Lecture 2: Introduction to Embedded Computing

Embedded Computing Systems

Mikko Lipasti, adapted from M. Schulte

Based on slides and textbook from Wayne Wolf

Topics

- Embedded Computing Landscape.
- Aspects of Embedded System Design.
  - Architectures
  - Applications
  - Methodologies
  - Modeling
- Embedded System Applications.

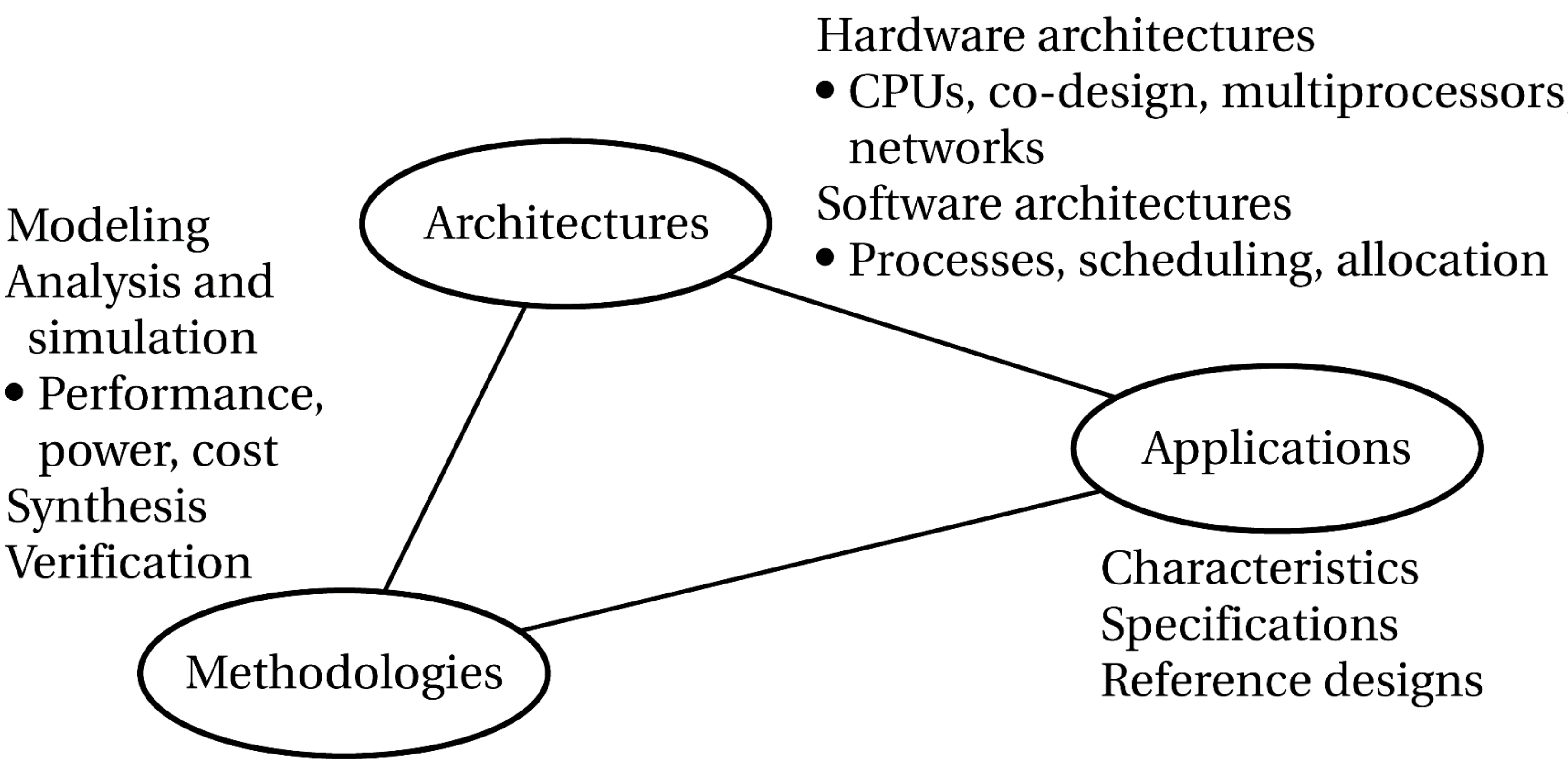
The landscape of embedded computing

- Lots of embedded applications require very high performance:
  - Communications.
  - Multimedia.
  - Graphics.
- Must also meet strict design goals:
  - Real-time performance.
  - Power/energy consumption.
  - Cost.
- How is power different than energy?
- How do embedded system design constraints differ from general purpose system design constraints?

Designing embedded systems

- No one architecture (hardware or software) can meet the needs of all applications.
- We need to be able to design a system for our target application or application domain:
  - Quickly and efficiently.
  - With reliable results.

Aspects of embedded system design



Architectures

- Both hardware and software architectures are important.
- The structure of the system determines cost, power, and performance.
- Different application requirements lead us to different architectures.

Applications

- You can't design the best embedded systems if you don't know anything about your application.
- You can't be an expert in everything.
  - But a little knowledge goes a long way.
- Domain expertise helps you make trade-offs:
  - Can the requirements be relaxed?
  - Can one requirement be traded for another?
  - How can the system be optimized?

Methodologies

- We must be able to reliably design systems:
  - Start from requirements/specification.
  - Build a system that is fast enough, doesn’t burn too much energy, and is cheap enough.
  - Be able to finish it on time.
  - And know before we start how difficult the project will be.
- Invention lets us get around some key technical barriers.
- Methodology keeps us going.

