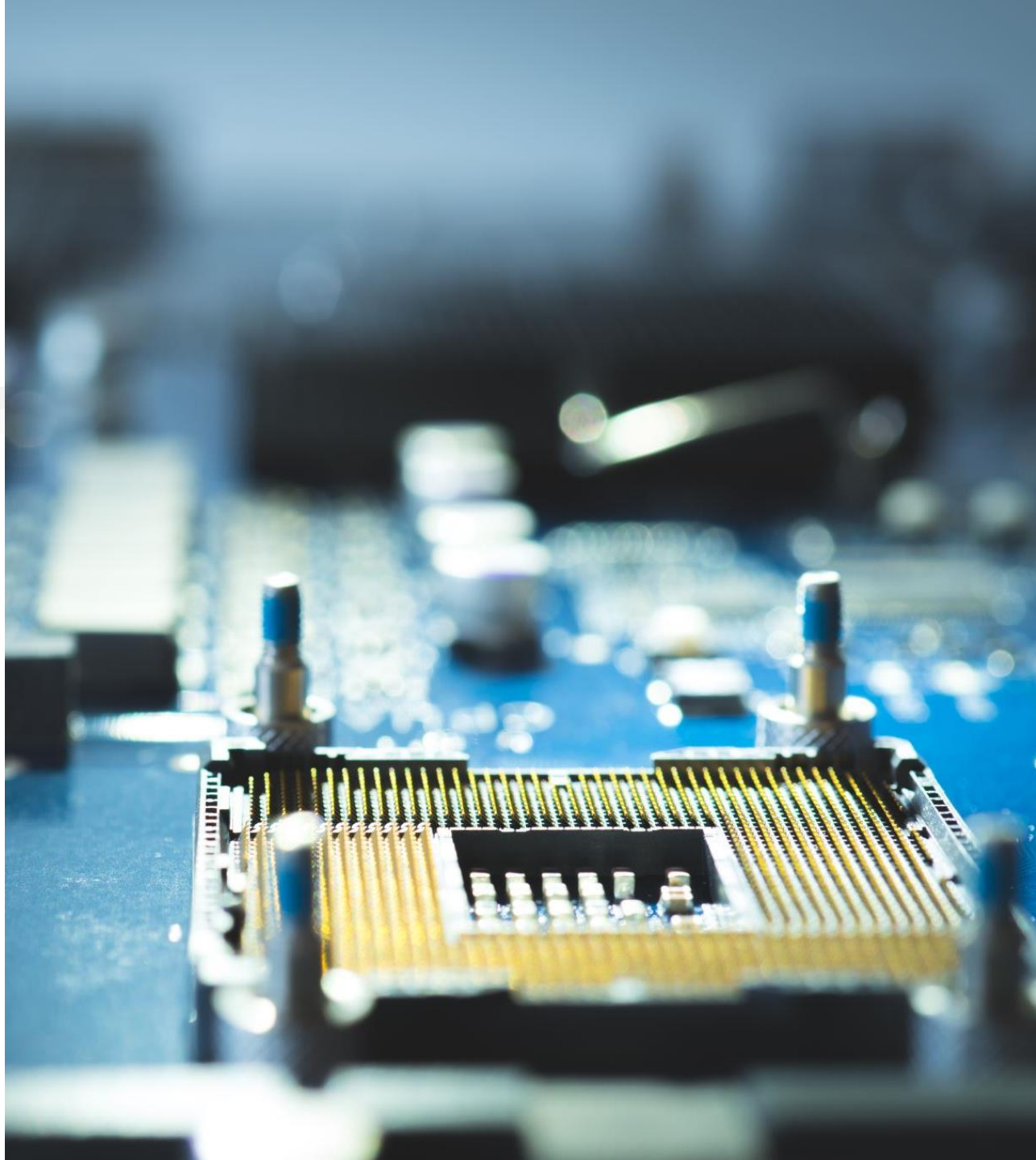
Memory Paging

Intel 32-bit Architecture(with Linux examples)

Memory Paging Overview

- Paging is a memory management scheme used by an operating system to allow the physical address space of a process to be noncontiguous
- Physical memory is broken into fixed size blocks called frames and logical memory from the cpu running a process is broken into blocks of the same size called pages
- Logical address space is separated from physical address space so processes can have larger address spaces than the system has physical memory
- Pages are mapped to physical memory frames through page tables

