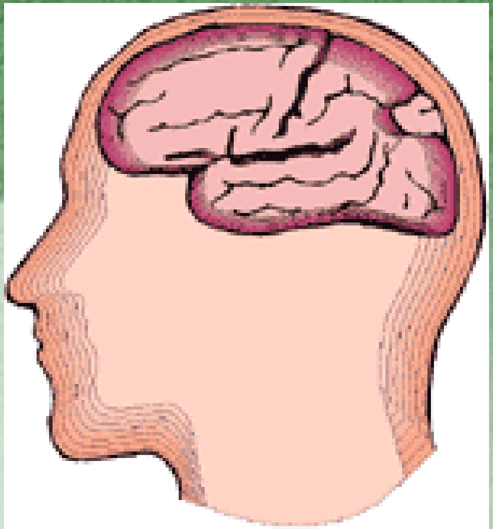
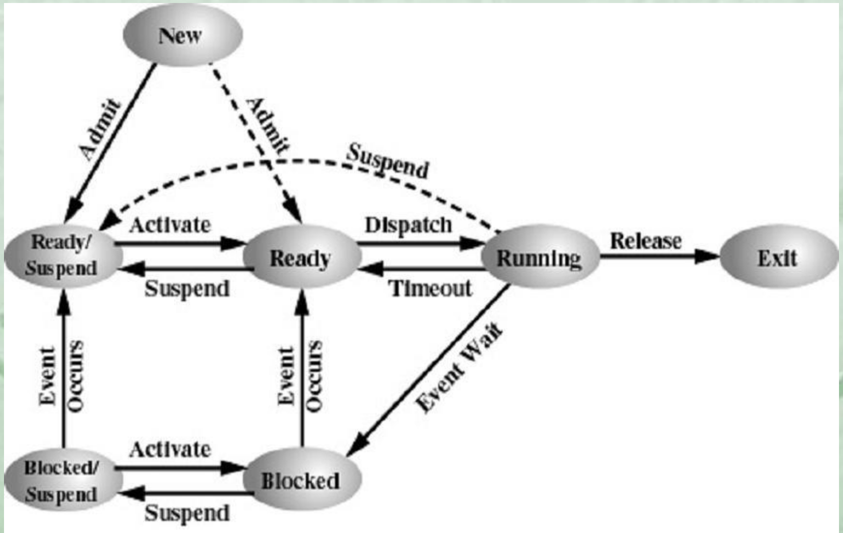
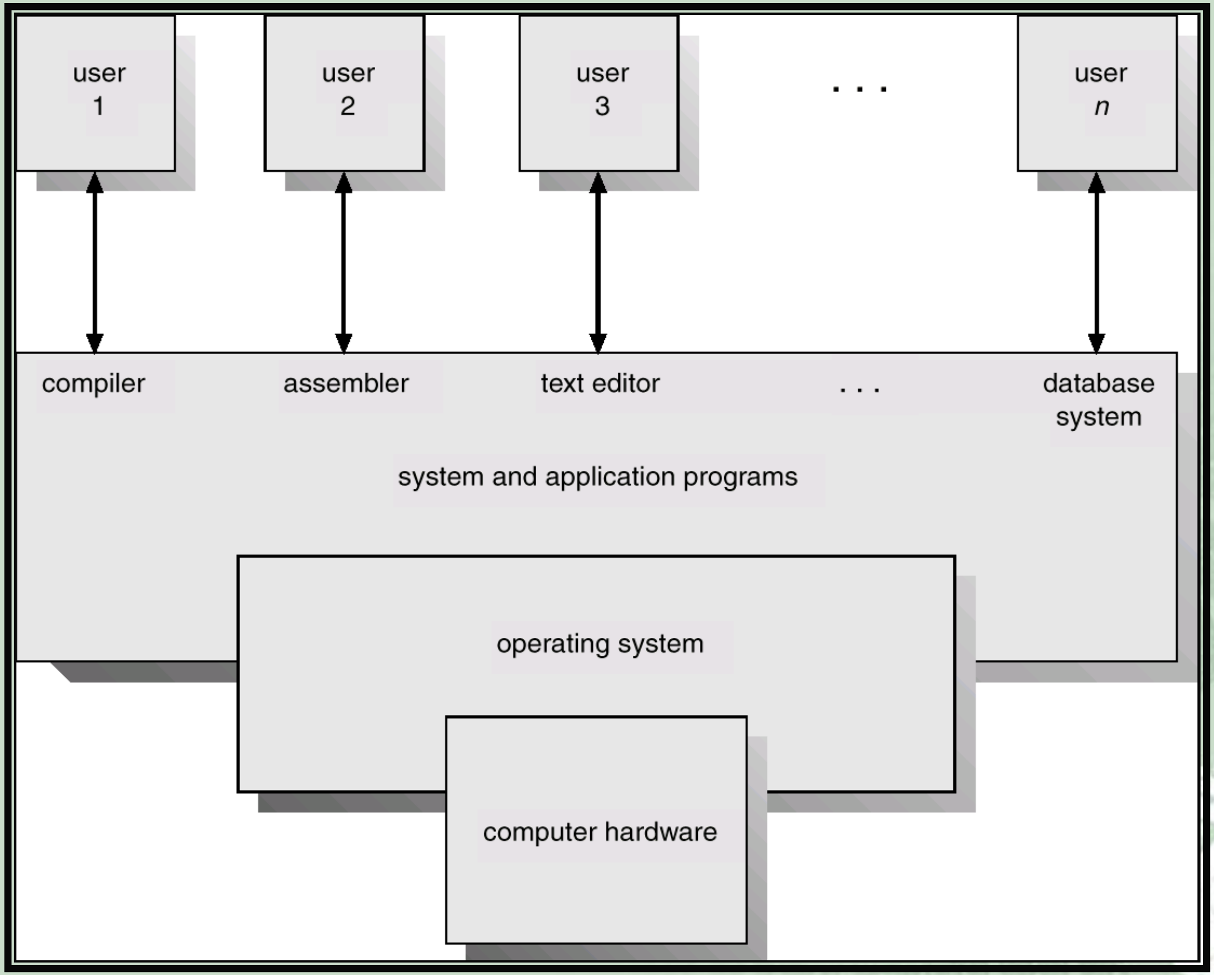


Operating Systems



Abstract View of System Components



Operating Systems

- What is an operating system?
  - Hard to define precisely, because operating systems arose historically as people needed to solve problems associated with using computers.
  - How about…
    - "Software that makes computing power available to users by controlling the hardware."
    - "Software executes when nothing else is happening."
    - "A collection of software modules including device drivers, libraries, and access routines."

What does a modern operating system do?

- **Provides Abstractions:**
  - Hardware has low-level physical resources with complicated, special-purpose interfaces.
  - OS provides abstractions that present clean interfaces.
  - Goal: make computer easier to use.
  - Examples: Processes, Unbounded Memory, Files, Synchronization and Communication Mechanisms.
- **Provides Standard Interface:**
  - Goal: portability.
  - Unix runs on many very different computer systems.

What does a modern operating system do?

- **Manages Resource Usage:**
  - Goal: allow multiple users to share resources fairly, efficiently, safely and securely.
  - Examples:
    - Multiple processes share one processor.
    - Multiple processes share multi-core processors.
    - Multiple programs share one physical memory.
    - Multiple users and files share one disk.
    - Multiple programs share a given amount of disk and network bandwidth.
- **Consumes Resources:**
  - Solaris takes up about 8 Mbytes physical memory.
  - Windows 7 has 50 million lines of code.

Where are OS's Used?

