PC and the x86

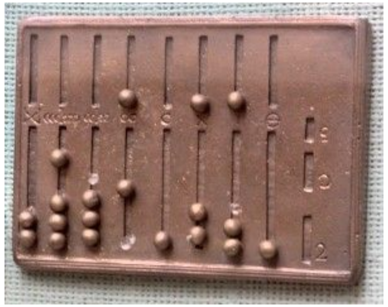
Lesson 3

Fall 2024 FI MU

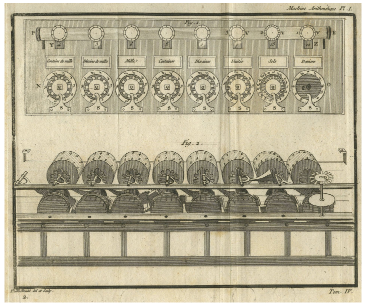
Rado Vrbovsky
<rvrbovsk@redhat.com>

- History of the computer
- IBM Personal Computer, model 5150
- PC architecture
- x86 CPU in a closer look

History of the computer

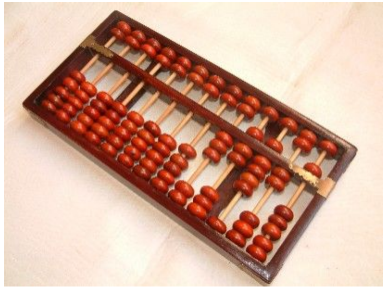


Roman Abacus

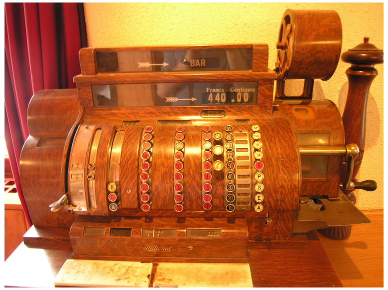


Pascaline

Curta



Zhusuan



Cash register

Mechanical calculators - source Wikipedia

Computers for the Explorer 1 trajectory - source Wikipedia

IBM Personal Computer 5150

Mainframe computer

- "The big iron"
- Large footprint (whole floor)
- Build for redundancy
- Large IO throughput (bulk data processing)
- Multiple OSes running at the same time in VMs
- Many modular peripherals
- Backward compatibility
- Used by US government, NASA, universities
- Still used today

IBM System/360

Mainframes - picture source Wikipedia

Minicomputers

- 16 /32 bit architecture
- Low cost version of a mainframe (but still size of a small truck or fridge)
- Eventually faded out
- Unix was written on DEC PDP-11

DEC PDP-11

Minicomputers - picture source Wikipedia

Microcomputers

- Small form factor
- Microchip CPU, 8 bit or 16 bit
- For home or office use
- Self assembled
- Single user
- Apple, Commodore, Tandy
- ČSSR - Didaktik, PMD

Commodore - 64

- Minicomputer market large enough to be noticed by IBM
- Super secret project inside IBM by a small team
- Use of "off the shelf" components
- Open hardware platform (ISA bus, option ROMs)
- Good open documentation including BIOS source code listing

IBM Personal Computer 5150

IBM Personal Computer – picture source Wikipedia

- Minicomputer market large enough to be noticed by IBM
- Super secret project inside IBM by a small team
- Use of "off the shelf" components
- Open hardware platform (ISA bus, option ROMs)
- Good open documentation including BIOS source code listing
- Easy to copy design -> Many cheap clones and 3rd party extension hardware -> Wide spread of platform

IBM Personal Computer 5150

IBM hardware (set of chips)

- 8086 - CPU
- 8237 - DMA
- 8259 - Interrupt controller
- 8253 - Timer
- 6843 - CRT controller
- 8042 - Keyboard controller
- 8255 - Cassette controller (PPI Programmable Peripheral Interface chip)

IBM Personal Computer 5150

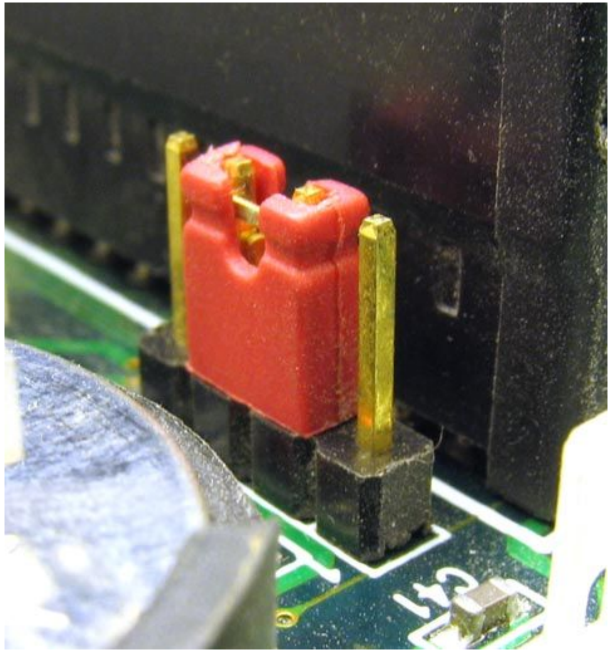
IBM hardware (set of chips)

- 8086 - CPU
- 8237 - DMA
- 8259 - Interrupt controller
- 8253 - Timer
- 6843 - CRT controller
- 8042 - Keyboard controller
- 8255 - Cassette controller (PPI Programmable Peripheral Interface chip)

IBM Personal Computer 5150

Most of the hardware still present in some form.

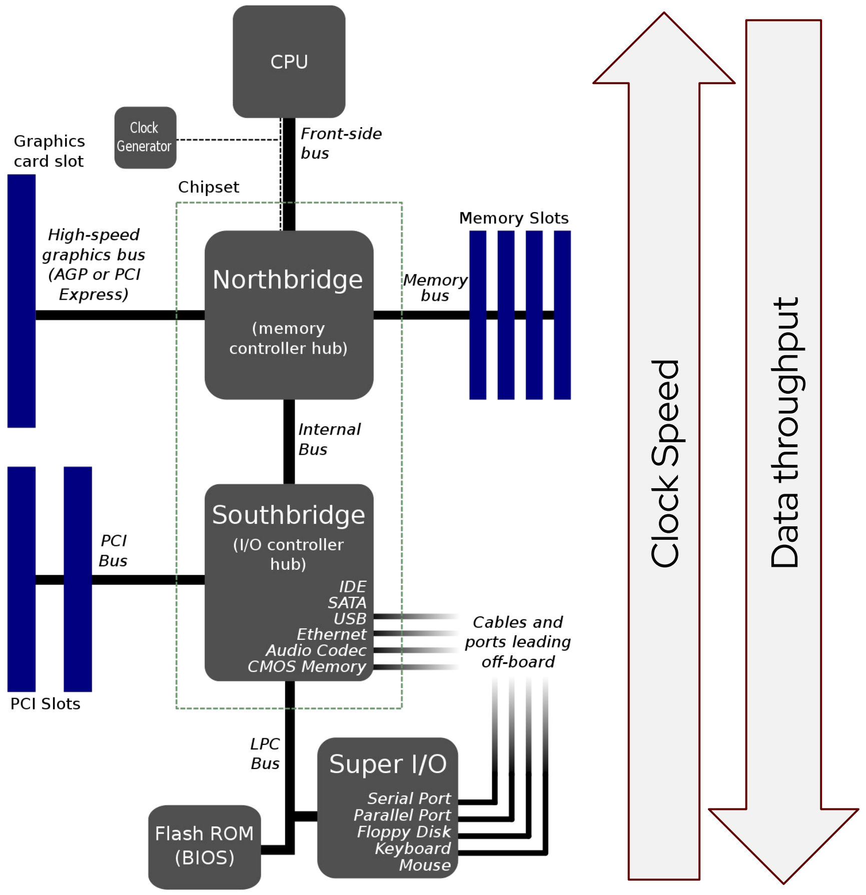
- Fixed boot order
  - When no bootable device was found, option ROM with BASIC was booted
- No interactive BIOS setup tool
- No resource management
  - Manual assignment of resource by hand
  - Conflicts hard to detect
- No power management
  - Hard to make a portable device



Jumper for HW configuration

Jumper – picture source Wikipedia

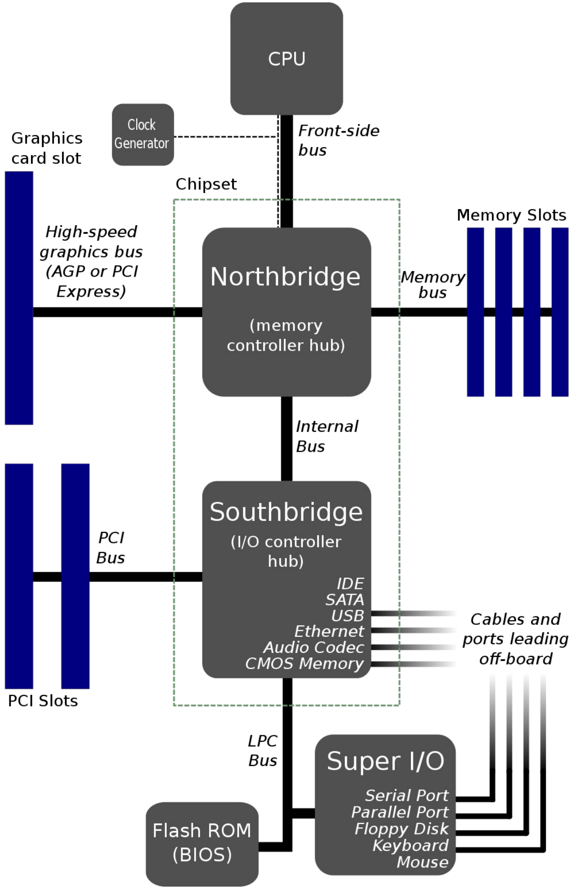
PC Architecture



Historical Insight

- Set of individual chips (8237, 8042, ...) integrated into one large component -> Chipset
- Later split again
  - Speed
  - Divide legacy hardware and the new PCI hardware
- Nowadays Memory Controller or the whole Northbridge controller integrated to CPU

