Embedded Systems

Advanced GPP/ASP

From:

“Sistemi Embedded: Sviluppo HW e SW per sistemi dedicati (Cap. 6)”

W. Fornaciari, C. Brandolese - Edizioni Pearson – Prentice Hall 2007

and other different sources.

Overview

- Introduction
- General Purpose Processors
  - CISC, RISC, Superscalar Architectures
  - CISC/RISC, EPIC/VLIW, CISC/VLIW Architectures
- Application Specific Processors
  - Digital Signal Processor
  - Network Processor
  - Microcontrollers
  - Graphical Processing Unit
  - MPSoC
- Conclusion

Introduction

Introduction

- A lot of embedded applications are implemented by exploiting SW
  - This is mainly due to higher flexibility (maintainability and extendibility), and reduced NRE costs and time-to-market
- However, SW normally has poor *performances* with respect to HW solutions since SW exploit generic processing architectures
  - Execution time, power/energy, memory, chip area, etc.
- So, adopting a SW solution requires to know very well what the current microprocessor technology is offering and compare this with F/NF requirements
  - Main choices: processor class and form factor

Introduction

- Processor Class
  - *General Purpose Processor vs Application Specific Processor*
    - Typical embedded architectures are composed of a GPP for supervision/management, and one/more ASP for specific functions
- Form Factor
  - COTS vs IP
    - COTS
      - Physical chip to be mounted on a board and interfaced with other devices
    - IP
      - Description (i.e., a model) of a processor mainly used with PLD or for ASIC design
        - » e.g. Micro/Pico Blaze di Xilinx, Nios-II di Altera, ARM di Arm Ltd., etc.
      - Different abstraction levels
        - » *soft macro* (RT level HDL) or *hard macro* (bitstream/layout)

Introduction

- Other aspects
  - Performance (execution time): CPI/IPC, MIPS, MFLOPS, …
  - Consumption: average power, peak power, mW/MHz, MIPS/mW
  - Memory: capacity, bandwidth, addressing
  - Peripherals: on-chip, on-board
  - SW components: libraries, OSs
  - Development enviroments: code generation and analysis
  - Packaging (COTS): dimension, pins, material
  - Certification: commercial, industrial, military, aerospace

General Purpose Processors

GPP

- GPP are generic processing architecture suitable to provide good average performances in different fields
  - Used if there are no strict performance requirements, the application is quite heterogeneous, or for management and supervision
- GPP architectures are mainly classified as
  - CISC: *Complex Instruction Set Computer*
    - A big number of very complex instructions
  - RISC: *Reduced Instruction Set Computer*
    - Few simple instructions

GPP

CISC Architectures

GPP

- CISC Architectures
  - First kind of architectures, mainly due to the high memory cost
    - Normaly provided of instructions able to perform *load-execute-store*
    - A lot of addressing modes
    - Instructions coded with different lenghts words
    - ALU offers a lot of variation in basic operations
      - Also *vectorial operations* able to work concurrently on more registers
        - » *Single Istruction Multiple Data (MMX,…, SSE, AVX, etc.)*
  - All this leads to complex (i.e. slower) fetch unit, decode unit, data-path and also stress the memory time access requirements

GPP

- CISC Architectures
  - So, to improve performances (i.e. *throughput*), they have been introduced very long and complex *pipelines*
  - However, their full exploitation requires ***out-of-oder execution***
    - But, given a very complex pipeline, it is very difficult for a compiler to really optimize execution time at compile-time since it is needed to know the state of the pipeline at run-time
    - So, modern CISC architecture are provided of an internal HW support to perform dynamical scheduling at run-time
    - Such HW components are very complex and increase both processor size and power consumption

GPP

RISC Architectures

GPP

- RISC Architectures
  - New kind of architectures, possible thanks to the reduced cost per KB in DRAM technologies at the end of ‘80
    - In the same period, the increased IC integration capabilities also offer the opportunity to realize more complex devices on chip
      - Pipelines and greater number of registers (reduced *memory spill*)
  - To have few simple RISC operations simplify also the processor architecture
    - Fixed size instructions, more balanced pipeline stages, load/store ISA with simplified addressing modes, optimal cache hierarchy, and higher frequencies
      - All is more exploitable thanks to uniform instruction execution time

GPP

- RISC Architectures
  - Moreover…
    - Thanks to the predictability of instruction execution time, a good compiler is able to optimize pipeline exploitation
      - No additional HW units strictly needed
  - Basic RISC are simple and powerful architectures with reduced power consumption

