

COMP427 Embedded Systems

Lecture 1. What is Embedded System?

Prof. Taeweon Suh
Computer Science & Engineering
Korea University

Embedded Systems

- Embedded System is a **special-purpose computer system** designed to perform one or a few dedicated functions -- Wikipedia
 - In general, it does not provide programmability to users, as opposed to general purpose computer systems like PC
 - Embedded systems are virtually everywhere in your daily life



Embedded Systems (Cont)

- Even though embedded systems cover a wide range of special-purpose systems, there are common characteristics
 - **Low cost**
 - Should be cheap to be competitive
 - Memory is typically very small compared to a general purpose computer system
 - Lightweight processors are used in embedded systems
 - **Low power**
 - Should consume low power especially in case of portable devices
 - Low-power processors are used in embedded systems



Embedded Systems (Cont)

▪ High performance

- Should meet the computing requirements of applications
 - Users want to watch video on portable devices
 - Audio should be in sync with video
 - Gaming gadgets like playstation should provide high performance



▪ Real-time property

- Job should be done within a time limit
 - Aerospace applications, Car control systems, Medical gadgets are critical in terms of time constraint – Otherwise, it could lead to catastrophe such as loss of life
- Will talk more about this



Embedded Systems (Cont)

- It is challenging to satisfy the characteristics
 - You may not be able to achieve high performance while utilizing cheap components and maintaining low power consumption
 - So, you got to do your best in a given circumstance to be competitive in the market

HW/SW Stack of Embedded Systems

- Identical to the general computer systems



Application Software

VxWorks



ANDROID

OS / Device Drivers

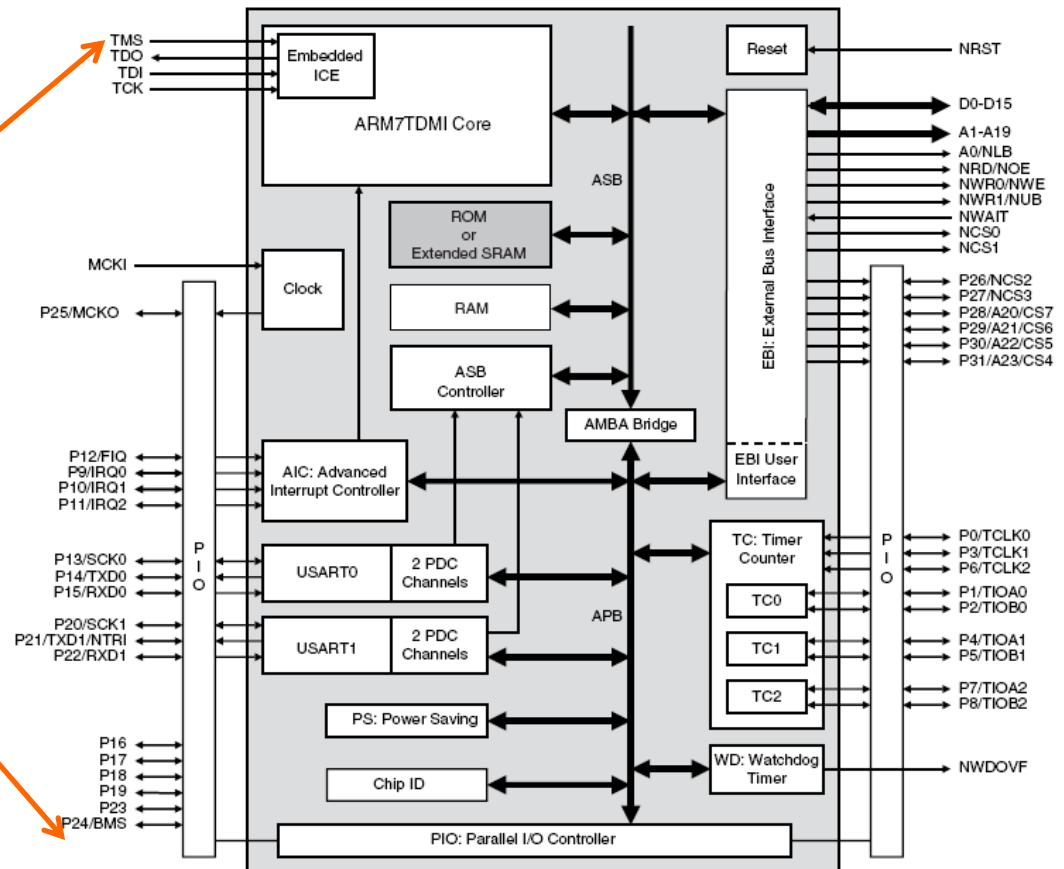
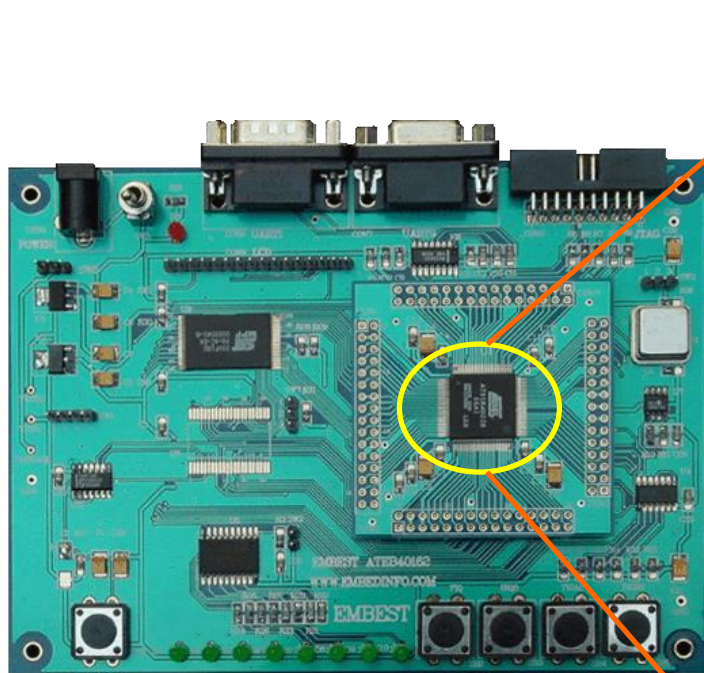


