Embedded Systems

Supplemental Notes

Yul Williams, D.Sc.

Goals

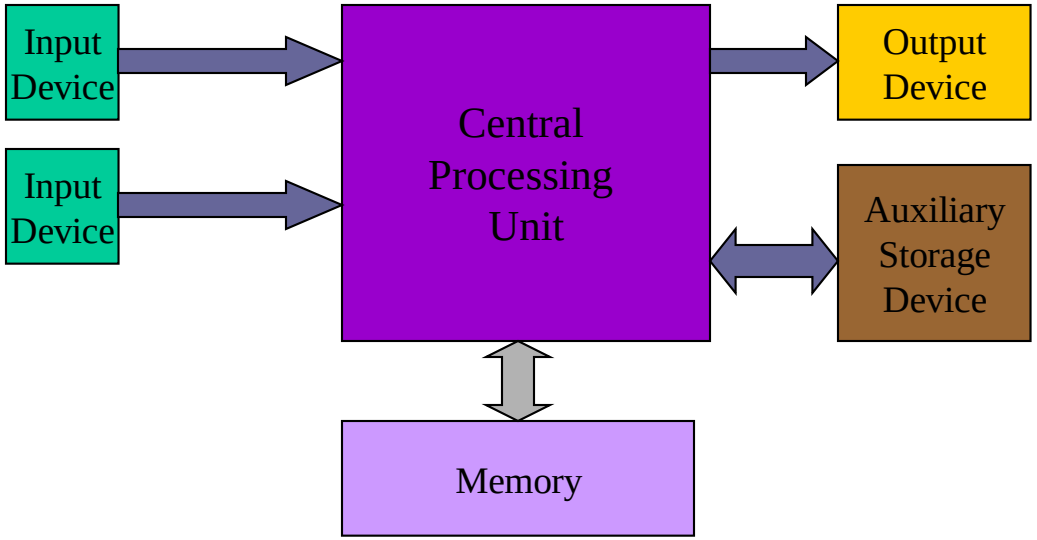
- Upon reviewing these supplemental notes, the student should be able to:
  - Recognize differences between general purpose computers and embedded computer systems
  - Identify modern embedded systems by their characteristics
  - Discuss key issues of embedded system design
  - Discuss the embedded systems development methodology

Outline

- The General-Purpose Computer
- The Embedded Computer System
- Modern Examples of Embedded Systems
- Embedded System Characteristics
- Embedded System Design
- Designing a GPS Moving Map System
- Summary

The General-Purpose Computer

A Computer

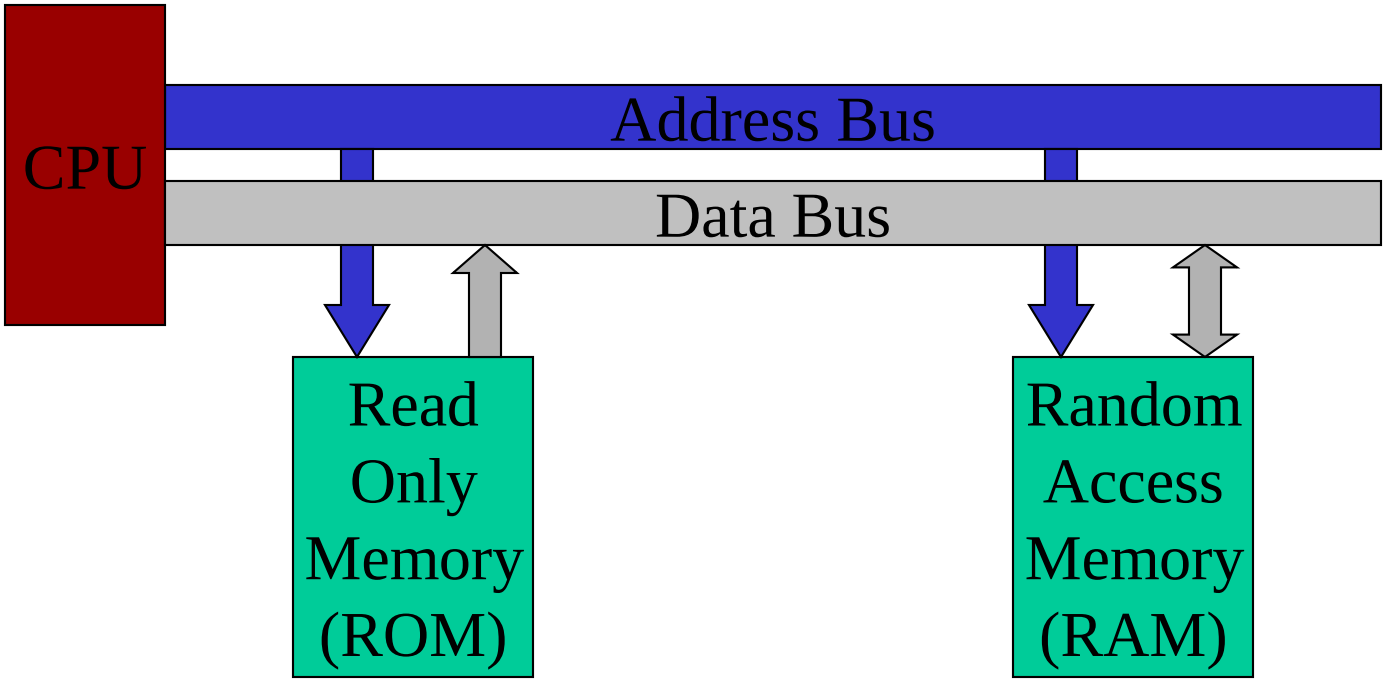


The computer is composed of input devices, a central processing unit, a memory unit and output devices.