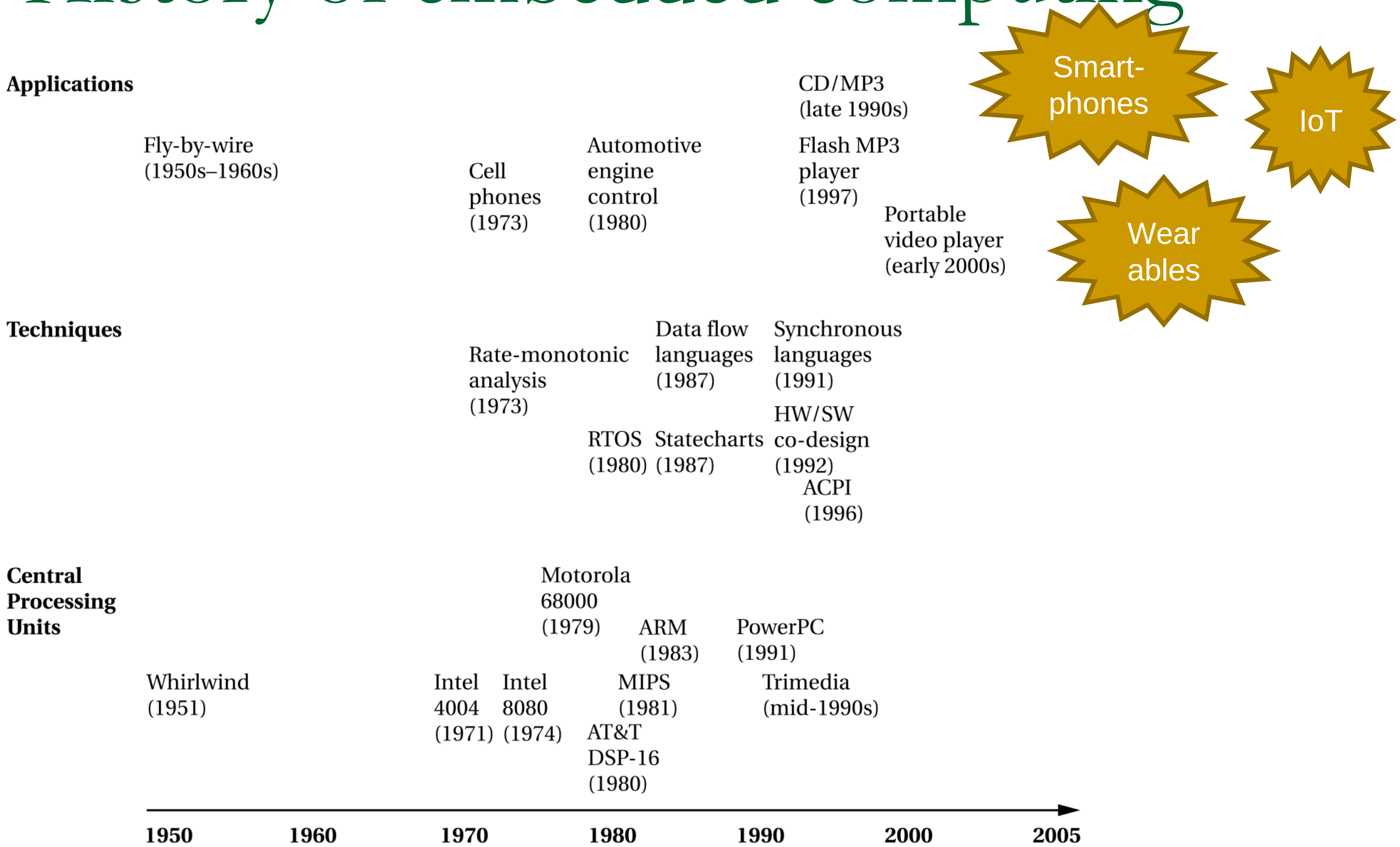
Modeling

- A key aspect of methodology is modeling.
  - Work with a simplified version of the object.
- Modeling helps us predict the consequences of design decisions.
- Models help us work faster (once we have the model).
- We can afford to use models if we can reuse them in several designs---methodology relies on and enables modeling.
- Pitfall: models can mislead

Disciplines in embedded computing

- Core areas:
  - Real-time computing : ECE 750
  - Hardware/software co-design.
- Closely related areas:
  - Computer architecture: ECE 552, 752, 757
  - Software engineering.
  - Low-power design.
  - Operating systems.
  - Programming languages and compilers.
  - Networking.
  - Secure and reliable computing.