Hardware

Components of Embedded Systems

- **Hardware**

- It is mainly composed of processor (1 or more), memory, I/O devices including network devices, timers, sensors etc.



Components of Embedded Systems

- **Software - System software**

- Operating systems

- Many times, a multitasking (multithreaded) OS is required, as embedded applications become complicated

- Networking, GUI, Audio, Video

- CPU is context-switched to process multiple jobs

- Operating system footprint should be small enough to fit into memory of an embedded system

- In the past and even now, real-time operating systems (RTOS) such as VxWorks and uC/OS-II have been used because they are light-weighted in terms of memory requirement

- Nowadays, heavy-weighted OSs such as iOS, Android, Windows Mobile, and embedded Linux (uClinux) are used, as embedded processors support computing power and advanced capabilities such as MMU (Memory Management Unit)

- Device drivers for I/O devices



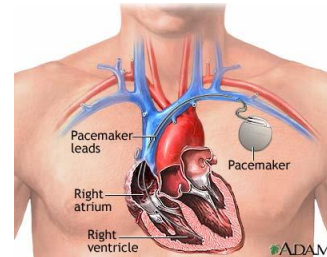
Components of Embedded Systems (Cont)

- **Software (cont.) - Application software**
 - Run on top of operating system
 - Execute tasks that users wish to perform
 - Web surfing, Social Network Service, Audio, Video playback