Memory Unit

- An ordered sequence of storage cells, each capable of holding a piece of data.
- Volatile Memory
  - RAM – Random Access Memory
- Non-volatile Memory
  - ROM – Read Only Memory

Memories



Factory programmed memory
Used to store BIOS*
Non-volatile memory

User-modifiable memory
Used to store program and data
Volatile memory

*BIOS – Basic Input/Output System

Central Processing Unit (CPU)

- The CPU has two components:
  - Arithmetic and Logic Unit (ALU)
    - Performs arithmetic operations
    - Performs logical operations
  - Control Unit
    - Controls the action of the other computer components so that instructions are executed in the correct sequence

Input/Output (I/O) Devices

- I/O devices are the components of a computer system that accepts data to be processed and presents the results of the processing
- Input Device Examples
  - Keyboard
  - Mouse
- Output Device Examples
  - Video display
  - Printer

Auxiliary Storage Devices

- An auxiliary storage device is sometimes referred to as a secondary storage device.
- Hard disk drives, floppy disk drives, zip drives, and tape drives are examples of auxiliary storage devices

Peripheral Devices

- Computers may be accessorized with attached devices that offer some value-added features
- Scanners, CD-ROM, DVD, and digital cameras are examples of computer peripherals

Interactive and Batch Systems

- An interactive system facilitates the direct communication between the computer and the computer user.
- A batch system requires that all data be entered prior to program execution and the results are viewable only after the program has completed executing

The Operating System

- The Operating System (OS) is a set of programs that manages all of the computer's resources
- Unix, Linux, Windows 98, Me, NT, 2000, XP, and MacOS are all examples of modern operating systems

The Embedded Computer System

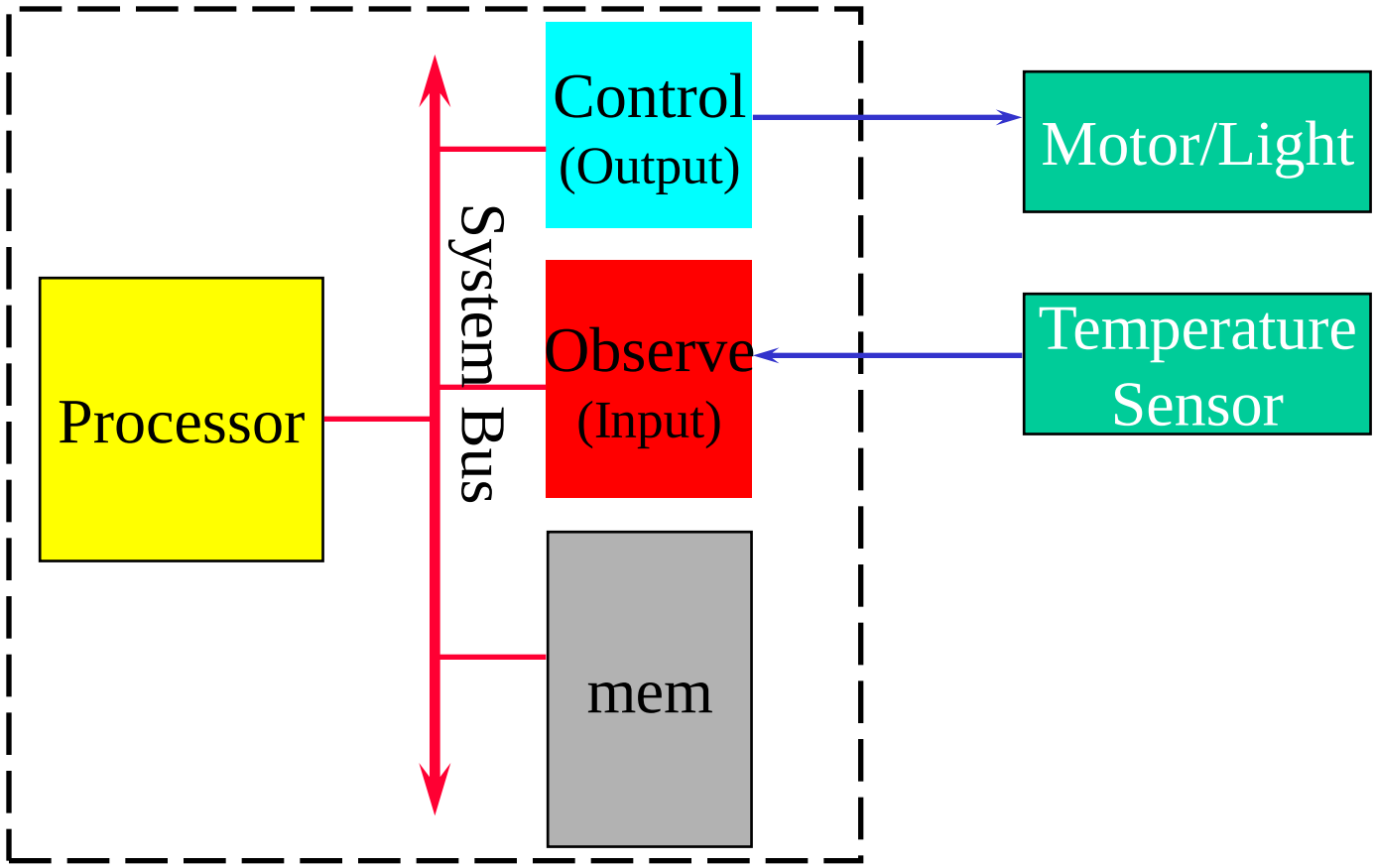
Definition:Embedded System

- “Any sort of device which includes a programmable computer but itself is not intended to be a general-purpose computer”
  - Wayne Wolf

Interfacing with the Real World

- The reason for designing and building embedded systems is to observe or control something in the “real world” (i.e. analog)
  - Determine the internal temperature of an oven
  - Control the motor speed of a robot
- Embedded systems must be able to translate between the digital and analog worlds in order to be effective

