Embedded Systems

Sribabu Doddapaneni

Christopher Paul Ramstetter

Embedded System

- Definition
- The Digital Signal Processor
- Embedded Benchmarks
- Embedded Multiprocessors
- Programming Embedded Systems
- Case Studies

Definition

- An Embedded System is a special purpose computer designed for a limited number of functions
- Key Characteristics
  - Usually real time
  - Processing information as "signals"
  - Limited memory and power conception

Examples

- Microwaves
- Washing machines
- Printers
- Networking devices
- Automobiles
- Cell phones
- PDAs
- Mp3 players
- Video game consoles
- TVs
- Children's toys

Cost comparison

Feature	Desktop	Server	Embedded
Price of system	$1000–$10,000	$10,000–$10,000,000	$10–$100,000 (including network routers at the high end)
Price of microprocessor module	$100–$1000	$200–$2000 (per processor)	$0.20–$200 (per processor)
Microprocessors sold per year (estimates for 2000)	150,000,000	4,000,000	300,000,000 (32-bit and 64-bit processors only)
Critical system design issues	Price-performance, graphics performance	Throughput, availability, scalability	Price, power consumption, application-specific performance

Digital Signal Processor

The Digital Signal Processor

- A special-purpose processor optimized for executing digital signal processing algorithms.
- Converts Analog to Digital
- Many perform multiply-accumulate (MAC)
  - A = A + B * C
- Usually fixed-point arithmetic
  - All data between -1 to +1
- Algorithms
  - Time-domain filtering
  - Convolution
  - Transforms
  - Forward error correction encodings

The TI 320C55

- Optimized for low-power, embedded applications
- 7-stage pipeline
  - Detects pipeline hazards and will stall on WAR and RAW
- Idle domains
- MACs

Pipeline

- **Fetch stage** – reads program data from memory into the instruction buffer queue
- **Decode stage** – decodes instructions and dispatches tasks
- **Address stage** – computes data addresses and branch addresses
- **Access 1/Access 2 stages** – send data read addresses to memory
- **Read stage** – transfers operand data
- **Execute stage** – executes and writes data

Idle domain

- Used to decrease power consumed
- Software programmable
- Six domains
  - CPU
  - DMA
  - Peripherals
  - Clock generator
  - Instruction cache
  - External memory interface

MACs

- Two macs
- 17-bit by 17-bit multiplier
- 40-bit adder
- 1 cycle to execute both multiple and add
- So two MACs are executed per cycle

The TI 320C6x

- Uses very long instruction word to exploit high level parallelism
- 11 stage pipeline
  - Four stages for instruction fetch
  - Two stages for instruction decode
  - Four stages for instruction execution

VLIW

- Very bad in code size
  - To over come they use a p bit that indicates if the instruction is in the current word or next one
  - No need for nops

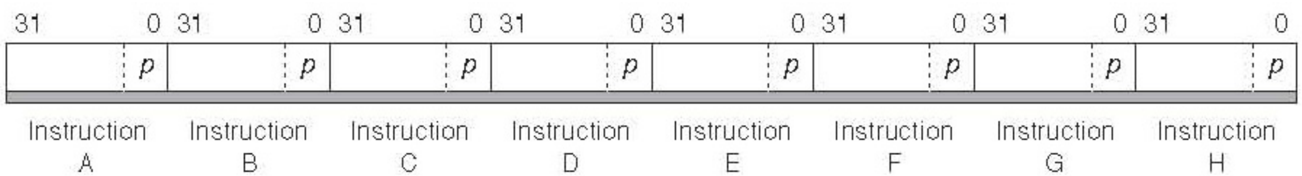


Figure D.6 Instruction packet of the TMS320C6x family of DSPs. The *p* bits determine whether an instruction begins a new VLIW word or not. If the *p* bit of instruction *i* is 1, then instruction *i* + 1 is to be executed in parallel with (in the same cycle as) instruction *i*. If the *p* bit of instruction *i* is 0, then instruction *i* + 1 is executed in the cycle after instruction *i*. (Courtesy Texas Instruments.)

Benchmarks

Benchmarks

- Tailored to tasks
  - Hard real time
  - Soft real time
  - Overall cost performance
- Broken up by type
  - Automotive
  - Consumer
  - Telecommunications
  - Networking
  - Office automation

Benchmark type ("subcommittee")	Number of kernels	Example benchmarks
Automotive/industrial	16	6 microbenchmarks (arithmetic operations, pointer chasing, memory performance, matrix arithmetic, table lookup, bit manipulation), 5 automobile control benchmarks, and 5 filter or FFT benchmarks
Consumer	5	5 multimedia benchmarks (JPEG compress/decompress, filtering, and RGB conversions)
Telecommunications	5	Filtering and DSP benchmarks (autocorrelation, FFT, decoder, encoder)
Digital entertainment	12	MP3 decode, MPEG-2 and MPEG-4 encode and decode (each of which are applied to five different data sets), MPEG Encode Floating Point, 4 benchmark tests for common cryptographic standards and algorithms (AES, DES, RSA, and Huffman decoding for data decompression), and enhanced JPEG and color-space conversion tests
Networking version 2	6	IP Packet Check (borrowed from the RFC1812 standard), IP Reassembly, IP Network Address Translator (NAT), Route Lookup, OSPF, Quality of Service (QOS), and TCP
Office automation version 2	6	Ghostscript, text parsing, image rotation, dithering, bezier

Figure D.8 The EEMBC benchmark suite, consisting of 50 kernels in six different classes. See *www.eembc.org* for more information on the benchmarks and for scores.

Benchmark Power

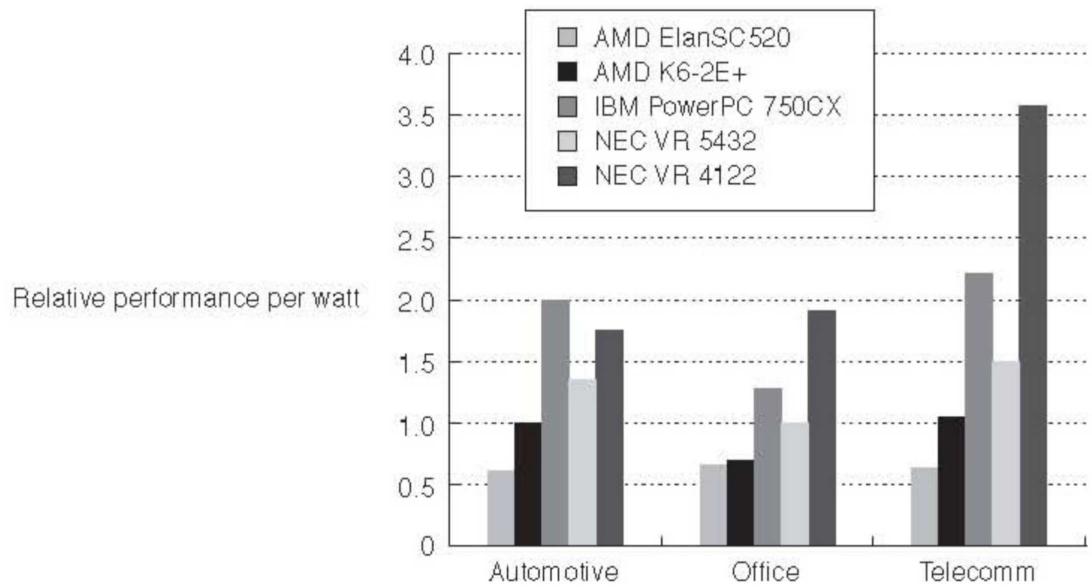


Figure D.9 Relative performance per watt for the five embedded processors. The power is measured as typical operating power for the processor and does not include any interface chips.