PC Architecture - Historical Insight - picture source Wikipedia



Topology overview

- List of nodes (devices) and links (buses) organized into a tree like structure
- Device
  - A hardware component
    - Storage device
    - Human interface device
    - CPU
    - Bus bridge
    - ...
- Bus
  - A communication system to transfer data between components inside a computer or between computers
  - Bus attributes:
    - Clock speed (Hz, MHz)
    - Style (Parallel/Serial)
    - Data width (8 bit, 16 bit, 32 bit, 64 bit, ...)
    - Duplex (Half duplex, Full duplex)

PC Architecture - Topology overview

ISA – Industry Standard Architecture, (PC Bus or IO Channel)

- Original IBM PC Bus
- Used for expansion cards
- Peripheral devices
- Parallel, 8/16 bit, 4, 8 to 20Mhz
- DMA support
- 16 IRQ (cannot be shared and at least 6 were already used by internal devices)
- 12V, 5V
- All devices are equal

PCI – Peripheral Component Interconnect

- Intel 1992
- Replaces ISA
- Parallel, 32/64 bit, up to 66 MHz
- Tree like structure (up to 255 buses, each bus 32 devices, each device 8 functions (logical device))
- Shared IRQs among PCI devices, IRQs stolen from ISA
- 5V, 3V
- Software accessible configuration space with a vendor and a device ID and a class code

AGP - Accelerated Graphics Port

- Intel 1997
- Parallel, 32 bit
- Specialized PCI bus for graphic card accelerators

PCIx - Peripheral Component Interconnect eXtended

- IBM, HP, and Compaq 1998
- Replaces PCI
- Parallel 32/64 bit, up to 533 MHz
- Hardware and software compatible with PCI
- Higher clock rates

PCIe - Peripheral Component Interconnect Express

- Intel, Dell, HP, IBM 2003
- Replaces PCI, AGP and PCIx
- High speed serial bus
- Not physically backward compatible
- Somewhat backward software compatible

LPC - Low Pin Count

- Intel 1994 as a substitute for ISA, 4 bit wide parallel
- Super IOs, BIOS ROM, Southbridge

eSPI - Enhanced Serial Peripheral Interface

- Substitute for LPC
- 1, 2, 4 bits wide

I2C - Inter-Integrated Circuit

- NXP in 1982
- 2 wire bus (SCL - clock, SDA - data)
- Sensors, EEPROMs, Fan control, EDID data in monitors and displays

SMBus - System Management Bus

- Intel and Duracell 1994
- Derived from I2C
- Used for motherboard devices, e.g. SPD (main memory configuration), laptop charging system (smart battery, embedded controller), sensors, fan, voltage regulator, clock generator

USB - Universal Serial Bus

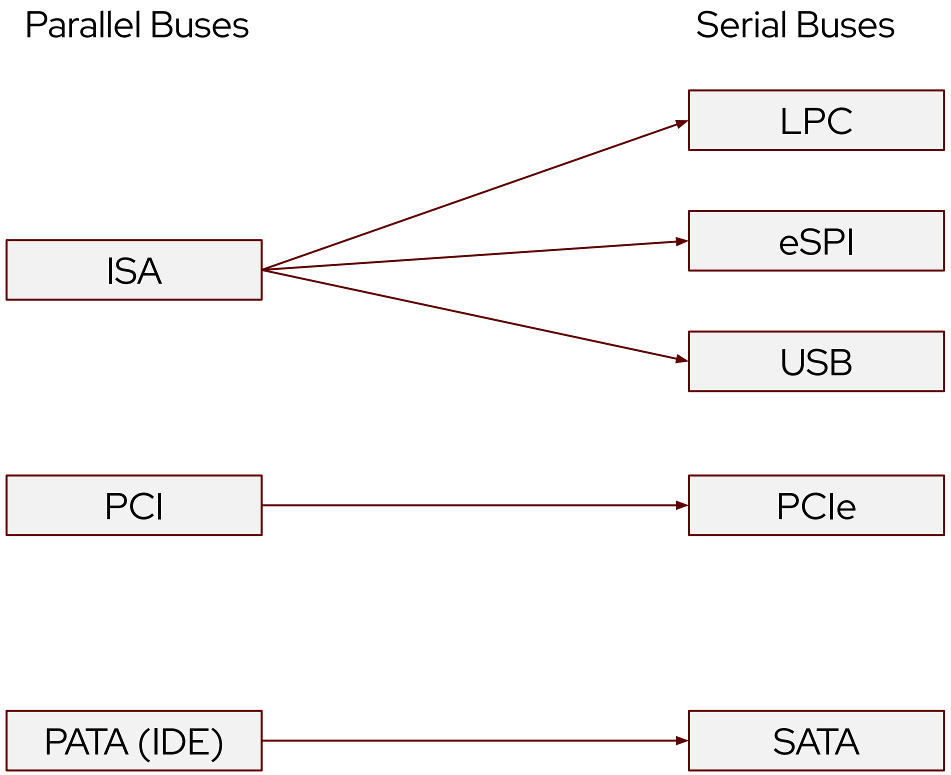
- Compaq, DEC, IBM, Intel, Microsoft, NEC, and Nortel 1996
- Standardize the connection of peripherals to computers
- Replacing UARTs, LPTs, game ports
- Nowadays support everything

ATA - AT Bus Attachment (Parallel ATA, IDE - Integrated Drive Electronics)

- Western digital and Compaq in 1986
- Direct connection to the 16 bit ISA bus for permanent storage (disks)
- Parallel

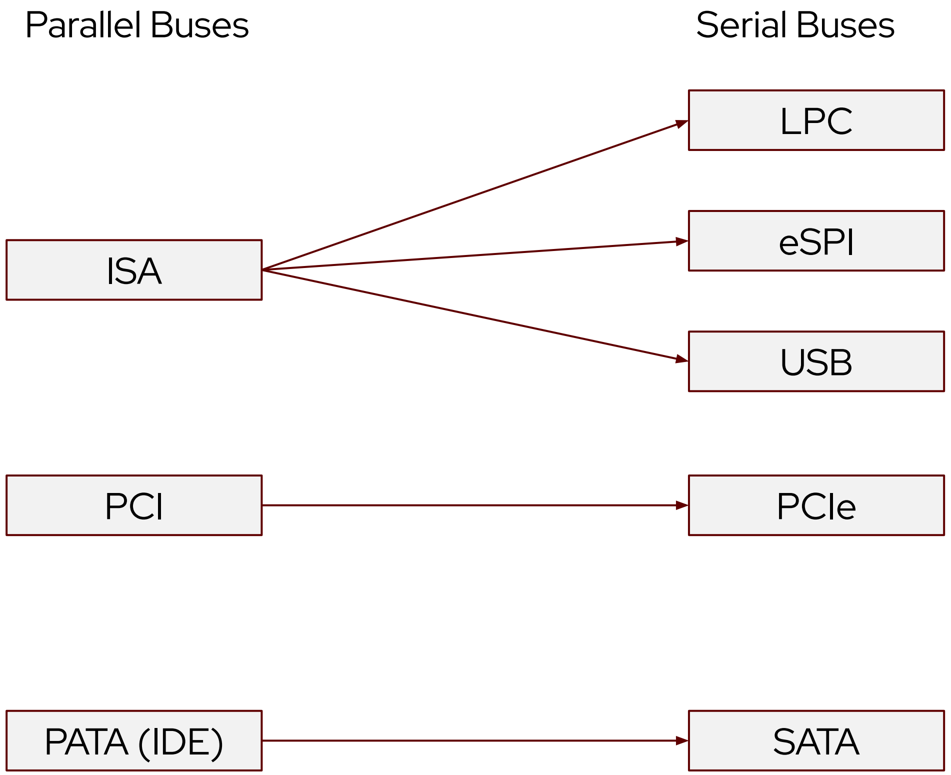
SATA - Serial AT Attachment

- 2000
- Substitute for PATA
- Higher data rates
- Native hot plug support



Over time width of bus is reduced from parallelism to serial.

WHY?



- Reduces risk of clock skew
- Higher clock rates
- Easier PCB design and layout
- Saves space on PCB
- Lower production costs

PC Architecture – Main PC Buses – Serialization of buses

Firmware

- Hardware specific low level software
  - CPU uCode, motherboard firmware, CDROMs, BT devices, hard drives, networking cards, graphic cards

- BIOS - Basic I/O System
  - PC specific firmware
  - Resides in E, F segments
  - POST - Power On Self Test
    - Boot up the computer
    - Enumerate resources for present devices
    - Scans for option ROMs
    - Detects bootable devices
    - Boots up the operating system
  - Abstraction layer between hardware and the OS
  - Provides runtime services for the OS

BIOS chip in a PLCC package

Legacy BIOS

- First IBM BIOS was very simple (fixed list of bootable devices, no HW or power management ...)
- Written in assembly language

UEFI - Unified Extensible Firmware Interface

- New standard since used 2010s
- Written mostly in C language
- Uses CAR (CPU Cache As Ram) for stack and heap before main memory is initialized
- Fully substitutes traditional BIOS, but can provide backward compatibility with Legacy BIOS if needed
- Modular
- Multiplatform (x86, ARM)
- Security by design
- Open Source implementations available

BIOS chip in a PLCC package

Many extensions since the first IBM BIOS

- BBS - BIOS Boot Specification
  - BIOS scans a list of bootable devices and lets user decide what to boot.
- DMI - Desktop Management Interface
  - Information about the hardware from specific vendors
  - Information about the present hardware and its resources
- PNP - Plug And Play
  - Microsoft in the 90s
  - PC was not aware of its own resources
  - Per device configuration with resource profiles
  - BIOS would find a suitable combination of profiles for all the devices
  - BIOS would resolve resource conflicts

