Programming Models for Various Computer Systems

Computer Architecture

Department of Computer Science and Engineering

Programming Models

- A sequence of instructions
  - Branching?

- Single program
  - Branching
  - Sub-routines / Function calls ?

- Interruptable programs?
  - Separate Interrupt service routine(s).
  - Processor’s ability to perform a context switch

Programming Models…

- Multi-threaded programs
  - Memory Safety?
  - Multiple programs originated from different sources
- Multi-core programs
  - Homogenous multi-core processors
  - How to write programs for multi core processors?
- Heterogenous systems
  - GPU-CPU co-processing
- System On Chip (SoC) architectures
- Virtualization architectures
  - Memory Safety?
  - Multiple virtual machines running on same shared hardware

Memory Safety

- Memory safety is based on Virtual Memory Page System
  - What happens if a program tries to access a memory location of another program?
    - If its for communication purpose, and coordinated well, it could be useful
    - However, it could be intentionally or unintentionally harmful as well
      - Corrupt the programs
      - Tamper with data
  - Who will manage the memory safety? And How?
    - Program is a collection of instructions
      - We have Load Store instructions
    - Somebody should avoid using unauthorized memory regions

Memory segmentation and paging (Virtual Memory)

- Caching mechanism between Main memory and hard disk
- **Doubles as a security mechanism**

Memory segmentation and paging (Virtual Memory)

- Caching mechanism between Main memory and hard disk
- **Doubles as a security mechanism**

Table 8.4 Analogous cache and virtual memory terms

Cache	Virtual Memory
Block	Page
Block size	Page size
Block offset	Page offset
Miss	Page fault
Tag	Virtual page number