History of embedded computing

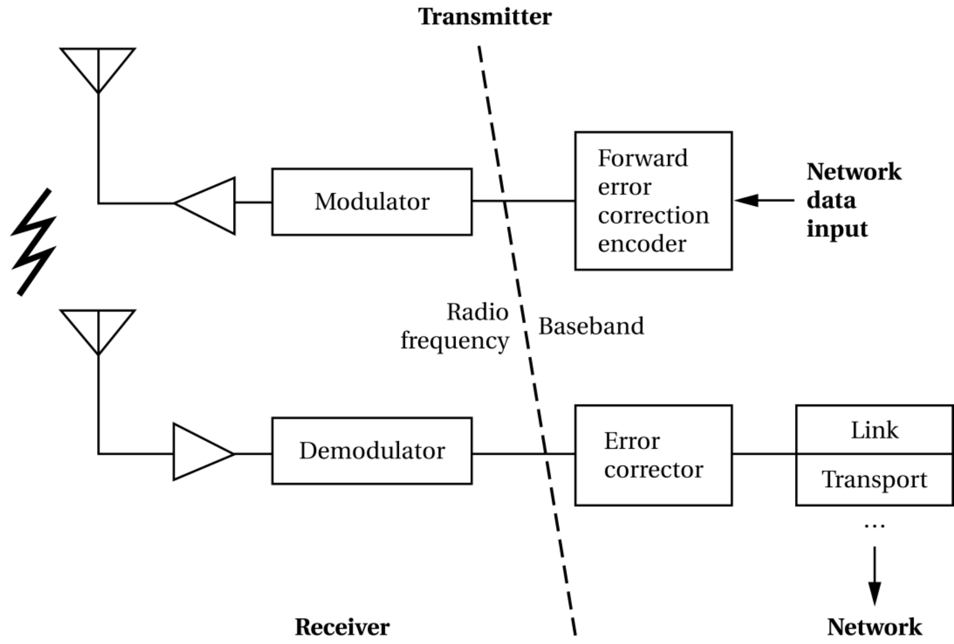


Sample Application domains

- Wireless communication systems
  - Radios and networking
- Multimedia
  - Image and video processing
- Real-time control
  - Automotive and avionic systems
- Smartphone SoCs

Radio and networking

- Modern radio systems carry digital information.
- Perform modulation/ demodulation and error correction.
- May also be closely tied to a networking stack.



Seven layers of the OSI network stack

1. Physical: Electrical, physical.
2. Data link: Access, error control across a single link.
3. Network: End-to-end service.
4. Transport: Connection-oriented service.
5. Session: Control activities.
6. Presentation: Data exchange formats.
7. Application: Interface to end use.

Networks and embedded systems

- An increasing number of embedded systems connect to the Internet (IoT)
  - Resource management.
  - Security.
- Many specialized networks have been developed for embedded systems:
  - Automotive.
  - Device control.
  - Personal area networks (Bluetooth LE)
  - IoT (Zigbee)

Radio and software radio

- Wireless receivers (radios) perform several basic functions:
  - Demodulate the signal.
  - Detection bits.
  - Correct errors.
- Software radio performs at least some of these functions using software on CPUs.
- Software defined radio (SDR) may be all software or a mix of HW and SW.

SDR Forum tiers of software-defined radio

0. Hardware radio, not programmable.
1. Software-controlled radio does not perform basic modulation/filtering in software.
2. Software-defined radio may cover a wide range of techniques and several modulation methods.
3. Ideal software-defined radio goes straight from A/D conversion to software.
4. Ultimate software radio is lightweight, low power/energy, requires no external antenna.

Radio operations

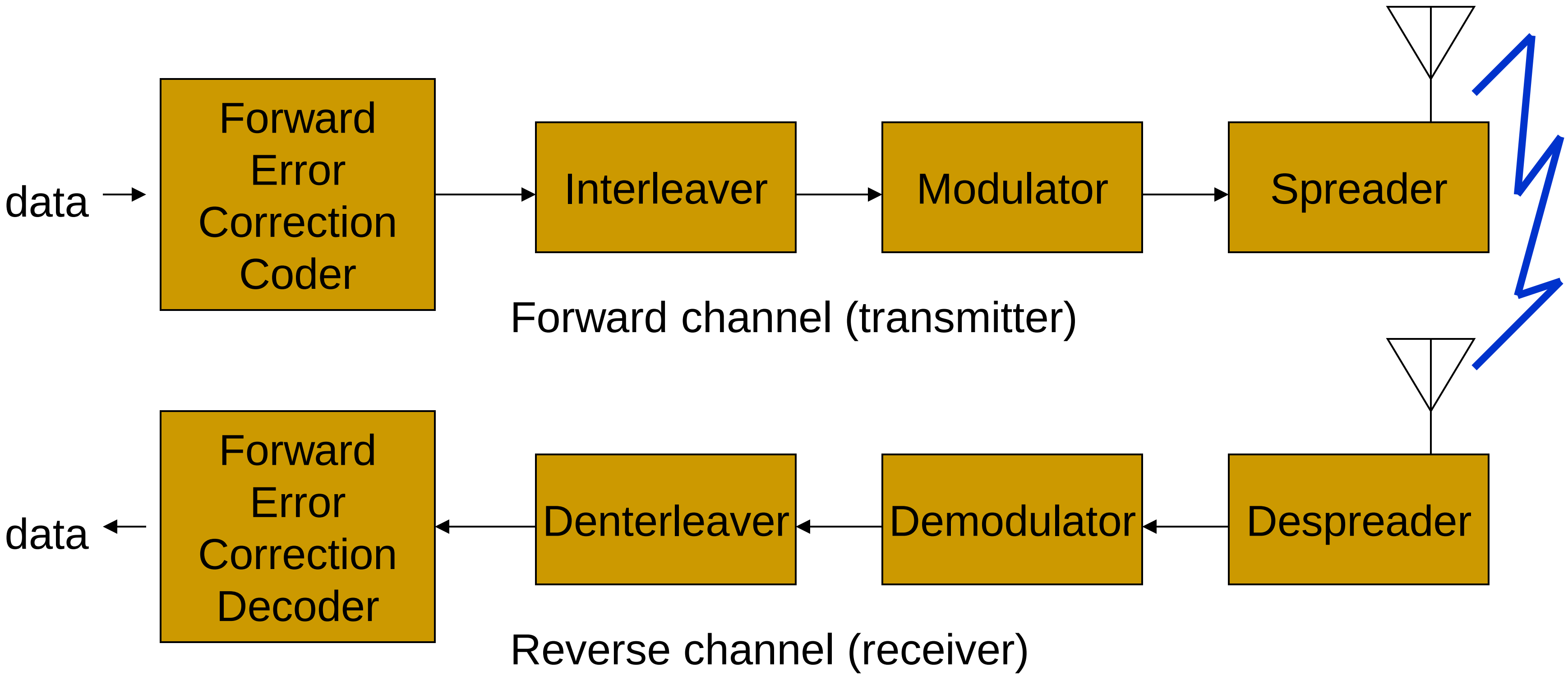
- Modulation:
  - Combinations of modulation variables (frequency, phase, amplitide) form symbols.
  - Symbols may be viewed as a constellation.
- Error correction:
  - Performed on raw bit stream to produce data payload.
  - Basic techniques like parity are often not powerful enough for noisy radio channels.
  - Viterbi method is widely used.
  - Example high-performance codes: turbo coding, low-density parity check (LDPC).

Radios and networks

- Radio may need to support an existing network (Internet, etc.).
- Radio may use its own network for coordination (cell phones).
- A data network may be designed to take advantage of the unique characteristics of radios (sensor networks).

Example: cdma2000

- Spread-spectrum for cell phones.
  - Uses direct-sequence spread spectrum.

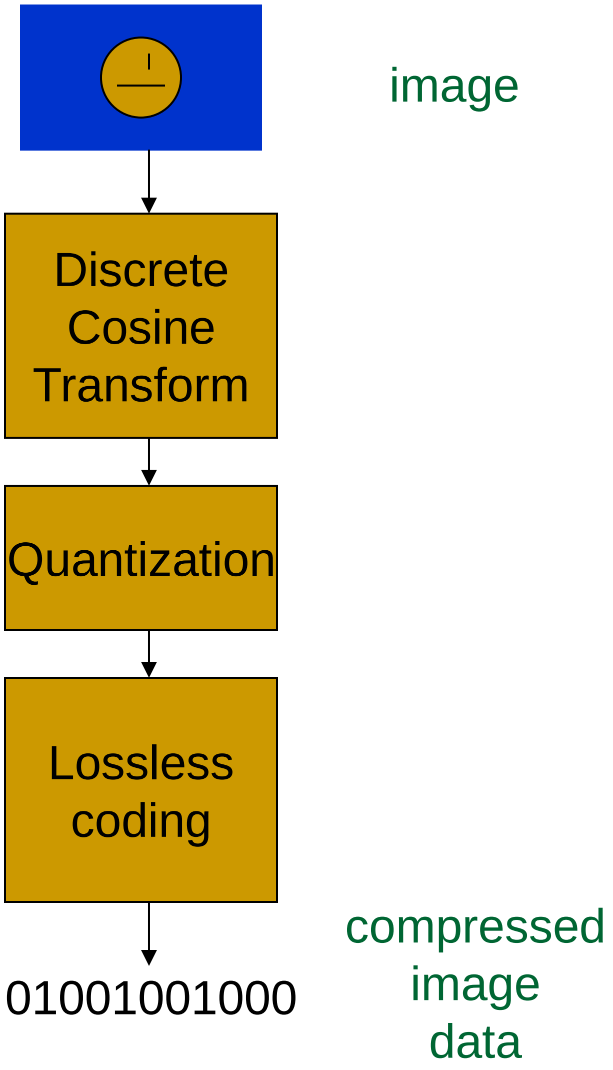


Multimedia

- Image compression: Each image is coded separately.
- Video compression: Takes advantage of correlation between successive frames.
- Perceptual coding: lossy coding, throws away information that will not be noticed.

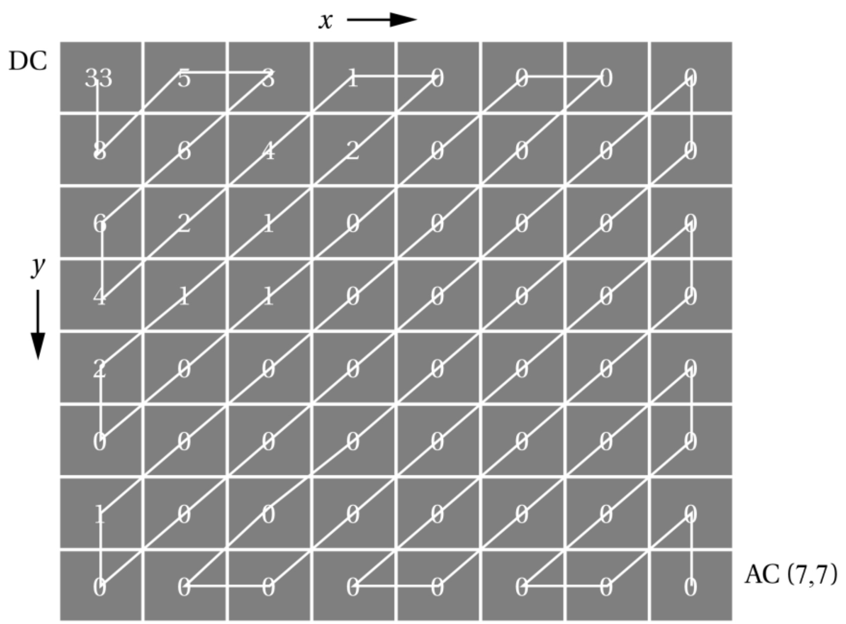
JPEG

- Discrete cosine transform (DCT) performed on 8 x 8 blocks typically, puts image into frequency domain.
- Quantization determines what image data to throw out.
- Lossless coding reduces the size of the representation.



JPEG zigzag pattern

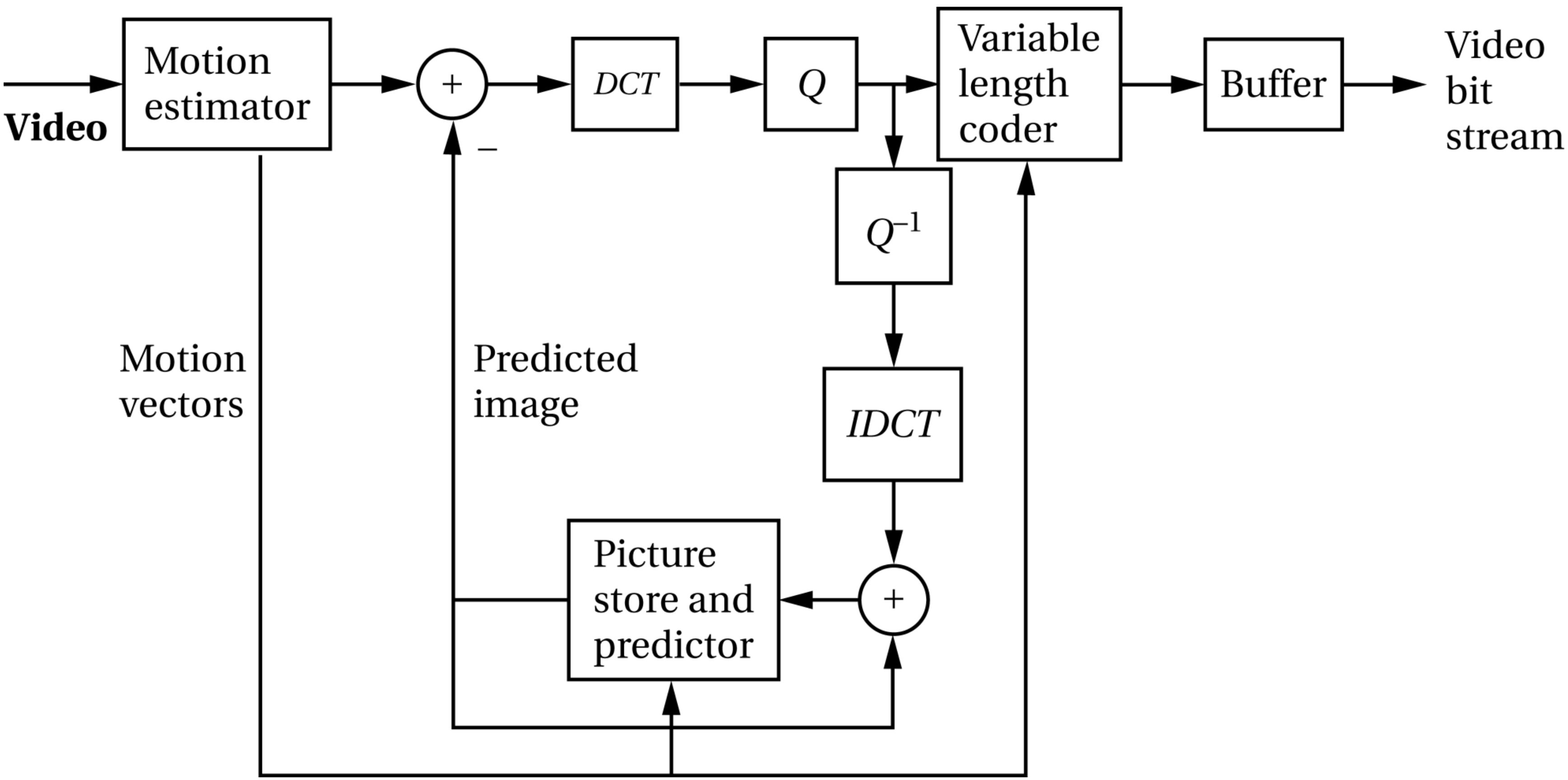
- After quantization, transform coefficients must be sent to lossless coder.
- Sending coefficients in zigzag pattern moves from low to high spatial frequencies.
- High frequency coefficients are more likely to be zero, producing strings that are easier to Huffman code.



Video compression standards

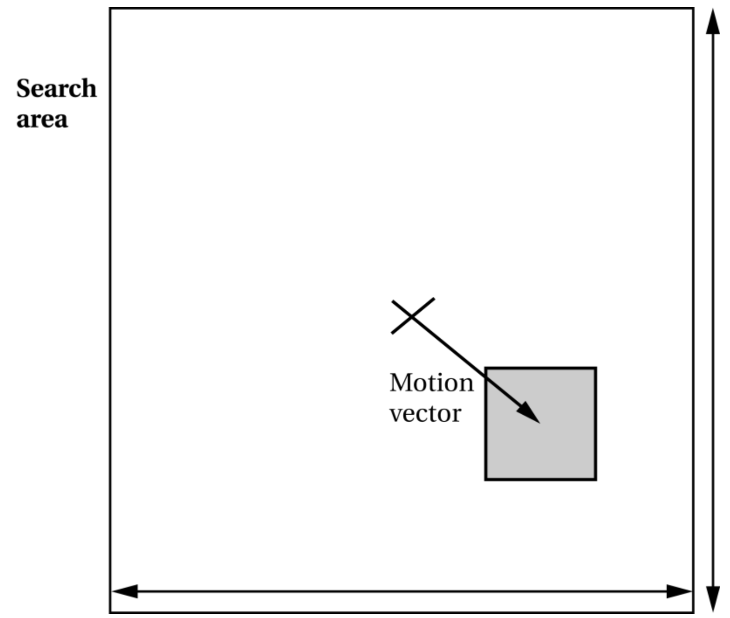
- Makes use of image compression techniques.
- Adds:
  - Support for frame-to-frame coding.
  - Audio streams, data, etc. controlled by a system stream.
- Two major families:
  - MPEG for broadcasting.
  - H.26x for videoconferencing.
- H.264/AVC combines techniques from both traditions.

MPEG-1/2 style compression



Motion estimation

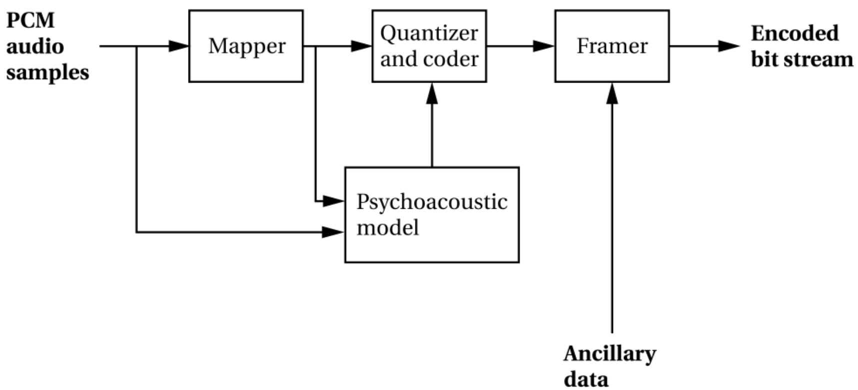
- Motion estimation compares one frame to another frame.
  - Generally performed on 16 x 16 macroblocks.
- Use 2-D correlation to find new position of a macroblock in the other frame.
- Transmit a motion vector to describe motion.



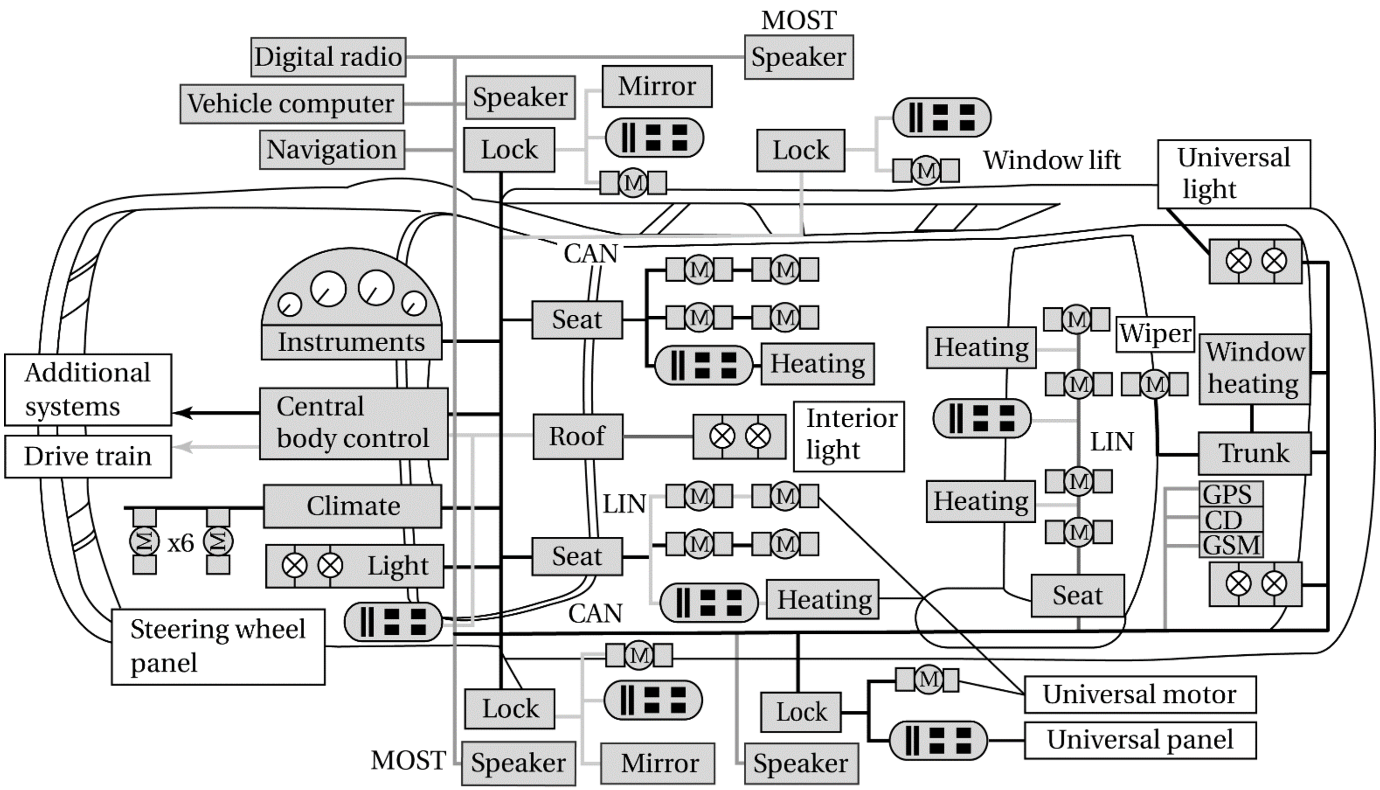
$$SAD = \Sigma_x \Sigma_y | S(x,y) - R(x,y) |$$

Audio encoding

- Perceptual coding models the human auditory system to predict what information can be thrown away.
- Subband decomposition helps improve the compression ratio.
- MP3 = MPEG-1 Audio Layer 3.



Automobiles as distributed embedded systems



CAN Controller Area Network
GPS Global Positioning System
GSM Global System for Mobile Communications
LIN Local Interconnect Network
MOST Media-Oriented Systems Transport

Automotive and aviation electronics

- Some functions are safety-critical.
- Must operate in real-time.
- Must fit within power budget (limited by generator).
- Must be lightweight to fit within vehicle weight budget.

- How would processors for these types of systems differ from those for multimedia systems?

Automotive electronics/avionics uses

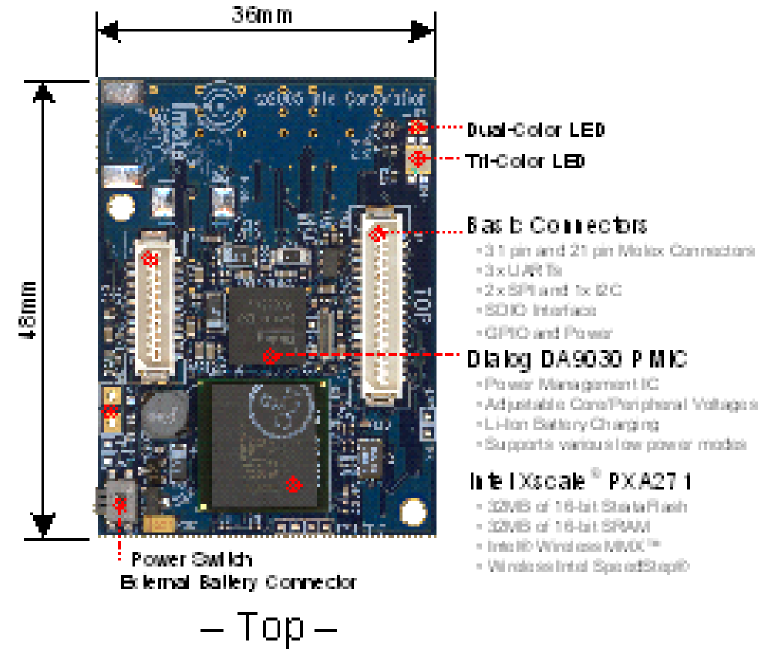
- Operator vs. passenger: Passenger operations are less critical, more varied (TV, Internet, etc.).
- Control vs. instrumentation: Instruments report on the vehicle, control closes the loop.
- Security: Jeep Cherokee hacked
  - https://www.youtube.com/watch?v=MK0SrxBC1xs
  - http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

Sensor networks

- Used to gather, process data in the field.
- Ad-hoc networks: must set themselves up without intervention of network manager.
- Often battery powered, very tight energy budget.
- Generally wirelessly networked.

Intel mote2

- Xscale processor.
- 256 KB of SRAM.
- 802.15.4 radio.
- Integrated antenna, etc.
- Can be programmed in C, nesC.
- TinyOS provides control functions.



Smartphone SoCs

- System-on-Chip (SoC)
  - Integrates peripherals with CPU on same die
  - Arguably first: Nintendo Gamecube
    - CPU, GPU, memory controller
- Smartphones with mixed-signal design
  - Single chip, low cost, incorporates most/all of analog signal processing hardware into CMOS
  - CMOS is not RF friendly
  - Initially simple ARM cores + RF

Smartphone SoCs today

- Integrate key components
  - CPU (multiple)
  - GPU (multiple cores)
  - Memory controller
  - RF pipeline: Baseband DSP, analog components
  - Media: Audio DSP, hardwired codecs
  - Camera: image signal processor
  - Sensors: sensor hub (always on, ultra low power)
  - …

High-IPC Processor Evolution

Desktop/Workstation Market

Scalar RISC Pipeline	2-4 Issue In-order	Limited Out-of-Order	Large ROB Out-of-Order
1980s: MIPS SPARC Intel 486	Early 1990s: IBM RIOS-I Intel Pentium	Mid 1990s: PowerPC 604 Intel P6	2000s: DEC Alpha 21264 IBM Power4/5 AMD K8