BIOS chip in a PLCC package

- APM - Advanced Power Management
  - Written by Microsoft and Intel in 1992
  - APM BIOS is in charge of devices and platform as whole
  - OS participates through APM driver and APM aware applications
  - System and device states can be controlled either by BIOS or APM aware OS through BIOS
- ACPI - Advanced Configuration and Power Interface
  - Released by Microsoft, Intel and Toshiba in 1996
  - All information about the hardware is provided through a single service
  - OS is put in charge of the whole platform management
  - Set of memory mapped tables containing information about the hardware
  - AML - ACPI Machine Language
    - Bytecode containing all hardware specific routines for hardware
    - Code in interpreted by the OS

BIOS chip in a PLCC package

Interrupts (Traps)

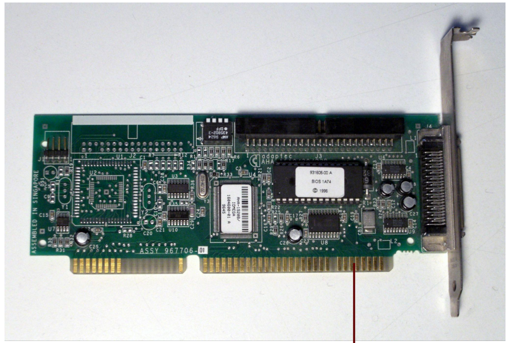
- Started as an alternative to to polling
- Latency - Delay between invoking (triggering, raising) and interrupt and running the software callback (handler)
- Can be masked (ignored) while running critical code
- Transparent to user space

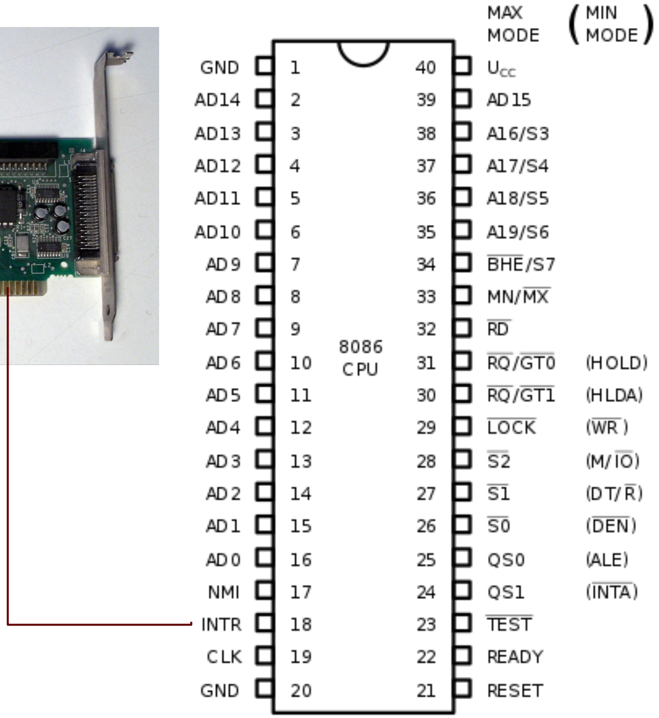
Hardware Interrupt (IRQs)

- Physical connection between a CPU and a device (SCSI, Net, Sound Card, …)
- Serves as a notification from the device to the CPU

Software Interrupt

- A callback function to handle a specific IRQ (interrupt handler)
- Can be also triggered by software by a special instruction
- 8086 had 256 interrupts, each interrupt mapped to a different callback





The 8086 pin assignments in min and max mode

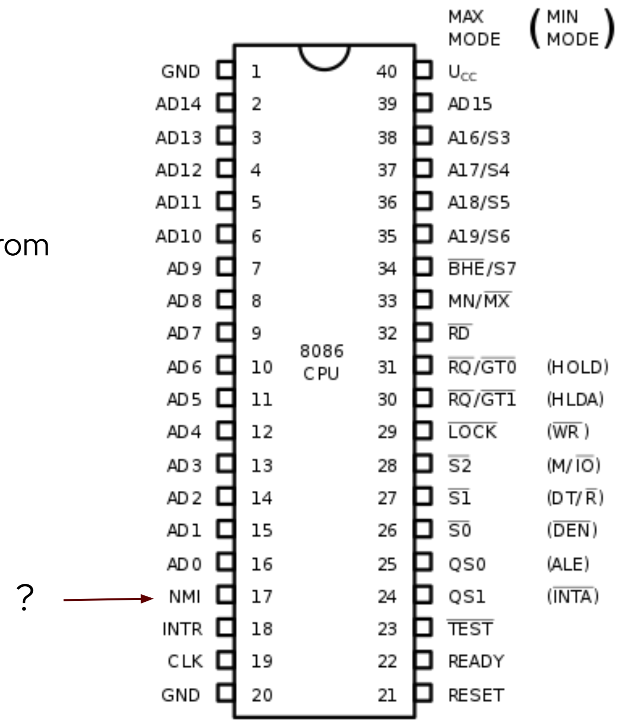
Red Hat

Non Maskable Interrupts

- Like regular interrupts, but cannot be masked (ignored)
- Can occur while handling a different interrupt!
- Once NMI is handled, another cannot be serviced until IRET (Return from Interrupt) instruction is executed
- Used for critical events
    - Critical failure is eminent
    - Data loss or data corruption
    - Watchdog is triggered

Exceptions

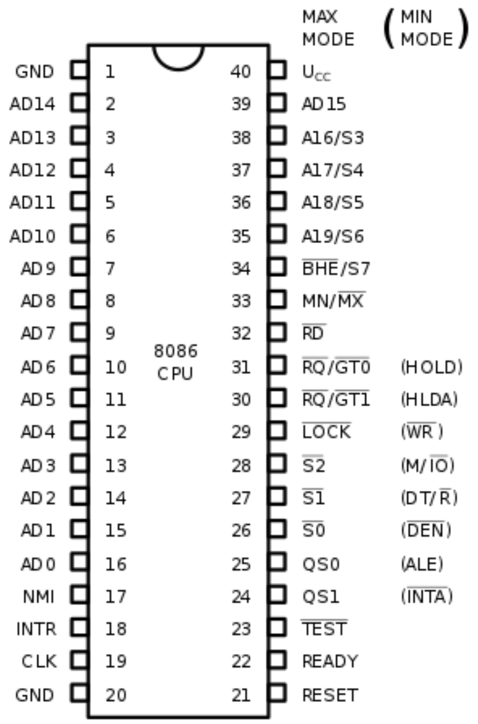
- Internal CPU events
    - Processor-detected program-error exceptions (Faults, Traps, Aborts)
    - Software-generated exceptions (INT 0, INT 1, INT3)
    - Machine-check exceptions



The 8086 pin assignments in min and max mode

DMA - Direct Memory Access

- Copy data over address space without involving CPU (its too slow for that anyway)
- DMA controller - device that copies the data over memory address bus
- Used mainly by devices (network cards, sound cards, storage devices, video cards, …)
- DMA16 - 16bit legacy address space used by ISA cards
- DMA32 - 32bit address space for PCI devices

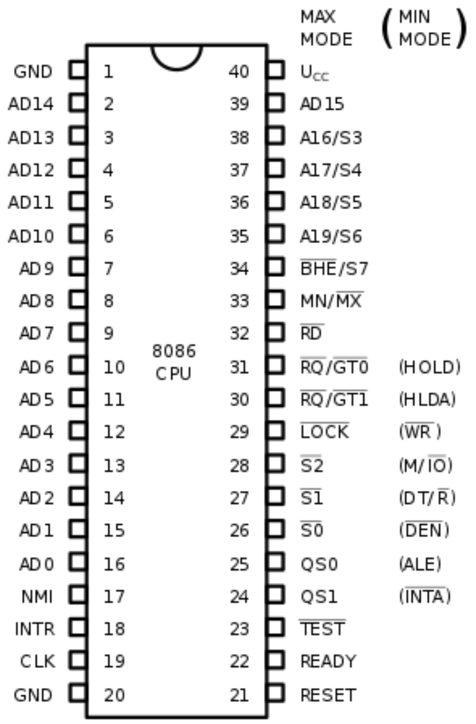


The 8086 pin assignments in min and max mode

x86 CPU in a closer look

Architecture and features

- CISC - Complex Instruction Set Computer
- Little Endian
- Instruction Pipelining
- Superscalar
- Speculative Execution
  - Branch Prediction
  - Out of Order Execution
- Privilege modes
- CPU modes
- Memory modes
- Paging
- Interrupts and Exceptions
- Registers



The 8086 pin assignments in min and max mode

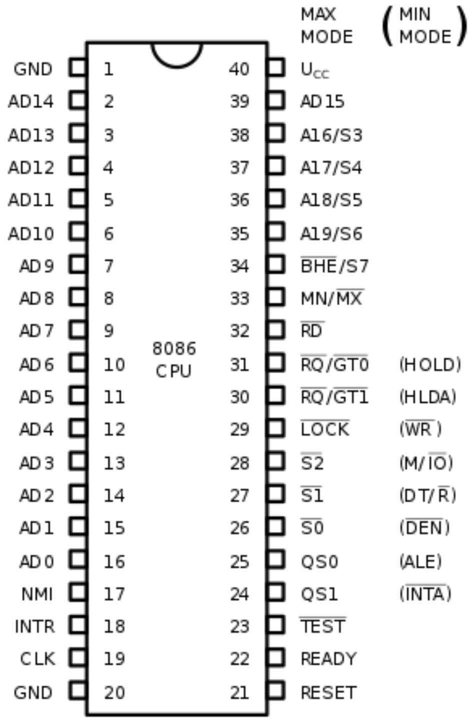
x86 CPU - picture source Wikipedia

CISC – Complex Instruction Set Computer

- Single instructions can execute several low-level operations (such as a load from memory, an arithmetic operation, and a memory store)

Intel:
```
add  rax,QWORD PTR [rbx+rbp*8+0xa]
```

AT&T:
```
add  0xa(%rbx,%rbp,8),%rax
```



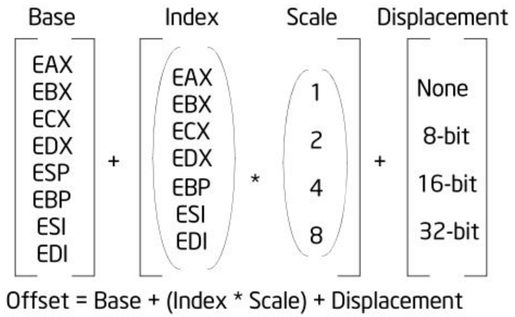
The 8086 pin assignments in min and max mode