GPP

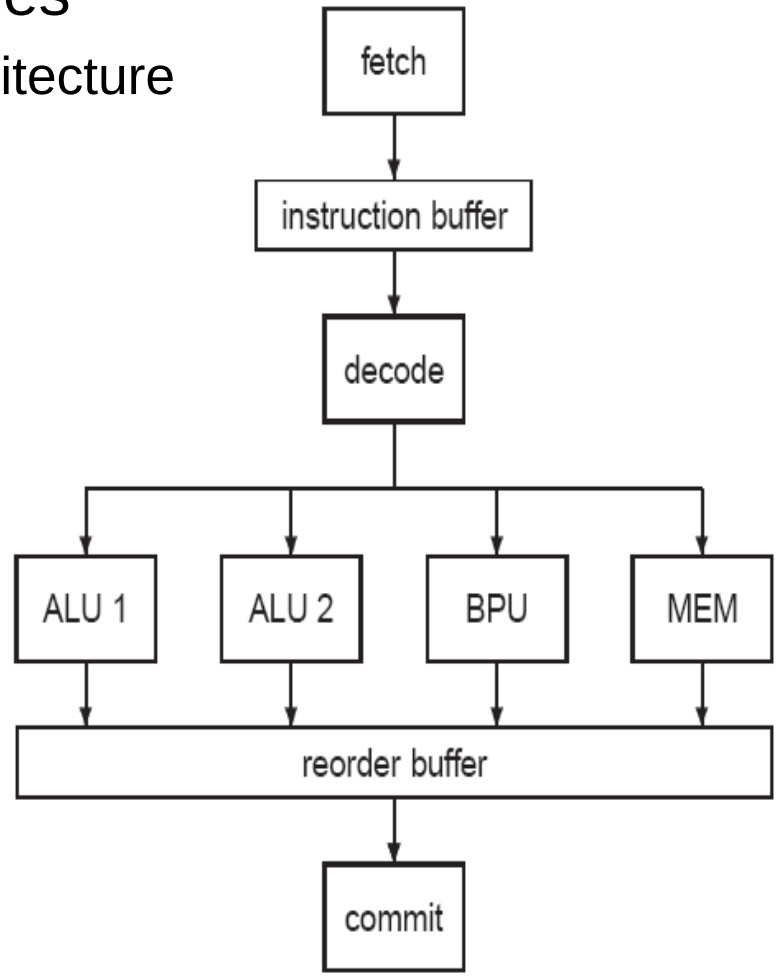
Superscalar Architectures

GPP

- Superscalar Architectures
  - Analysis of assembly code shows that it exist some **instruction level parallelism** (ILP) between non dependent instructions
    - Such instructions could be executed in different order and, if possible, also concurrently
      - This is the main fact that justify the design of superscalar architectures and their efficiency
  - A superscalar architecture is composed of more processing units
    - ALUs and/or BPUs (*Branch Processing Unit*)
  - Thanks to the parallelism offered by the *exe* stage, it is possible to execute more than one instruction for each clock cycle
    - However, control unit complexity limits frequency and increases power consumption

GPP

- Superscalar Architectures
  - Typical superscalar architecture



GPP

- Superscalar Architectures
  - Moreover, it is the processor to dynamically schedule instructions
    - Pro
      - Architecture-independent compilers
        - » Only ISA dependent
    - Con
      - A relevant % of processor size is dedicated to management
        - » Scheduling and consistency check
  - To better exploit HW resources
    - *Simmetric Multi-Threading (SMT: Intel "Hyperthreading" or similar)*
      - To fecth assemby instructions belonging to different processes/threads
        - » Need for some duplication also in the control unit