Real-Time Systems

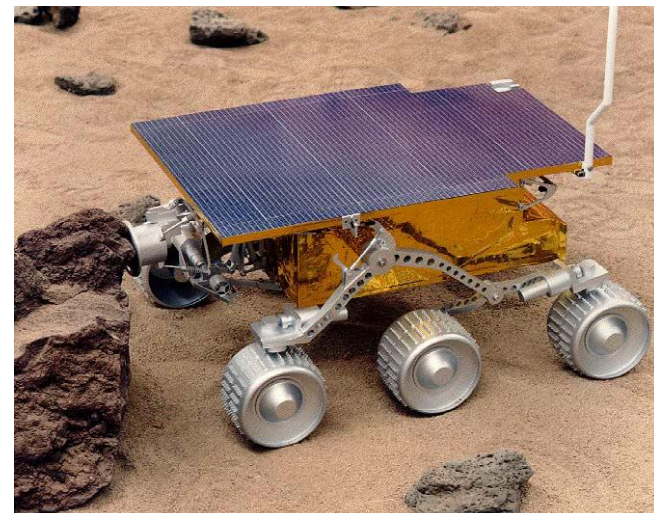
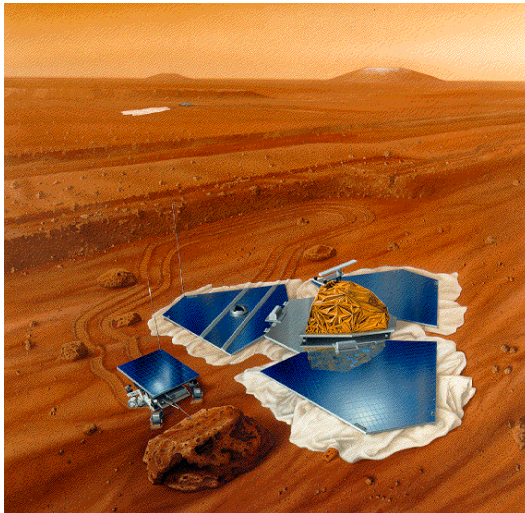
- Real-time operating system (RTOS):
Multitasking operating system for real-time applications
 - RTOS is valued for how quickly and/or predictably respond to a particular event
 - **Hard real-time systems** are required to complete a critical task within a guaranteed amount of time
 - **Soft real-time systems** are less restrictive
 - Implementing real-time system requires a careful design of scheduler
 - System must have the priority-based scheduling
 - Real-time processes must have the highest priority
 - **Priority inheritance (next slide)**
 - **Solve the priority inversion problem**
 - Process dispatch latency must be small