CISC – Complex Instruction Set Computer

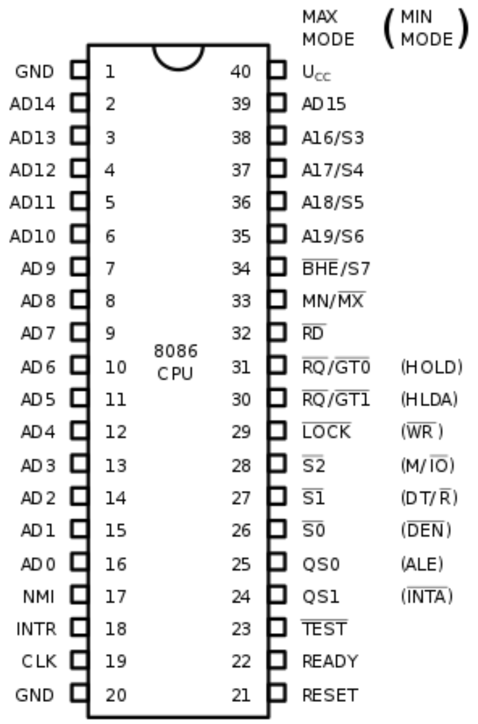
- Single instructions can execute several low-level operations (such as a load from memory, an arithmetic operation, and a memory store)

Intel: `add  rax,QWORD PTR [rbx+rbp*8+0xa]`

AT&T: `add  0xa(%rbx,%rbp,8),%rax`



1. Multiply register and constant
2. Add result with a constant
3. Add result with a register
4. Fetch QWORD from memory
5. Add register and a constant
6. Store result in a register



The 8086 pin assignments in min and max mode

Picture source – Wikipedia, Intel documentation

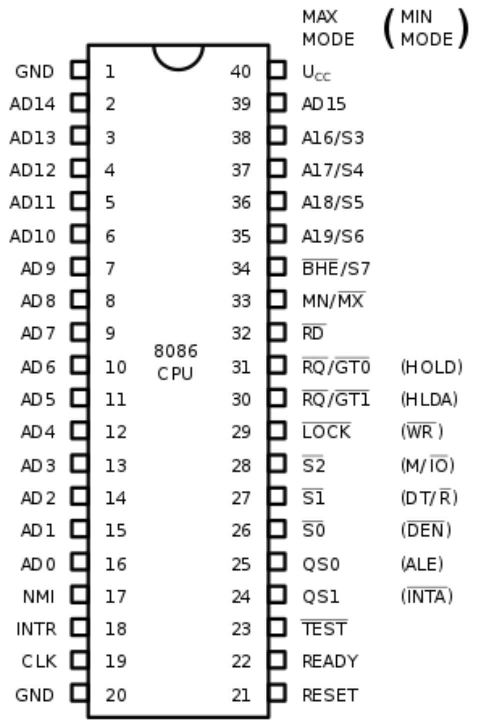
CISC – Complex Instruction Set Computer

- Variable length of instructions
- Instructions take several CPU cycles to execute, depending on the instructions and its operands

Intel:
```
48 c7 45 f8 78 56 34 12    mov    QWORD PTR [rbp-0x8], 0x12345678
c3                         ret
```

AT&T:
```
48 c7 45 f8 78 56 34 12    movq    $0x12345678,-0x8(%rbp)
c3                         ret
```

- Great for assembler developers (and virus writers)
- Nightmare for CPU designers (x86 is backward compatible to 1978)
- Modern x86 CPUs are internally RISC (Reduced Instruction Set Computers), CISC instructions are translated to RISC instructions internally



The 8086 pin assignments in min and max mode

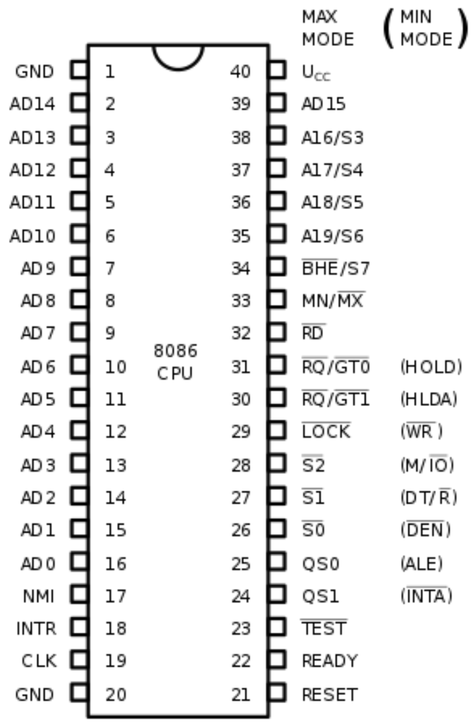
Little Endian

- Order in which bytes are stored in computer memory

```
unsigned long value = 0x12345678;
```

Offset	**Little Endian**	**Big Endian**
0x1000	0x78	0x12
0x1001	0x56	0x34
0x1002	0x34	0x56
0x1003	0x12	0x78

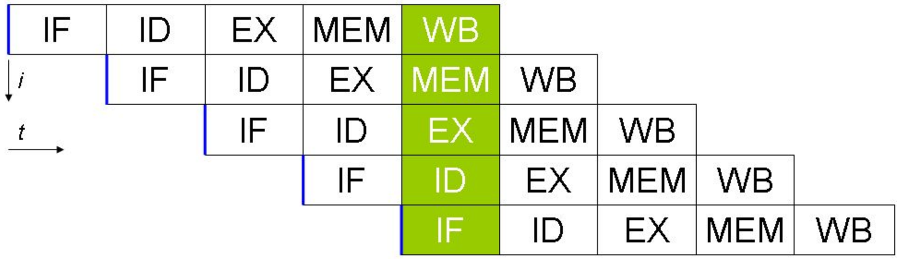
- BE - From historical architectures like IBM, still used by networking protocols
- LE - Simplifies silicon design and arithmetic operations on integers



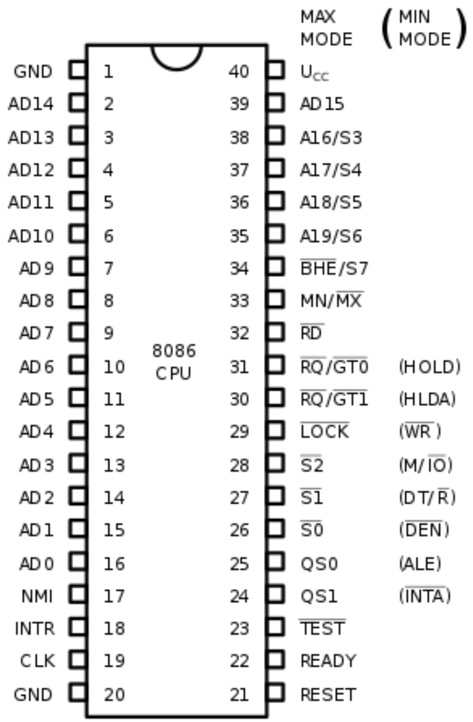
The 8086 pin assignments in min and max mode

Instruction Pipelining

- Idea originated in RISC CPU designs
- Split execution of instructions into stages
- A new instruction can be processed while previous instruction is being processed in a later stage of the pipeline
- Simplifies CPU design
- Makes sure each part of the CPU is busy



Basic five-stage pipeline in a RISC machine (IF = Instruction Fetch, ID = Instruction Decode, EX = Execute, MEM = Memory access, WB = Register write back)



The 8086 pin assignments in min and max mode

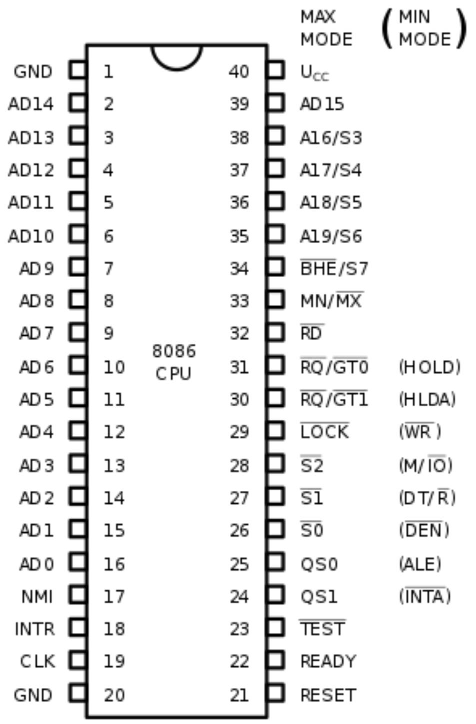
Speculative Execution - Branch Prediction

- On conditional branching, the CPU starts to execute both branches without knowing the result (using separate pipelines). Upon knowing the result, the pipeline with wrong chosen path is flushed
- Reduces risk of choosing the wrong code branch of execution and refilling the whole pipeline