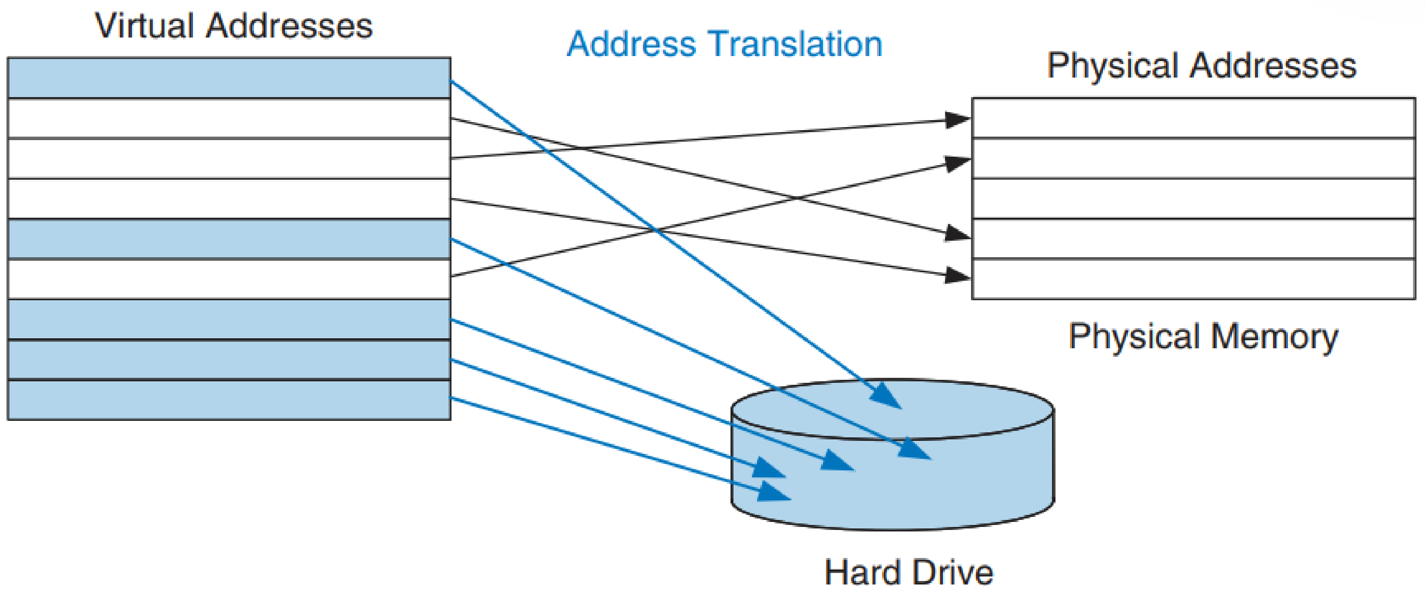
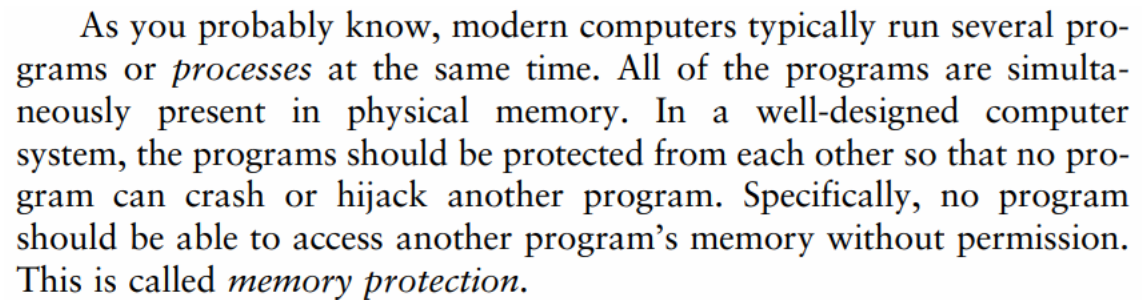
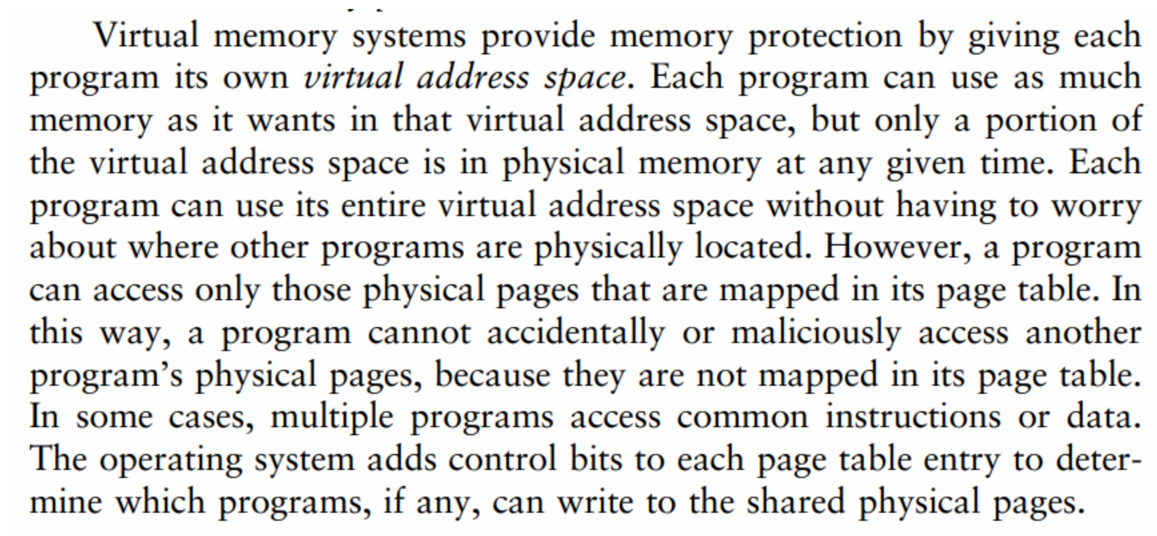


Figure 8.20 Virtual and physical pages


As you probably know, modern computers typically run several programs or *processes* at the same time. All of the programs are simultaneously present in physical memory. In a well-designed computer system, the programs should be protected from each other so that no program can crash or hijack another program. Specifically, no program should be able to access another program's memory without permission. This is called *memory protection*.


Virtual memory systems provide memory protection by giving each program its own *virtual address space*. Each program can use as much memory as it wants in that virtual address space, but only a portion of the virtual address space is in physical memory at any given time. Each program can use its entire virtual address space without having to worry about where other programs are physically located. However, a program can access only those physical pages that are mapped in its page table. In this way, a program cannot accidentally or maliciously access another program's physical pages, because they are not mapped in its page table. In some cases, multiple programs access common instructions or data. The operating system adds control bits to each page table entry to determine which programs, if any, can write to the shared physical pages.