Hard real-time systems



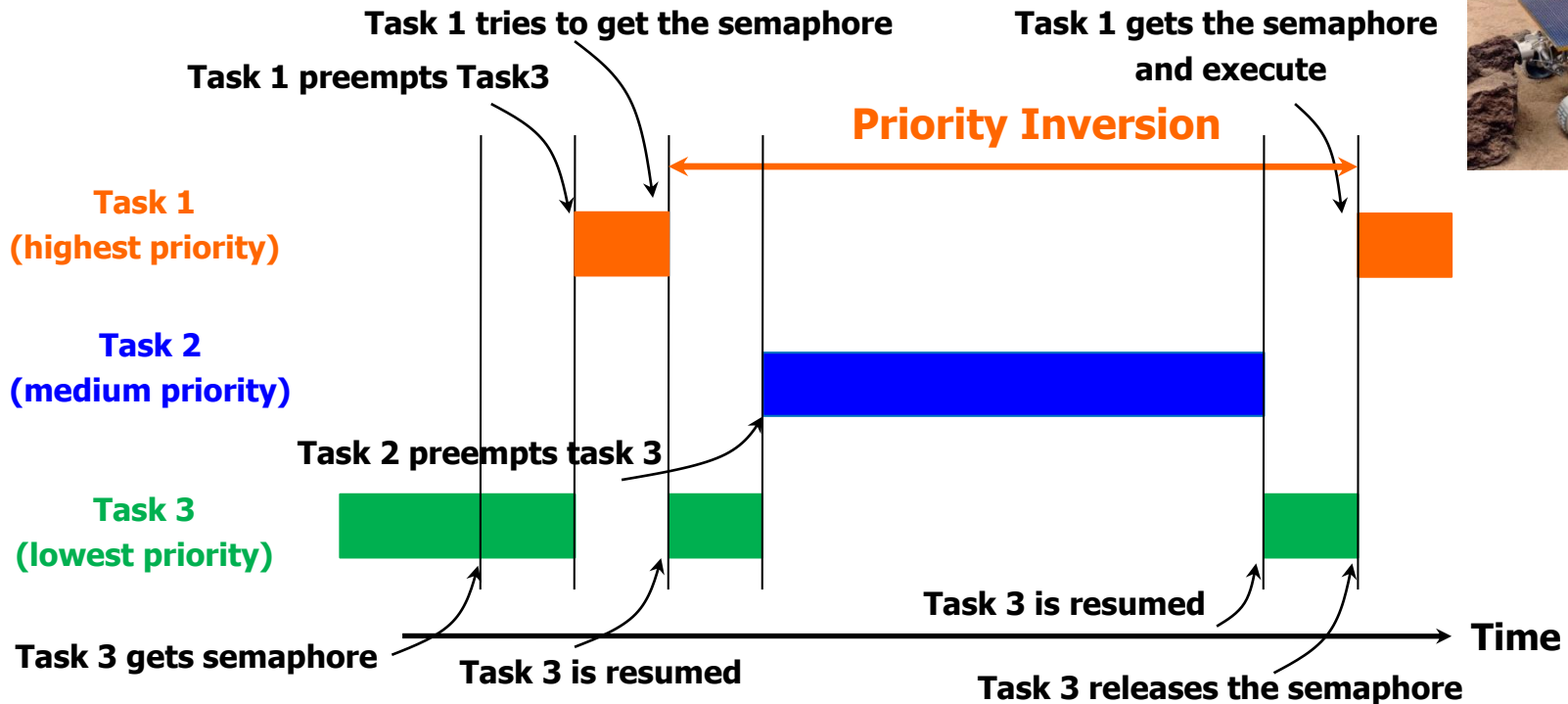
Priority Inversion Problem

- Pathfinder mission on Mars in 1997
 - Used VxWorks, an RTOS kernel, from WindRiver
 - Software problems caused the total system resets of the Pathfinder spacecraft in mission
 - Watchdog timer goes off, informing that something has gone dramatically wrong and initiating the system reset