Benchmark Preformance

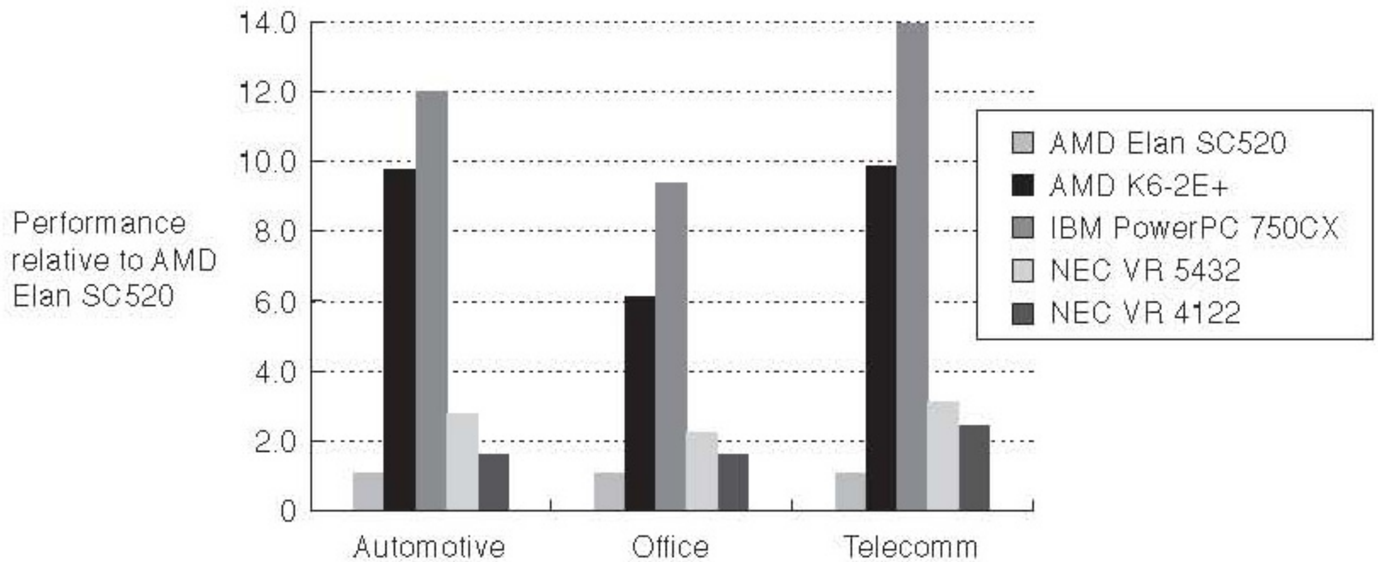


Figure D.10 Raw performance for the five embedded processors. The performance is presented as relative to the performance of the AMD ElanSC520.

Multiprocessors

- Growing
  - Most embedded systems are writen form scratch
  - Natural paralleism
- Example MXP

MXP

- Made for high-end telecommunications and networking
- Features
  - Interface to serial voice streams
    - Support for handing jitter
  - Fast packet routing and channel lookup
  - Ethernet interface
  - For MIPS processors
    - Used for
      - Code for maintaining voice-over-IP channels
      - Quality of service
      - Echo cancellation
      - Simple compression
      - Packet encoding

Building Embedded Systems & Case Studies

the rest of the talk

- **Issues** (why is it difficult)
- Classification based on **Processor** used
- **Programmable** Embedded Systems
- **Case Studies**

Design Constraints

Problem Matrix is Multi-dimensional

Power, performance, code size, weight, etc.

Stringent

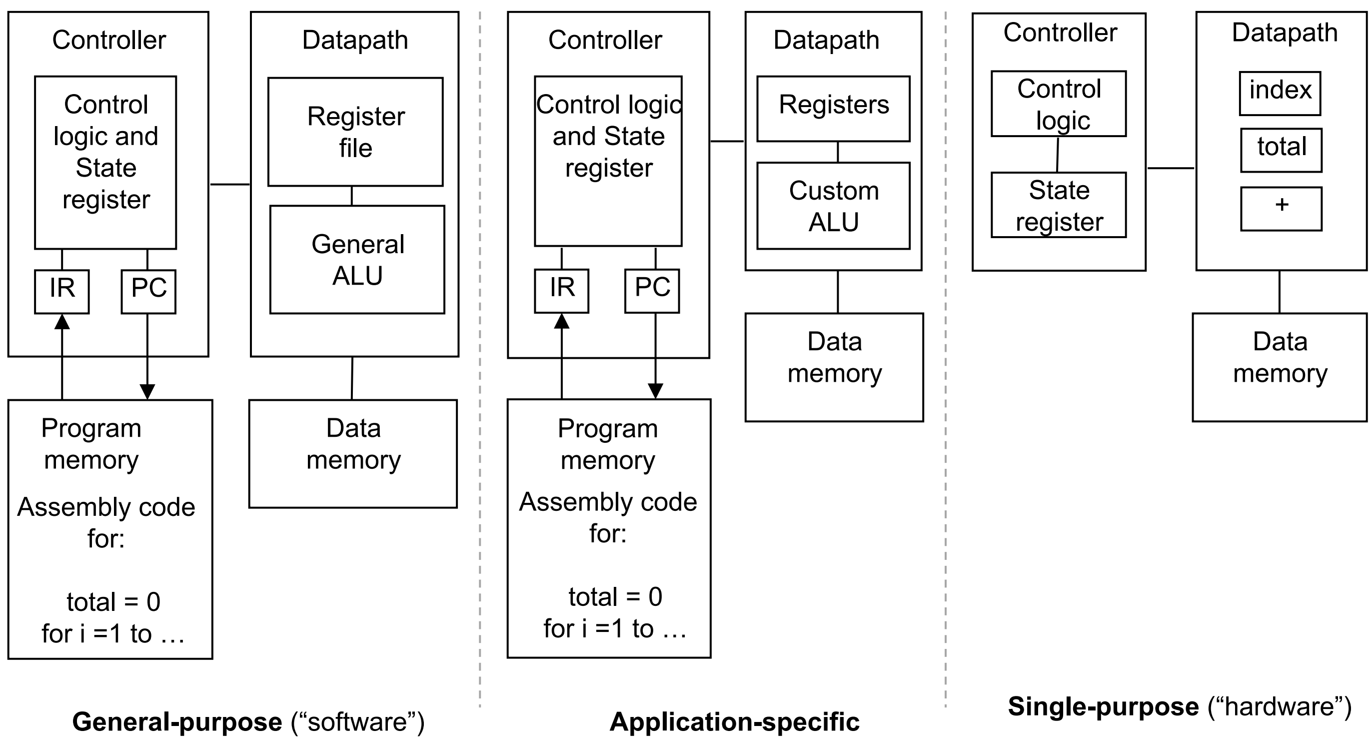
Tighter **constraints**, extreme **resource** constraint