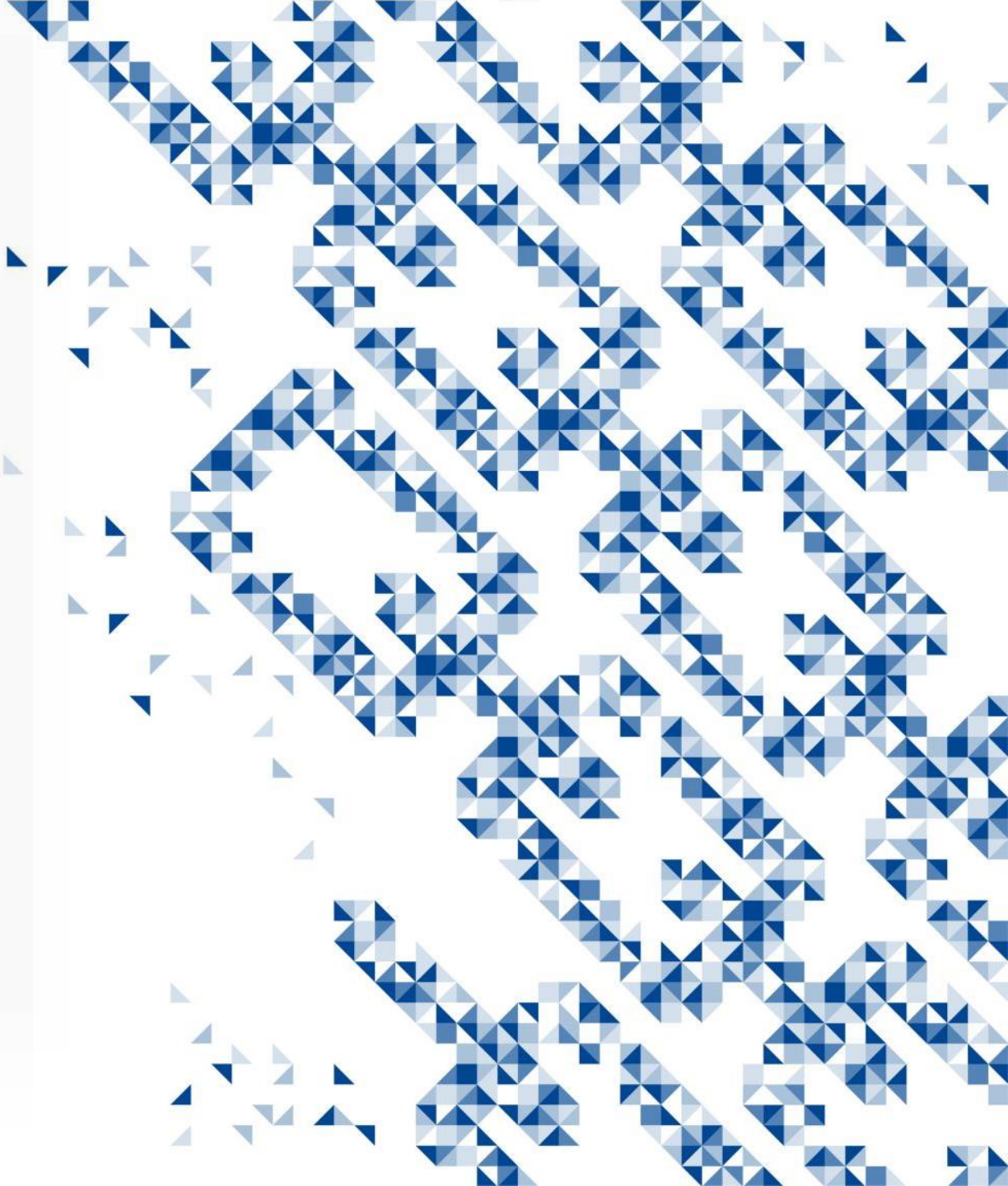


Intel & IA-32 Architecture

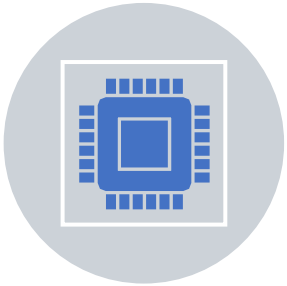
- Intel produced 32-bit chips which use the IA-32 architecture
- Memory management in systems that use IA-32 architecture is divided in two components which are segmentation and paging
- This involves logical addresses generated by the cpu being sent to a segmentation unit that produces linear addresses
- These linear addresses are then sent to a paging that generates the physical memory address
- CPU (logical address) => segmentation unit(linear address) => paging unit(physical address) => physical memory