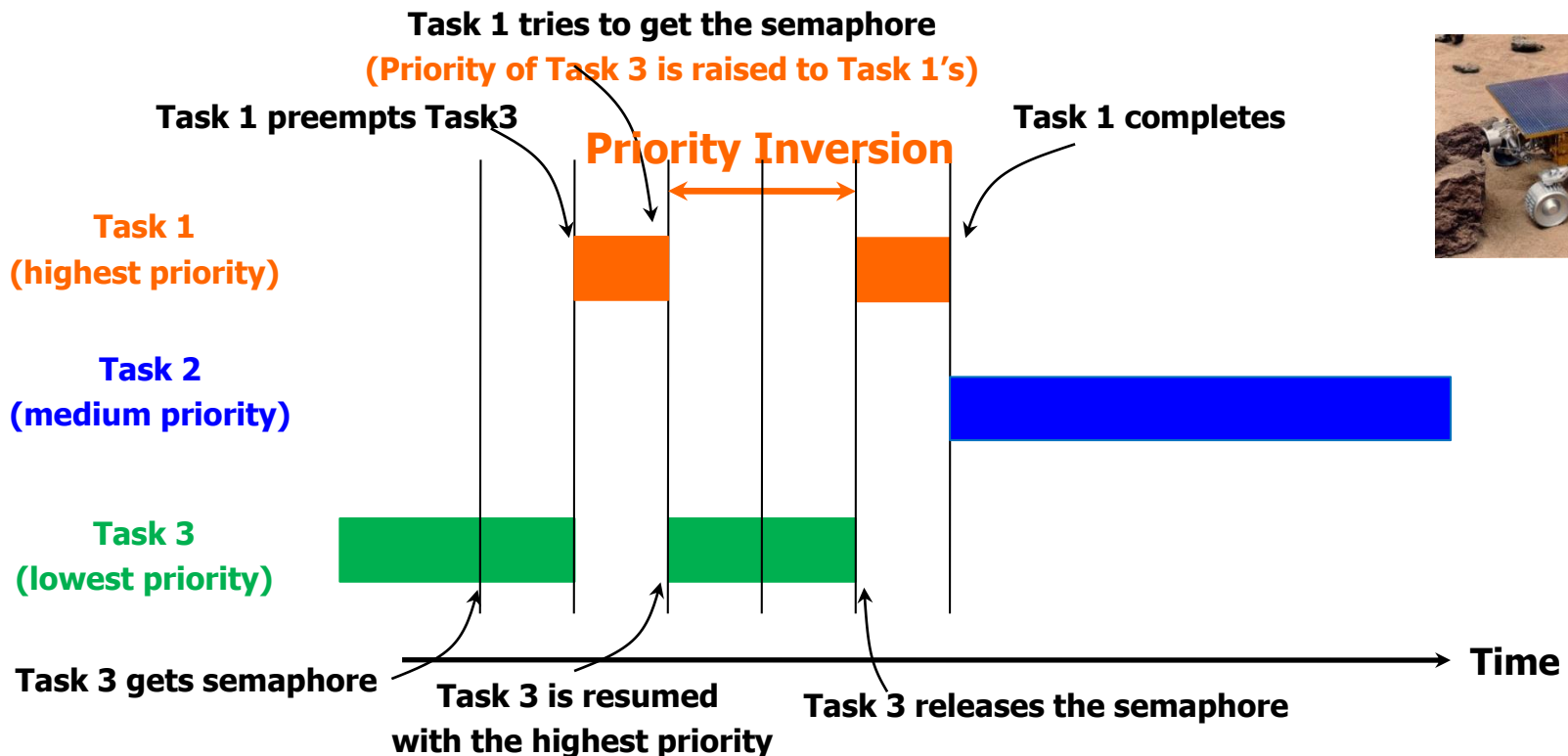
Priority Inversion Problem

- VxWorks provides **preemptive priority scheduling** of threads
 - Tasks on the Pathfinder spacecraft were executed as threads with priorities that were assigned in the usual manner reflecting the relative urgency of these tasks.



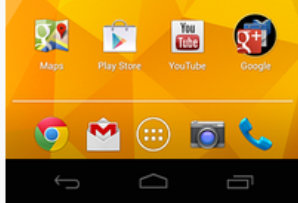
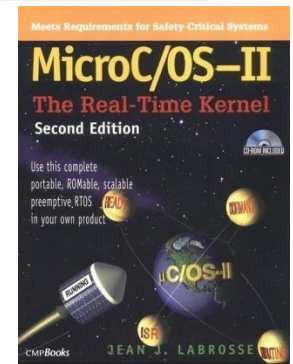
Priority Inheritance

- A lower priority process could be accessing a critical section (a shared resource) that the higher priority process needs
 - The process with a lower priority inherits the higher priority until they are done with the resource
 - When they are finished, its priority reverts to its original value



Operating Systems for Embedded Systems

- RTOSs
 - pSOS
 - VxWorks
 - VRTX (Versatile Real-Time Executive)
 - uC/OS-II
- Palm OS & Symbian OS(source: Wikipedia)
 - Palm OS: Embedded operating system initially developed by U.S. Robotics-owned Palm Computing, Inc. for personal digital assistants (PDAs) in 1996
 - Symbian OS: Proprietary operating system designed for mobile devices by Symbian Ltd. A descendant of Psion's EPOC and runs exclusively on ARM processors
- Android (<http://www.android.com/>)
 - Open Handset Alliance Project
 - Based on modified version of Linux 2.6 kernel
 - Currently supporting ARM, MIPS, and x86