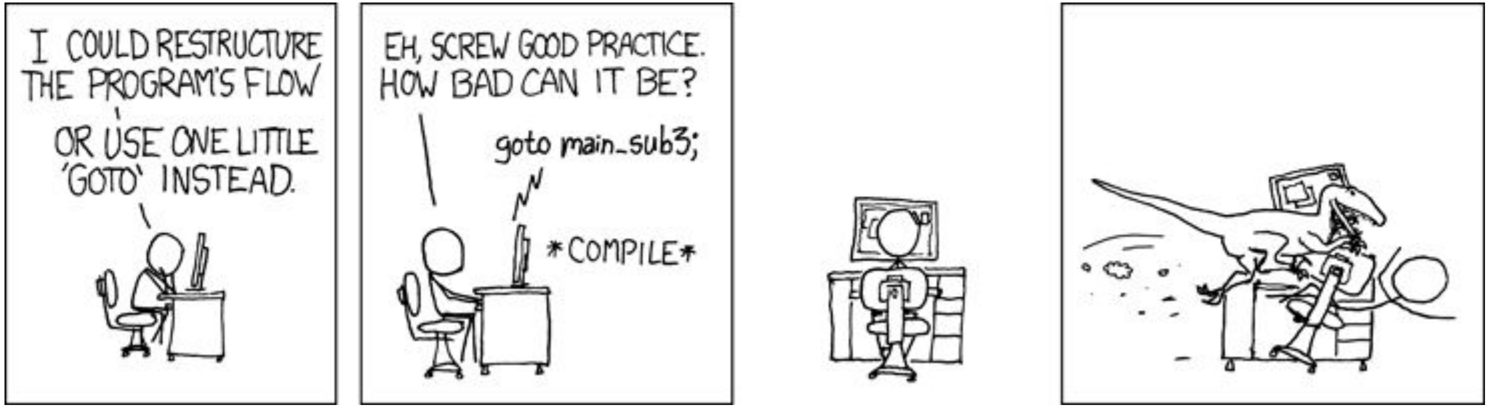
```
int do_stuff(int a, int b)
{
     if (a > b)
               goto _exit;

     printf("Hello world!\n");
     return 1;

_exit:
     return 0;
}
```



The 8086 pin assignments in min and max mode


I COULD RESTRUCTURE THE PROGRAM'S FLOW
OR USE ONE LITTLE 'GOTO' INSTEAD.
EH, SCREW GOOD PRACTICE. HOW BAD CAN IT BE?
goto main_sub3;
COMPILE

xkcd.com

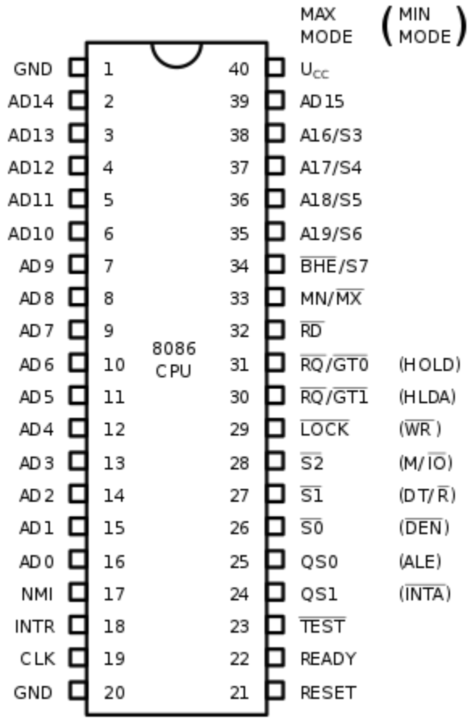
- Yes, there was a goto in the last example.
- The kernel code is full of them.
- A goto generates a relative jump and branch predictor makes sure that at least one pipeline is not completely flushed.
- Absolute jumps flush instruction pipeline completely.

Speculative Execution - Out of order execution

- The CPU executes instructions in a order depending on the availability of pipelines, disregarding order of instructions in the program
- Reduces idleness of pipelines

```
Intel:  mov     rax,QWORD PTR [rbx]
        add     rcx,rdx

AT&T:   mov     (%rbx),%rax
        add     %rdx,%rcx
```



The 8086 pin assignments in min and max mode

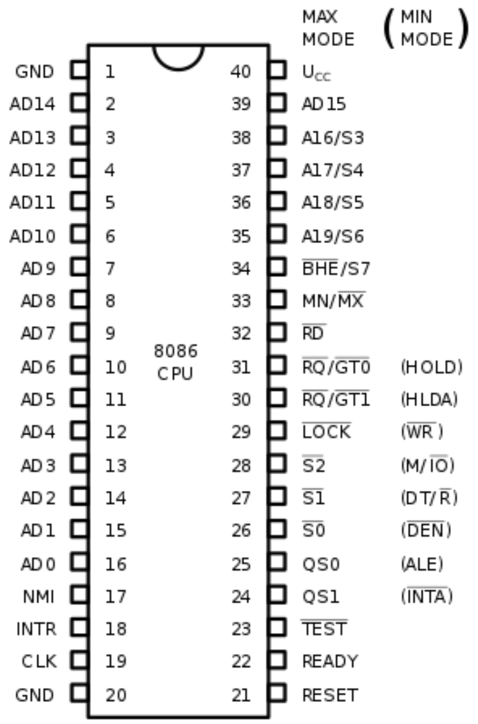
Speculative Execution - Out of order execution

- The CPU executes instructions in a order depending on the availability of pipelines, disregarding order of instructions in the program
- Reduces idleness of pipelines

```
Intel:   mov       rax,QWORD PTR [rbx]
         add       rcx,rdx

AT&T:    mov       (%rbx),%rax
         add       %rdx,%rcx
```

The addition will be executed first. It is not dependent on result of previous operation.
Fetch from memory is guaranteed to take longer than addition on registers.



The 8086 pin assignments in min and max mode

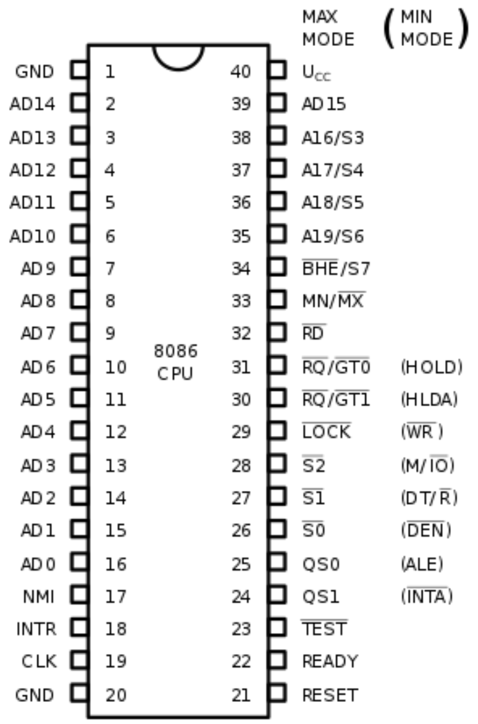
Red Hat

Register

- Device specific storage

X86 CPU

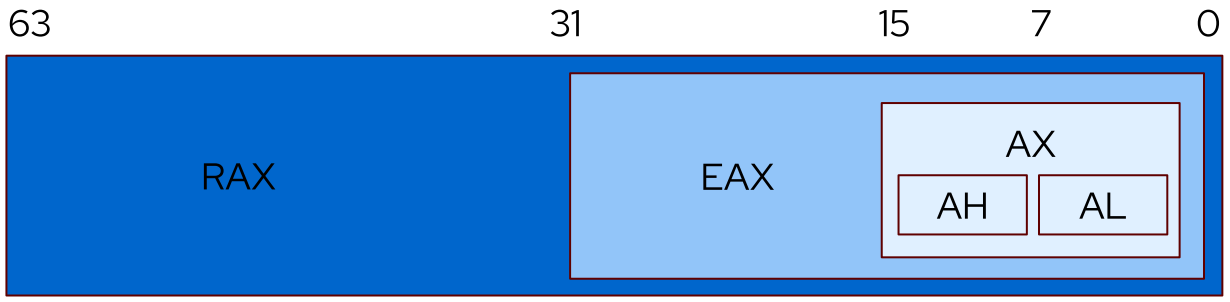
- General Purpose Registers
- Segment Registers
- Flag Register
- Instruction Pointer Register
- Control Registers
- Memory Registers
- FPU Registers
- Registers For Multimedia Extensions
- Debug Registers
- ...



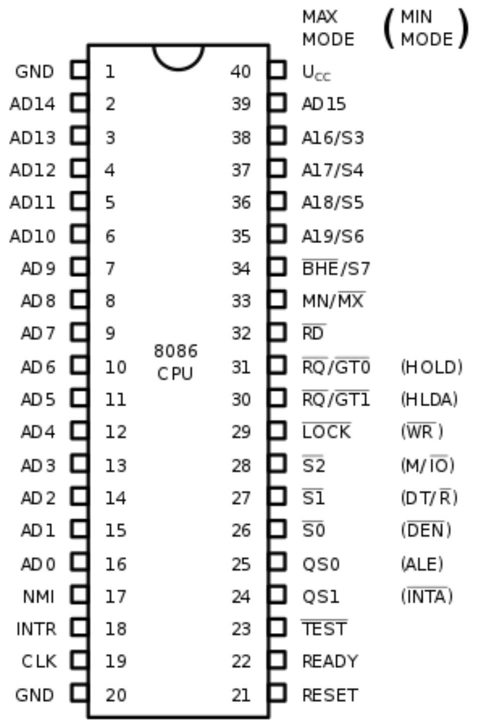
The 8086 pin assignments in min and max mode

General Purpose Registers

- Operands for logical and arithmetic operations
- Operands for address calculations
- Memory pointers
- There are 64 bit, 32 bit, 16 bit and 8 bit general purpose registers but …



- The same goes for RBX, RCX, RDX, RSI, RDI, RSP, RBP, R8 - R15, except:
    - R8 - R15 don't have a 32 bit and 16 bit version
    - RSI, RDI, RSP, RBP, R8-R15 have only low 8 bit register (no high)



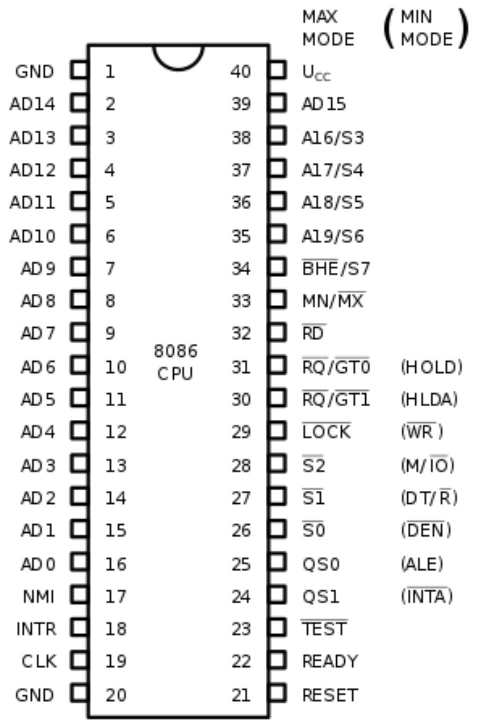
The 8086 pin assignments in min and max mode

General Purpose Registers

- AX - Accumulator
- BX - Base or pointer register
- CX - Counter for loop operations
- DX - Data pointer
- SI - Stack index
- DI - Data index
- SP - Stack pointer
- BP - Base pointer

- E prefix is a 32 bit extension (AX -> EAX)
- R prefix is a 64 bit extension (AX -> RAX)

- Instructions are bound to specific registers (in, out, mul, div, ...)