- In more and more places!
  - Desktop and Server Computers
  - DOS + Windows 95/98/ME
  - Windows NT/2000/XP/7/8
  - Free Unix variants: Linux, FreeBSD, NetBSD, etc.
  - Commercial Unix variants: Solaris, HP-UX, AIX, etc.
  - MacOS (1-9), OS X
- Some Game Consoles
  - Xbox: Cut-down Windows 2000

Where are OS’s Used?

- Personal Digital Assistants (PDAs)
  - PalmOS
  - Windows CE Windows Mobile
  - Embedded Linux
- Mobile Phones
  - Symbian OS
  - Windows Mobile
- Cars (fancy ones)

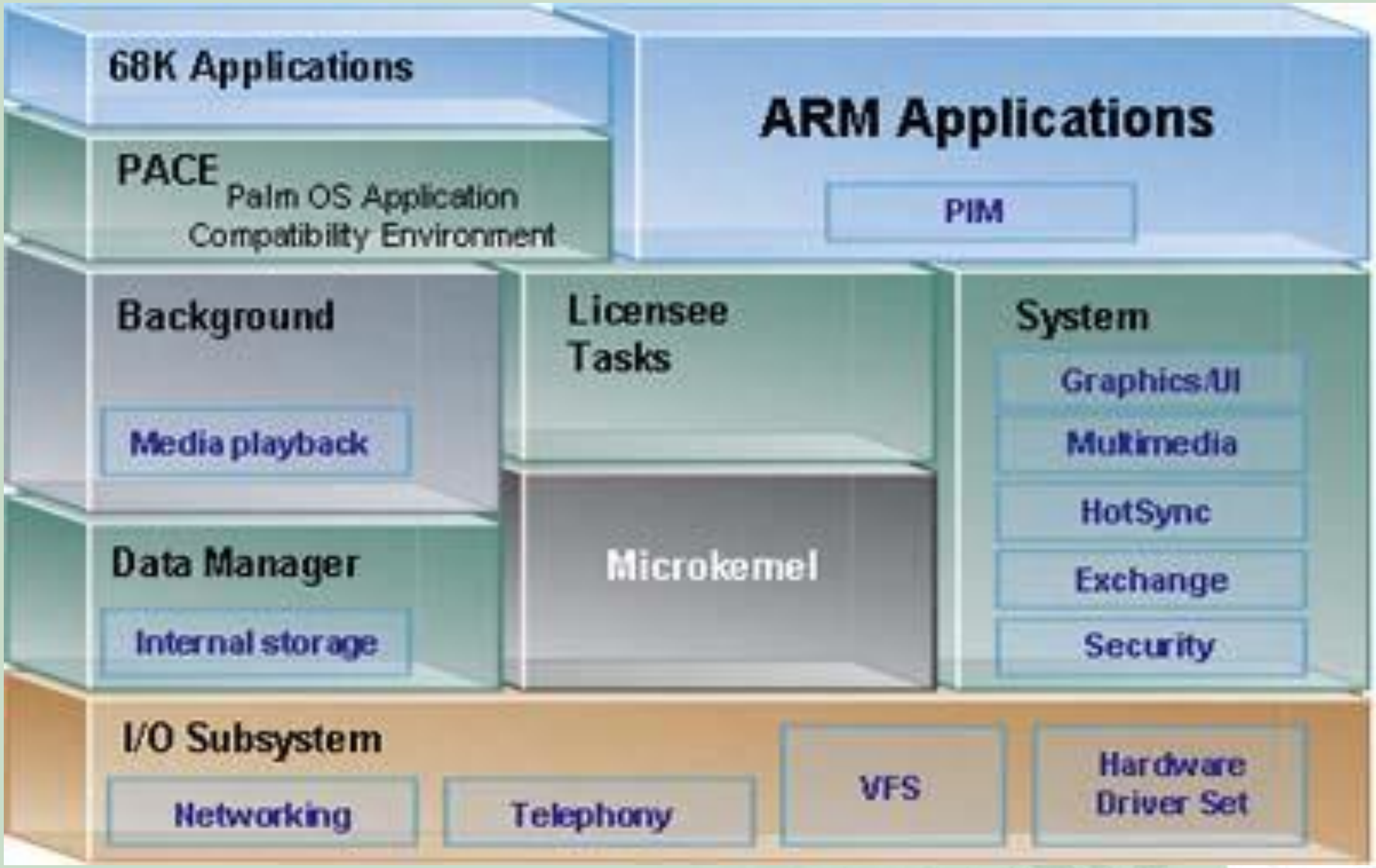
Where are OS’s Used?

- In the future also:
  - Digital Cameras (fancy ones)
  - MP3 Players (iPods, etc.)
  - Refrigerators!
  - Others?

Example OS: PalmOS

- Used for PalmPilot PDAs and successors
- Multitasking since PalmOS 5
- CPUs: Intel XScale,Texas Instruments OMAP, Motorola Dragonball MX
- Wireless: 802.11b, Bluetooth, GSM, CDMA
- 320×320+ displays
- Good battery utilisation