Android 4.2 "Jelly Bean" on the Nexus 4

Company / developer	Google
	Open Handset Alliance
	Android Open Source Project
Programmed in	C, C++, Java ^[1]
OS family	Unix-like
Working state	Current
Source model	Open source ^[2]
Initial release	September 23, 2008 ^[3]
Latest stable release	4.2.2 Jelly Bean / February 11, 2013; 21 days ago ^{[4][5]}
Marketing target	Smartphones Tablet computers
Available language(s)	Multi-lingual
Package manager	Google Play, APK
Supported platforms	ARM, MIPS, ^[6] x86 ^[7]

Operating Systems for Embedded Systems



- uClinux (source: Wikipedia) - as of 2009
 - The use of a Linux operating system in embedded computer systems
 - According to survey conducted by Venture Development Corporation, Linux was used by 18% of embedded engineers
 - Embedded versions of Linux are designed for devices with relatively limited resources, such as cell phones and set-top boxes
 - Due to concerns such as cost and size, embedded devices usually have much less RAM and secondary storage than desktop computers, and are likely to use flash memory instead of a hard drive
 - Since embedded devices are used for specific purposes rather than general purposes, developers optimize their embedded Linux distributions to target specific hardware configurations and usage situations
 - These optimizations can include reducing the number of device drivers and software applications, and modifying the Linux kernel to be a real-time operating system
 - Instead of a full suite of desktop software applications, embedded Linux systems often use a small set of free software utilities such as busybox, and replace the glibc C standard library with a more compact alternative such as dietlibc, uClibc, or Newlib.

FAQ: Linux in Embedded Systems?

- Is Linux too large?
 - Linux is highly modular and has an excellent component selection mechanism
 - Based on your system configuration, you can selectively choose software components
 - How about memory requirement?
 - A minimal working embedded Linux system with networking and file system support needs around 4MB of SDRAM and 2MB of flash
- Is Linux real-time enough?
 - A lot of work going on in the embedded Linux to enable real-time
 - Enhancements are in the form of a preemptive kernel or real-time-capable scheduler



- A port of Linux to systems without a Memory Management Unit (MMU)
 - <http://www.uclinux.org/>
- BTW, what's MMU? Let's stop here to review MMU (Memory Management Unit)
 - MMU is an essential hardware component to support and implement virtual memory
 - MMU provides a fast translation from virtual address to physical address
 - Otherwise, the translation from virtual address to physical address will slow down the execution of your applications a lot