The 8086 pin assignments in min and max mode

Segment Registers

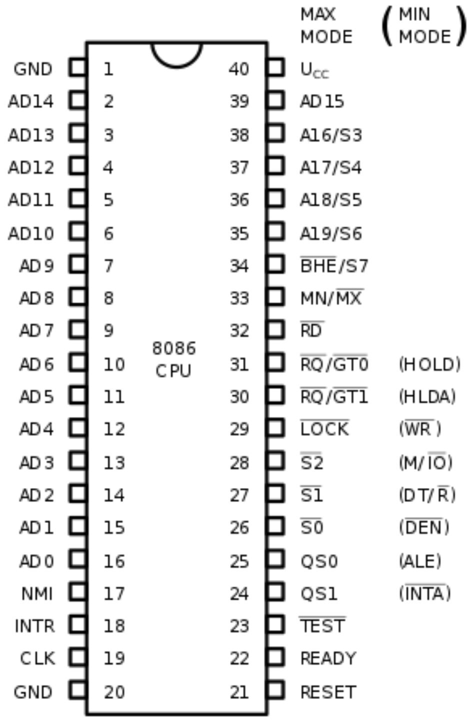
- 16 bit registers (upper parts are directly not accessible)
- Used to address different locations in address space as direct memory pointers or used with segmentation

```
Intel:  mov      rax, es:[rbx]
        mov      gs:[rbp],rax

AT&T:   es mov   (%rdx),%rax
        mov      %rax,%gs:0x0(%rbp)
```

- Not Really used in 64bit mode, but needed to properly set up protected mode

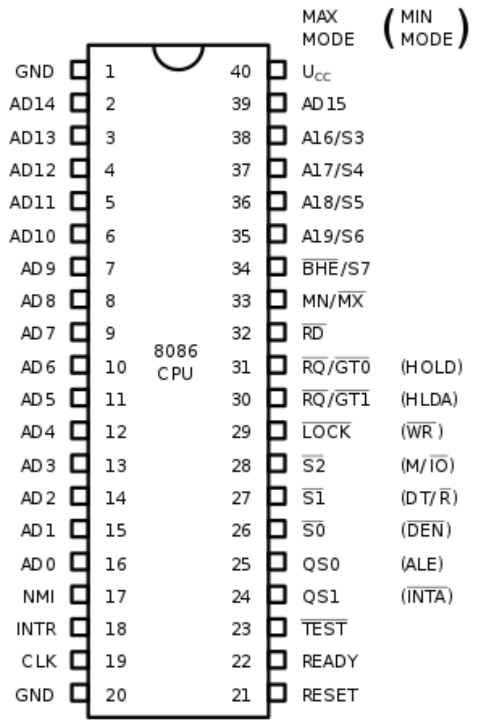
- CS – Code Segment – Code is always fetched from here
- DS – Data Segment
- SS – Stack Segment
- ES – Extra segment
- FS, GS – Additional Extra segments



The 8086 pin assignments in min and max mode

EFlags - Flag register

- Union of 1 bit registers in one 64 bit register
- 3 main groups
  - Status
    - Results of arithmetic operations
      - Evaluated by conditional jumps
      - Parity, Overflow, Zero, Carry, Sign bits
  - Control
    - Direction of auto incrementation of string instructions
      - MOVS, CMPS, SCAS, LODS, and STOS
  - System
    - IRQ handling flags
      - Traps, Interrupts, virtual 8086 interrupt, ..



The 8086 pin assignments in min and max mode

IP – Instruction Pointer

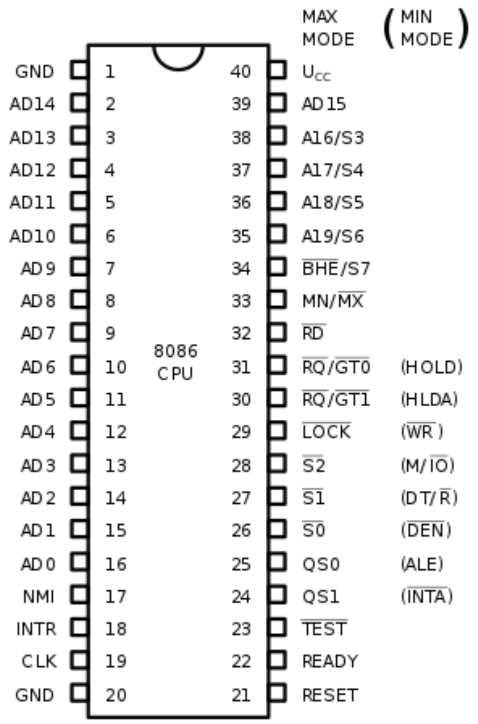
- Points to next instruction to be executed
- Cannot be accessed directly
  - Indirectly through jump, call, syscall and ret instructions

CRX – Control Registers

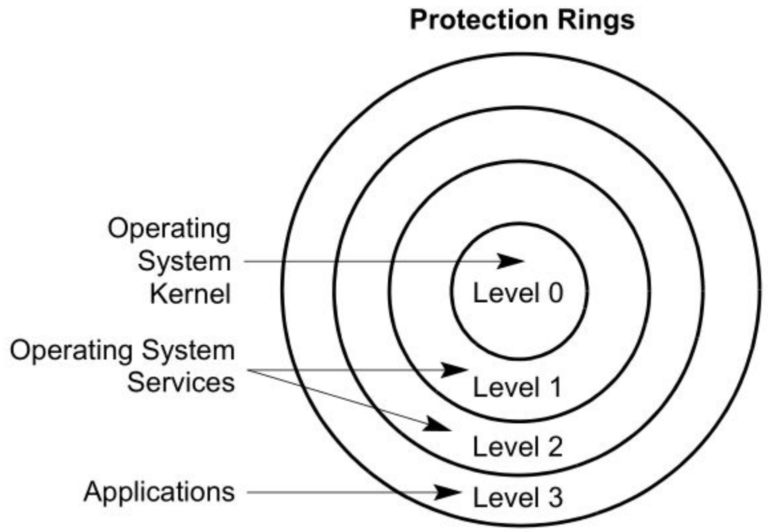
- CR0, CR1, CR2, CR3, CR8
- Enable/Disable protected mode
- CPU Cache control
- Page Fault addresses when used with paging

XDTR – Memory Registers

- GDTR – Global Descriptor Table
  - Define segments and their properties
- LDTR – Local Descriptor Table
  - Define custom segments with less functionality (e.g. no TSS)
- IDTR – Interrupt Descriptor Table
  - Table with callback functions for interrupts, exceptions and traps



The 8086 pin assignments in min and max mode



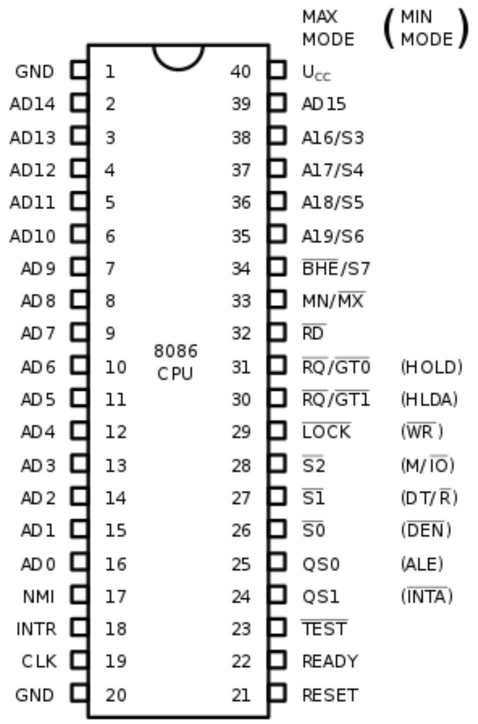
There are 4 privilege modes on x86 also known as rings or levels:

- Ring 0
  - Supervisor mode
  - Can read/write anything in the system
  - Used for OS kernels
- Ring 3
  - Restricted mode, used for user space applications
  - Memory address access restriction
  - Access to certain registers is restricted (e.g Memory control or CPU control registers)
  - Restricted access to specific instructions (e.g. RDMSR, WRMSR)
- Ring 1 and 2 are not used in Linux

Figure source - Intel documentation

x86 CPU Modes

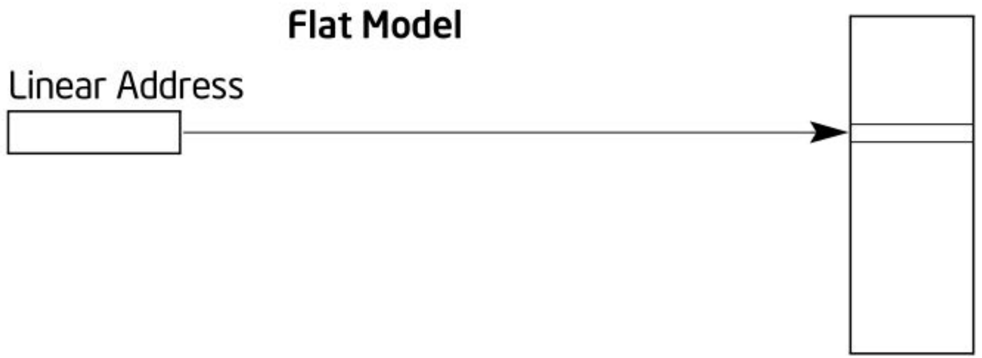
- Real Mode
  - 8086 mode with possibility to switch to protected mode
  - Processor is placed in this mode after power up or a reset
- System Management Mode (SMM)
  - Transparent to OS
  - Hardware emulation
  - Power management
  - Silicon fixes
  - USB support for DOS
- Protected Mode
  - This is the native operating mode of the processor
  - Modern OSes run in this mode
  - Memory paging can be enabled
- Virtual 8086 Mode
  - Emulation of 8086 mode inside protected mode for backward compatibility
- IA-32e mode
  - 32 bit compatibility mode in 64 bit CPU environment