Embedded System Composition



Modern Examples of Embedded Systems

Modern Embedded Systems

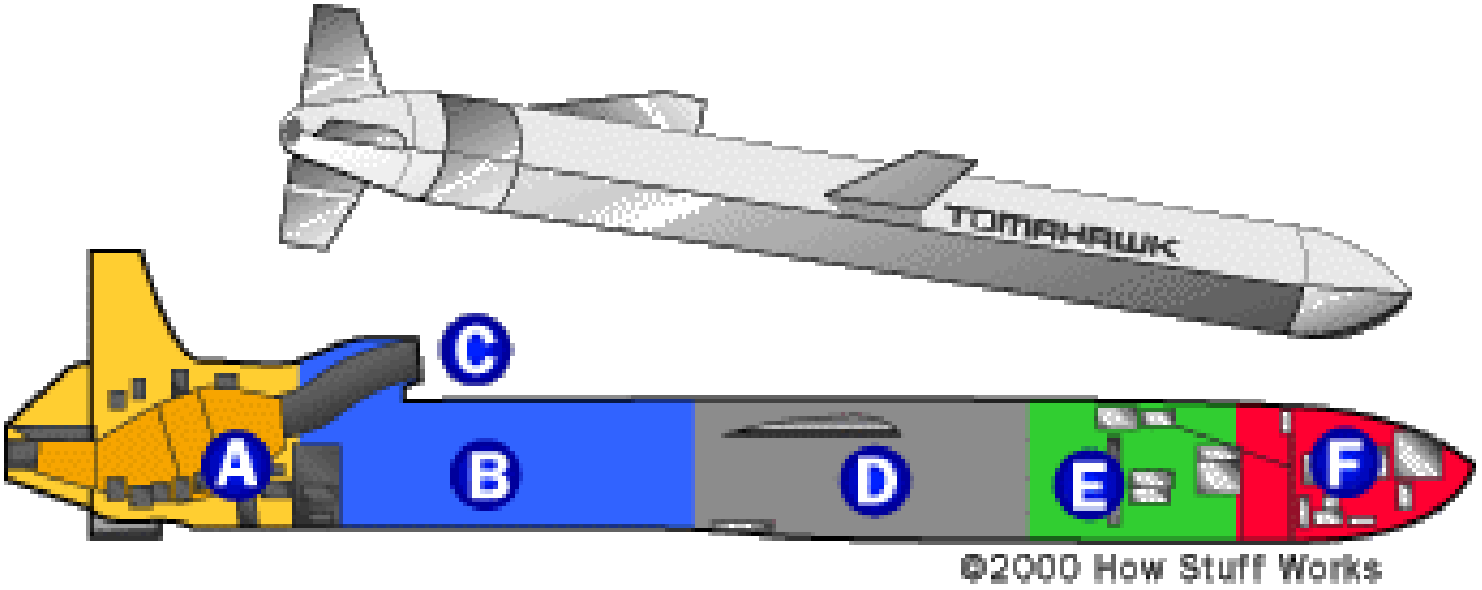
- Personal digital assistant (PDA).
- Consumer electronics (e.g. digital cameras and household appliances)
- Mobile phone
- Automobile engines, fuel control, etc.
- Grocery store check-out scanners
- Global Positioning System (GPS) units