Motivation of Virtual Memory

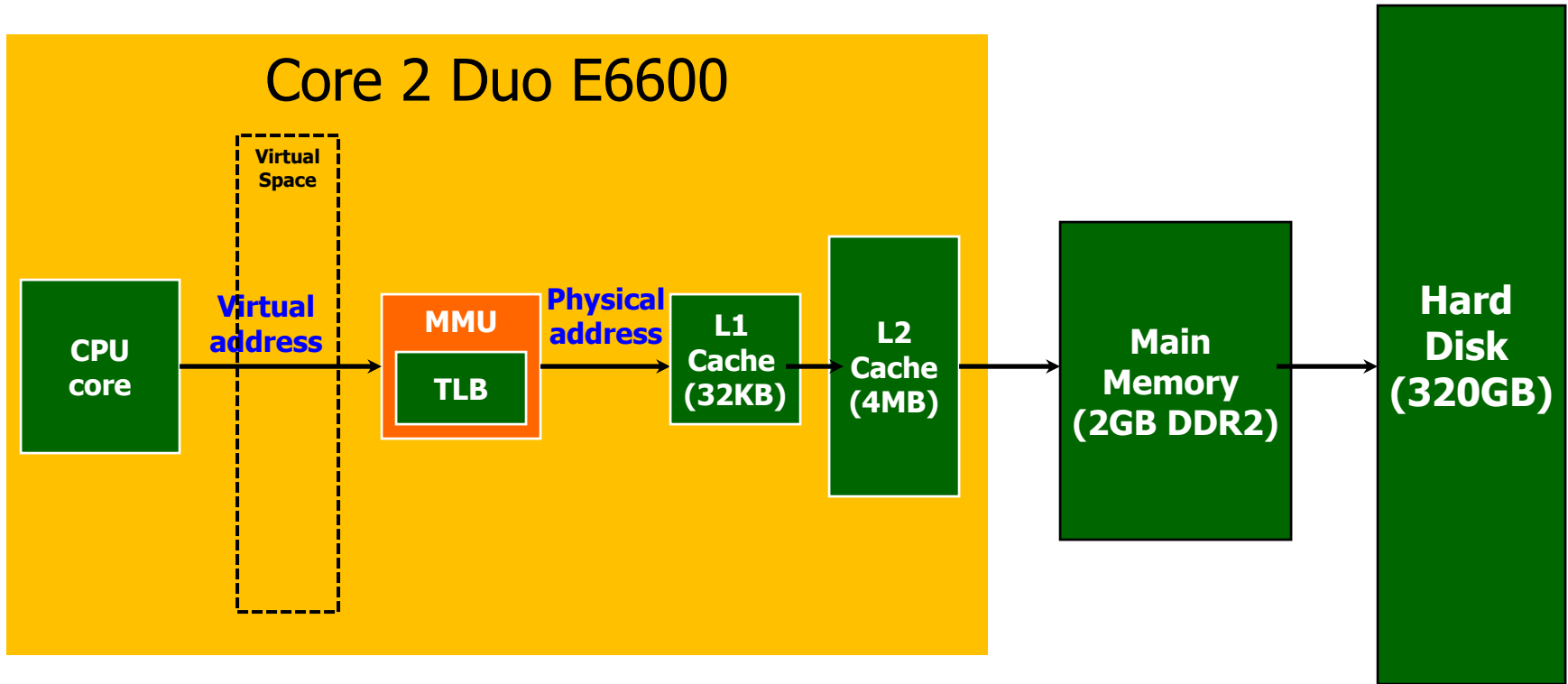
- Virtual memory (VM) was invented to relieve programmers from burdens
 1. VM allows efficient and safe sharing of main memory among multiple programs
 - Consider a collection of programs running all at once on a computer
 - We don't want to know which programs will share main memory with other programs when we compile them
 - In fact, the programs sharing main memory change dynamically while the programs are running
 - Because of this dynamic interaction, we would like to compile each program into **its own address space (virtual address space)**
 - VM (implemented in Operating System) dynamically manages the translation of the program's address space (virtual address space) to the physical address space
 2. VM provides the ability to easily run programs larger than the size of physical memory
 - In old days, if a program is too large for memory, it was the programmers' responsibility to make it fit
 - Programmers divided programs into pieces and then load and unload pieces into main memory under user's program control