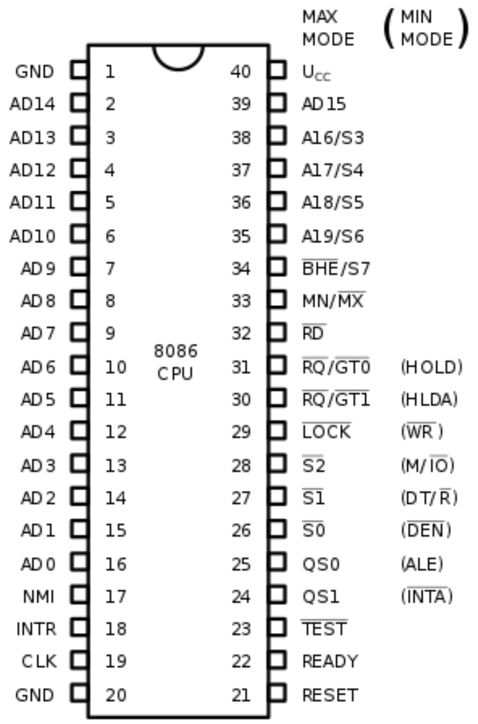


The 8086 pin assignments in min and max mode

x86 Memory Modes

- Flat memory, linear address space
  - Linear address on CPU has one-to-one mapping with the physical address
  - Code, stack and data share the same address space



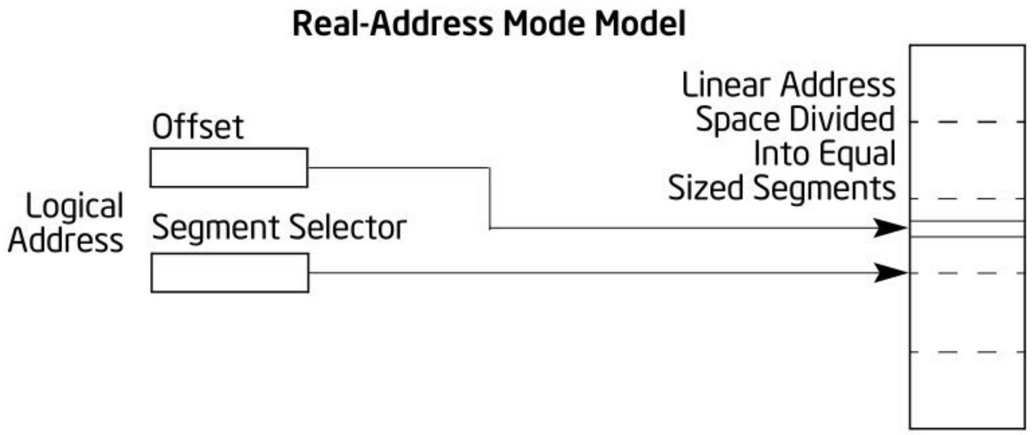


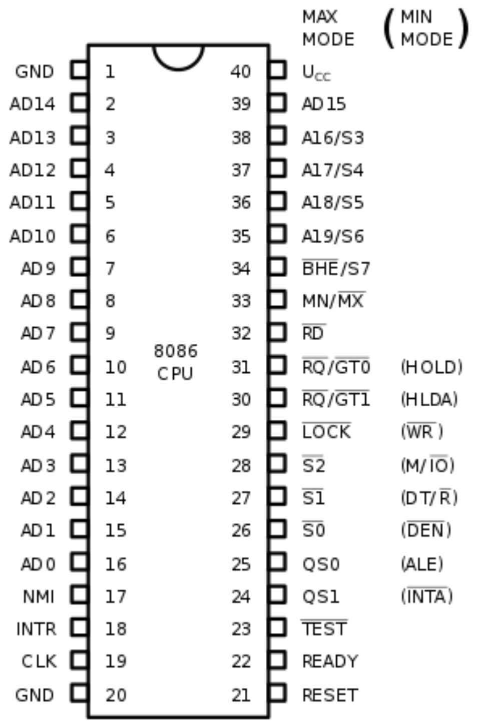
The 8086 pin assignments in min and max mode

Figure source - Intel documentation

x86 Memory Modes

- Real Address Mode
  - 8086 Address space 20 bit wide, registers are 16 bit wide
    - A combination of two 16 bit registers is used for addressing
    - A segment register and a general purpose register
    - The segment register contains a direct memory address



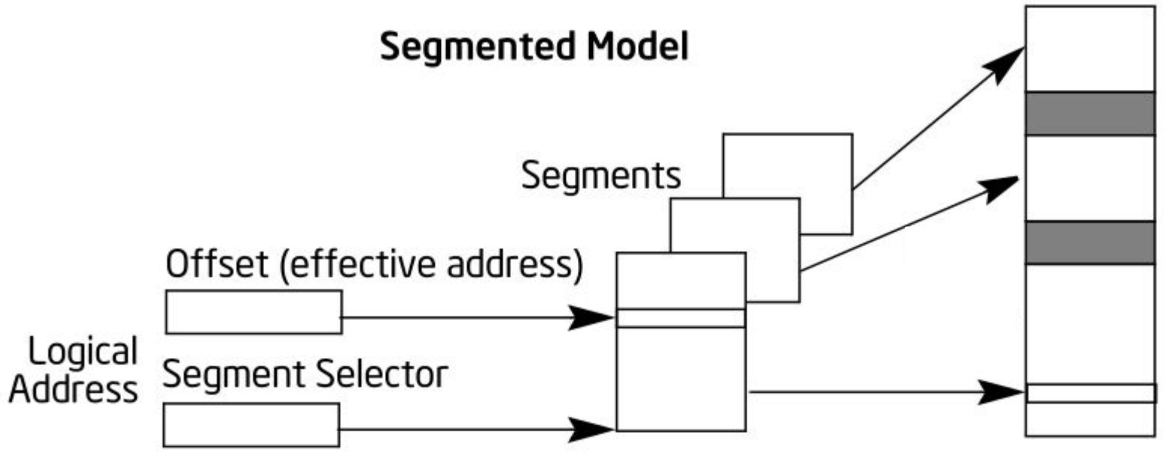


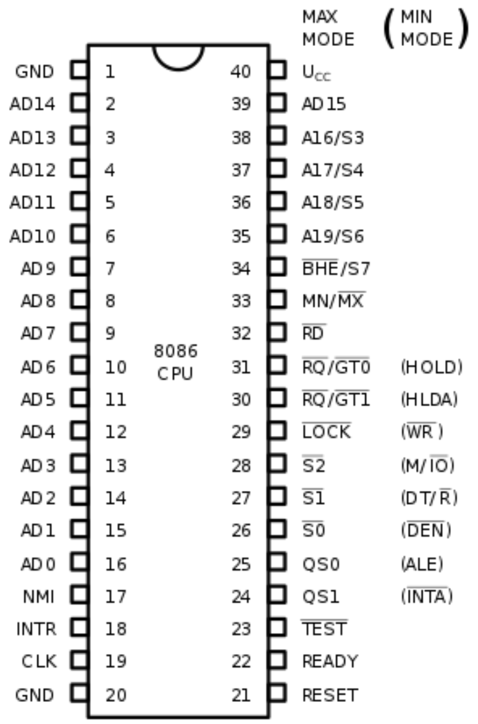
The 8086 pin assignments in min and max mode

Figure source – Intel documentation

x86 Memory Modes

- Segmented memory model
  - Memory is split into regions of variable size -> segments
  - Each segment is described using a CPU structure - Segment Descriptor
  - Segments are stored in a table





The 8086 pin assignments in min and max mode

Figure source - Intel documentation

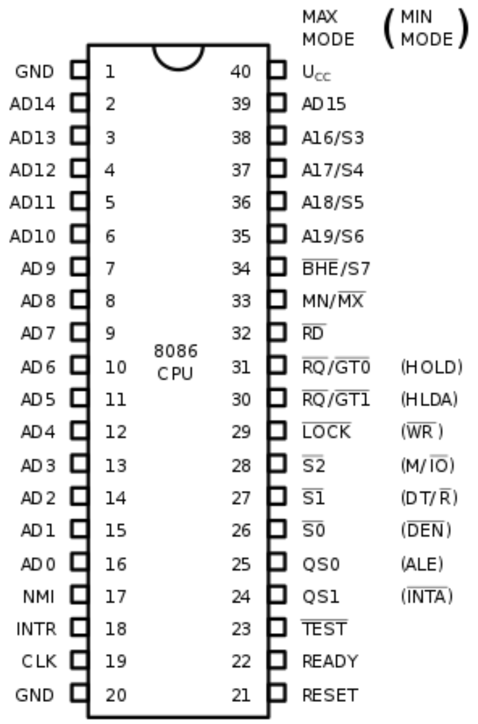
Segmentation

- Hardware enforced mechanism of isolating individual code, data, and stack modules
- Multiple programs (or tasks) can run on the same processor without interfering with one another.
- Each program can be assigned its own segments.
- Each segment is described by a descriptor
  - Base address - Where does the segment start in address space
  - Limit - Size of segment
  - Access rights
  - Privilege level
  - Segment type
  - Offset in linear address space (start of its first byte)
- Linux uses segmentation in a very limited way, required by hardware