Application Specific

Radiation level, operating **temperature**, available **wattage**

Processors

Processor technology



Processor does not have to be programmable

General-purpose processors

Programmable device used in a variety of applications Also known as “**microprocessor**”

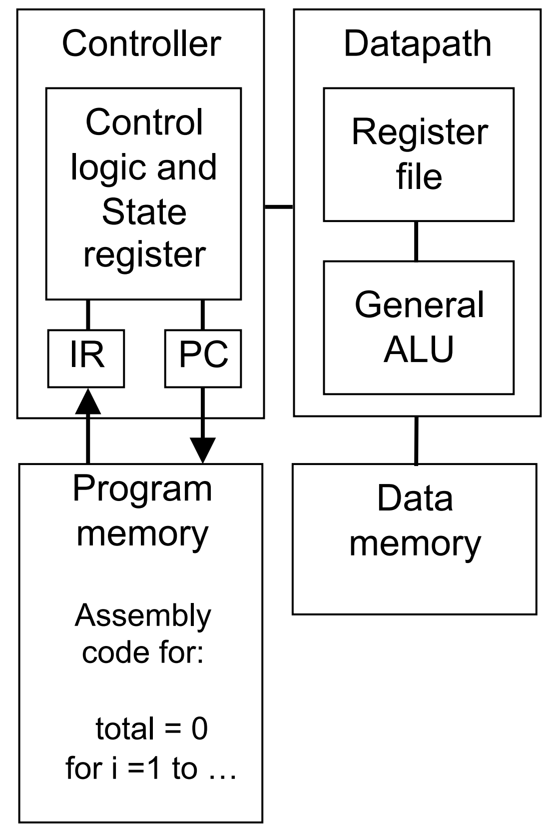
Features

- Program memory
- General datapath with **large** register file and **general** ALU

User benefits

- Low time-to-market and NRE costs
- High flexibility

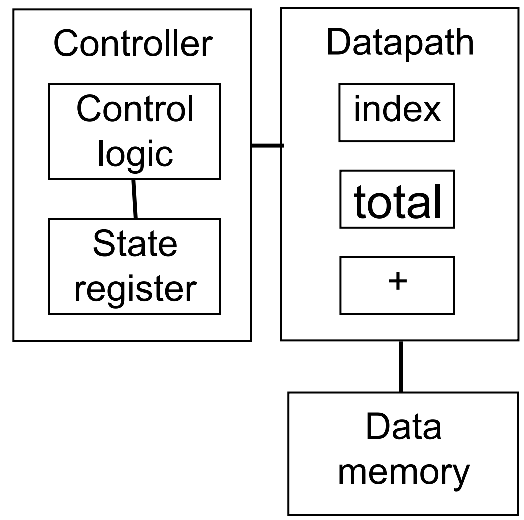
“**Pentium**” the most well-known, but there are hundreds of others



Single-purpose processors

Digital circuit designed to execute exactly one program/task

a.k.a. coprocessor, accelerator or peripheral



Features

- Contains only the components needed to execute a single program
- No program memory

Benefits

- Fast
- Low power
- Small size

e.g. DSPs.

Application-specific processors

Programmable processor optimized for a particular class of applications having common characteristics

Compromise between general-purpose and single-purpose processors

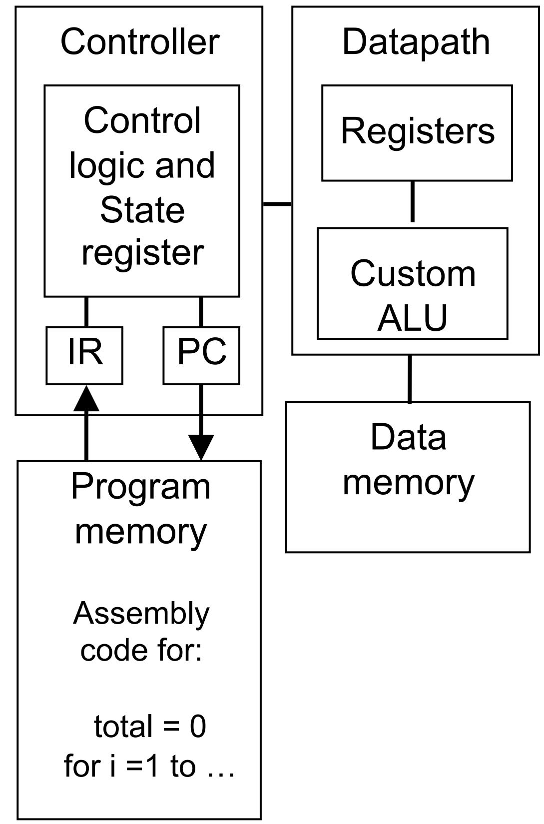
Features

- Program memory
- Optimized datapath
- Special functional units

Benefits

Some flexibility, good performance, size and power

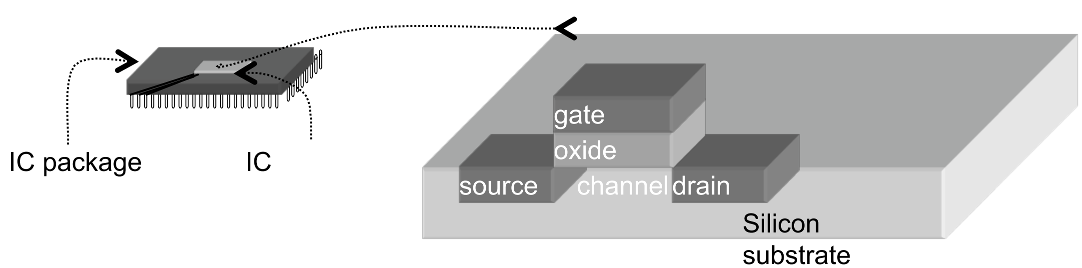
e.g. PIC, MSP430



IC technology

The manner in which a digital (gate-level) implementation is mapped onto an IC

- IC: Integrated circuit, or "chip"
- IC technologies differ in their customization to a design
- IC's consist of numerous layers (perhaps 10 or more)
  - IC technologies differ with respect to who builds each layer and when



IC technology

Three types of IC technologies

1. Full-custom/VLSI
2. Semi-custom ASIC (gate array and standard cell)
3. Programmable Logic Device (PLD)

Full-custom/VLSI

All layers are optimized for an embedded system's particular digital implementation

- **Placing** transistors
- **Sizing** transistors
- Routing wires

Benefits

Excellent performance, small size, low power

Drawbacks

High Initial cost (e.g., $300k), long time-to-market

Semi-custom

Lower layers are fully or partially built

- Designers are left with **routing of wires** and maybe placing some blocks

Benefits

- Good performance, good size, lesser initial cost than a full-custom implementation (perhaps $10k to $100k)

Drawbacks

- Still require weeks to months to develop

PLD (Programmable Logic Device)

All layers already exist

- Designers can purchase an IC
- Connections on the IC are either created or destroyed to implement desired functionality
- Field-Programmable Gate Array (FPGA) very popular

Benefits

Low initial costs, almost instant IC availability

Drawbacks

Bigger, expensive (perhaps $30 per unit), power hungry, slower

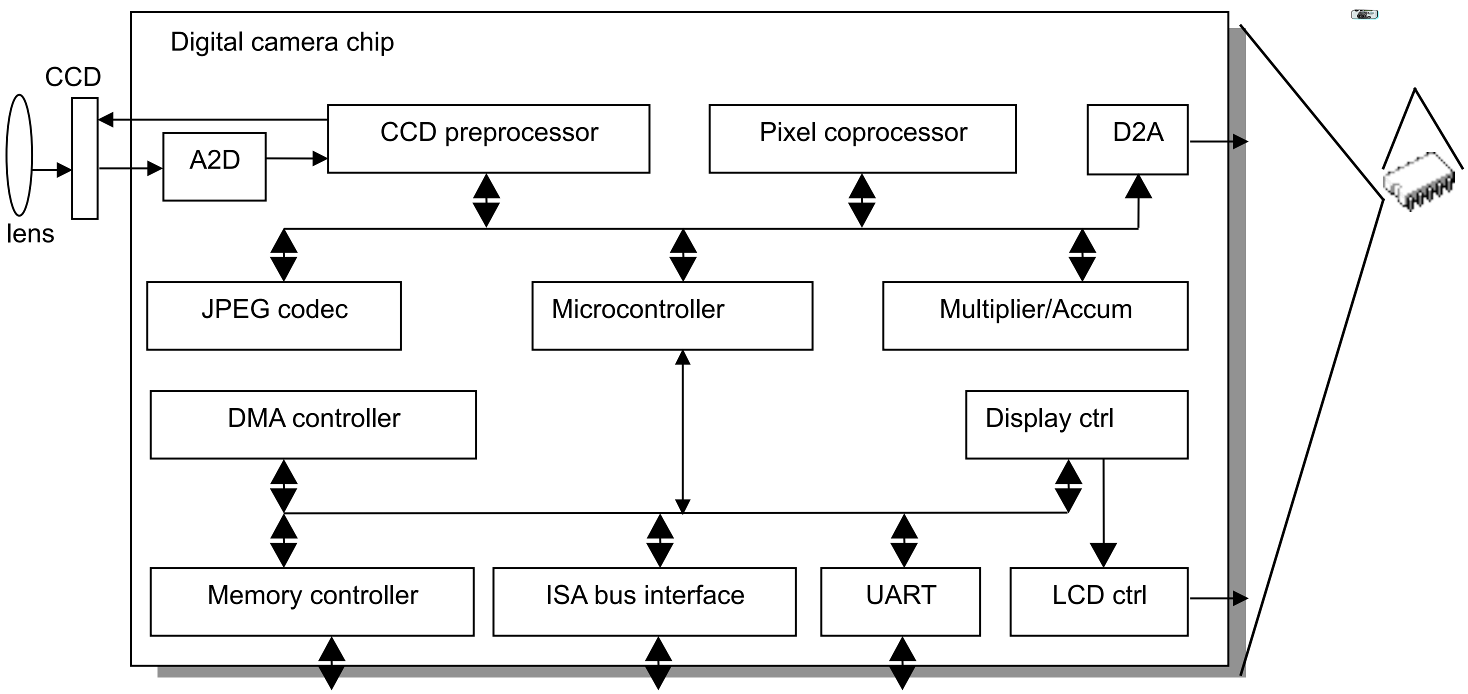
Peripherals

- Serial Communication Interfaces (SCI): RS-232, RS-422, RS-485 etc
- Synchronous Serial Communication Interface: I2C, SPI, SSC and ESSI
- Universal Serial Bus (USB)
- Networks: Ethernet, Controller Area Network etc
- Timers: PLL(s), Capture/Compare and Time Processing Units
- Discrete IO: aka General Purpose Input/Output (GPIO)
- Analog to Digital/Digital to Analog (ADC/DAC)
- Debugging: JTAG, ISP, BDM Port

Case Study 1:

Digital Camera

An embedded system example -- a digital camera

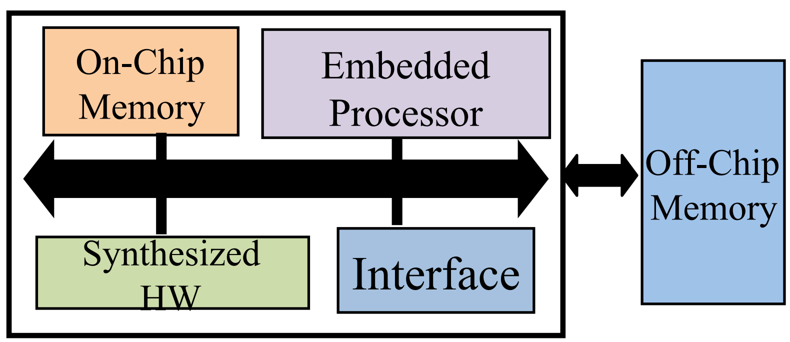


- Single-functioned -- always a digital camera
- Tightly-constrained -- Low cost, low power, small, fast
- Reactive and real-time -- only to a small extent

Programmable Embedded system

Programmable ES

- Increasing Complexity
- Shrinking Time-to-market
- Programmable Embedded Systems
  - Increase designer productivity
    - Software provides
      - faster development
      - easier reusability and upgrade-ability



Can I use existing Compilers?

Design Constraints
- Multi-dimensional
  - Power, performance, code size, weight, etc.
- Stringent
  - Tighter constraints, highly resource constraint
- Application-specific
  - Radiation level, operating temperature, available wattage

"Highly-customized" designs of embedded systems
- Different ISAs
  - e.g. ARM, MIPS 16, micro-controllers
- Missing architectural features
  - e.g. missing caches, branch predictors
- Design idiosyncrasies
  - e.g partitioned register file, hardware loop counters
- "Light-weight" versions of standard architectural features
  - e.g partial register renaming, limited support for prefetching, partial predication

Functionality – Maybe (Different ISAs)
Optimizations – NO

Compiler for Embedded Systems
Meet all design constraints simultaneously

Compiler Issues

Code size

Performance

Power, Energy

Real-time guarantees

Security

Reliability

Robustness

Compiler for ES

Highly Customized embedded processor architecture

Compiler's job is tough

- Limited compiler technology
- Difficult and costly analysis

2-fold job of Compiler

- Exploit existing design features
- Avoid loss due to **missing design** features

However, Compiler can be very effective

Significant impact on power, performance etc.

Embedded software architectures

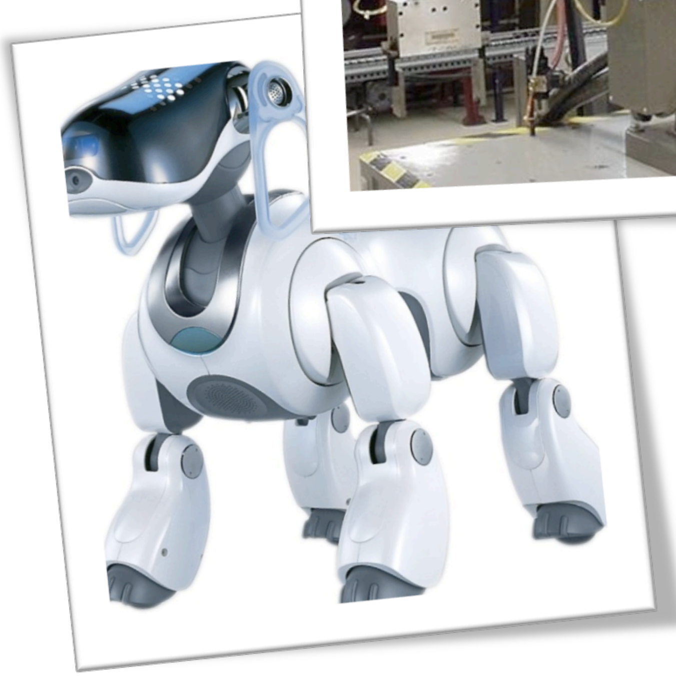
- Simple control loop
- Interrupt controlled system
- Preemptive multitasking or multi-threading
- Monolithic kernel
- MicroKernel
- Exotic custom operating systems
- Additional software components

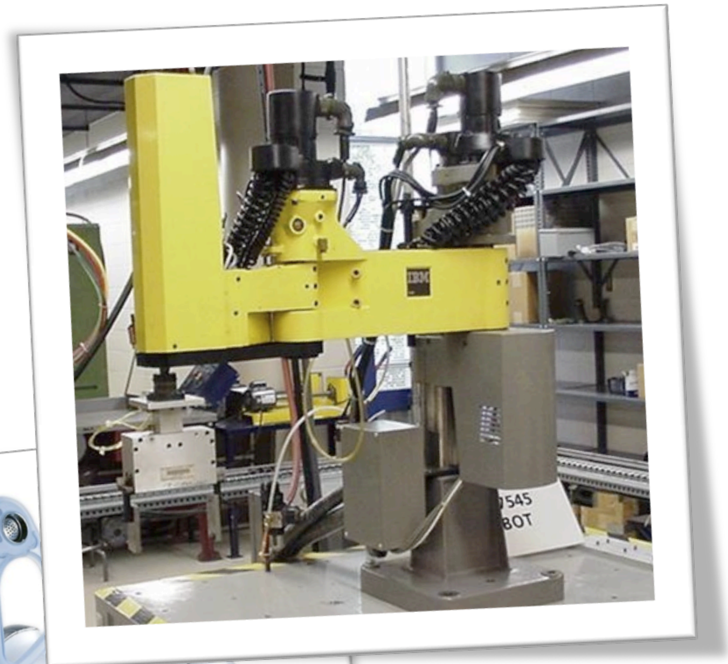
Debugging

- **Interactive resident debugging :** using the simple shell provided by the embedded operating system
- **External debugging** using logging or serial port output to trace operations
- **An in-circuit debugger (ICD)**, a hardware device that connects to the microprocessor via a **JTAG** or **NEXUS** interface.
  - Replaces the microprocessor with a simulated equivalent
  - Provides full control over all aspects of the microprocessor

Firmware vs RTOS

v/s

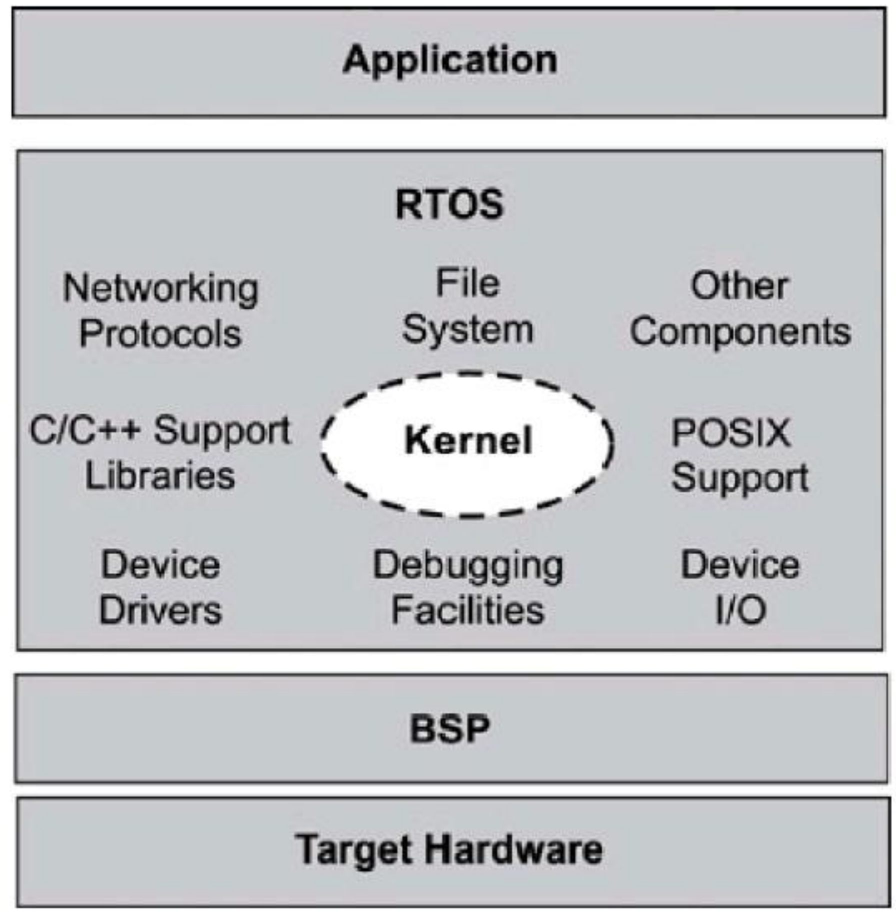




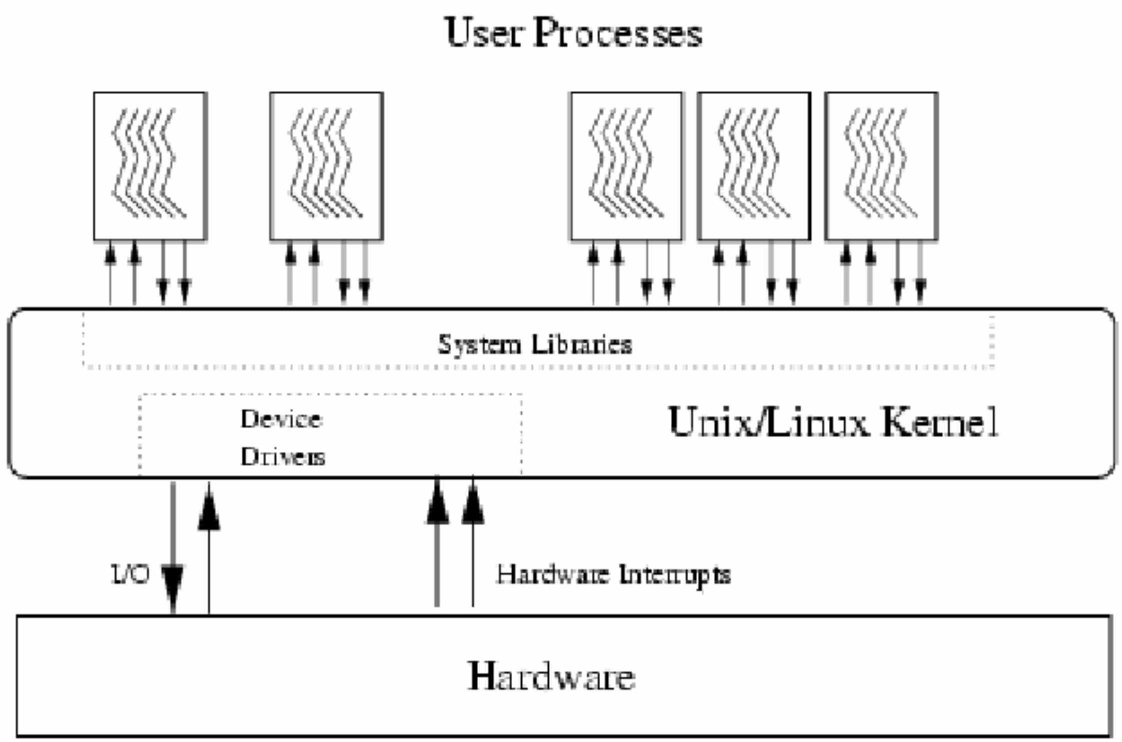
RealTimeOS

Features

- Scheduling
- Events
- queues
- low interrupt latencies
- small memory footprint
- fast context switching
- Timer Resolution

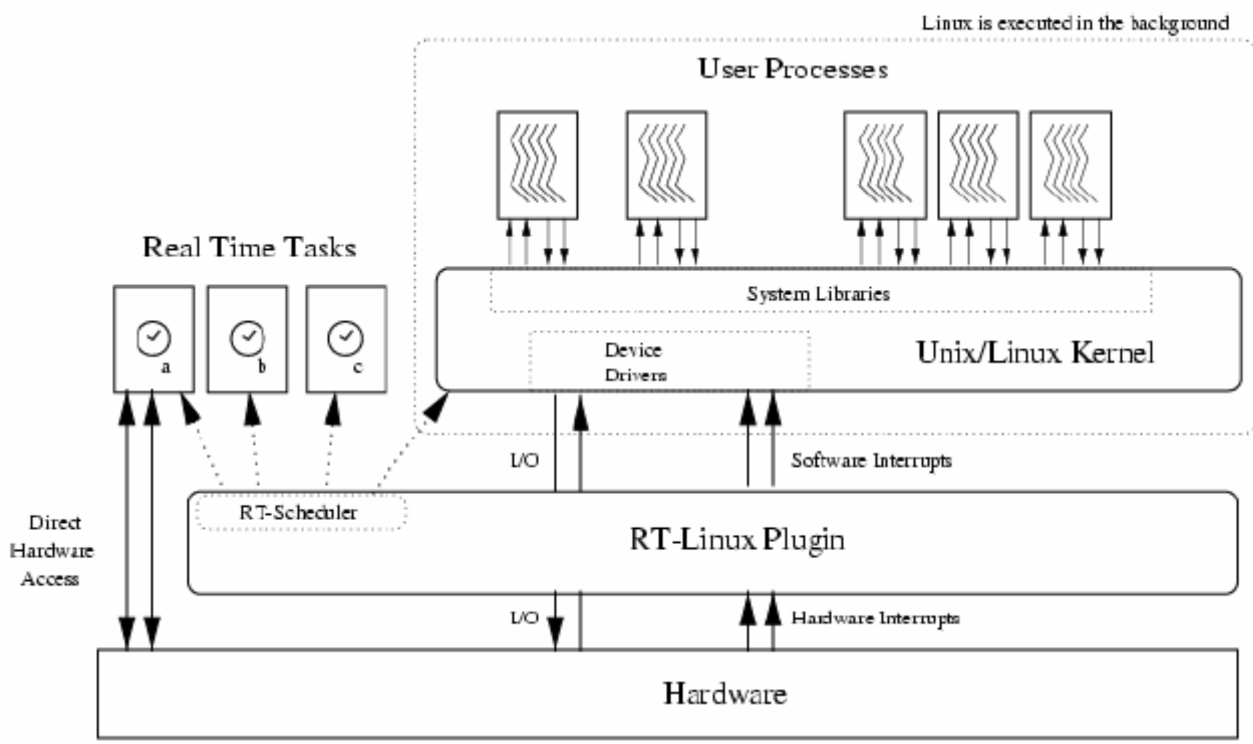


Linux Bare kernel



Detail of the bare Linux kernel

RTLinux Kernel



Detail of the RTLinux kernel

Case study 2

Inside a Cell Phone

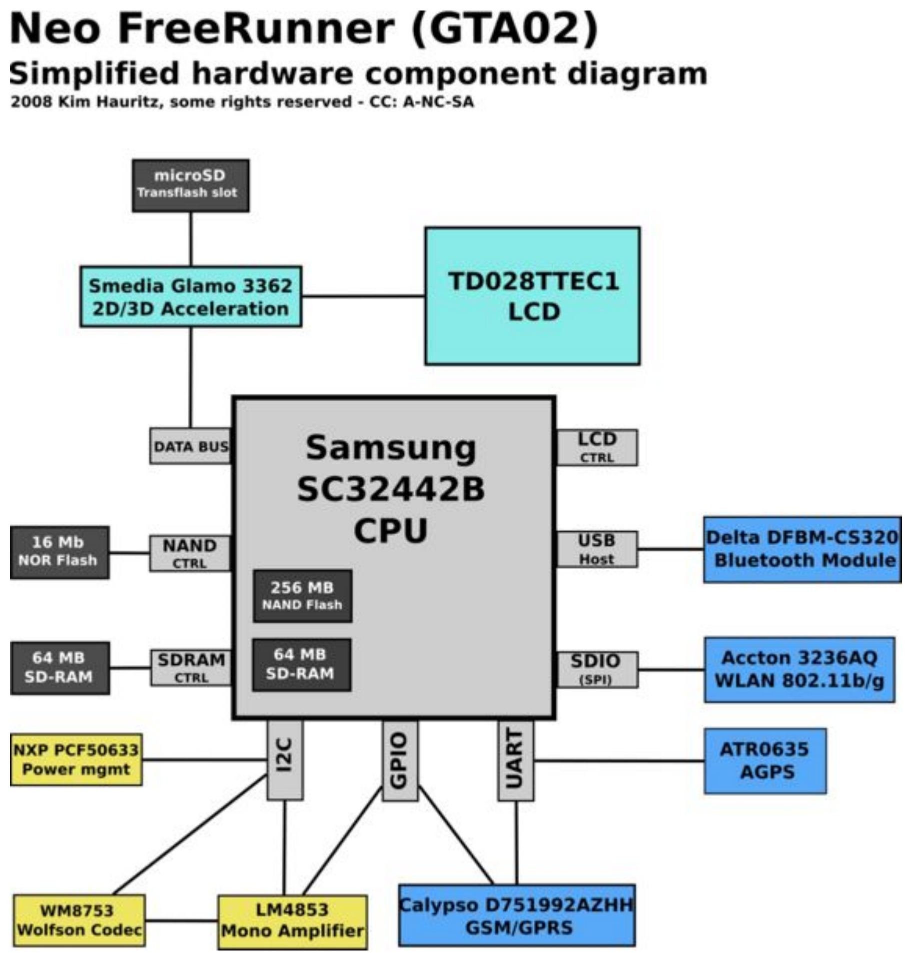
Case Study: OpenMoko

- Linux kernel
- GUI with **X.Org** Server, **GTK+** toolkit, and the **Matchbox** window manager and also support for **Qt** toolkit and **Enlightenment** 17.
- Native applications can be developed and compiled using various languages including **C and C++.**

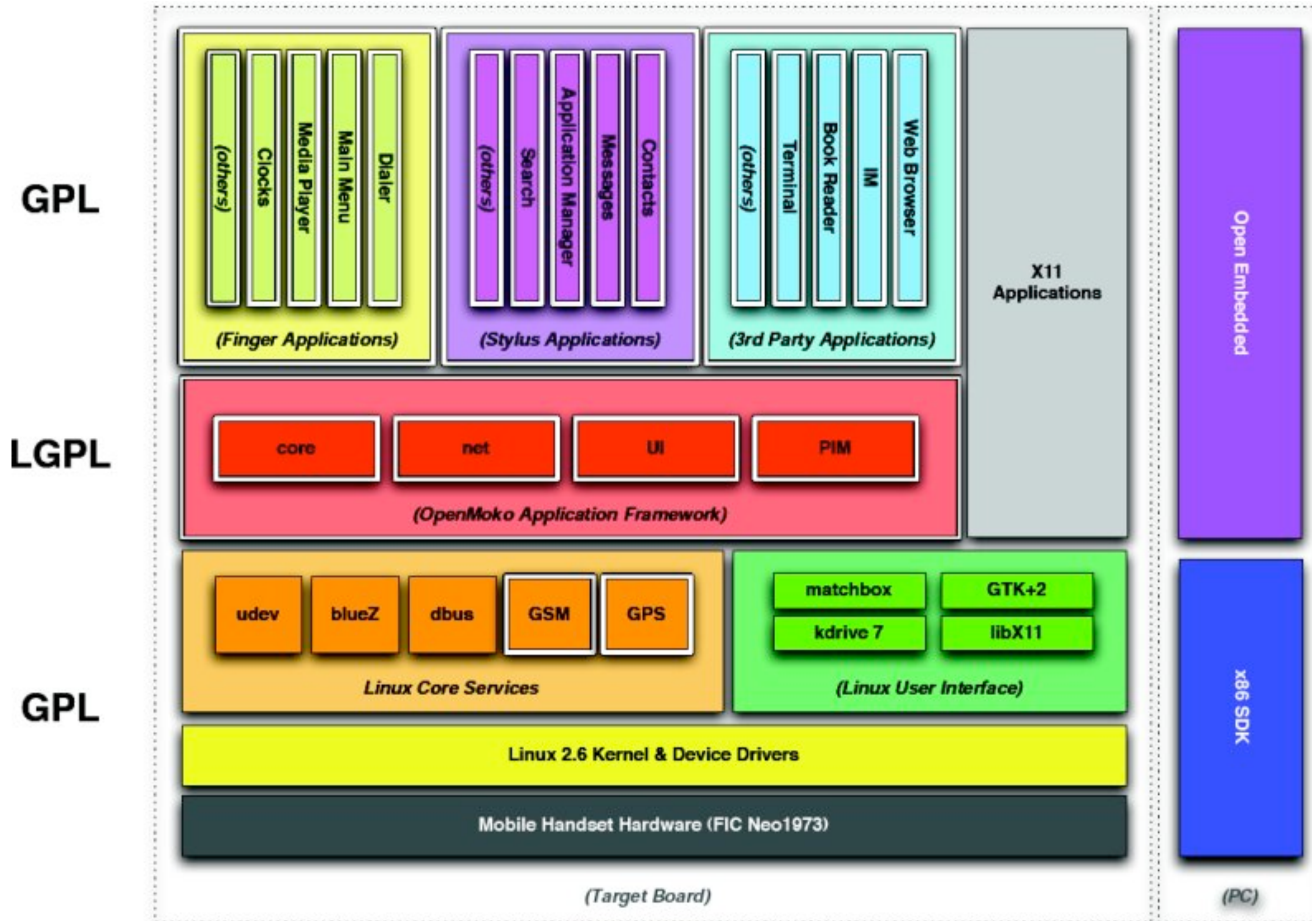
Development Tools

- toolchain to enable code compilation
- **MokoMakeFile** which is a tool that greatly simplifies build process
- The QEMU emulator can be used to run Openmoko
- Full GTA02 hardware emulation
- Phone can be flashed directly

Block Diagram



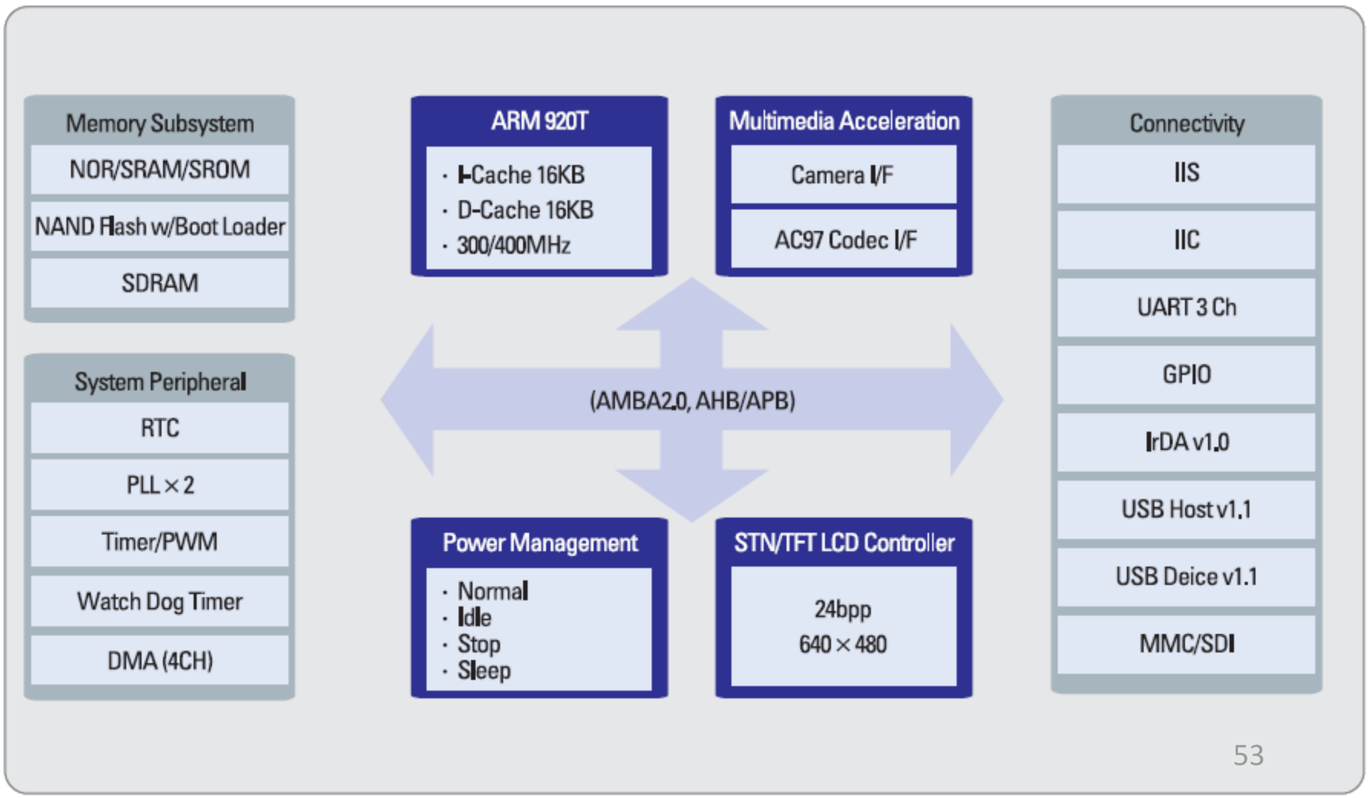
Software Stack



Samsung S3C2442B (Multi Stacked Package)

separate **16 KB instruction cache** and **16 KB data cache**, **MMU** to handle virtual memory Management, **TFT** and **STN LCD** controller, **NAND flash boot loader**, 3-ch **UART**, 4-ch **DMA**, 4-ch timers with PWM, I/O ports, RTC, 8-ch 10-bit ADC and touch screen interface, camera interface, IIC-BUS interface, IIS-BUS interface, USB host, USB device, SD host and multimedia card interface, **2-ch SPI** and **PLL** for clock generation.

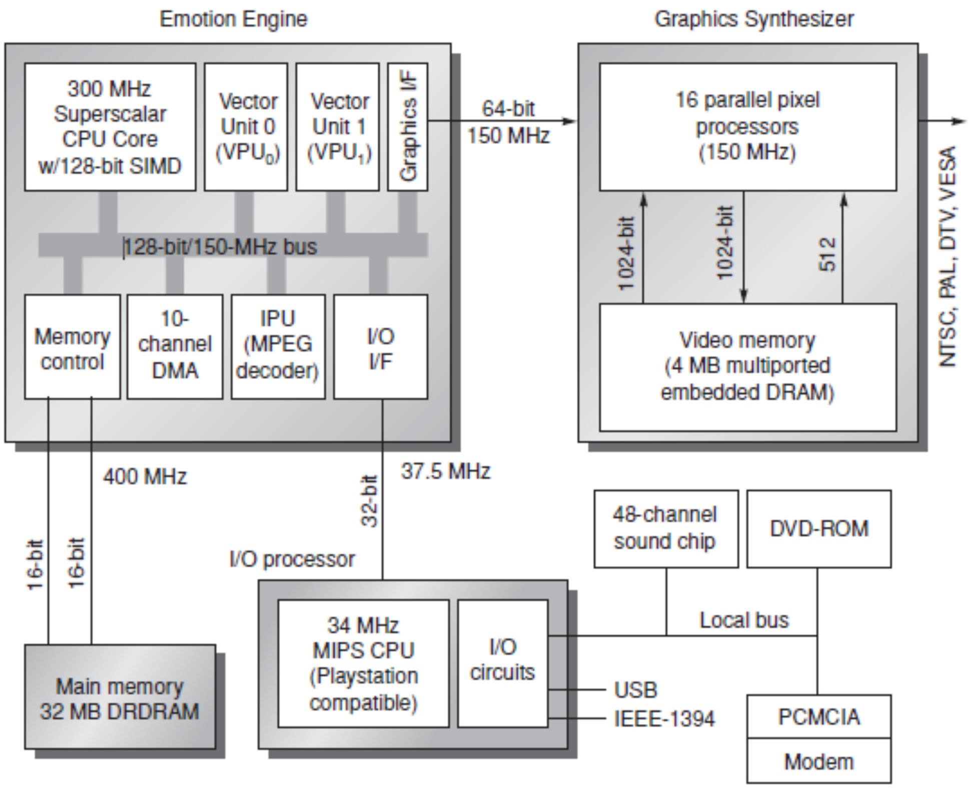
- Specification:
- Core: ARM920T
- Instruction Set: ARMv4

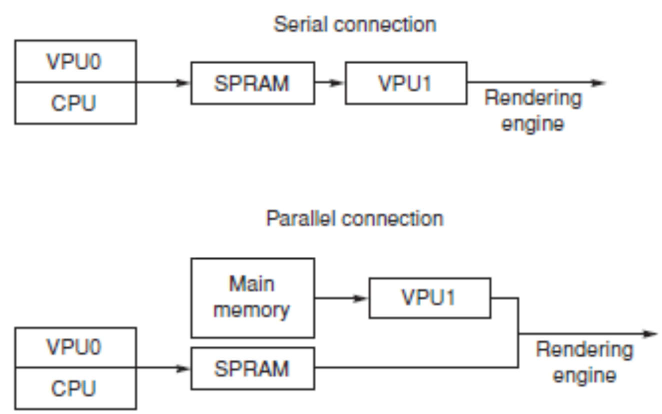


Case Study 3

Emotion Engine for Sony Playstation 2

Emotion Engine of the Sony Playstation 2





References

- Appendix D, Computer Architecture by John Hennessy and David Patterson
- *Openmoko.org*
- John Catsoulis, Designing Embedded Hardware, O'Reilly, May 2005, ISBN 0-596-00755-8
- A few slides adopted from *www.public.asu.edu/~ashriva6/teaching/CES/CES_Spring_2008*

Thank you

?s