustrates the translation process when it uses a 4-KByte page; Figure 4-3 covers the case of a 4-MByte page. The following items describe the 32-bit paging process in more detail as well as how the page size is determined:

1. Bits in the range 39:32 are 0 in any physical address used by 32-bit paging except those used to map 4-MByte pages. If the processor does not support the PSE-36 mechanism, this is true also for physical addresses used to map 4-MByte pages. If the processor does support the PSE-36 mechanism and MAXPHYADDR < 40, bits in the range 39:MAXPHYADDR are 0 in any physical address used to map a 4-MByte page. (The corresponding bits are reserved in PDEs.) See Section 4.1.4 for how to determine MAXPHYADDR and whether the PSE-36 mechanism is supported.

Table 4-2. Paging Structures in the Different Paging Modes

<table>
<thead>
<tr>
<th>Paging Structure</th>
<th>Entry Name</th>
<th>Paging Mode</th>
<th>Physical Address of Structure</th>
<th>Bits Selecting Entry</th>
<th>Page Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>PML4 table</td>
<td>PML4E</td>
<td>32-bit, PAE</td>
<td>N/A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IA-32e</td>
<td>32-bit, PAE</td>
<td></td>
<td>N/A (PS must be 0)</td>
<td></td>
</tr>
<tr>
<td>Page-directory-pointer table</td>
<td>PDTE</td>
<td>32-bit</td>
<td>N/A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PAE</td>
<td>32-bit, CR3</td>
<td></td>
<td>N/A (PS must be 0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>IA-32e</td>
<td>32-bit, CR3</td>
<td>31:30</td>
<td>1-GByte page if PS=1</td>
<td></td>
</tr>
<tr>
<td>Page directory</td>
<td>PDE</td>
<td>32-bit, CR3</td>
<td></td>
<td>4-MByte page if PS=1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PAE, IA-32e</td>
<td>32-bit, CR3</td>
<td>31:22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page table</td>
<td>PTE</td>
<td>32-bit, PDE</td>
<td>21:12</td>
<td>4-KByte page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PAE, IA-32e</td>
<td>32-bit, PDE</td>
<td>20:12</td>
<td>4-KByte page</td>
<td></td>
</tr>
</tbody>
</table>

NOTES:
1. Not all processors allow the PS flag to be 1 in PDPTES; see Section 4.1.4 for how to determine whether 1-GByte pages are supported.
2. 32-bit paging ignores the PS flag in a PDE (and uses the entry to reference a page table) unless CR4.PSE = 1. Not all processors allow CR4.PSE to be 1; see Section 4.1.4 for how to determine whether 4-MByte pages are supported with 32-bit paging.
PAGING

Table 4-3. Use of CR3 with 32-Bit Paging

<table>
<thead>
<tr>
<th>Bit Position(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>2:0</td>
<td>Ignored</td>
</tr>
<tr>
<td>3 (PWT)</td>
<td>Page-level write-through; indirectly determines the memory type used to access the page directory during linear-address translation (see Section 4.9)</td>
</tr>
<tr>
<td>4 (PCD)</td>
<td>Page-level cache disable; indirectly determines the memory type used to access the page directory during linear-address translation (see Section 4.9)</td>
</tr>
<tr>
<td>11:5</td>
<td>Ignored</td>
</tr>
<tr>
<td>31:12</td>
<td>Physical address of the 4-KByte aligned page directory used for linear-address translation</td>
</tr>
<tr>
<td>63:32</td>
<td>Ignored (these bits exist only on processors supporting the Intel-64 architecture)</td>
</tr>
</tbody>
</table>

![Figure 4-2. Linear-Address Translation to a 4-KByte Page using 32-Bit Paging](image)

- A 4-KByte naturally aligned page directory is located at the physical address specified in bits 31:12 of CR3 (see Table 4-3). A page directory comprises 1024 32-bit entries (PDEs). A PDE is selected using the physical address defined as follows:
  - Bits 39:32 are all 0.
  - Bits 31:12 are from CR3.
Because a PDE is identified using bits 31:22 of the linear address, it controls access to a 4-Mbyte region of the linear-address space. Use of the PDE depends on CR.PSE and the PDE’s PS flag (bit 7):

- If CR4.PSE = 1 and the PDE’s PS flag is 1, the PDE maps a 4-MByte page (see Table 4-4). The final physical address is computed as follows:
  - Bits 39:32 are bits 20:13 of the PDE.
  - Bits 31:22 are bits 31:22 of the PDE.\(^1\)
  - Bits 21:0 are from the original linear address.