Wireless Communications

Hand-held GPS Units

Telematics System for Automobiles

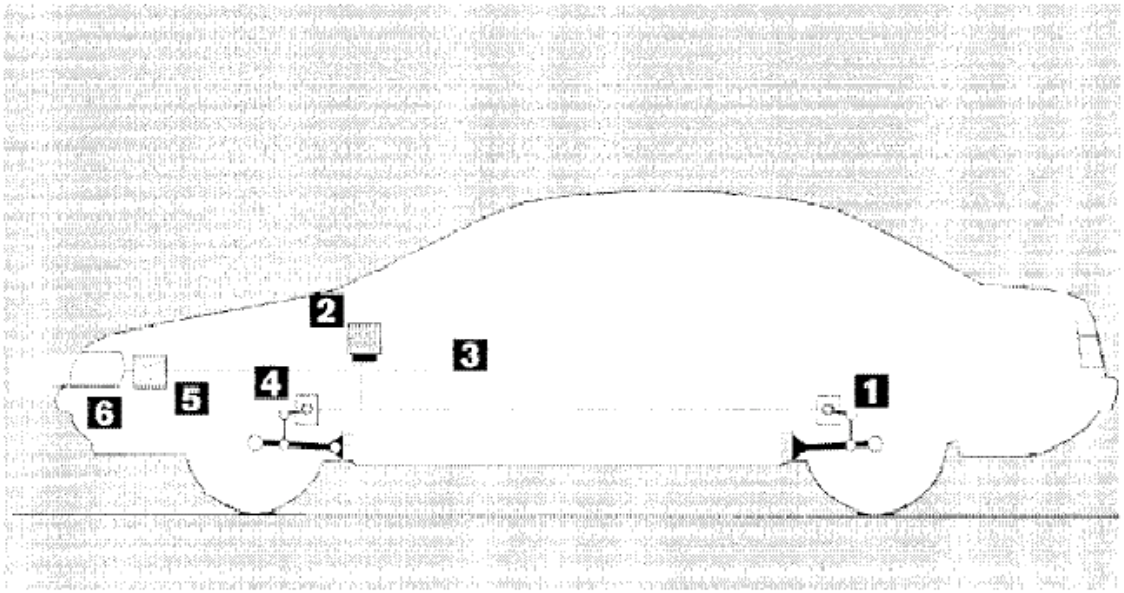
Cruise Missile Guidance



Robotics Control

Spider robot – constructed with LEGO Mindstorms Components

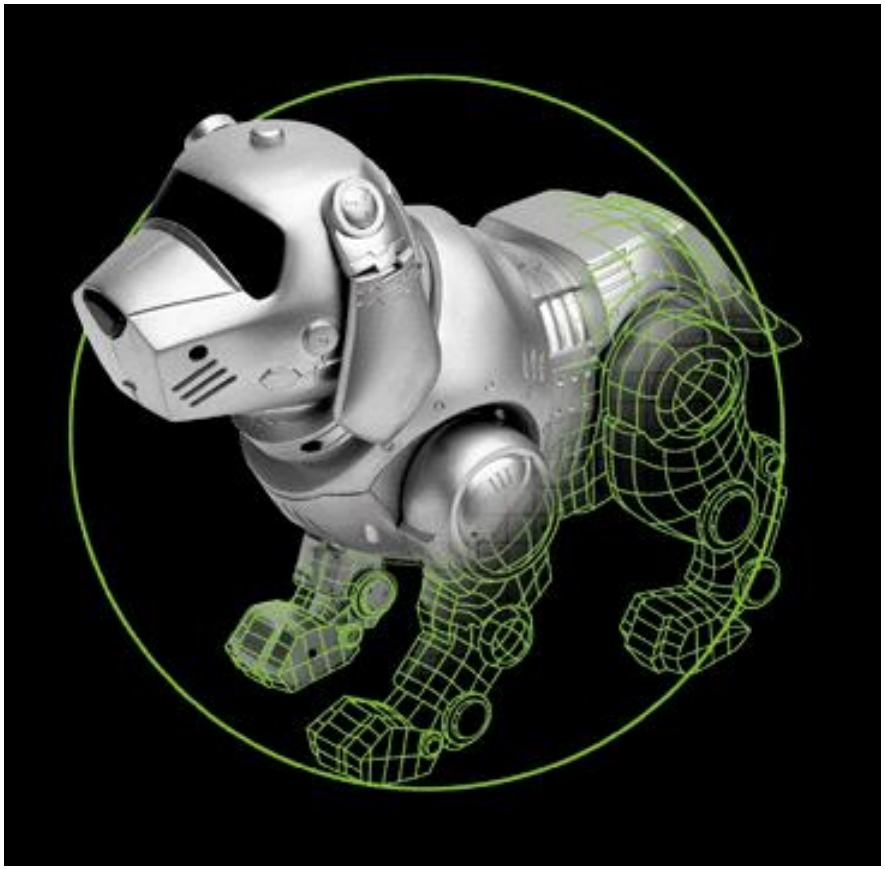
Automotive Control

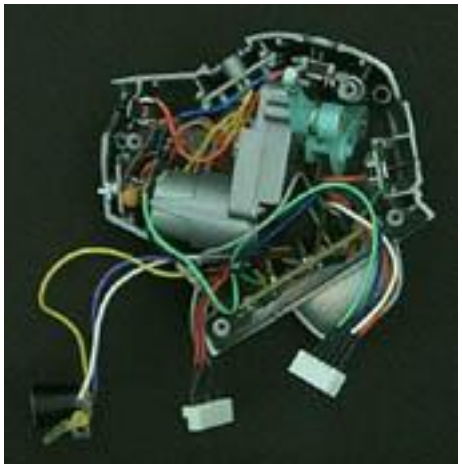


Car with an automatic headlight leveling system. 1: Rear distance Sensor, 2: Control unit, 3: Speed signal, 4: Front distance sensor, 5: Motor, 6: Lamps.

Source:Gmehlich, R., Specification and Validation of Embedded Systems using LUSTRE and ARGOS. Case Study:The Automatic Headlight Leveling System, Design Automation for Embedded Systems, 6, 151–175 (2001)