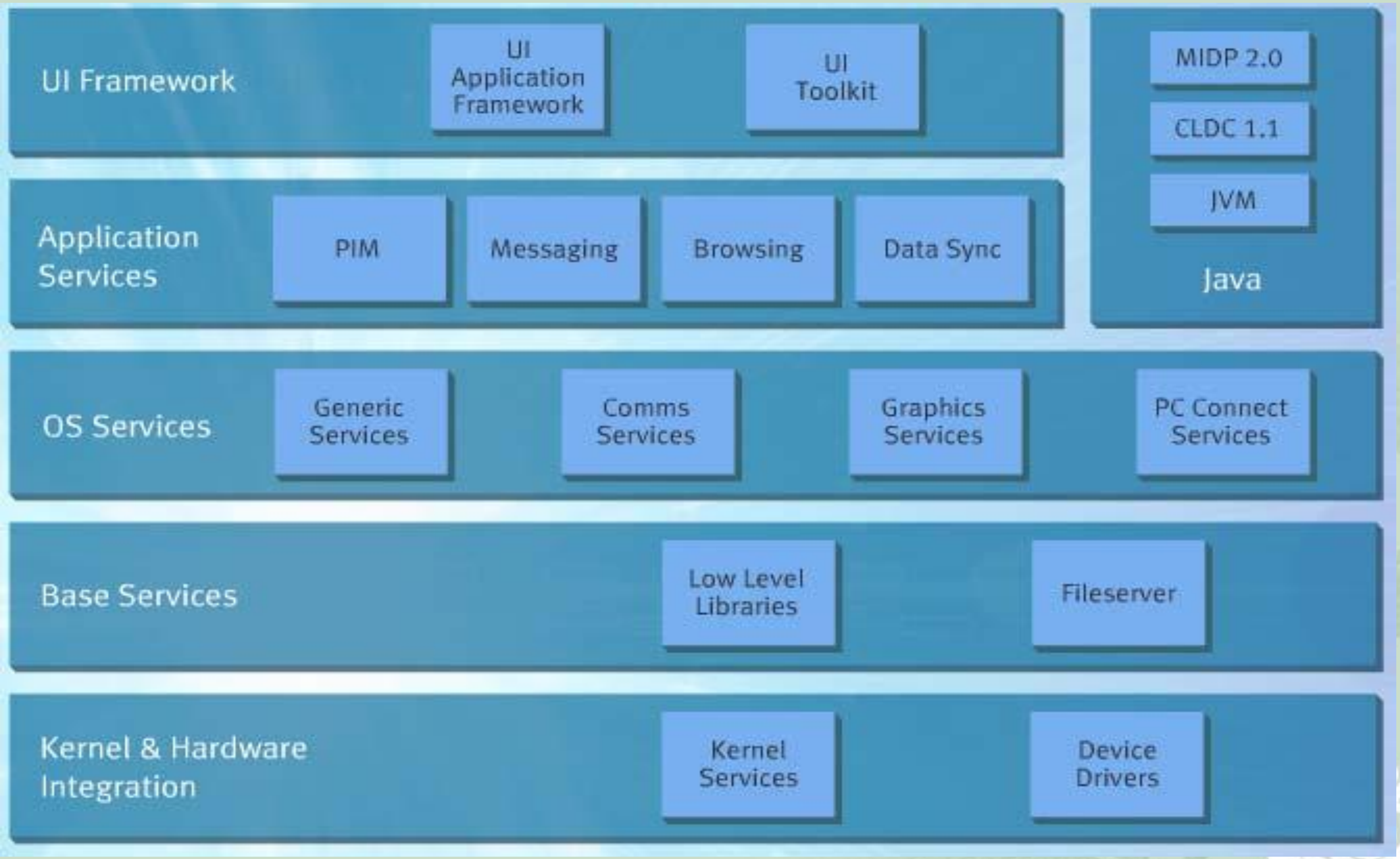
Example OS: PalmOS



Example OS: Symbian OS

- Designed for mobile phones
- Gives access to graphics, multimedia, networking, telephony, crypto,
- PC connectivity, etc.

Example OS: Symbian OS



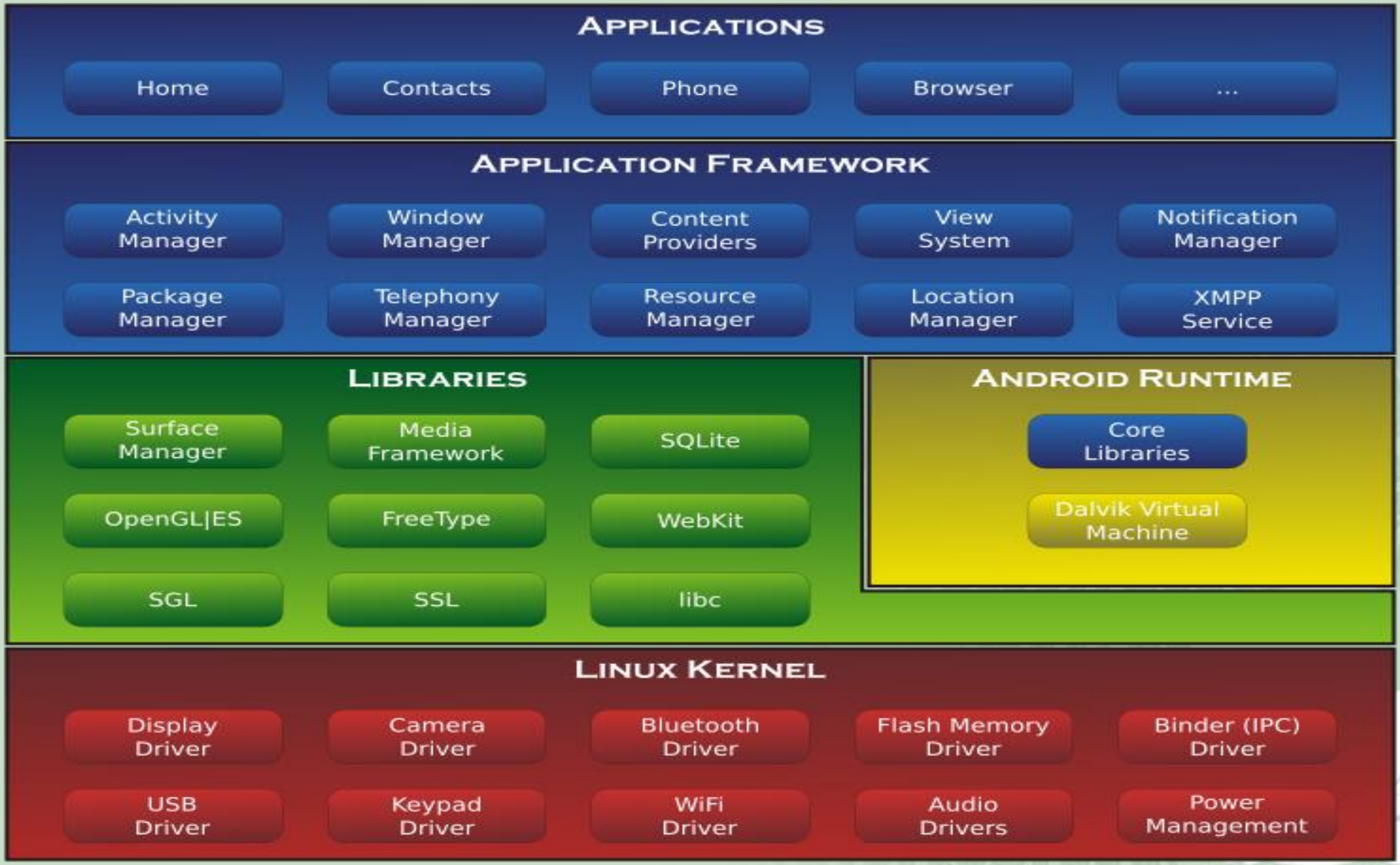
Example OS: Samsung Galaxy S III

- Android™ 4.1 Jelly Bean with access to apps, games & more at Google Play™
- 8 MP camera with 1.9 MP webcam

Example OS: SmartWatch

- Sony: SmartWatch; MN2; Android ™ 2.1 and later;
- multi-level touchscreen;
- bluetooth 3.0;
- OLED 1.3 *display*

Android-System-Architecture



APPLICATIONS

Home, Contacts, Phone, Browser, …

APPLICATION FRAMEWORK

Activity Manager, Window Manager, Content Providers, View System, Notification Manager

Package Manager, Telephony Manager, Resource Manager, Location Manager, XMPP Service

LIBRARIES

Surface Manager, Media Framework, SQLite

OpenGL|ES, FreeType, WebKit

SGL, SSL, libc

ANDROID RUNTIME

Core Libraries

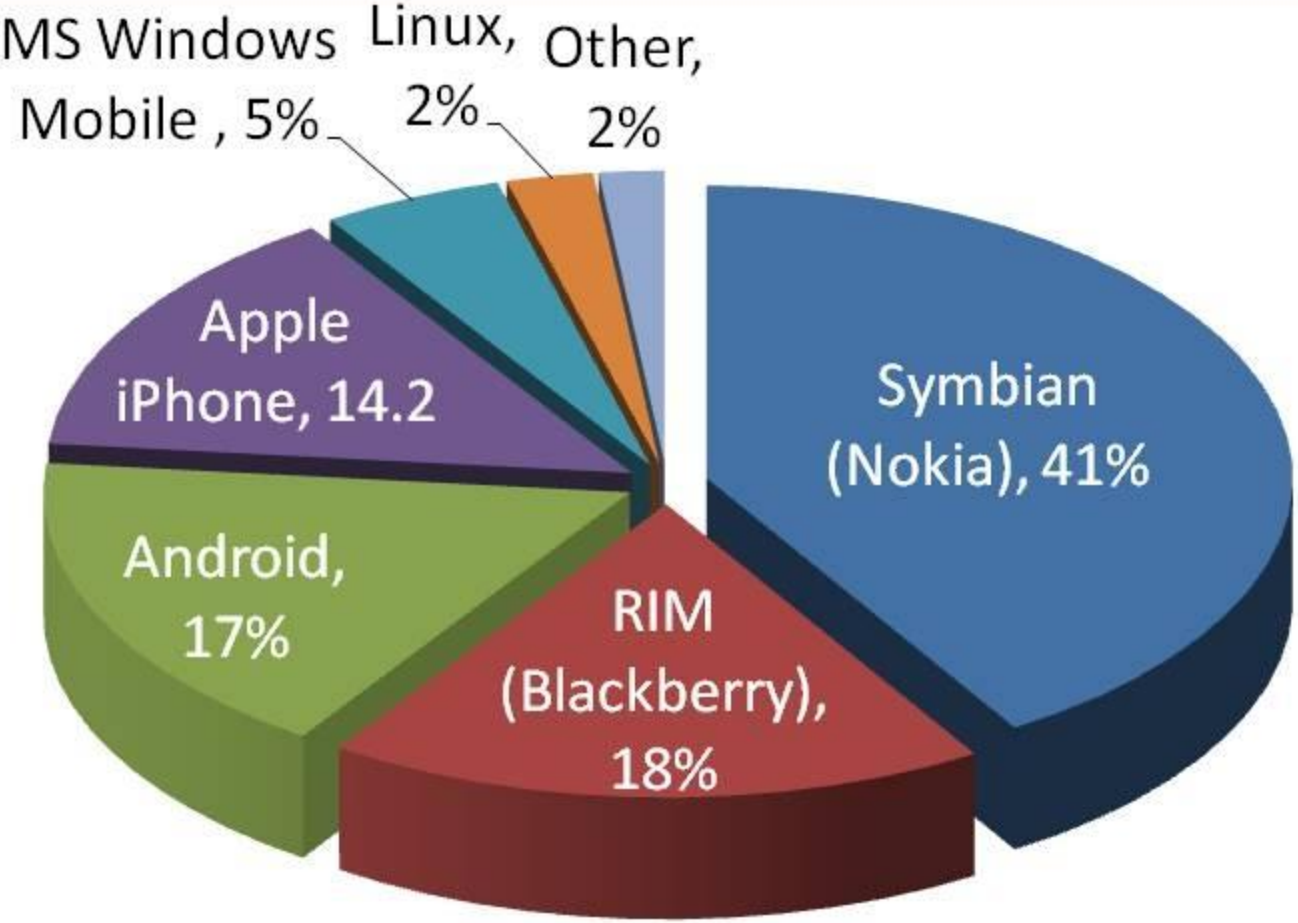
Dalvik Virtual Machine

LINUX KERNEL

Display Driver, Camera Driver, Bluetooth Driver, Flash Memory Driver, Binder (IPC) Driver

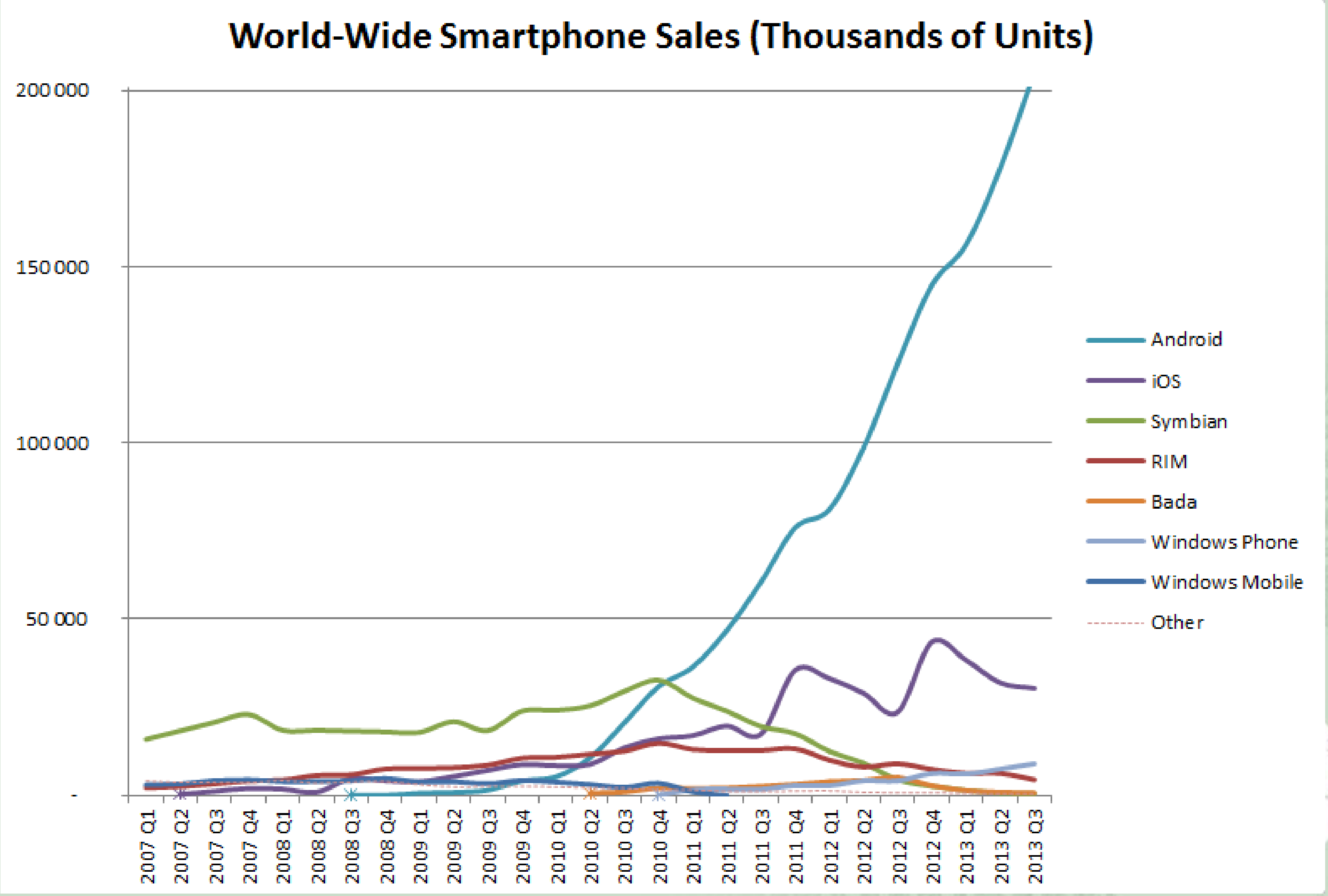
USB Driver, Keypad Driver, WiFi Driver, Audio Drivers, Power Management

Global Smartphone Market Share by Operating System



Source: *Financial Times*, August 13, 2010

What Happens Now?



The Future…

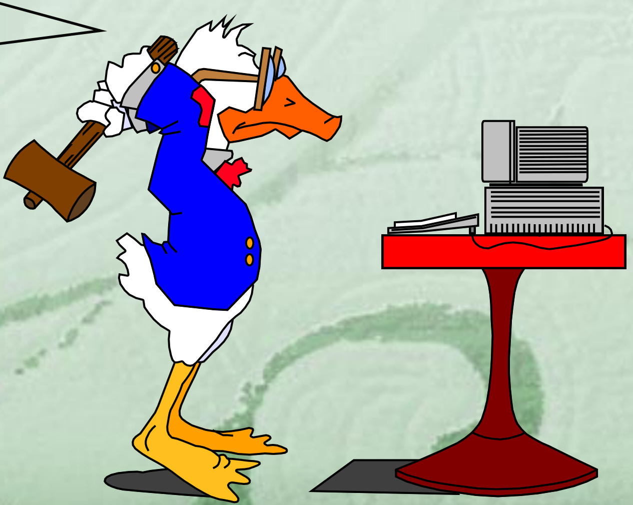
- In the future, computers will continue to become physically smaller and more portable.
- Operating systems have to deal with issues like disconnected operation and mobility.
- Media rich information within the grasp of common people - information with psuedo-real time components like voice and video.
- Operating systems will have to adjust to deliver acceptable performance for these new forms of data.

Finally

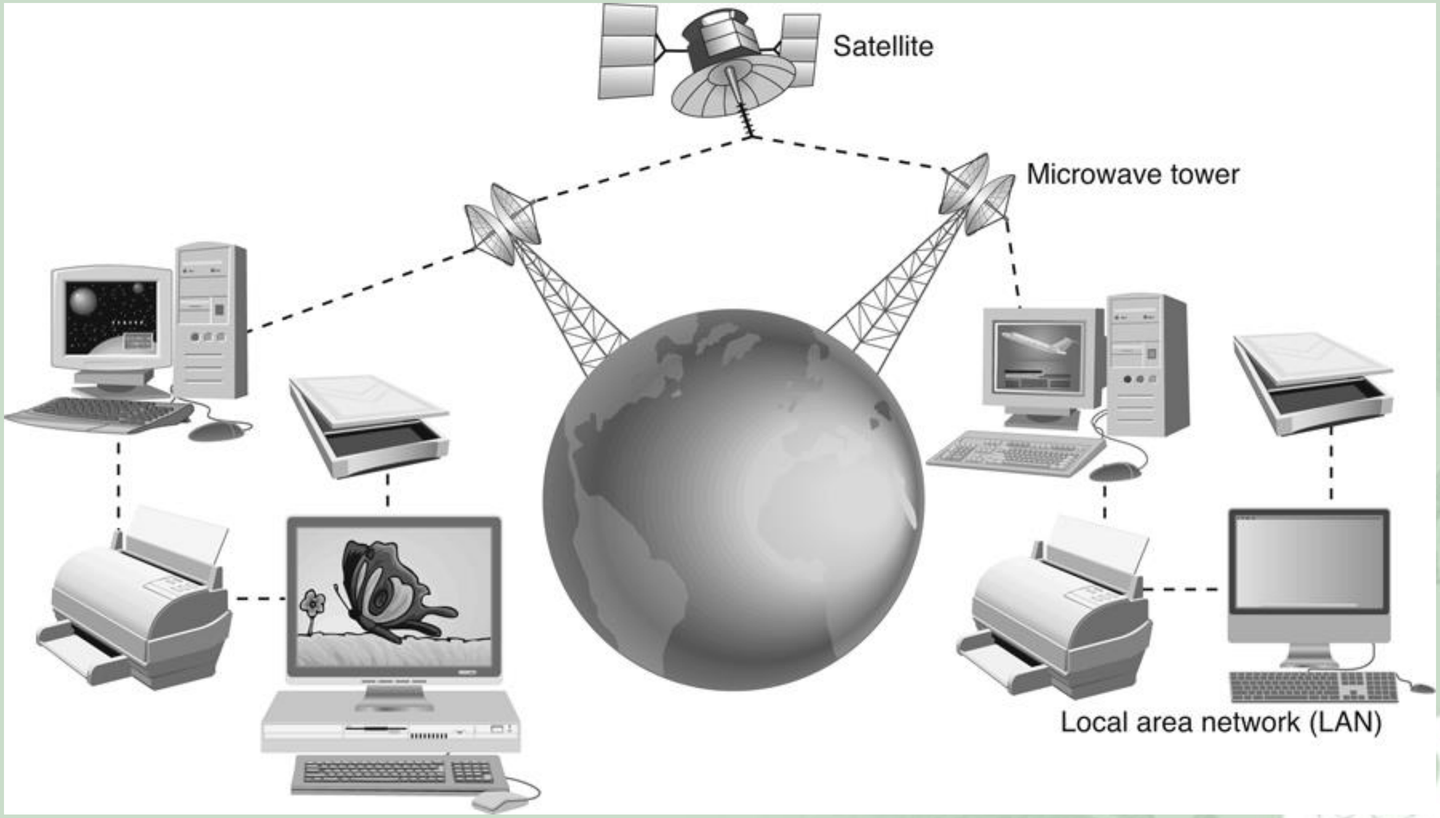
- Operating systems are so large no one person understands whole system. Outlives any of its original builders.
- The major problem facing computer science today is how to build large, reliable software systems.
- Operating systems are one of very few examples of existing large software systems, and by studying operating systems we may learn lessons applicable to the construction of larger systems.

Computer System Overview

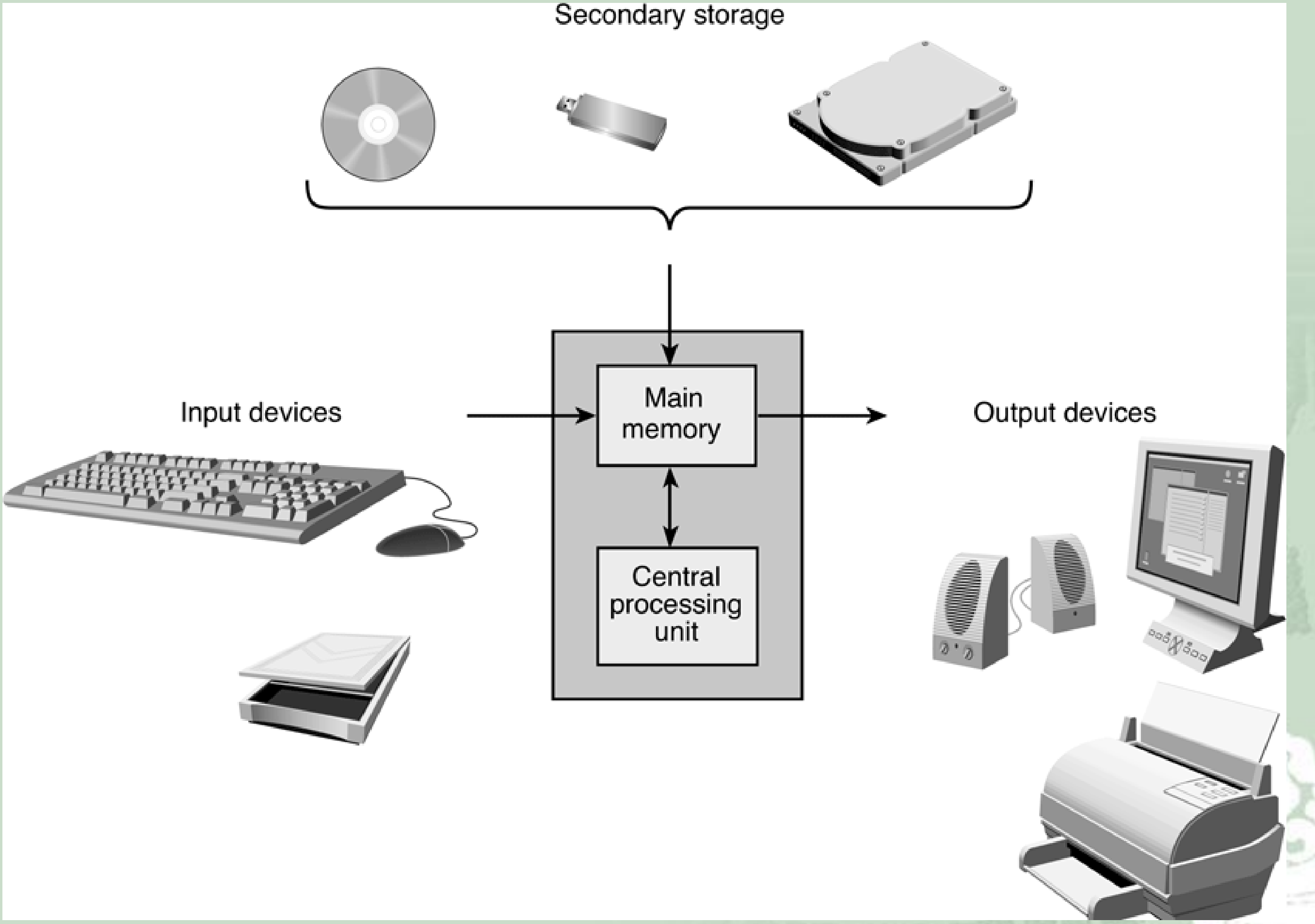
- Let’s figure out
- what’s inside
- this thing...



Network



Components of a Computer

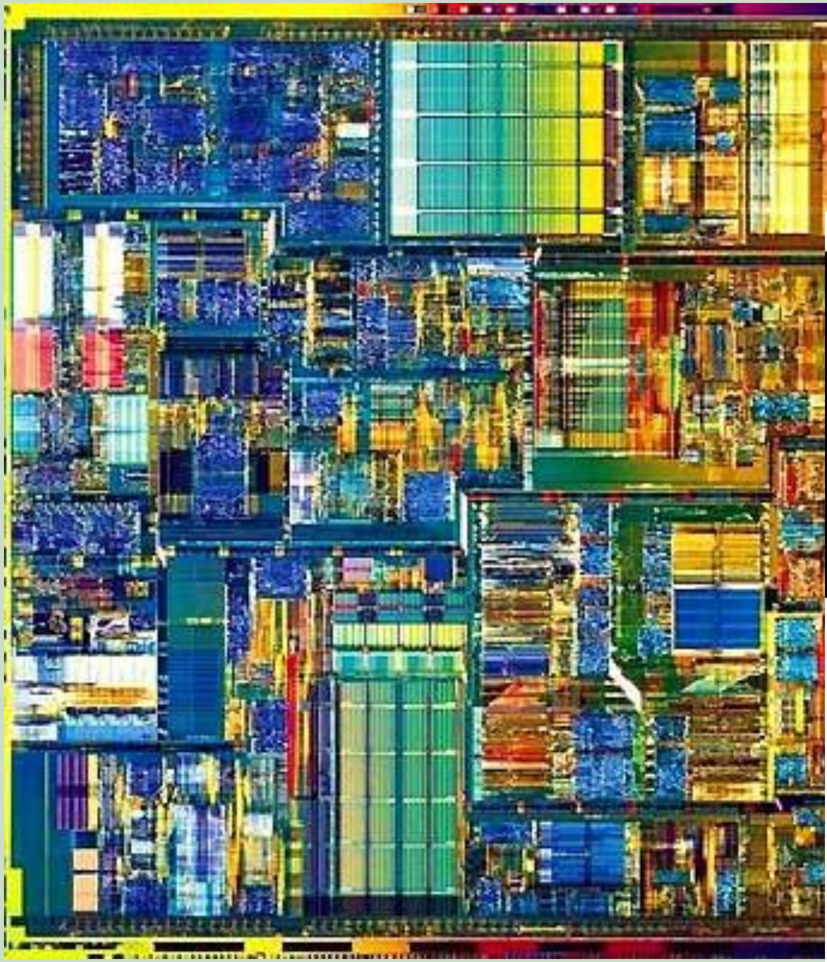


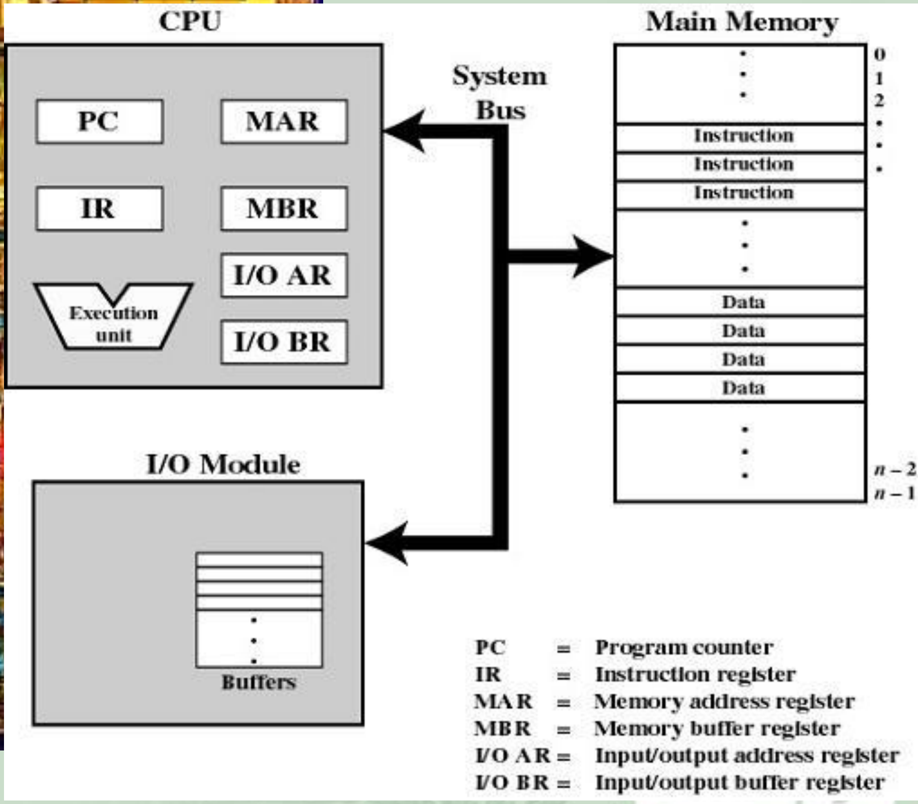
Computer Systems

- Registers
- Interrupts
- Caching
- Input/Output
- Protection
- Summary

Registers

CPU





PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

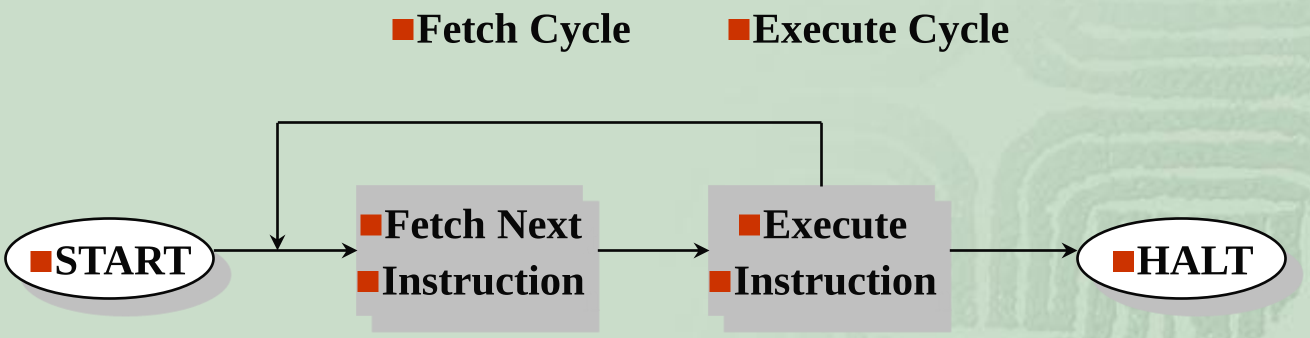
Processor Registers

- User-visible registers
  - May be referenced by machine language
  - Available to all programs - application programs and system programs
    - Data Registers – can be changed by user
    - Address Registers – could be separate from data register
    - Stack Registers – user / supervisor stacks
    - Condition Codes – results of operations
- Control and status registers
  - May or may not be visible
    - Program Counter (PC) – address of next instruction
    - Instruction Register (IR) – most recently fetched instruction
    - MAR/MBR – memory reference registers
    - Program Status Word (PSW) – condition codes, interrupts, mode

Registers

Instruction Execution

- Processor executes instructions in a program
- Instructions are fetched from memory on at a time



Fetch Cycle

Execute Cycle

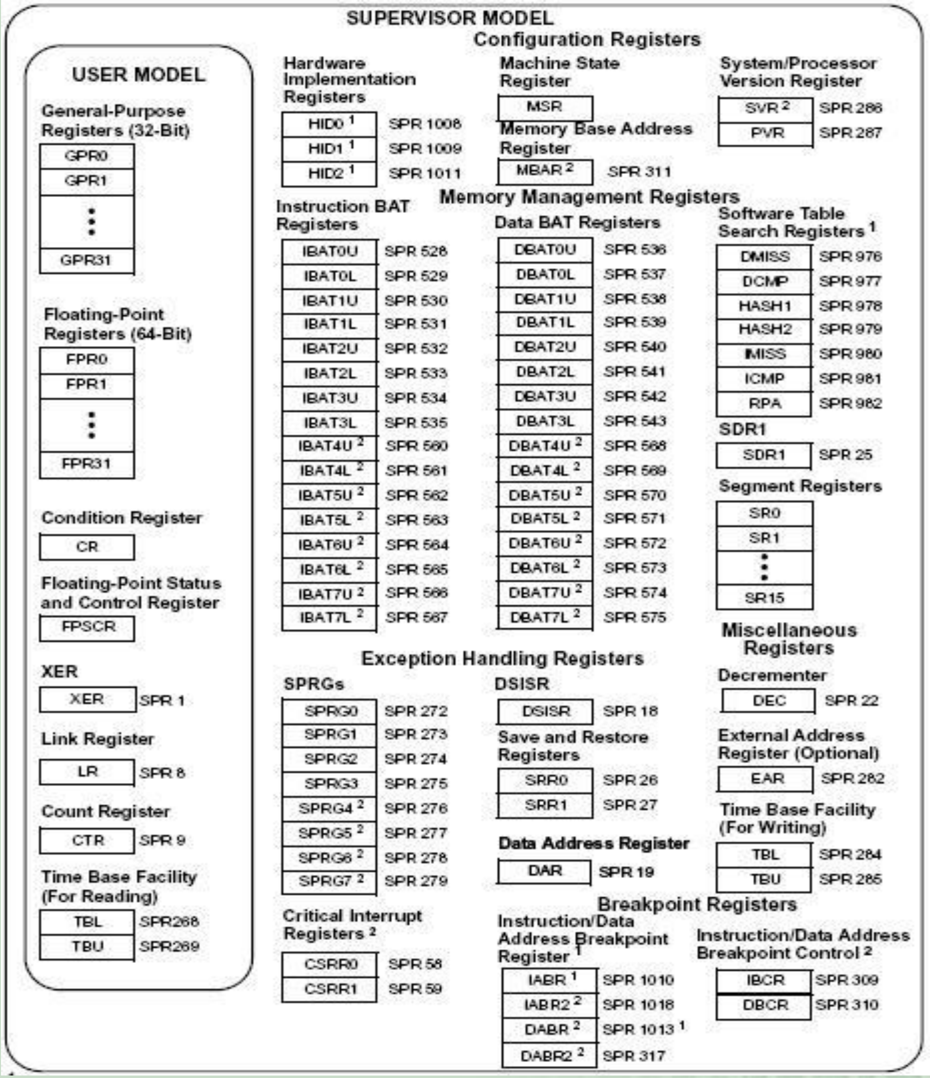
START

Fetch Next Instruction

Execute Instruction

HALT

Lots of Registers…



USER MODEL

General-Purpose Registers (32-Bit)

GPR0
GPR1
⋮
GPR31

Floating-Point Registers (64-Bit)

FPR0
FPR1
⋮
FPR31

Condition Register

CR

Floating-Point Status and Control Register

FPSCR

XER

XER	SPR 1

Link Register

LR	SPR 8

Count Register

CTR	SPR 9

Time Base Facility (For Reading)

TBL	SPR268
TBU	SPR269

SUPERVISOR MODEL

Configuration Registers

Hardware Implementation Registers

HID0[1]	SPR 1008
HID1[1]	SPR 1009
HID2[1]	SPR 1011

Machine State Register

MSR

Memory Base Address Register

MBAR[2]	SPR 311

System/Processor Version Register

SVR[2]	SPR 286
PVR	SPR 287

Memory Management Registers

Instruction BAT Registers

IBAT0U	SPR 528
IBAT0L	SPR 529
IBAT1U	SPR 530
IBAT1L	SPR 531
IBAT2U	SPR 532
IBAT2L	SPR 533
IBAT3U	SPR 534
IBAT3L	SPR 535
IBAT4U[2]	SPR 560
IBAT4L[2]	SPR 561
IBAT5U[2]	SPR 562
IBAT5L[2]	SPR 563
IBAT6U[2]	SPR 564
IBAT6L[2]	SPR 565
IBAT7U[2]	SPR 566
IBAT7L[2]	SPR 567

Data BAT Registers

DBAT0U	SPR 536
DBAT0L	SPR 537
DBAT1U	SPR 538
DBAT1L	SPR 539
DBAT2U	SPR 540
DBAT2L	SPR 541
DBAT3U	SPR 542
DBAT3L	SPR 543
DBAT4U[2]	SPR 568
DBAT4L[2]	SPR 569
DBAT5U[2]	SPR 570
DBAT5L[2]	SPR 571
DBAT6U[2]	SPR 572
DBAT6L[2]	SPR 573
DBAT7U[2]	SPR 574
DBAT7L[2]	SPR 575

Software Table Search Registers[1]

DMISS	SPR 976
DCMP	SPR 977
HASH1	SPR 978
HASH2	SPR 979
IMISS	SPR 980
ICMP	SPR 981
RPA	SPR 982

SDR1

SDR1	SPR 25

Segment Registers

SR0
SR1
⋮
SR15

Exception Handling Registers

SPRGs

SPRG0	SPR 272
SPRG1	SPR 273
SPRG2	SPR 274
SPRG3	SPR 275
SPRG4[2]	SPR 276
SPRG5[2]	SPR 277
SPRG6[2]	SPR 278
SPRG7[2]	SPR 279

DSISR

DSISR	SPR 18

Save and Restore Registers

SRR0	SPR 26
SRR1	SPR 27

Data Address Register

DAR	SPR 19

Critical Interrupt Registers[2]

CSRR0	SPR 58
CSRR1	SPR 59

Miscellaneous Registers

Decrementer

DEC	SPR 22

External Address Register (Optional)

EAR	SPR 282

Time Base Facility (For Writing)

TBL	SPR 284
TBU	SPR 285

Breakpoint Registers

Instruction/Data Address Breakpoint Register[1]

IABR[1]	SPR 1010
IABR2[2]	SPR 1018
DABR[2]	SPR 1013[1]
DABR2[2]	SPR 317

Instruction/Data Address Breakpoint Control[2]

IBCR	SPR 309
DBCR	SPR 310

Computer Systems

- Registers
- Interrupts
- Caching
- Input/Output
- Protection
- Summary

Interrupts

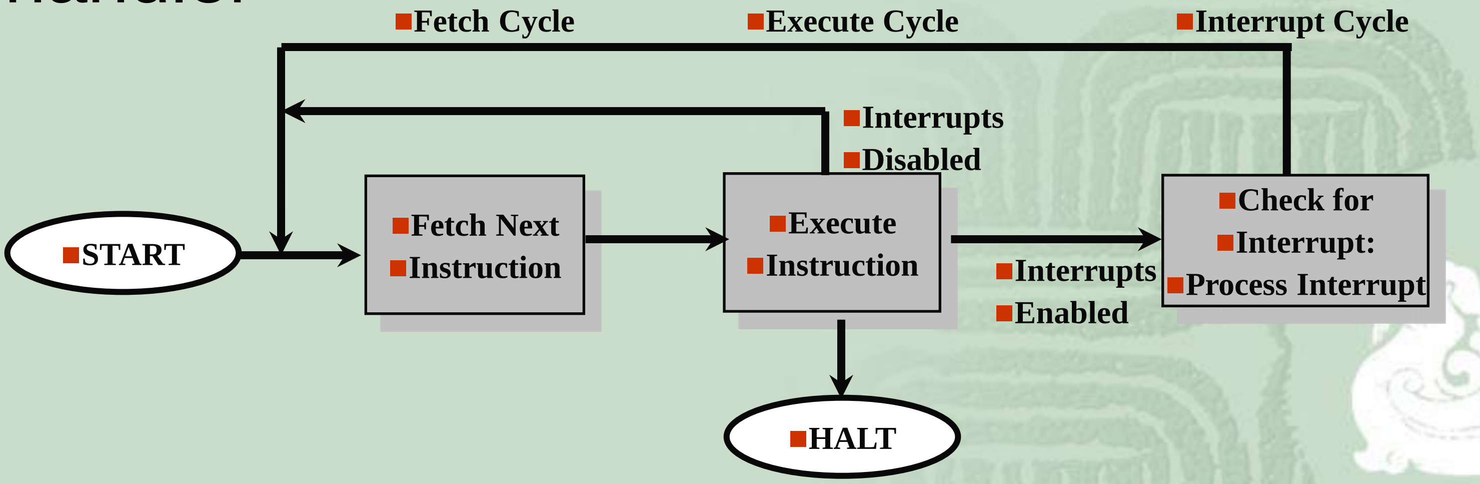
- The interrupt was the principle tool available to system programmers in developing multi-tasking systems!
- Improves processing efficiency by allowing the processor to execute other instructions while an I/O operation is in progress
- A suspension of a process caused by an event external to that process and performed in such a way that the process can be resumed

Processing of Interrupts

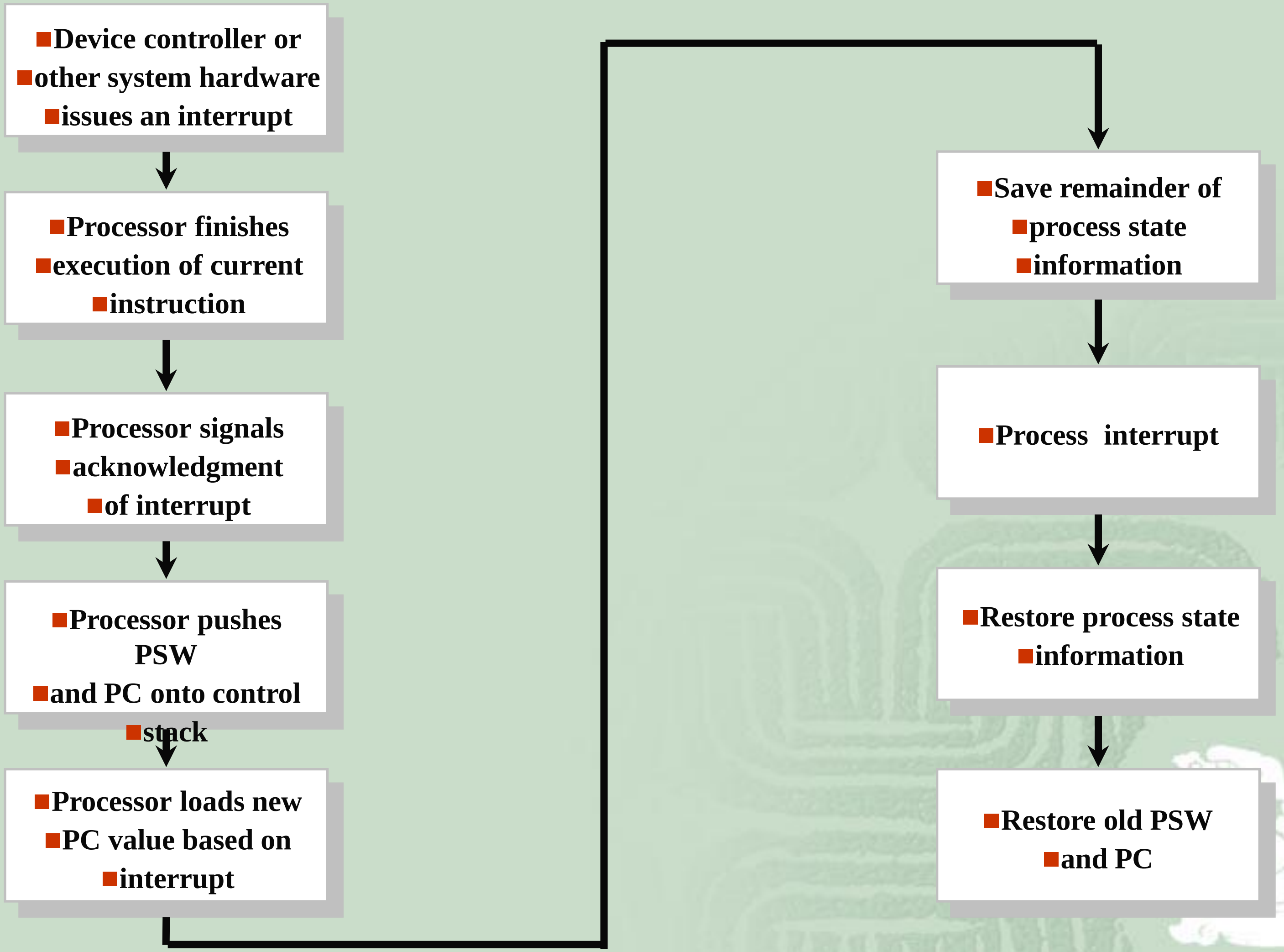
- Classes of Interrupts
  - Program
    - arithmetic overflow
    - division by zero
    - execute illegal instruction
    - reference outside user's memory space
  - Timer
  - I/O
  - Hardware failure
- An interrupt handler determines nature of the interrupt and performs whatever actions are needed
  - Control is transferred to this program
  - Generally part of the operating system

Interrupt Cycle

- Processor checks for interrupts
- If no interrupts fetch the next instruction for the current program
- If an interrupt is pending, suspend execution of the current program, and execute the interrupt handler



Simple Interrupt Processing



Device controller or
other system hardware
issues an interrupt

Processor finishes
execution of current
instruction

Processor signals
acknowledgment
of interrupt

Processor pushes PSW
and PC onto control
stack

Processor loads new
PC value based on
interrupt

Save remainder of
process state
information

Process interrupt

Restore process state
information

Restore old PSW
and PC