- If CR4.PSE = 0 or the PDE’s PS flag is 0, a 4-KByte naturally aligned page table is located at the physical address specified in bits 31:12 of the PDE (see Table 4-5). A page table comprises 1024 32-bit entries (PTEs). A PTE is selected using the physical address defined as follows:
  - Bits 39:32 are all 0.
  - Bits 31:12 are from the PDE.
  - Bits 11:2 are bits 21:12 of the linear address.
  - Bits 1:0 are 0.

\(^1\) The upper bits in the final physical address do not all come from corresponding positions in the PDE; the physical-address bits in the PDE are not all contiguous.
Table 4-4. Format of a 32-Bit Page-Directory Entry that Maps a 4-MByte Page

<table>
<thead>
<tr>
<th>Bit Position(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (P)</td>
<td>Present; must be 1 to map a 4-MByte page</td>
</tr>
<tr>
<td>1 (R/W)</td>
<td>Read/write; if 0, writes may not be allowed to the 4-MByte page referenced by this entry (depends on CPL and CR0.WP; see Section 4.6)</td>
</tr>
<tr>
<td>2 (U/S)</td>
<td>User/supervisor; if 0, accesses with CPL=3 are not allowed to the 4-MByte page referenced by this entry (see Section 4.6)</td>
</tr>
<tr>
<td>3 (PWT)</td>
<td>Page-level write-through; indirectly determines the memory type used to access the 4-MByte page referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>4 (PCD)</td>
<td>Page-level cache disable; indirectly determines the memory type used to access the 4-MByte page referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>5 (A)</td>
<td>Accessed; indicates whether software has accessed the 4-MByte page referenced by this entry (see Section 4.8)</td>
</tr>
<tr>
<td>6 (D)</td>
<td>Dirty; indicates whether software has written to the 4-MByte page referenced by this entry (see Section 4.8)</td>
</tr>
<tr>
<td>7 (PS)</td>
<td>Page size; must be 1 (otherwise, this entry references a page table; see Table 4-5)</td>
</tr>
<tr>
<td>8 (G)</td>
<td>Global; if CR4.PGE = 1, determines whether the translation is global (see Section 4.10); ignored otherwise</td>
</tr>
<tr>
<td>11:9</td>
<td>Ignored</td>
</tr>
<tr>
<td>12 (PAT)</td>
<td>If the PAT is supported, indirectly determines the memory type used to access the 4-MByte page referenced by this entry (see Section 4.9); otherwise, reserved (must be 0)(^1)</td>
</tr>
<tr>
<td>(M–20):13</td>
<td>Bits (M–1):32 of physical address of the 4-MByte page referenced by this entry(^2)</td>
</tr>
<tr>
<td>21:(M–19)</td>
<td>Reserved (must be 0)</td>
</tr>
<tr>
<td>31:22</td>
<td>Bits 31:22 of physical address of the 4-MByte page referenced by this entry</td>
</tr>
</tbody>
</table>

**NOTES:**
1. See Section 4.1.4 for how to determine whether the PAT is supported.
2. If the PSE-36 mechanism is not supported, M is 32, and this row does not apply. If the PSE-36 mechanism is supported, M is the minimum of 40 and MAXPHYADDR (this row does not apply if MAXPHYADDR = 32). See Section 4.1.4 for how to determine MAXPHYADDR and whether the PSE-36 mechanism is supported.
Because a PTE is identified using bits 31:12 of the linear address, every PTE maps a 4-KByte page (see Table 4-6). The final physical address is computed as follows:

- Bits 39:32 are all 0.
- Bits 31:12 are from the PTE.
- Bits 11:0 are from the original linear address.

If a paging-structure entry’s P flag (bit 0) is 0 or if the entry sets any reserved bit, the entry is used neither to reference another paging-structure entry nor to map a page. A reference using a linear address whose translation would use such a paging-structure entry causes a page-fault exception (see Section 4.7).

With 32-bit paging, there are reserved bits only if CR4.PSE = 1:

- If the P flag and the PS flag (bit 7) of a PDE are both 1, the bits reserved depend on MAXPHYADDR whether the PSE-36 mechanism is supported.\(^1\)

---

Table 4-5. Format of a 32-Bit Page-Directory Entry that References a Page Table

<table>
<thead>
<tr>
<th>Bit Position(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (P)</td>
<td>Present; must be 1 to reference a page table</td>
</tr>
<tr>
<td>1 (R/W)</td>
<td>Read/write; if 0, writes may not be allowed to the 4-MByte region controlled by this entry (depends on CPL and CR0.WP; see Section 4.6)</td>
</tr>
<tr>
<td>2 (U/S)</td>
<td>User/supervisor; if 0, accesses with CPL=3 are not allowed to the 4-MByte region controlled by this entry (see Section 4.6)</td>
</tr>
<tr>
<td>3 (PWT)</td>
<td>Page-level write-through; indirectly determines the memory type used to access the page table referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>4 (PCD)</td>
<td>Page-level cache disable; indirectly determines the memory type used to access the page table referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>5 (A)</td>
<td>Accessed; indicates whether this entry has been used for linear-address translation (see Section 4.8)</td>
</tr>
<tr>
<td>6</td>
<td>Ignored</td>
</tr>
<tr>
<td>7 (PS)</td>
<td>If CR4.PSE = 1, must be 0 (otherwise, this entry maps a 4-MByte page; see Table 4-4); otherwise, ignored</td>
</tr>
<tr>
<td>11:8</td>
<td>Ignored</td>
</tr>
<tr>
<td>31:12</td>
<td>Physical address of 4-KByte aligned page table referenced by this entry</td>
</tr>
</tbody>
</table>

---

1. See Section 1.1.5 for how to determine MAXPHYADDR and whether the PSE-36 mechanism is supported.

\(^1\)
Table 4-6. Format of a 32-Bit Page-Table Entry that Maps a 4-KByte Page

<table>
<thead>
<tr>
<th>Bit Position(s)</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (P)</td>
<td>Present; must be 1 to map a 4-KByte page</td>
</tr>
<tr>
<td>1 (R/W)</td>
<td>Read/write; if 0, writes may not be allowed to the 4-KByte page referenced by this entry (depends on CPL and CR0.WP; see Section 4.6)</td>
</tr>
<tr>
<td>2 (U/S)</td>
<td>User/supervisor; if 0, accesses with CPL=3 are not allowed to the 4-KByte page referenced by this entry (see Section 4.6)</td>
</tr>
<tr>
<td>3 (PWT)</td>
<td>Page-level write-through; indirectly determines the memory type used to access the 4-KByte page referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>4 (PCD)</td>
<td>Page-level cache disable; indirectly determines the memory type used to access the 4-KByte page referenced by this entry (see Section 4.9)</td>
</tr>
<tr>
<td>5 (A)</td>
<td>Accessed; indicates whether software has accessed the 4-KByte page referenced by this entry (see Section 4.8)</td>
</tr>
<tr>
<td>6 (D)</td>
<td>Dirty; indicates whether software has written to the 4-KByte page referenced by this entry (see Section 4.8)</td>
</tr>
<tr>
<td>7 (PAT)</td>
<td>If the PAT is supported, indirectly determines the memory type used to access the 4-KByte page referenced by this entry (see Section 4.9); otherwise, reserved (must be 0)</td>
</tr>
<tr>
<td>8 (G)</td>
<td>Global; if CR4.PGE = 1, determines whether the translation is global (see Section 4.10); ignored otherwise</td>
</tr>
<tr>
<td>11:9</td>
<td>Ignored</td>
</tr>
<tr>
<td>31:12</td>
<td>Physical address of the 4-KByte page referenced by this entry</td>
</tr>
</tbody>
</table>

NOTES:
1. See Section 4.1.4 for how to determine whether the PAT is supported.

   — If the PSE-36 mechanism is not supported, bits 21:13 are reserved.
   — If the PSE-36 mechanism is supported, bits 21:(M–19) are reserved, where M is the minimum of 40 and MAXPHYADDR.

• If the PAT is not supported: 1
   — If the P flag of a PTE is 1, bit 7 is reserved.
   — If the P flag and the PS flag of a PDE are both 1, bit 12 is reserved.

(If CR4.PSE = 0, no bits are reserved with 32-bit paging.)

1. See Section 4.1.4 for how to determine whether the PAT is supported.
A reference using a linear address that is successfully translated to a physical address is performed only if allowed by the access rights of the translation; see Section 4.6.

Figure 4-4 gives a summary of the formats of CR3 and the paging-structure entries with 32-bit paging. For the paging structure entries, it identifies separately the format of entries that map pages, those that reference other paging structures, and those that do neither because they are "not present"; bit 0 (P) and bit 7 (PS) are highlighted because they determine how such an entry is used.

![Figure 4-4. Formats of CR3 and Paging-Structure Entries with 32-Bit Paging](Figure 4-4.png)

**NOTES:**
1. CR3 has 64 bits on processors supporting the Intel-64 architecture. These bits are ignored with 32-bit paging.
2. This example illustrates a processor in which MAXPHYADDR is 36. If this value is larger or smaller, the number of bits reserved in positions 20:13 of a PDE mapping a 4-MByte will change.

### 4.4 PAE PAGING

A logical processor uses PAE paging if CR0.PG = 1, CR4.PAE = 1, and IA32_EFER.LME = 0. PAE paging translates 32-bit linear addresses to 52-bit physical addresses. Although 52 bits corresponds to 4 PBytes, linear addresses are limited to 32 bits; at most 4 GBytes of linear-address space may be accessed at any given time.