Smart Toys

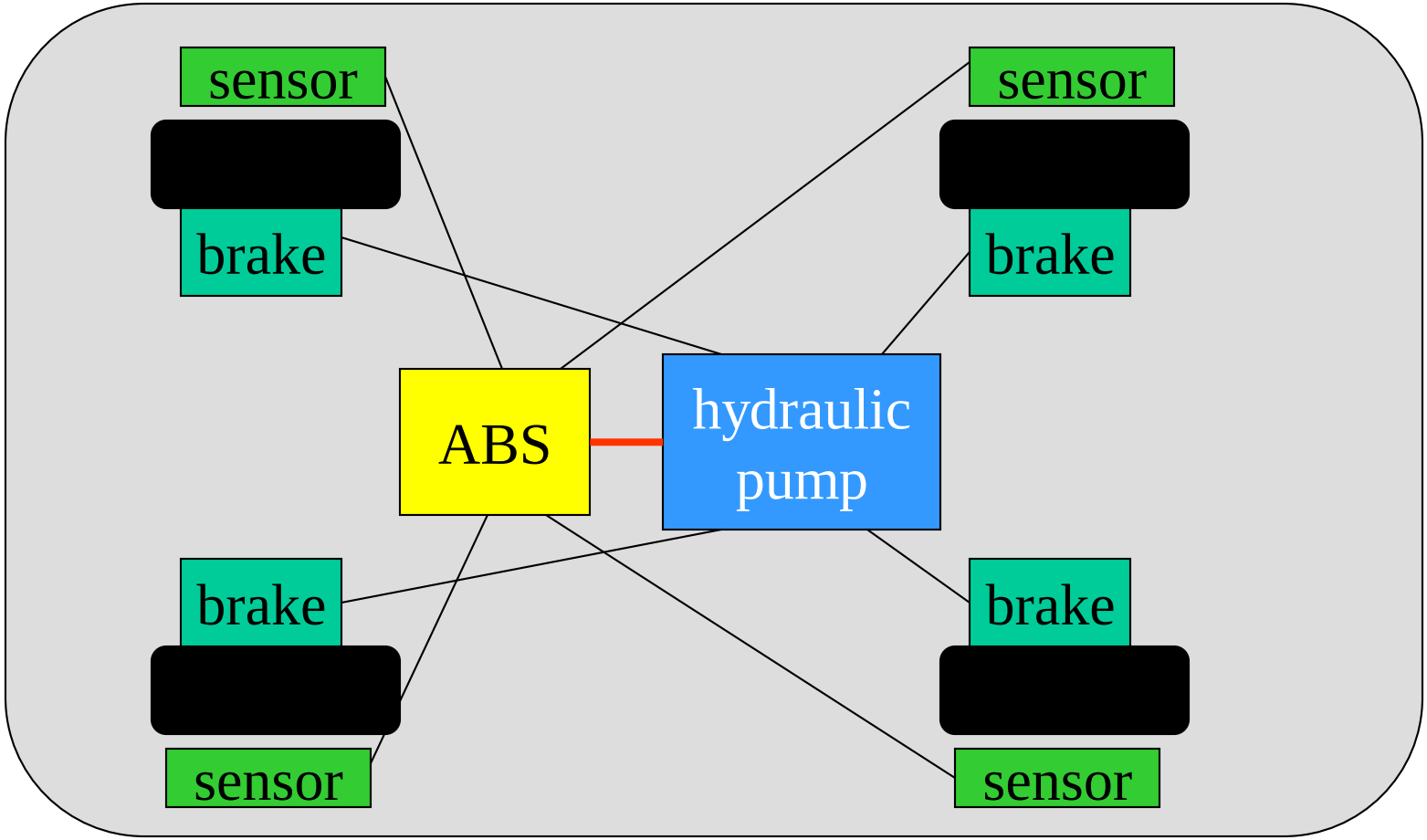




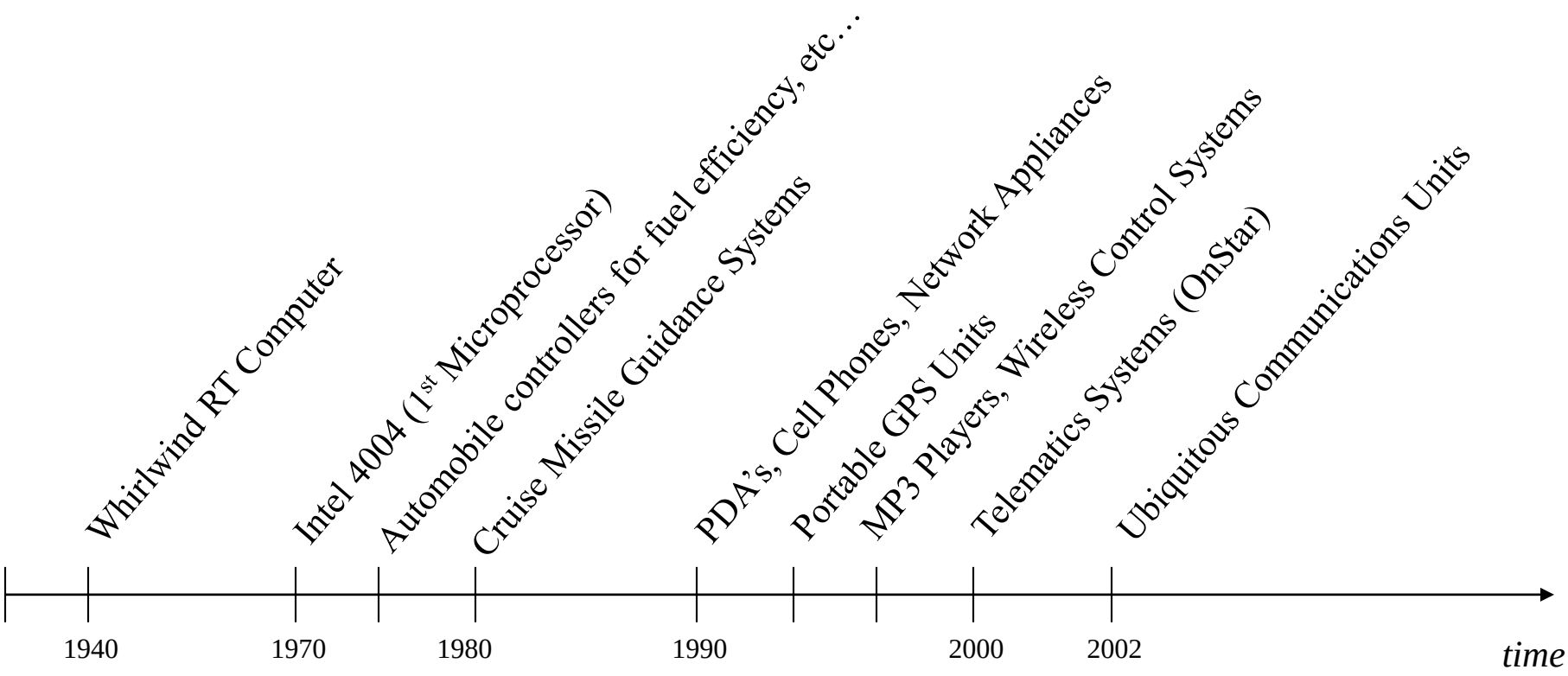
BMW 850i brake and stability control system

- Anti-lock brake system (ABS): pumps brakes to reduce skidding.
- Automatic stability control (ASC+T): controls engine to improve stability.
- ABS and ASC+T communicate.
  - ABS was introduced first---needed to interface to existing ABS module.

BMW 850i, cont'd.



Embedded Systems Evolution



Embedded System Characteristics

Characteristics of Embedded Systems

- Application-specific functionality – specialized for one or one class of applications
- Deadline constrained operation – system may have to perform its function(s) within specific time periods to achieve successful results
- Resource challenged – systems typically are configured with a modest set of resources to meet the performance objectives
- Power efficient – many systems are battery-powered and must conserve power to maximize the usable life of the system.
- Form factor – many systems are light weight and low volume to be used as components in host systems
- Manufacturable – usually small and inexpensive to manufacture based on the size and low complexity of the hardware.

Functional complexity

- Often have to run sophisticated algorithms or multiple algorithms.
  - Cell phone, laser printer.
- Often provide sophisticated user interfaces.

Real-time operation

- Must finish operations by deadlines.
  - Hard real time: missing deadline causes failure.
  - Soft real time: missing deadline results in degraded performance.
- Many systems are multi-rate: must handle operations at widely varying rates.

Non-functional requirements

- Many embedded systems are mass-market items that must have low manufacturing costs.
  - Limited memory, microprocessor power, etc.
- Power consumption is critical in battery-powered devices.
  - Excessive power consumption increases system cost even in wall-powered devices.

Design teams

- Often designed by a small team of designers.
- Often must meet tight deadlines.
  - 6 month market window is common.
  - Can’t miss back-to-school window for calculator.

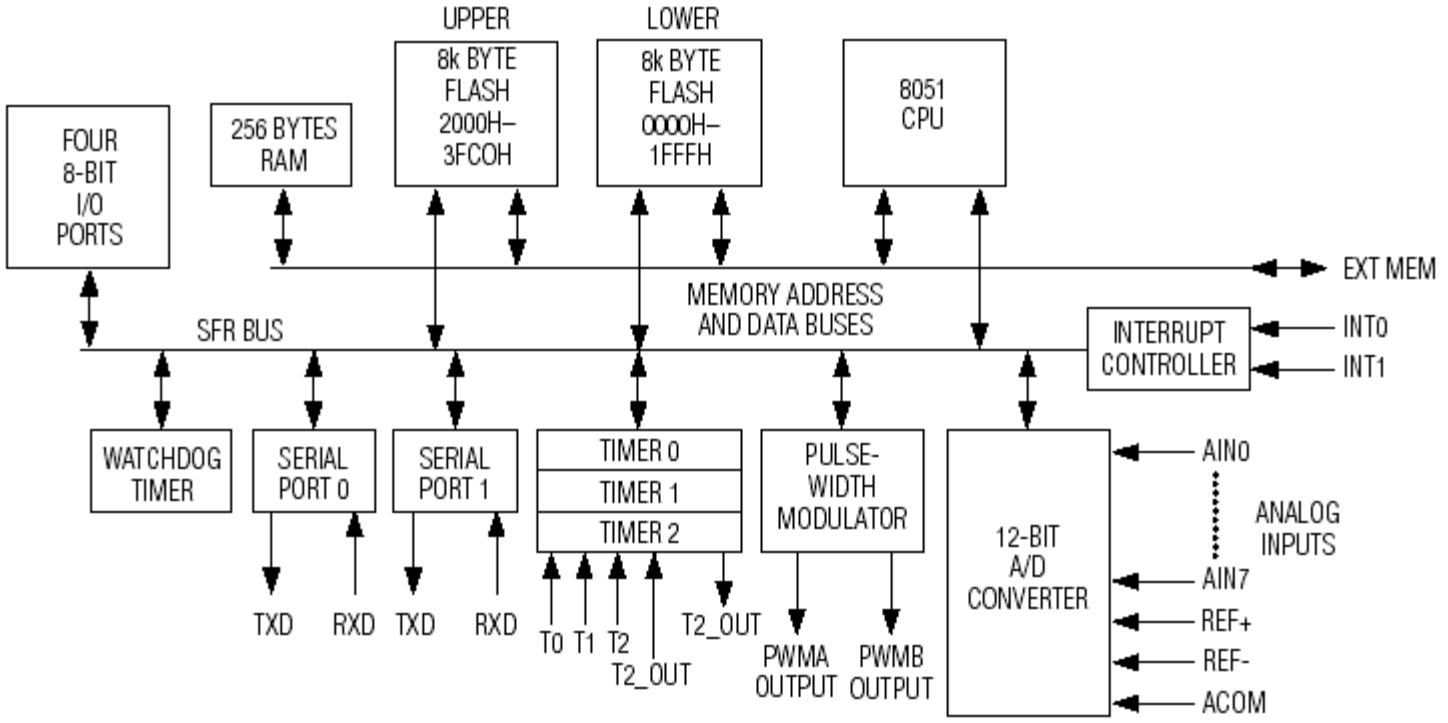
Why use microprocessors?

- Alternatives: field-programmable gate arrays (FPGAs), custom logic, etc.
- Microprocessors are often very efficient: can use same logic to perform many different functions.
- Microprocessors simplify the design of families of products.

Microprocessor varieties

- Microcontroller: includes I/O devices, on-board memory.
- Digital signal processor (DSP): microprocessor optimized for digital signal processing.
- Typical embedded word sizes: 8-bit, 16-bit, 32-bit.

Microcontroller



The performance paradox

- Microprocessors use much more logic to implement a function than does custom logic.
- But microprocessors are often at least as fast:
  - heavily pipelined;
  - large design teams;
  - aggressive VLSI technology.

Power

- Custom logic is a clear winner for low power devices.
- Modern microprocessors offer features to help control power consumption.
- Software design techniques can help reduce power consumption.

Embedded System Design

Challenges in embedded system design

- How much hardware do we need?
  - How big is the CPU? Memory?
- How do we meet our deadlines?
  - Faster hardware or cleverer software?
- How do we minimize power?
  - Turn off unnecessary logic? Reduce memory accesses?

Challenges, etc.

- Does it really work?
  - Is the specification correct?
  - Does the implementation meet the spec?
  - How do we test for real-time characteristics?
  - How do we test on real data?
- How do we work on the system?
  - Observability, controllability?
  - What is our development platform?

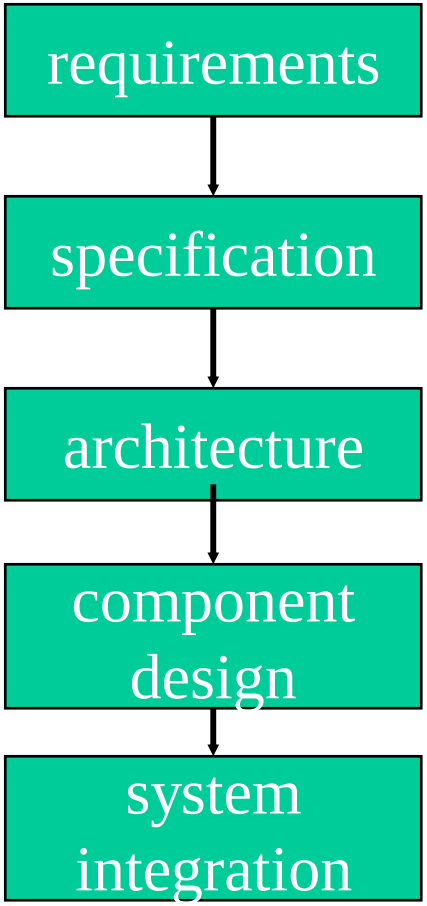
Design methodologies

- A procedure for designing a system.
- Understanding your methodology helps you ensure you didn’t skip anything.
- Compilers, software engineering tools, computer-aided design (CAD) tools, etc., can be used to:
  - help automate methodology steps;
  - keep track of the methodology itself.

Design goals

- Performance.
  - Overall speed, deadlines.
- Functionality and user interface.
- Manufacturing cost.
- Power consumption.
- Other requirements (physical size, etc.)

Levels of abstraction



Top-down vs. bottom-up

- Top-down design:
  - start from most abstract description;
  - work to most detailed.
- Bottom-up design:
  - work from small components to big system.
- Real design uses both techniques.

Stepwise refinement

- At each level of abstraction, we must:
  - analyze the design to determine characteristics of the current state of the design;
  - refine the design to add detail.

Requirements

- Plain language description of what the user wants and expects to get.
- May be developed in several ways:
  - talking directly to customers;
  - talking to marketing representatives;
  - providing prototypes to users for comment.

Functional vs. non-functional requirements

- Functional requirements:
  - output as a function of input.
- Non-functional requirements:
  - time required to compute output;
  - size, weight, etc.;
  - power consumption;
  - reliability;
  - etc.

Our requirements form

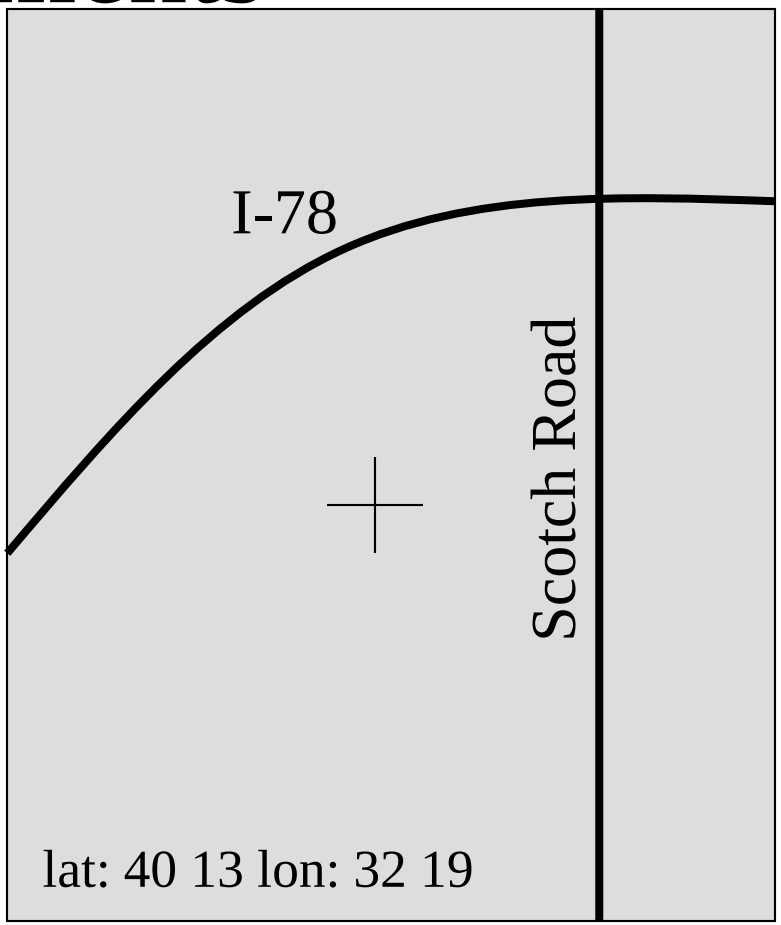
name
purpose
inputs
outputs
functions
performance
manufacturing cost
power
physical size/weight

Designing a GPS Moving Map System

A case study

Example: GPS moving map requirements

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



GPS moving map needs

- Functionality: For automotive use. Show major roads and landmarks.
- User interface: At least 400 x 600 pixel screen. Three buttons max. Pop-up menu.
- Performance: Map should scroll smoothly. No more than 1 sec power-up. Lock onto GPS within 15 seconds.
- Cost: $500 street price = approx. $100 cost of goods sold.

GPS moving map needs, cont'd.

- Physical size/weight: Should fit in hand.
- Power consumption: Should run for 8 hours on four AA batteries.

GPS moving map requirements form

name	GPS moving map
purpose	consumer-grade moving map for driving
inputs	power button, two control buttons
outputs	back-lit LCD 400 X 600
functions	5-receiver GPS; three resolutions; displays current lat/lon
performance	updates screen within 0.25 sec of movement
manufacturing cost	$100 cost-of-goods-sold
power	100 mW
physical size/weight	no more than 2: X 6:, 12 oz.

Specification

- A more precise description of the system:
  - should not imply a particular architecture;
  - provides input to the architecture design process.
- May include functional and non-functional elements.
- May be executable or may be in mathematical form for proofs.

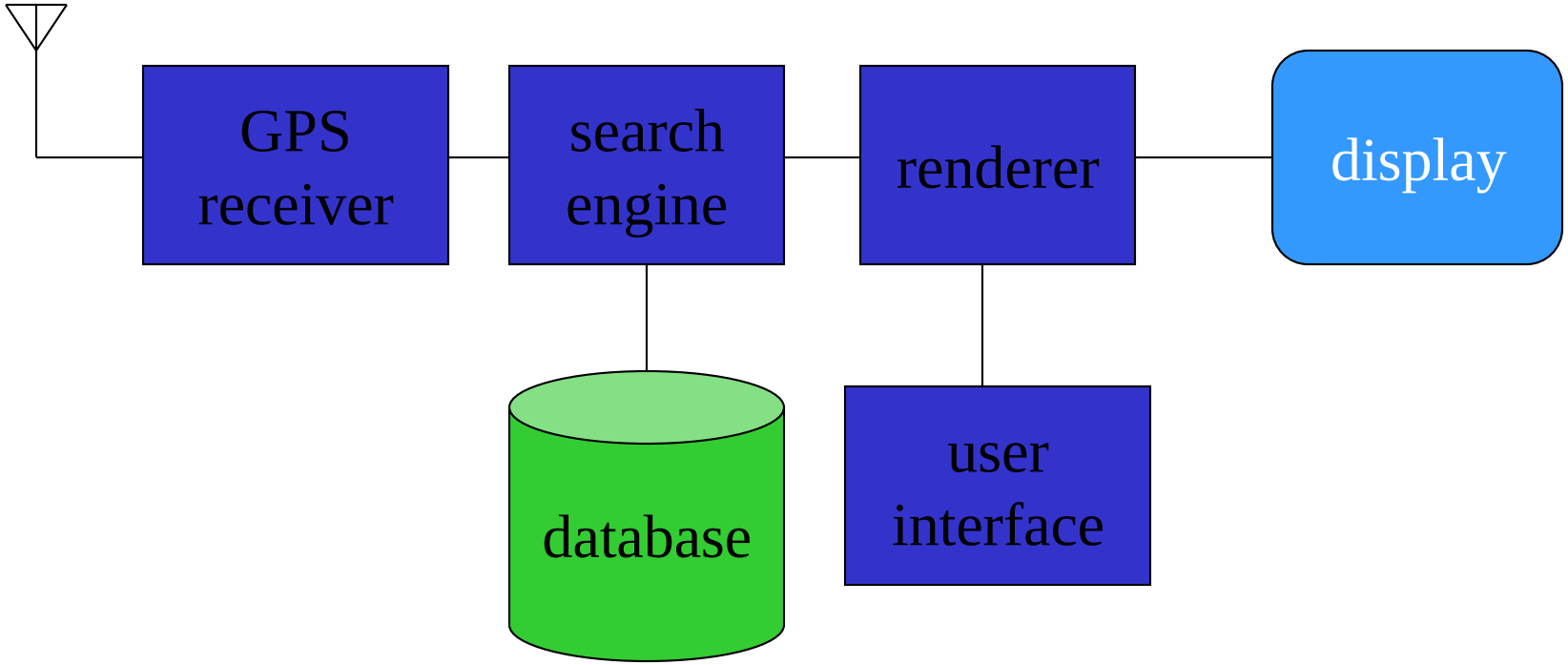
GPS specification

- Should include:
  - What is received from GPS;
  - map data;
  - user interface;
  - operations required to satisfy user requests;
  - background operations needed to keep the system running.

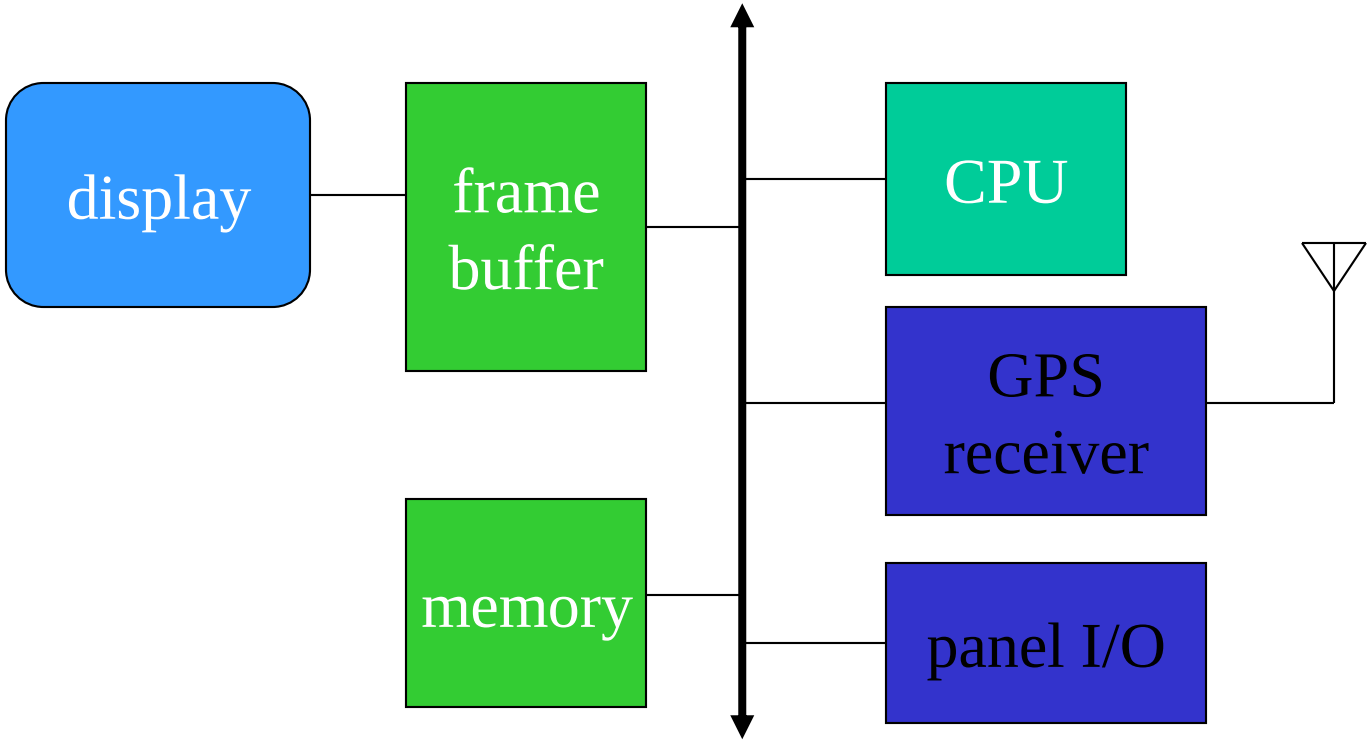
Architecture design

- 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.

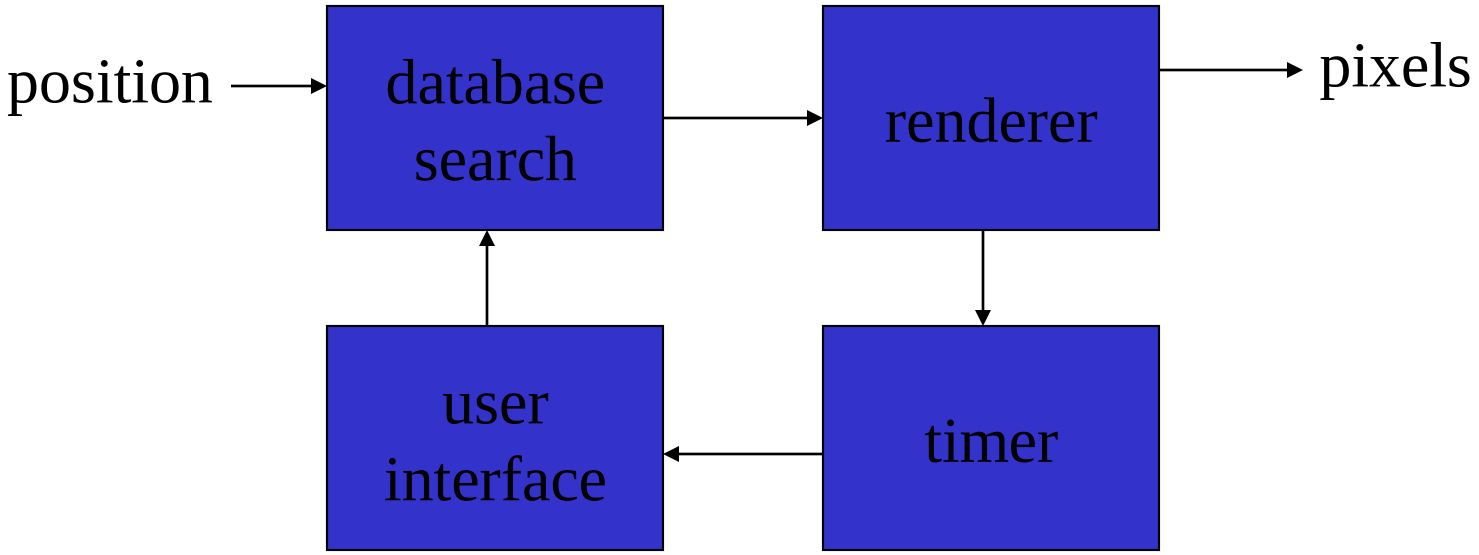
GPS moving map block diagram



GPS moving map hardware architecture



GPS moving map software architecture



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.

System integration

- Put together the components.
  - Many bugs appear only at this stage.
- Have a plan for integrating components to uncover bugs quickly, test as much functionality as early as possible.

Summary

- Embedded computers are all around us.
  - Many systems have complex embedded hardware and software.
- Embedded systems pose many design challenges: design time, deadlines, power, etc.
- Design methodologies help us manage the design process.

Acknowledgements

- Much of the presentation material was provided by Morgan Kaufmann Publishers and is copyrighted.
- Additional material was developed by the course instructor.