1985 – 2005: 20 years, 100x frequency

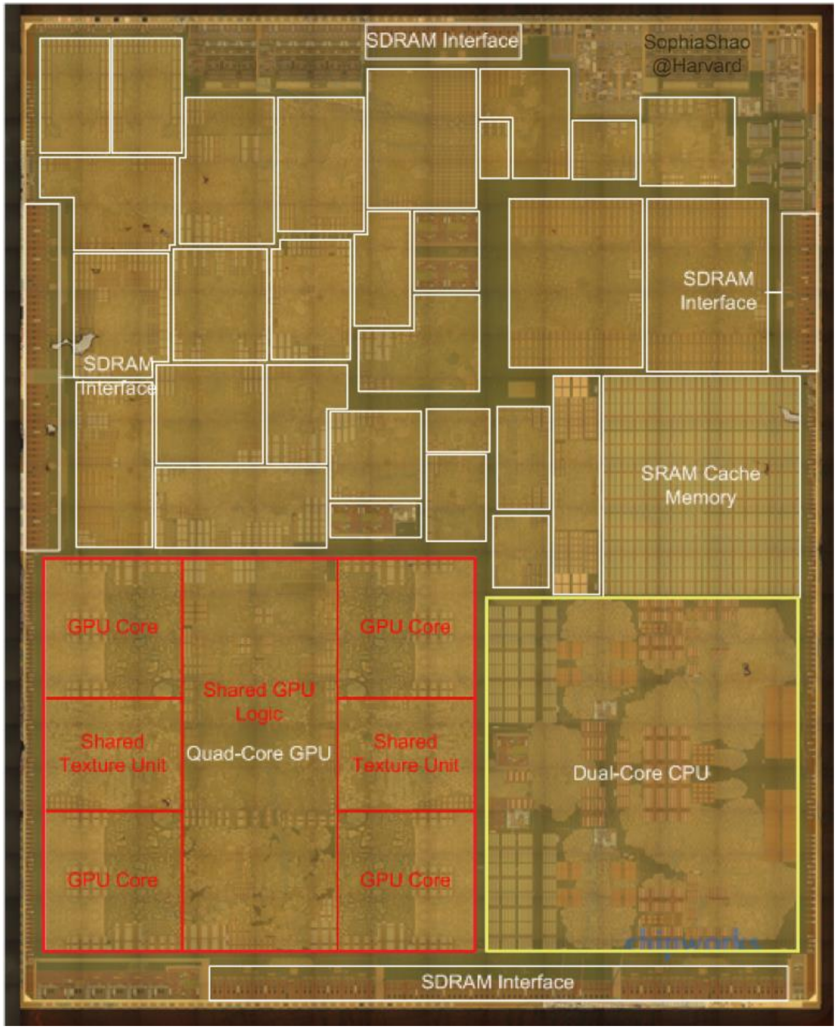
Mobile Market

Scalar RISC Pipeline	2-4 Issue In-order	Limited Out-of-Order	Large ROB Out-of-Order
2002: ARM11	2005: Cortex A8	2009: Cortex A9	2011: Cortex A15

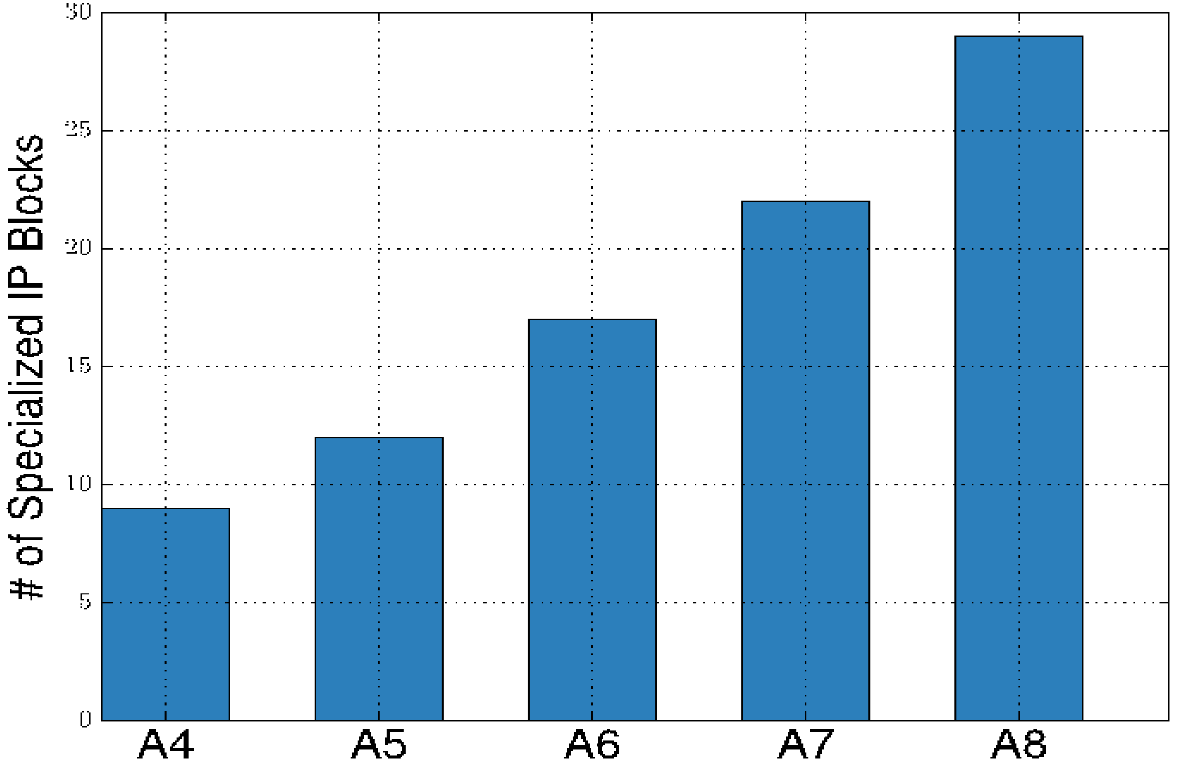
2002 – 2011: 10 years, 10x frequency

SoC IP Cores/Accelerators

- Annotated die photos reveal how little area CPU requires
- Apple A8 shown on right
- From: http://vlsiarch.eecs.harvard.edu/accelerators/die-photo-analysis



SoC IP Cores/Accelerators



- Growth in specialized IP blocks
  - Many of these are unknown outside Apple

Summary

- Embedded Computing Landscape.
- Aspects of Embedded System Design.
  - Architectures
  - Applications
  - Methodologies
  - Modeling
- Embedded System Applications.