IA-32 Segmentation

- IA-32 segmentation divides the logical address space into segments
- The logic address now has a segment selector and an offset
- The selector contains information on the segment number and whether the segment's information is in the local descriptor table(ldt) or the global descriptor table(gdt)
- Each entry in these tables have a descriptor with information about a particular segment such as base location and limit for the segment

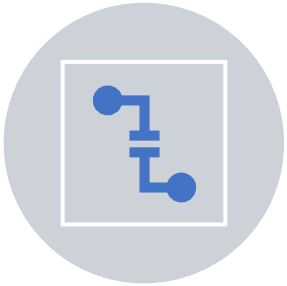


IA-32 Segmentation Continued



The offset is a 32-bit number that contains the location of the byte of a particular segment

The segment register points an appropriate entry in ldt or gdt and the base and limit information generates a linear address



The limit information about the segment is used to determine if the address if valid, if not valid a memory fault is generated

If it is valid the value of the offset is added to the value of the base which produces the 32-bit linear address

IA-32 Paging

IA-32 architecture allows a page to be either 4KB or 4MB in size

It uses a 2-level paging scheme that divides the 32-bit linear address into a higher order 10 bits, innermost 10 bits and a lower order 12 bits that refers to in the 4KB page

The higher order 10 bits points to an outer level page table called the page directory which points to an inner level page table that then points to an offset in a page

The page directory contains a page size flag which has the necessary information to know if the page size is 4MB or 4KB

This is important because the page directory points directly to the 4MB page bypassing the need for the inner page and the last 22 bits are used for an offset in a 4MB page

Efficiency & Page Address Extension(PAE)

IA 32-page tables can be swapped to the disk, and an invalid bit is used to whether the table is in memory or is on the disk

If the table is on the disk the other 31 bits are used to locate the table on the disk and bring it into memory

Page Address Extensions allows 32-bit processors to access a larger physical address space than 4GB

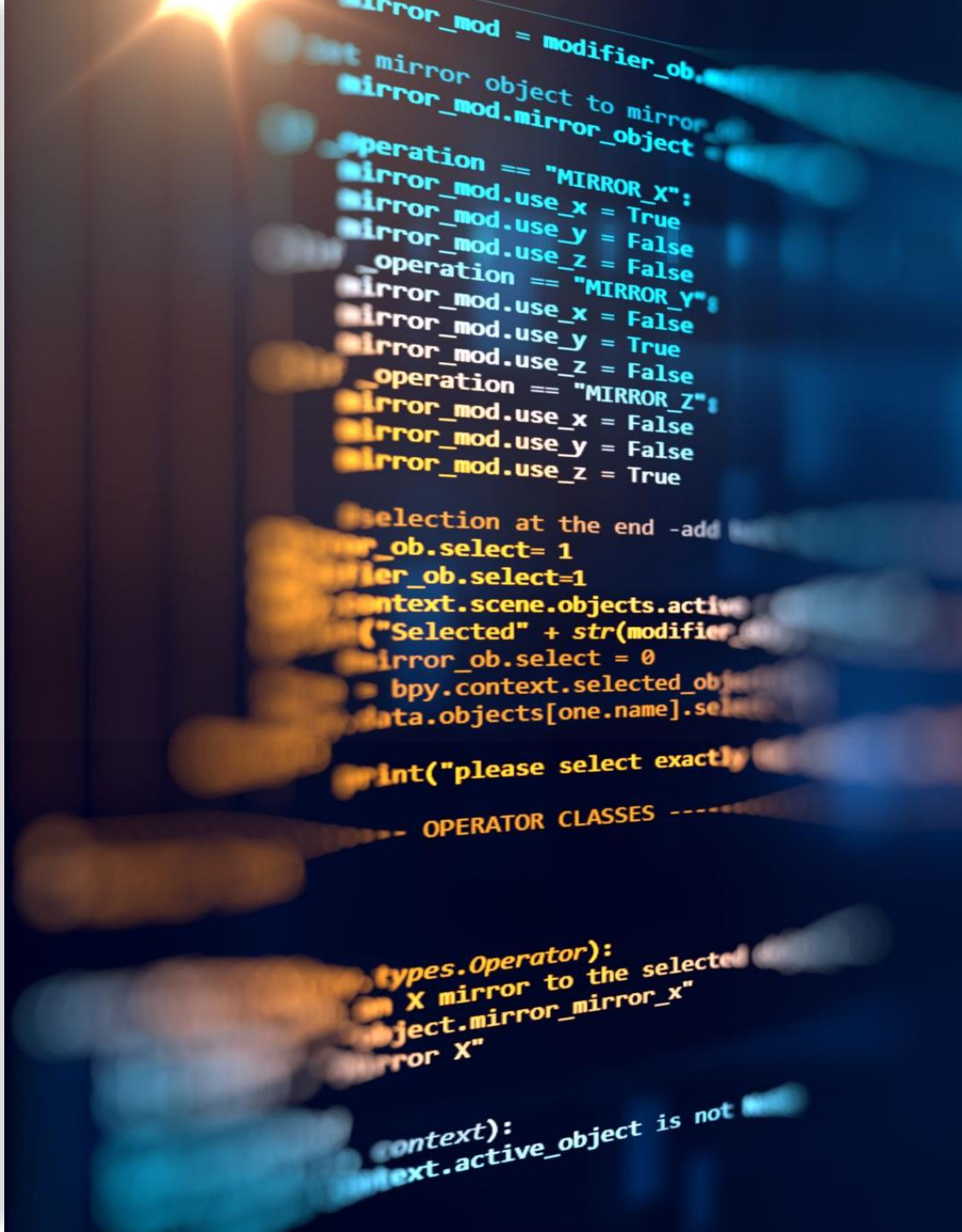
In PAE paging goes from a 2-level scheme to a 3-level scheme where the top two bits refer to a page directory pointer table