Warning, intel documentation and Linux source code use the terms “linear address”, “physical address”, “virtual address” very loosely. They are interchanged and should be understood by context of topic

Paging

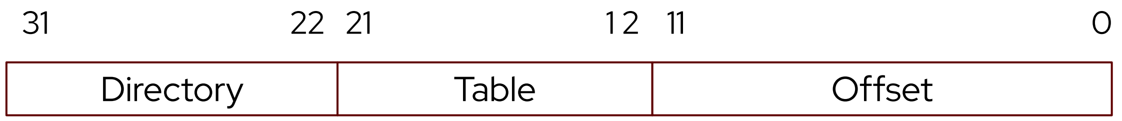
- Linear address space is divided into blocks of same size -> **pages or frames**
- Size of page is architecture dependant
  - Default 4KB
  - X86 supports also 2MB or 1GB
- OS and CPU keep a track of pages with page specific metadata
  - Physical address
  - Access rights
  - Present
  - Dirty
  - Global
  - ...
- Pages are kept in a hierarchical structure **Page Directory**



The 8086 pin assignments in min and max mode

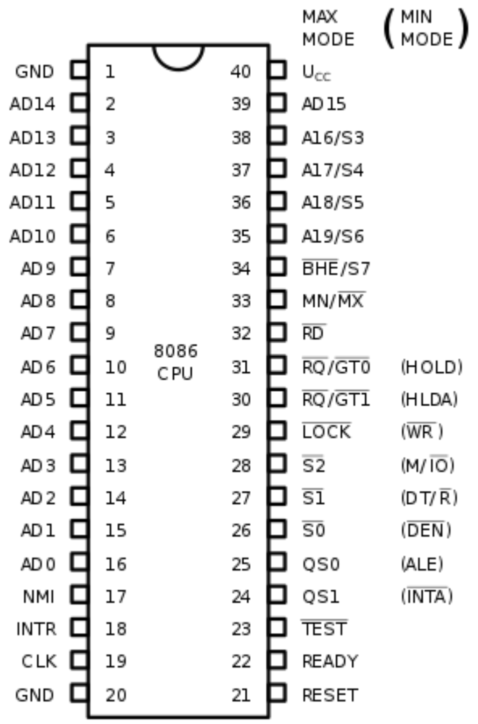
Page Hierarchy

- Frames are stored in a hierarchical structure **Page Directory**
- Linear (virtual!) address is cut into chunks (count and length is CPU mode and architecture specific)



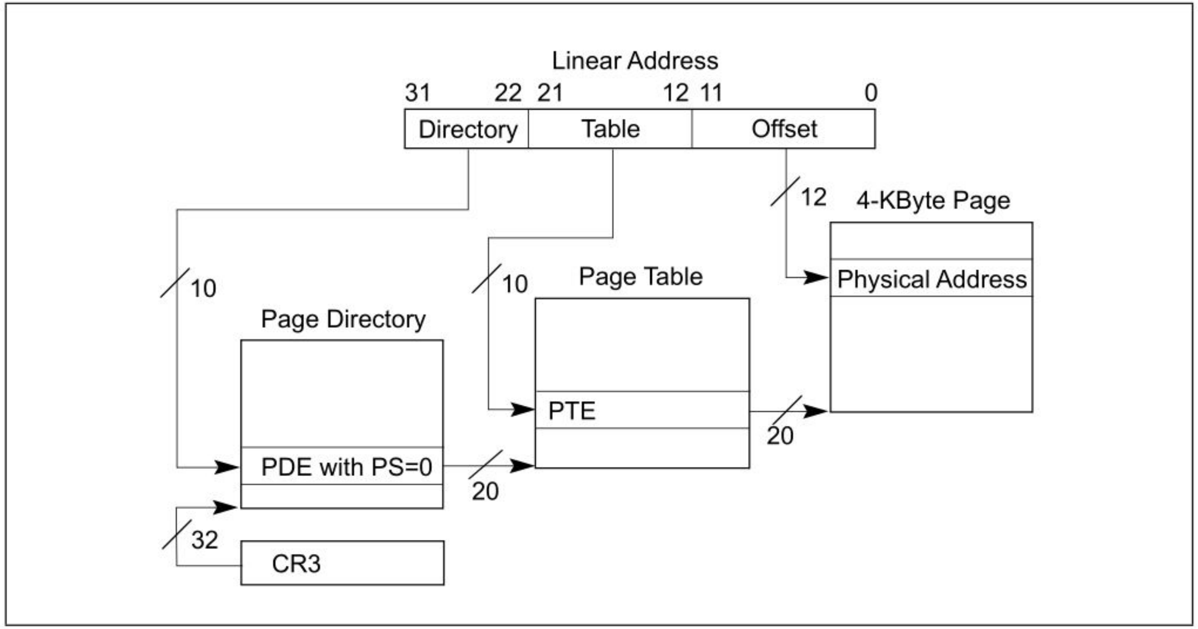
32 bit linear address in a 4-KByte page using 3 level paging.

- Chunks act as indexes into tables -> **Page Tables** and **Page Directories**
- Table entries contain parts of physical memory address or indexes to another paging structure
- Last chunk of the address (low part) contains offset in to the page -> **Page Offset**
- Depending on the size of the CPU address space, more page tables are needed to describe pages -> **Page Levels**
- On x86 page tables are limited to 4096 bytes -> count of entries changes depending on on CPU paging mode, architecture, …



The 8086 pin assignments in min and max mode

Figure source - Intel documentation



Linear address translation to a 4-KByte page using 32 bit paging

```
Physical address =
((Page Directory Entry & 0xFFFFF) << 20 | Page table Entry & 0xFF000) + (Offset & 0xFFF)
```

Figure source - Intel documentation

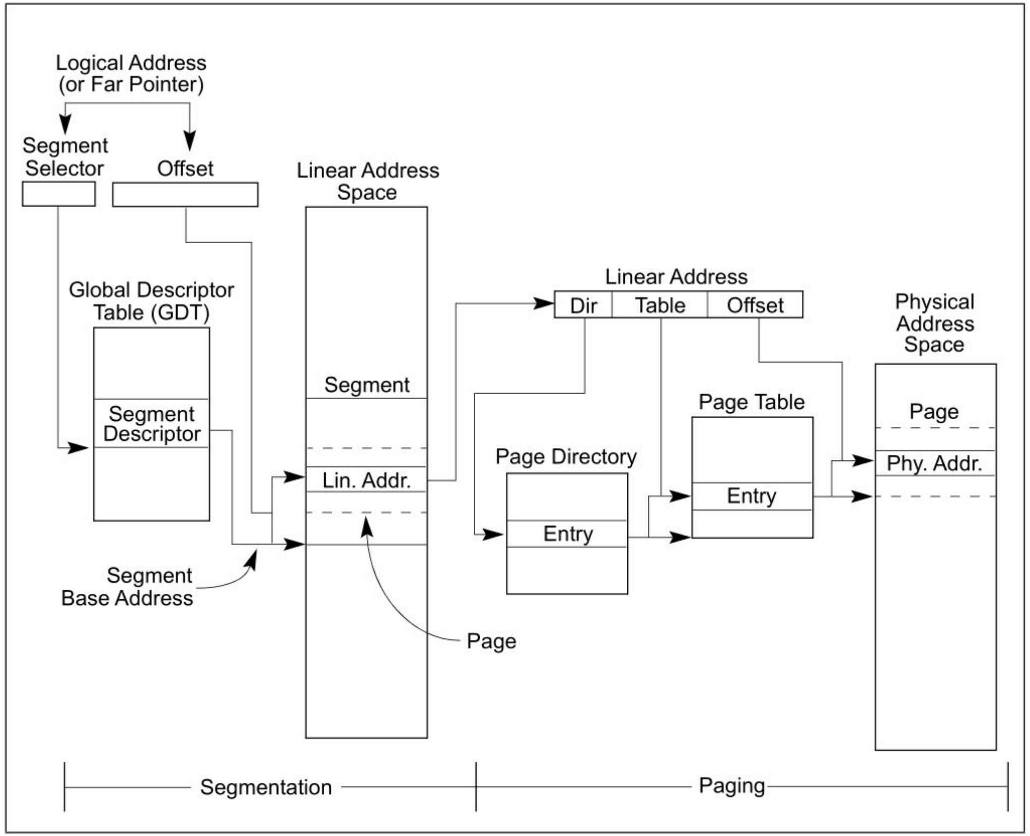
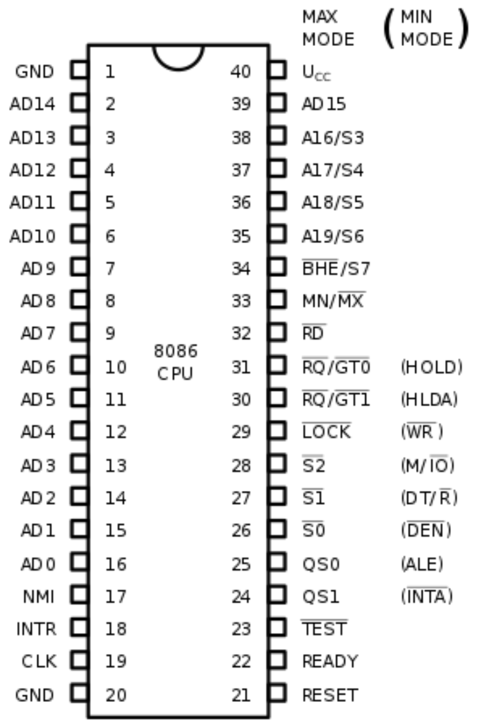


Figure source – Intel documentation

Paging and segmentation are the main workhorses of the memory management, protection and isolation in a modern operating system, including Linux.

Intel Documentation	Linux source code
PML5 Table	Page Global Directory, pgd
PLM4 Table	Page Level 4 Directory, pd4
Page-directory-pointer table	Page Upper Directory, pud
Page Directory	Page Middle Directory, pmd
Page Table	Page Table Entry, pte



The 8086 pin assignments in min and max mode

Thank you!

Questions?

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat