嵌入式系统设计方法

主要内容

- 嵌入式系统面临的挑战
- 嵌入式系统的设计过程
- 嵌入式系统设计方法学

嵌入式系统设计所面临的挑战

- 需要多少硬件?
- 如何满足时限要求，如何处理多项功能在时间上的协调一致关系?
- 如何降低系统的功耗?
- 如何设计以保证系统可升级?
- 如何保证系统可靠地工作?

嵌入式系统设计者要求

- 懂得系统的整个构架
- 详细了解硬件的细节
- 软件设计满足：
  - 实时要求
  - 低功耗
  - 代码量小
- 详细了解领域知识

Design objectives

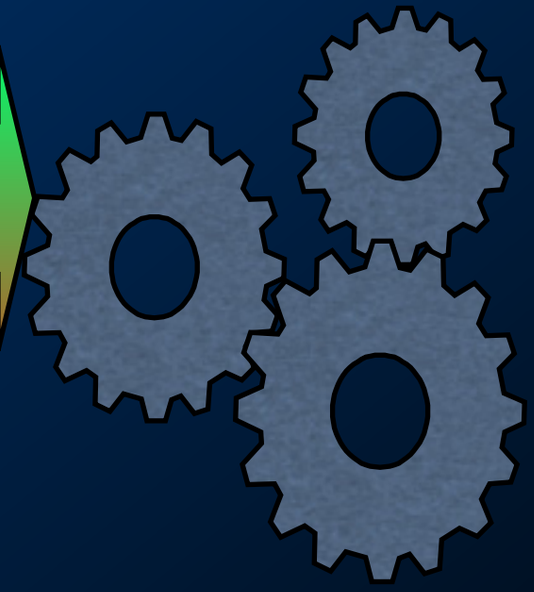
- Cost
- Performance
- Power
- Area
- Scalability and reusability
- Fault tolerance
- Thermal characteristics
- …

嵌入式系统的设计过程

问题：
功能、
性能要求、
价格、
开发周期等
约束

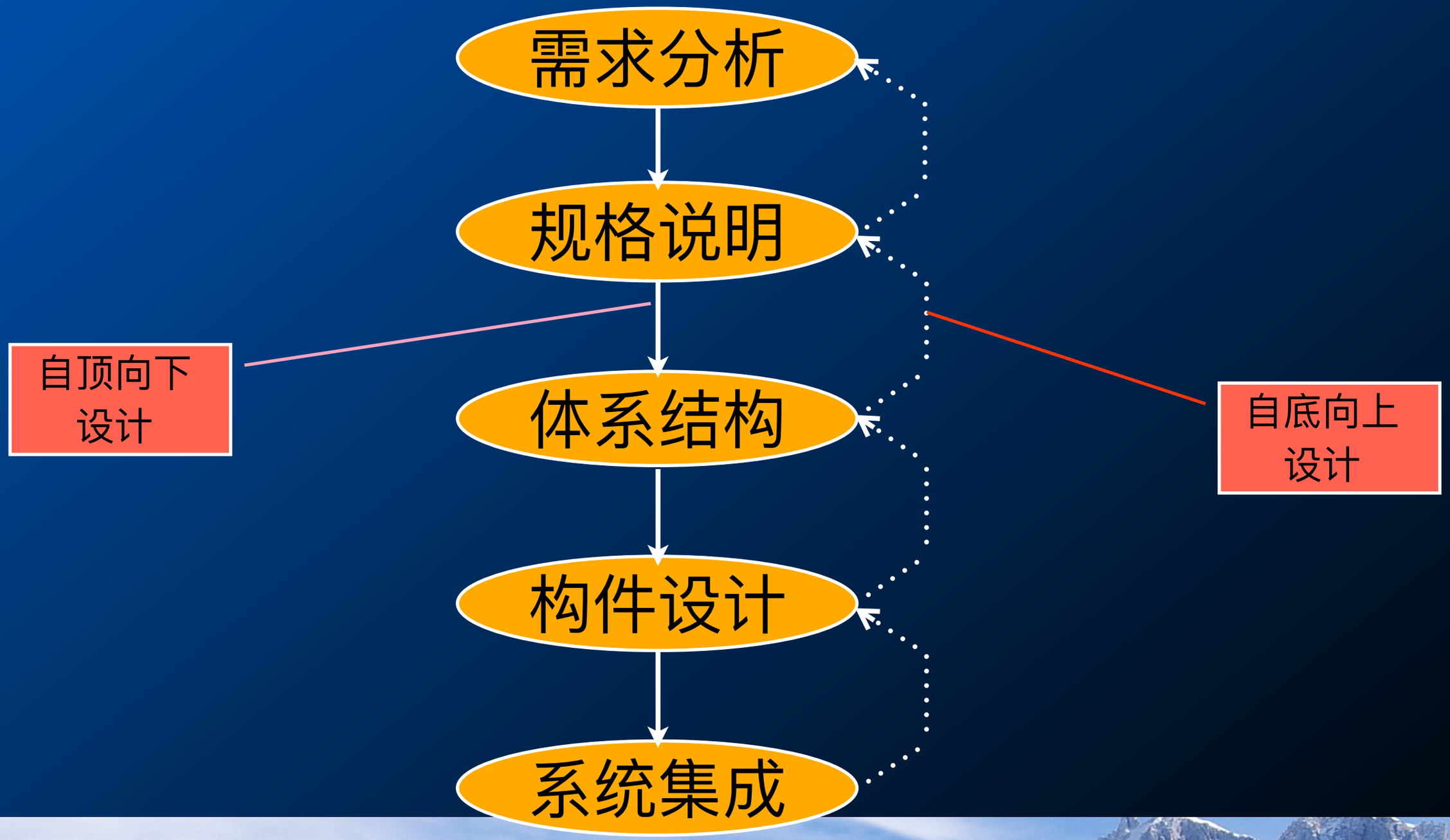
设计：
选择
折衷
分析比较
计算
评价

嵌入式系统



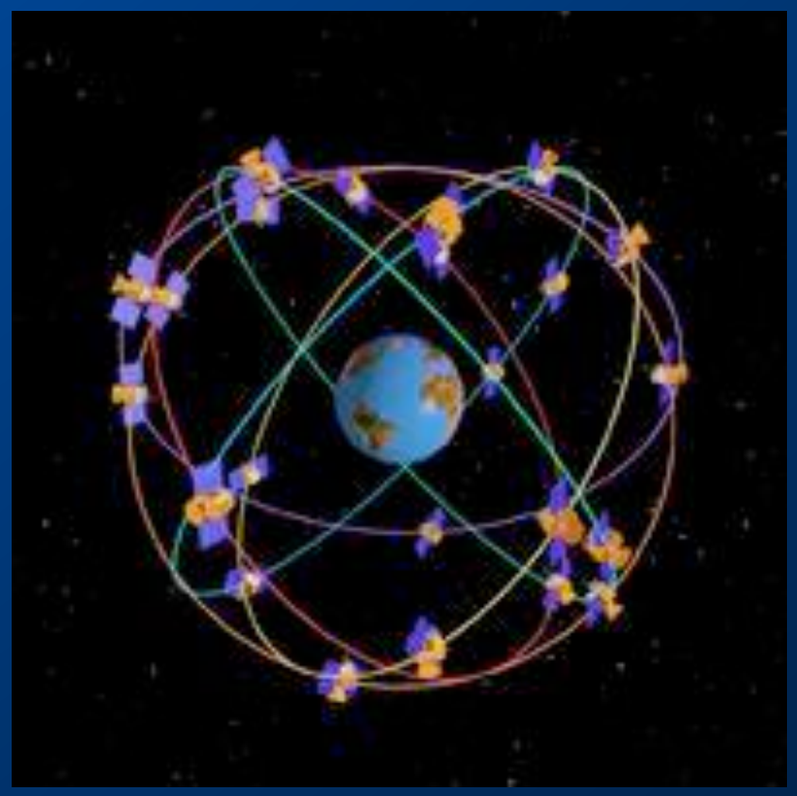
方法 ⇢ 工具

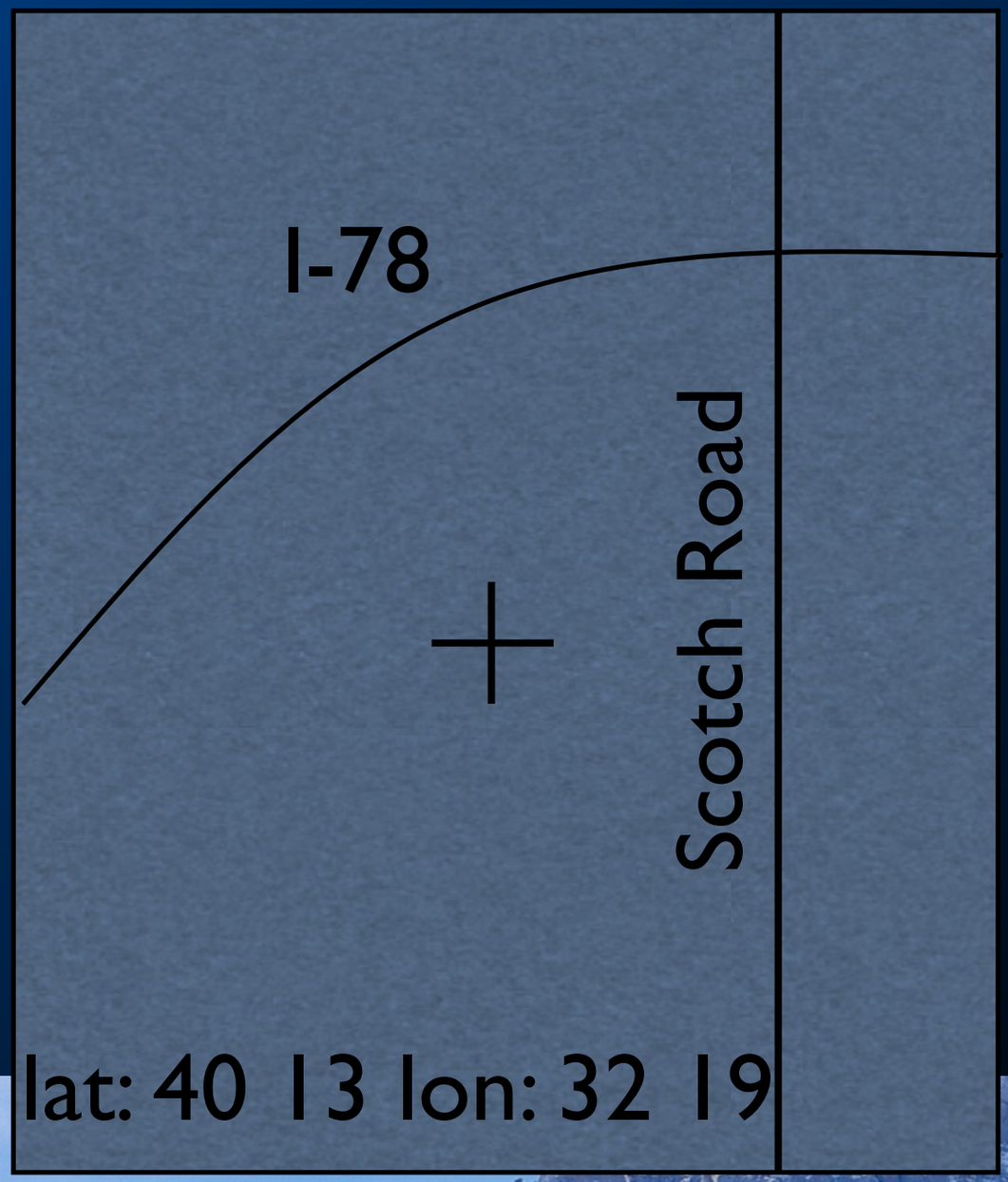
嵌入式系统的设计过程的基本流程



Example: GPS moving map requirements

- Moving map obtains position from GPS, paints map from local database.





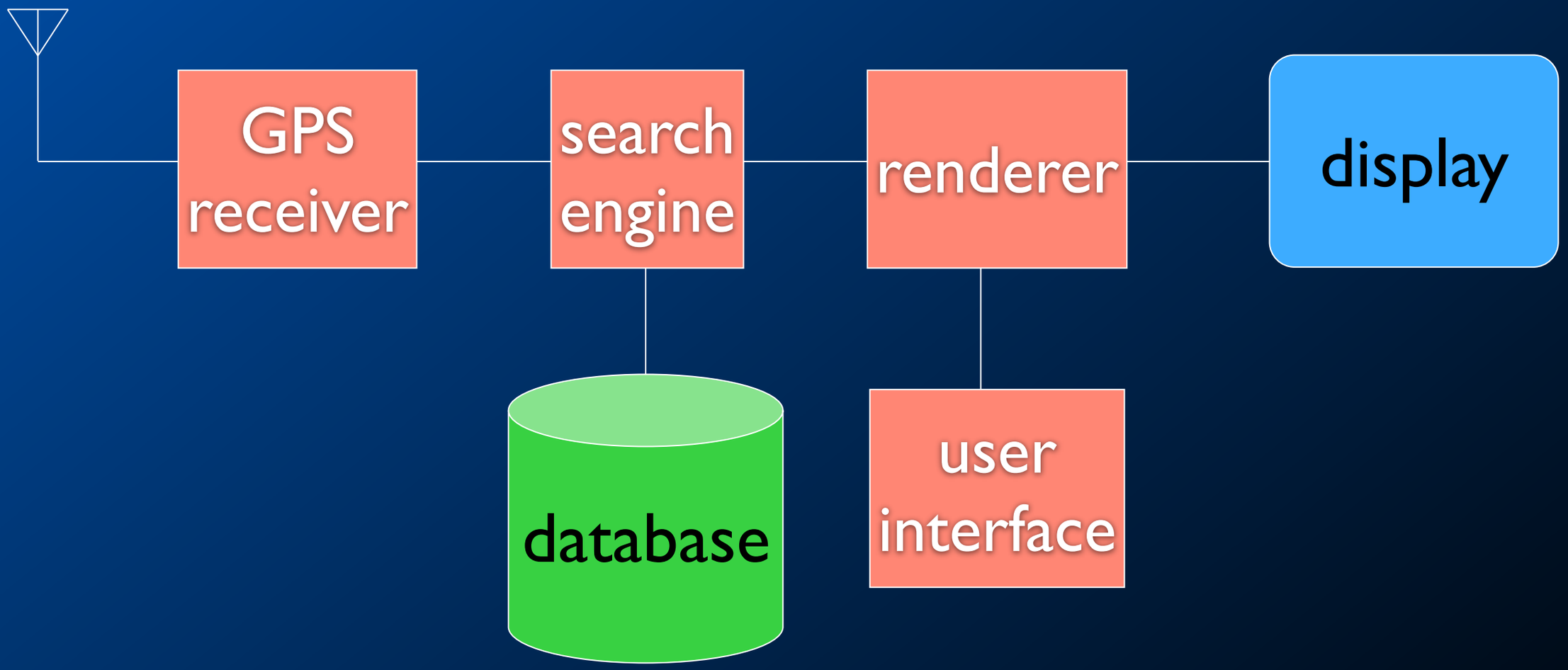
Architecture/Design

- What major components go satisfying the specification?
- Hardware components:
  - CPUs, peripherals, etc.
- Software components:
  - major programs and their operations.
- Must take into account functional and non-functional specifications.

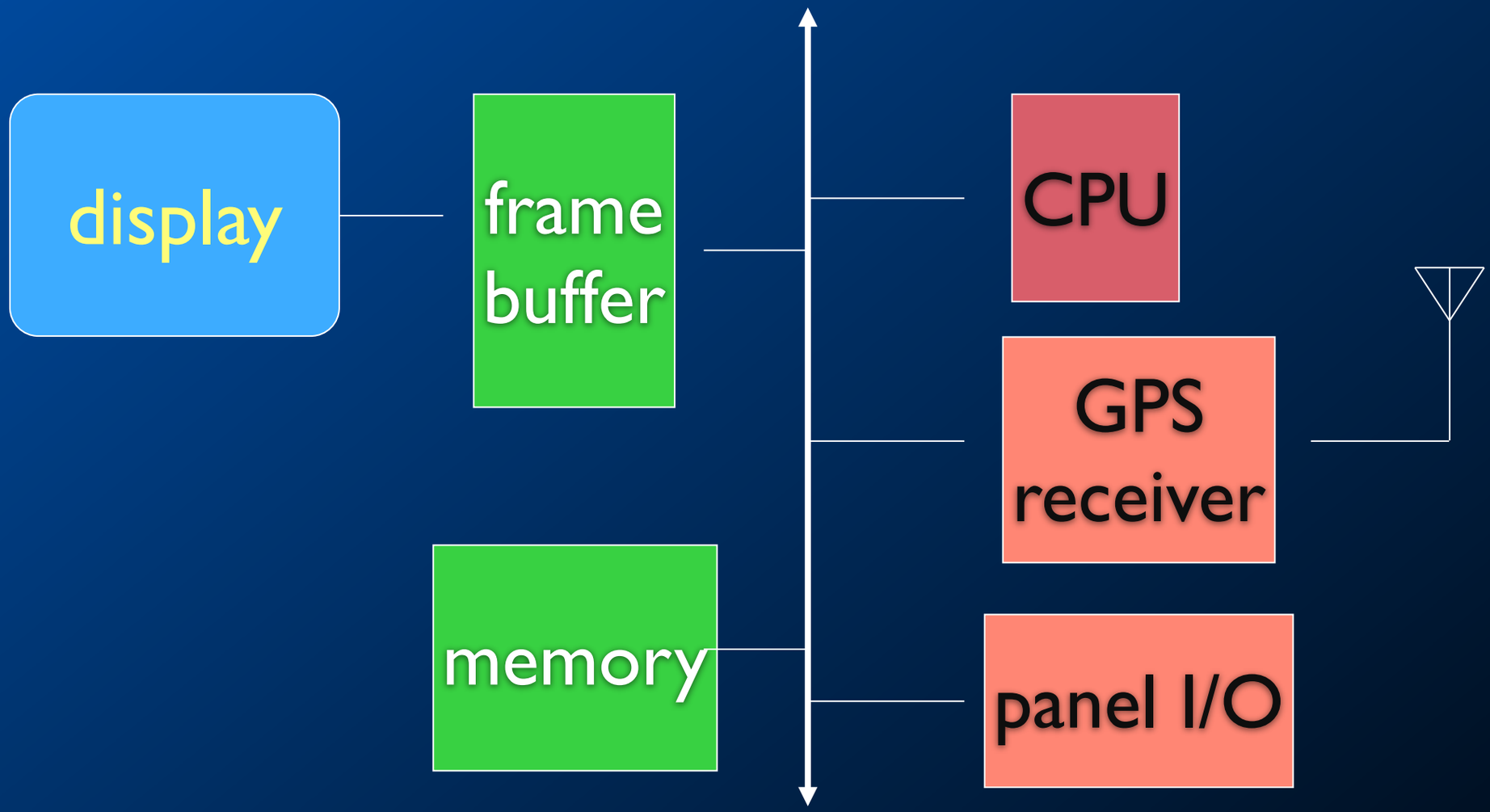
Designing hardware and software components

- Must spend time architecting the system before you start coding.
- Some components are ready-made, some can be modified from existing designs, others must be designed from scratch.

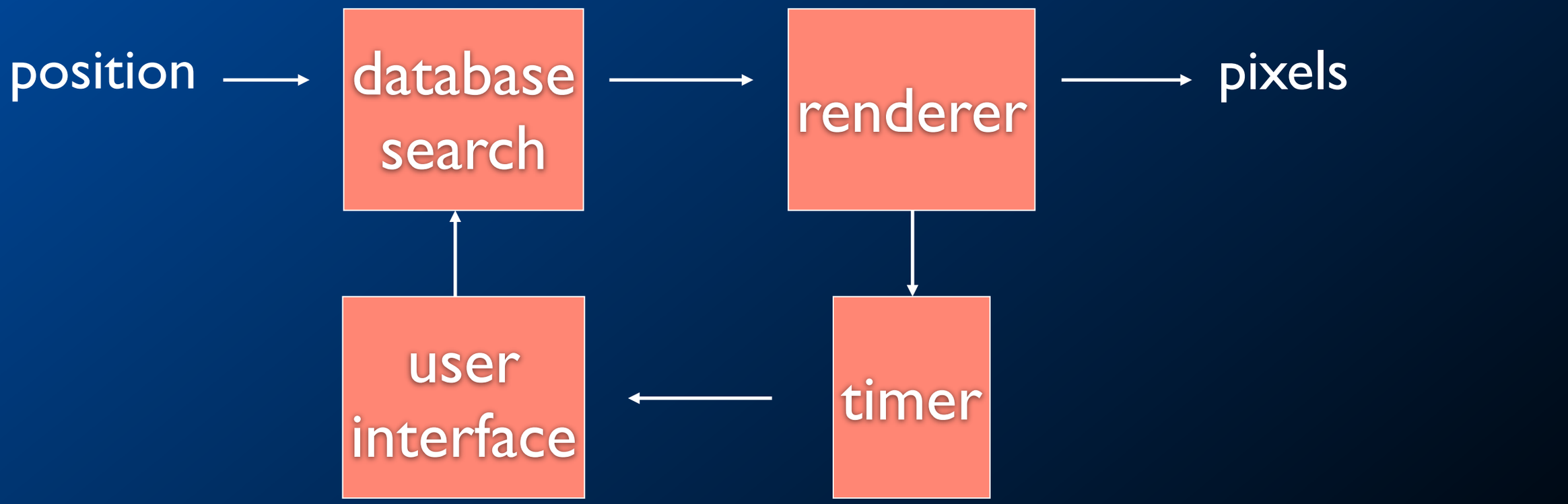
GPS moving map block diagram



GPS moving map hardware architecture



GPS moving map software architecture



软硬件的划分

- 嵌入式系统的设计涉及硬件与软件部件，设计中必须决定什么功能由硬件实现，什么功能由软件实现。
- 硬件和软件具有双重性
- 软硬件变动对系统的决策造成影响
- 划分和选择需要考虑多种因素
- 硬件和软件的双重性是划分决策的前提

通常由软件实现的部分

- 操作系统功能
  - 任务调度
  - 资源管理
  - 设备驱动
- 协议栈
  - TCP / IP
- 应用软件框架
- 除基本系统、物理接口、基本逻辑电路，许多由硬件实现的功能都可以由软件实现。

双重性部分

- 算法
  - 加密 / 解密
  - 编码 / 解码
  - 压缩 / 解压
  - ……
- 数学运算
  - 浮点运算， FFT， ……
  - ……

软硬件技术对系统结构的影响

- 硬软件设计的趋势——融合、渗透
  - 硬件设计的软件化
    - VHDL,Verilog
    - HANDEL-C
  - 软件实现的硬件化
    - 各种算法的ASIC
- 对系统设计的影响——协同设计
  - 增加灵活性
  - 增加了风险

嵌入式系统设计方法学

嵌入式系统设计方法的演变

1

以PCB、CAD和在线仿真器为主要工具

2

EDA和EOS为开发平台

3

以IP内核库为设计基础，用软硬件协同设计技术的系统级设计方法

传统的嵌入式系统设计过程

- 传统软硬件设计过程的基本特征:
  - 系统在一开始就被划分为软件和硬件两大部分
  - 软件和硬件独立进行开发设计
  - “Hardware first” approach often adopted
- 隐含的一些问题:
  - 软硬件之间的交互受到很大限制
    - 凭经验划分软硬件
    - 软硬件之间的相互性能影响很难评估
  - 系统集成相对滞后，NRE较大
- 因此:
  - Poor quality designs（设计质量差）
  - Costly modifications（设计修改难）
  - Schedule slippages（研制周期不能有效保障）

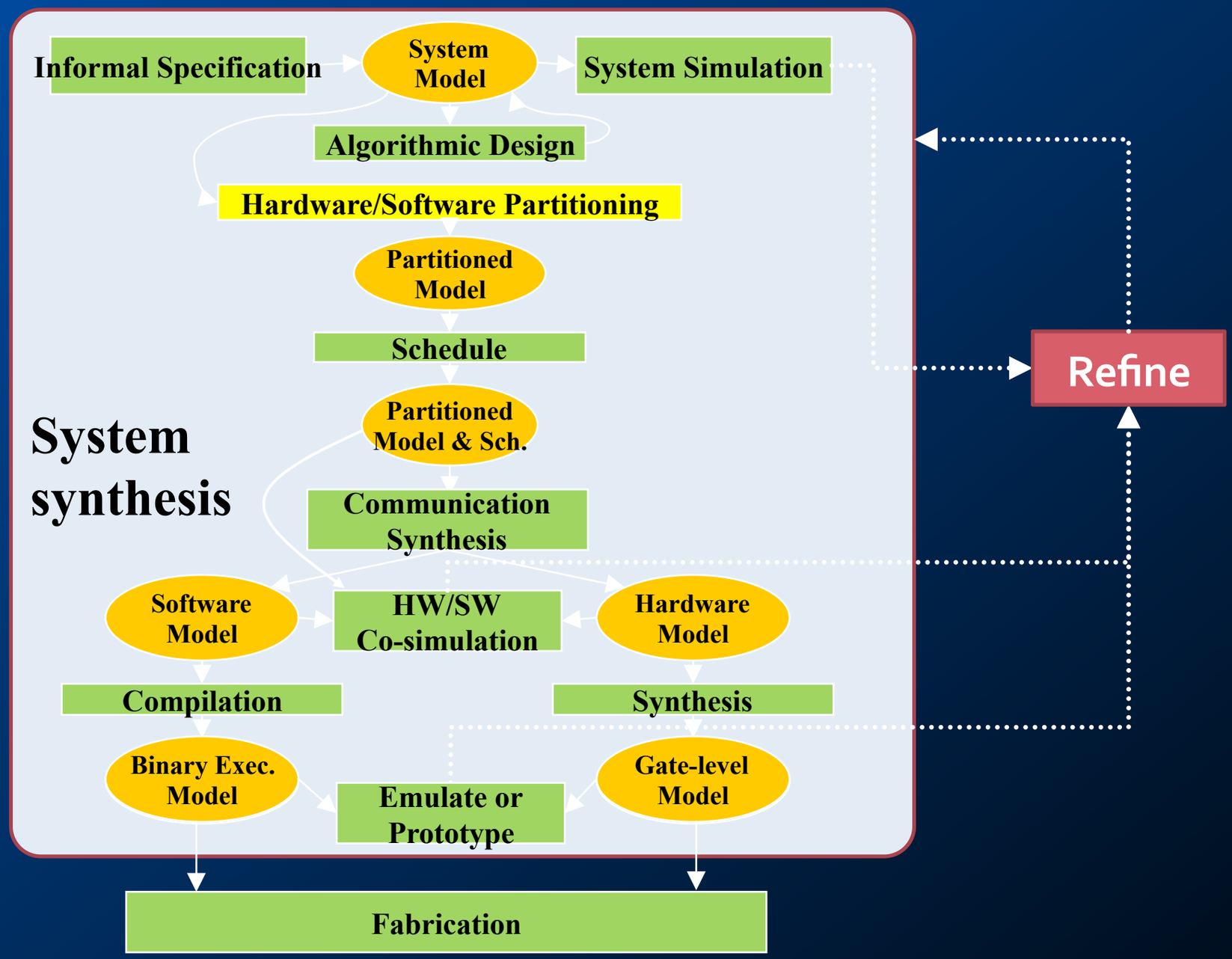
HW/SW Co-design

- HW/SW Co-design means the design of a special-purpose system composed of a few application-specific ICs that cooperate with software procedures on general-purpose processors (1994)
- HW/SW Co-design means meeting system-level objectives by exploiting the synergism of hardware and software through their concurrent design (1997)
- HW/SW Co-design tries to increase the predictability of embedded system design by providing analysis methods that tell designers if a system meets its performance, power, and size goals and synthesis methods that let designers rapidly evaluate many potential design methodologies (2003)
- It moved from an emerging discipline (early '90s) to a mainstream technology (today)

Simplified design flow: part in HW, part in SW

Decision based on hardware/software partitioning, a special case of hardware/software co-design.

HW/SW Co-design
- **Task concurrency management**
- **High-level transformations**
- **Design space exploration**
- **HW/SW partitioning**
- **Compilation, scheduling**


Informal Specification
System Model
System Simulation
Algorithmic Design
Hardware/Software Partitioning
Partitioned Model
Schedule
Partitioned Model & Sch.
Communication Synthesis
System synthesis
Software Model
HW/SW Co-simulation
Hardware Model
Compilation
Synthesis
Binary Exec. Model
Emulate or Prototype
Gate-level Model
Refine
Fabrication

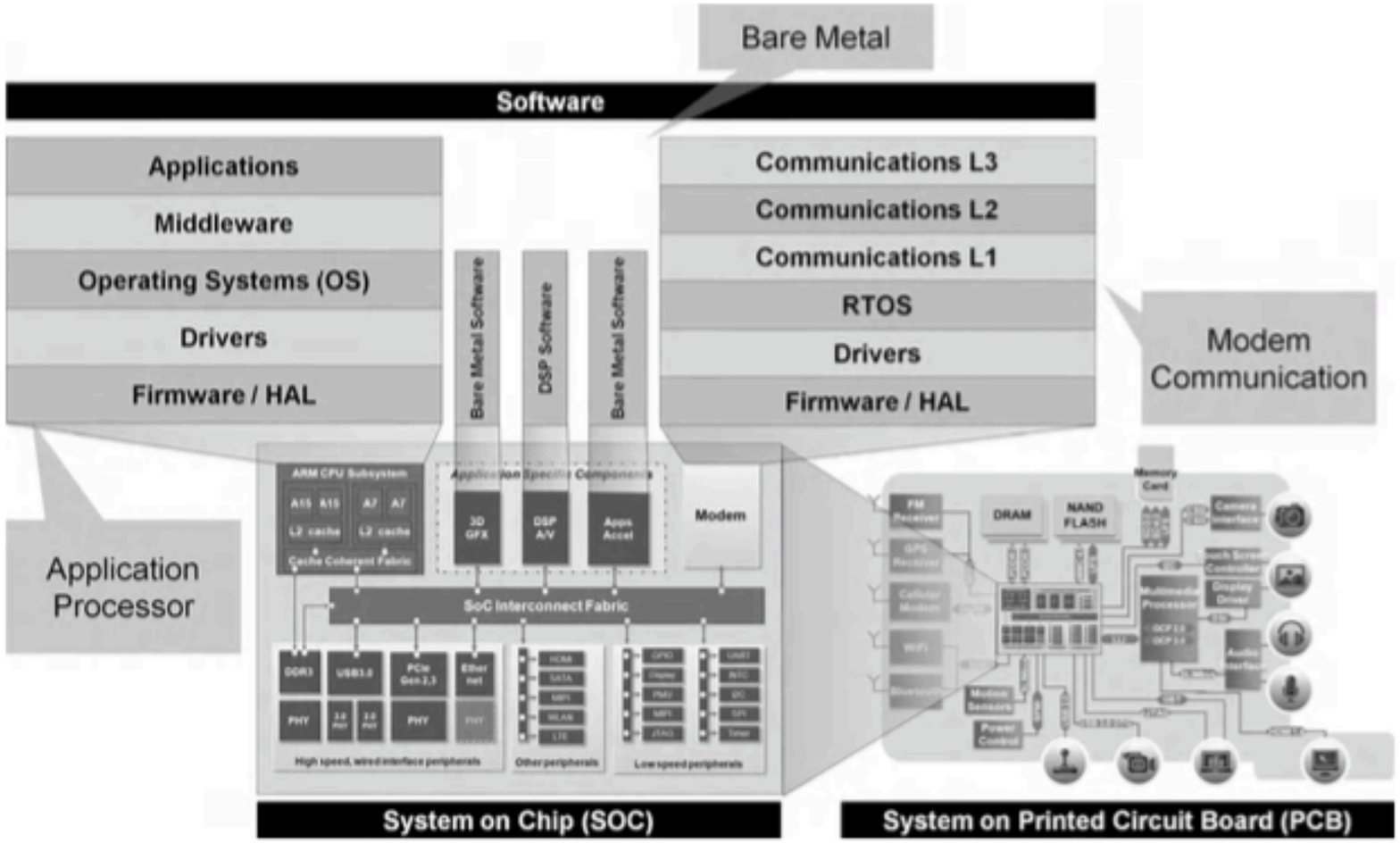


Figure 2.1:
An ARM® core-based system on chip in its system context.

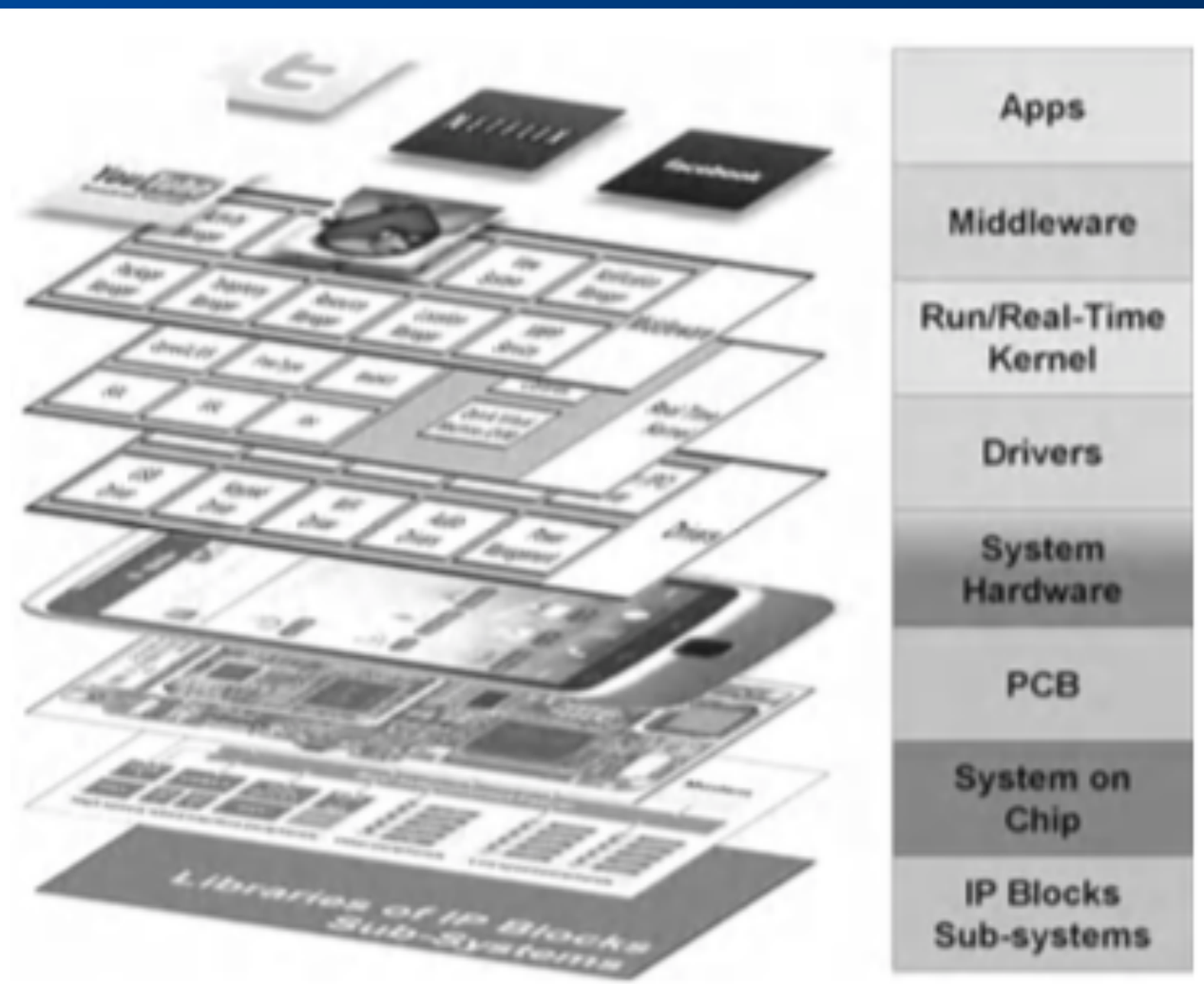


Figure 2.2:
Layers of hardware and software in an embedded system.

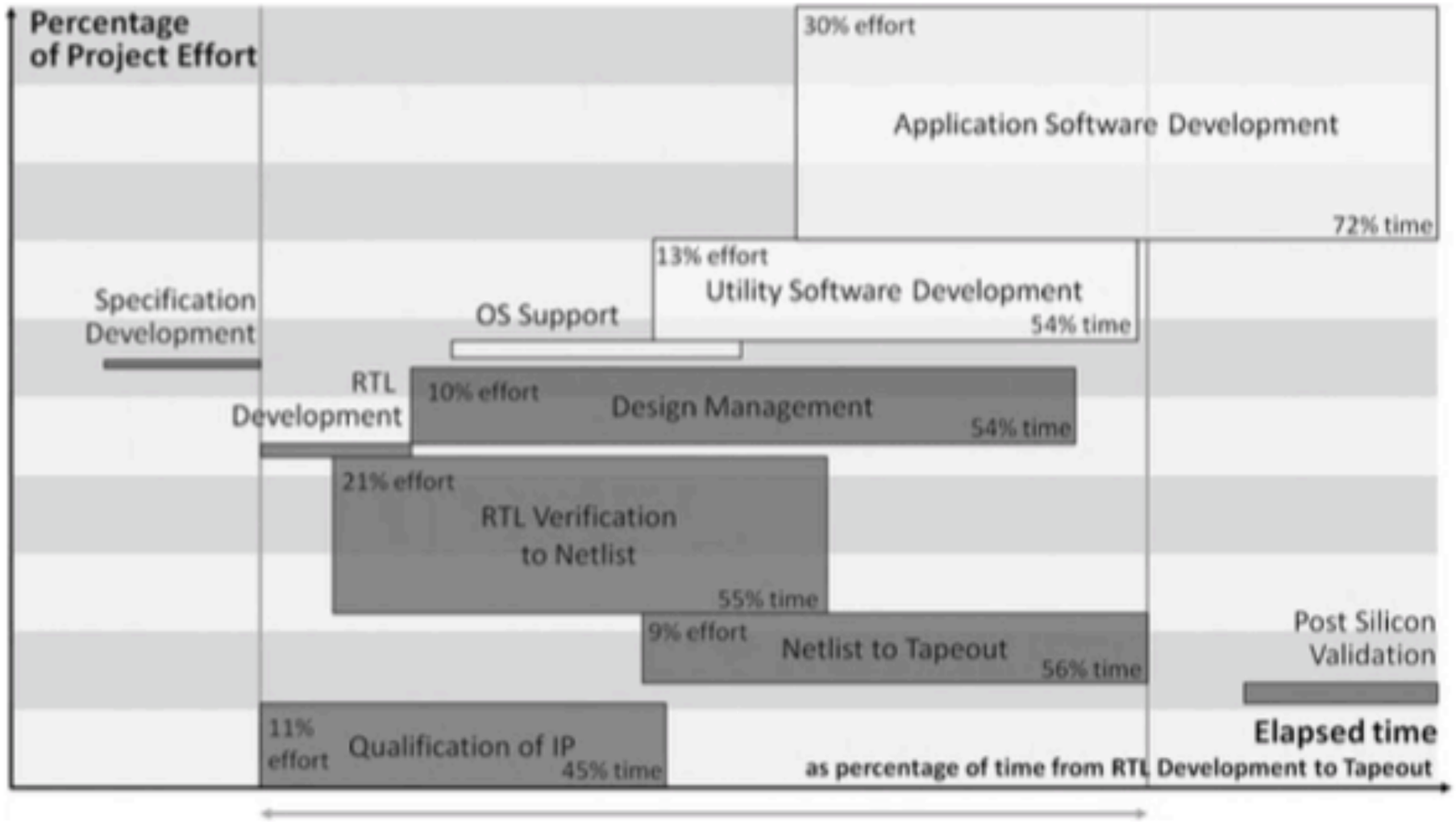


Figure 2.3:
Typical embedded system design flow and percentage of project effort.

Figure 2.4:
Prototyping options — there is no "one fits all".

	Virtual Prototyping	RTL Simulation	Acceleration Emulation	FPGA Based Prototyping	Silicon
Early Availability	++	+		-	--
Speed	++	--	-	+	++
HW Accuracy	--	++	++	++	++
HW Debug	--	++	+	-	--
SW Debug	++	--		+	++
Execution Control	++	++	+		--
Effort of Extra Development	--	++	+		++
Cost of Replication	++	++	--	-	+

Figure 2.5:
Prototyping user needs vs. prototyping capabilities.

Thank you!