On reset, processor is initialized in physical address mode (real mode in x86).

Linux requires virtual addressing to enable process protection and isolation - protected mode.

Processor must be switched to protected mode and turn on virtual address support before the kernel kicks in.

For this,

-MMU chipset must be initialized by setting up appropriate core data structures.

-These are architecture specific and are implemented in arch/ branch of the kernel source tree.

-These are compiled and linked as a header to protected mode kernel image. This header is called kernel bootstrap or real mode kernel.

For x86,

arch/x86/boot/main.c

/\* arch/x86/boot/main.c \*/

void main(void)

{

/\* First, copy the boot header into the "zeropage" \*/

copy\_boot\_params();

/\* Initialize the early-boot console \*/

console\_init();

if (cmdline\_find\_option\_bool("debug"))

puts("early console in setup coden");

/\* End of heap check \*/

init\_heap();

/\* Make sure we have all the proper CPU support \*/

if (validate\_cpu()) {

puts("Unable to boot - please use a kernel appropriate "

"for your CPU.n");

die();

}

/\* Tell the BIOS what CPU mode we intend to run in. \*/

set\_bios\_mode();

/\* Detect memory layout \*/

detect\_memory();

/\* Set keyboard repeat rate (why?) and query the lock flags \*/

keyboard\_init();

/\* Query Intel SpeedStep (IST) information \*/

query\_ist();

/\* Query APM information \*/

#if defined(CONFIG\_APM) || defined(CONFIG\_APM\_MODULE)

query\_apm\_bios();

#endif

/\* Query EDD information \*/

#if defined(CONFIG\_EDD) || defined(CONFIG\_EDD\_MODULE)

query\_edd();

#endif

/\* Set the video mode \*/

set\_video();

/\* Do the last things and invoke protected mode \*/

go\_to\_protected\_mode();

}

Once paging is enabled, system begins to see/treat RAM as an array of blocks of fixed size, called page frames.

Most MMUs support 4k, 8k, 16k, 64k upto 4MB options for size of a page frame.

Linux kernel’s default page frame size is 4k.

### Page Descriptor

Page frames would be required for

- mapping physical memory to virtual addresses

- kernel code and its data structures

- processing dynamic allocation requests raised by process/kernel.

Architecture independent data structure - struct page.

Holds all metadata related to a page frame.

For each physical page frame, there will be an instance of struct page.

Kernel has to maintain a list of page instances in memory all the time.

**Important members**