Virtual Memory

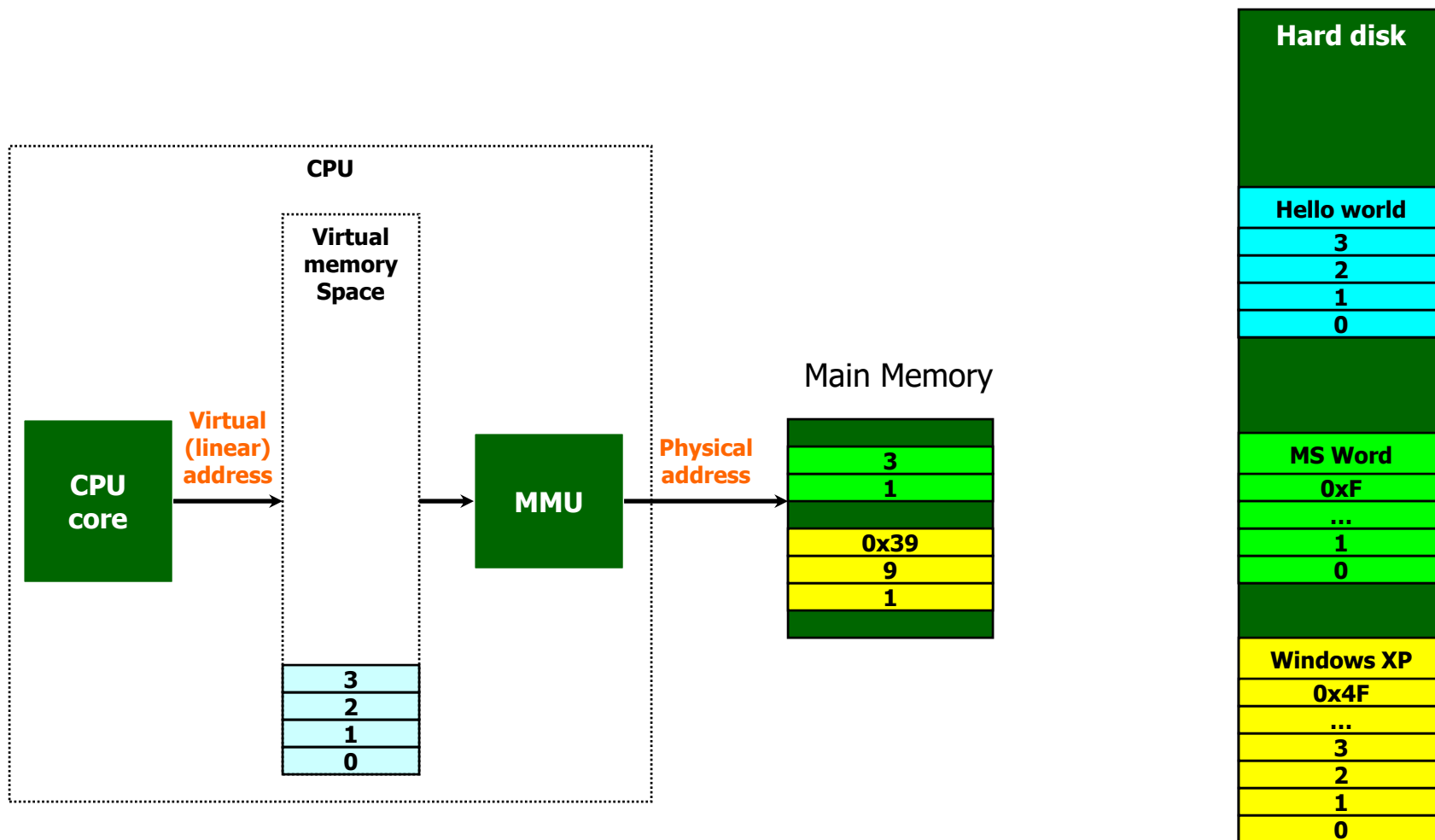
- Virtual memory is a technique provided by operating systems such as Windows and Linux
- Virtual memory uses main memory as a “cache” for secondary storage
 - Virtual memory automatically manages the 2 levels of memory hierarchy: main memory and secondary storage (HDD)
 - Virtual space is split into fixed-sized blocks, which are called **pages** (typically 4KB)
 - Load only required pages for execution to physical memory
 - Operating systems create **page tables**, which contain the translation information from virtual page to physical page

Memory Subsystem in x86

- MMU translates from virtual address to physical address
 - Operating system creates page tables
 - TLB (Translation Lookaside Buffer) inside MMU caches recently used page table entries



Visualizing Virtual Memory





- It is hard to support real-time property with virtual memory
 - Paging in and out takes a huge amount of time
 - BTW, embedded systems generally do not have hard disk though
 - Instead, flash memory is used many times



- uClinux has been ported to many microcontrollers
 - ARM7TDMI (ARM)
 - Dragonball, ColdFire, 68K Derivatives, QUICC (Motorola)
 - Blackfin (ADI)
 - i960 (Intel)
 - Microblaze (Xilinx)
 - V850E (NEC)
- I myself don't know much about uClinux
 - I'd like you guys to dig into the source code , hack it, and join the uClinux community!
- Anyway, we are going to use uClinux for Labs