PAE increased address space to 36 bits which support 64GB of physical memory

Operating system support is required to use PAE and Linux and Mac operating systems support it

Memory Management in Linux

- Linux relies heavily on memory management to handle concurrent running processes
- Every process on a linux system requires its own memory space to store its code, data, and stack and the allocation of this memory is handled by the kernel
- A running process’s memory is divided into different segments such as the text segment which contains code, the data segment which contains globala and static variables, the heap which has dynamically allocated memory, and the stack which has function calls and local variables



Virtual Memory in Linux

Virtual memory is a vital part of Linux's memory management strategy because the processes can get a memory space much larger than the physical RAM available

Paging is one of the methods through which this is achieved

It divides virtual address space into fixed size pages and they can be swapped between RAM and secondary storage as needed

Pages that are not accessed frequently are moved to secondary storage to free up RAM for the processes

Paging In Linux

Paging involves dividing virtual memory into fixed size blocks called pages

A page table maps the virtual address to a physical address

When a process tries to access a page that is not in memory a page fault occurs and the kernel brings the page into RAM from secondary storage

Swapping moves larger portions of memory from RAM to secondar storage and is slower than paging

Efficient management of both are needed for the system to run smoothly

Paging in Linux IA-32

- The mechanism is divided into 2 parts segmentation and paging
- The logical address bit contains two parts one is a segment selector, and the other is an offset
- The segment selector points to an entry in the GDT table which is a segment descriptor and the segment descriptor combines with the offset from the logical address to form the linear address
- The linear address is part of the page that is formed after segmentation and lives in linear address space
- The linear address is then used to entry into the page directory

Paging in Linux IA-32 Continued

- The paging portion begins once a page and linear address have been created
- The linear address consists of three parts, the directory, the table and the offset
- The directory is the specific portion that references a page directory entry
- The selected entry along with the table portion of the linear address is used to reference an entry in a page table
- The selected entry in the page table along with the offset from the linear address is used to select a page in physical address space
- In this mechanism the paging process is broken into two parts segmentation and paging and the logical address is ultimately translated into a physical memory address

Sources

- Section 9.3 - CSCI 431: Operating Systems | zyBooks
- Section 9.6 - CSCI 431: Operating Systems | zyBooks
- Why Linux has 4 layers of “page tables” and how it works exactly | by Mayin Dev | Medium
- Segmentation in Intel x64(IA-32e) architecture - explained using Linux