- https://wiki.riscv.org/display/HOME/Recently+Ratified+Extensions

- https://riscv.org/wp-content/uploads/2017/05/riscv-privileged-v1.10.pdf

Kernals

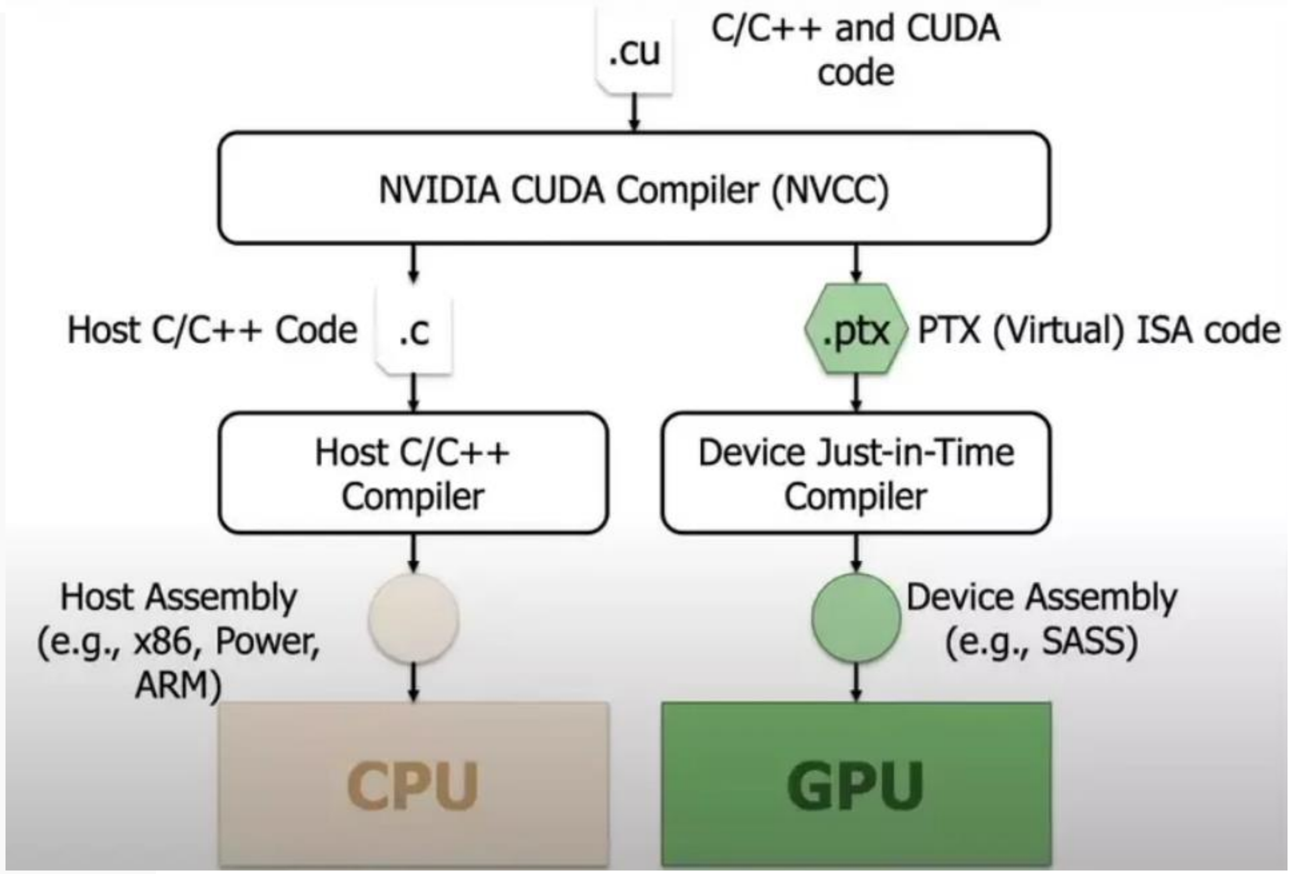
```
#include <stdio.h>


__global__ void kernel()
{

    printf("hello world");
}

int main()
{
    kernel<<<1,1>>>();
    cudaDeviceSynchronize();

    return 0;
}
```


.cu
C/C++ and CUDA code
NVIDIA CUDA Compiler (NVCC)
Host C/C++ Code
.c
.ptx
PTX (Virtual) ISA code
Host C/C++ Compiler
Device Just-in-Time Compiler
Host Assembly (e.g., x86, Power, ARM)
Device Assembly (e.g., SASS)
CPU
GPU

- What are the programming models for System On Chip Designs?