GPP

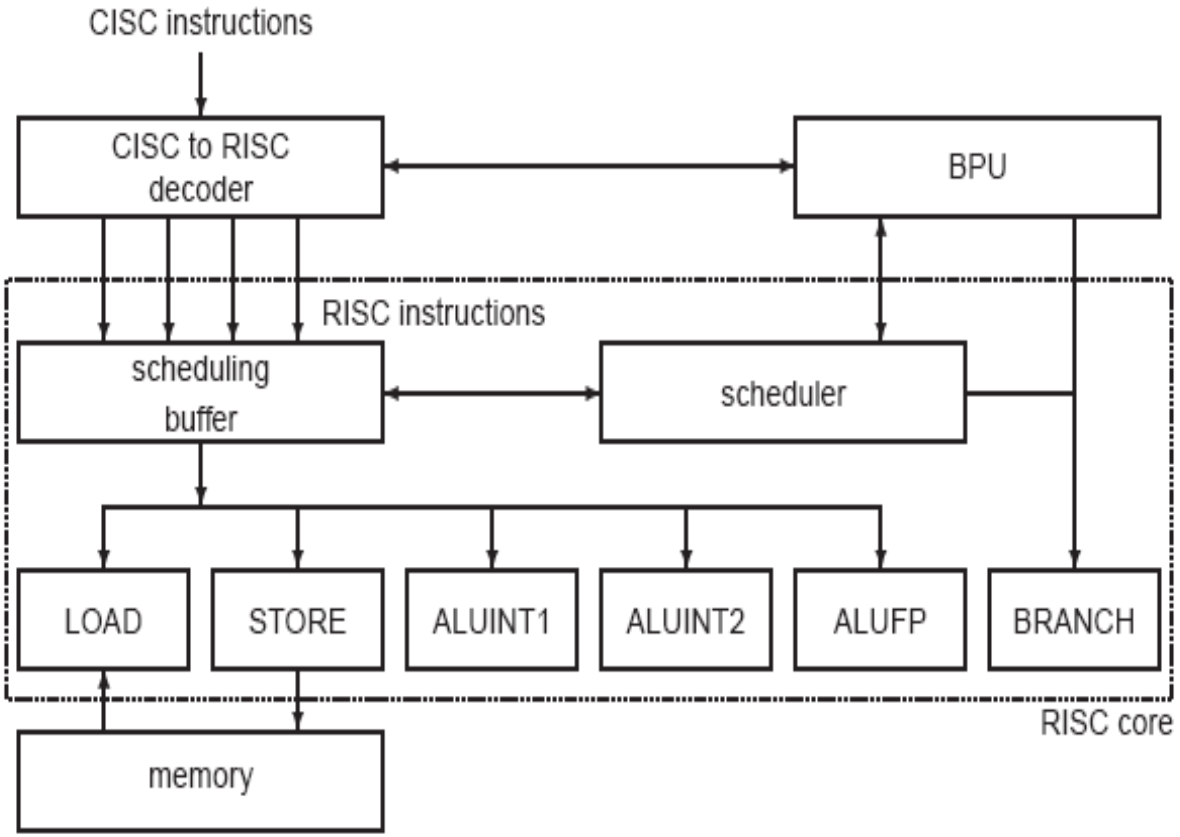
CISC/RISC Architectures

GPP

- CISC/RISC Architectures
  - Approach that combines *out-of-order execution* e superscalarity to keep ISA-level compatibility with older processors
    - x86 ISA (IA-32/Intel-64) is a *de facto* standard for desktop computing
  - Each CISC instruction is replaced at run-time by an equivalent sequence of RISC instructions executed by an effective and efficient RISC core
    - This technique is really HW demanding (and power consuming) but it allows retro-compatibility and high performances

GPP

- CISC/RISC Architectures
  - CISC processor with RISC core

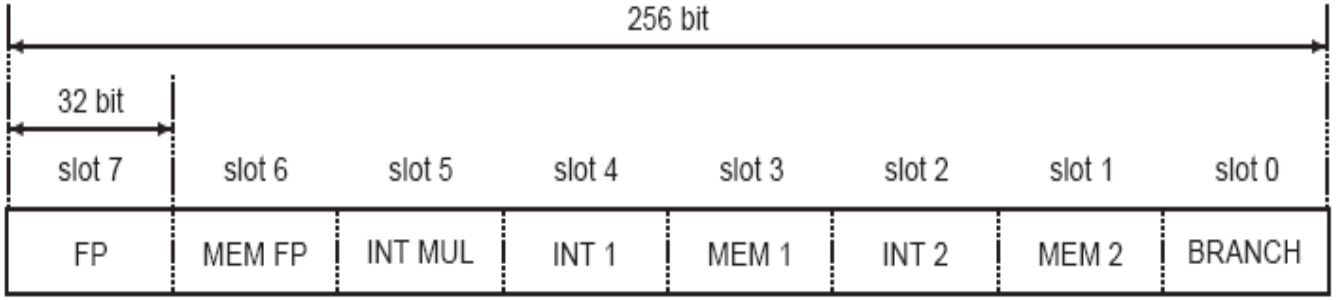


GPP

EPIC/VLIW Architectures

GPP

- EPIC/VLIW Architectures
  - To overcome the limitations related to HW scheduling complexity, new architectures have been introduced
    - VLIW: Very Long Instruciton Word
    - EPIC: Explicit Parallel Instruction Computer
      - Intel Itanium/Itanium2 (IA-64)
  - They are able to execute more instructions concurrently under the control of the program
    - A VLIW instruction is typically composed of a group of RISC instructions in a single *very large word* with fixed structure

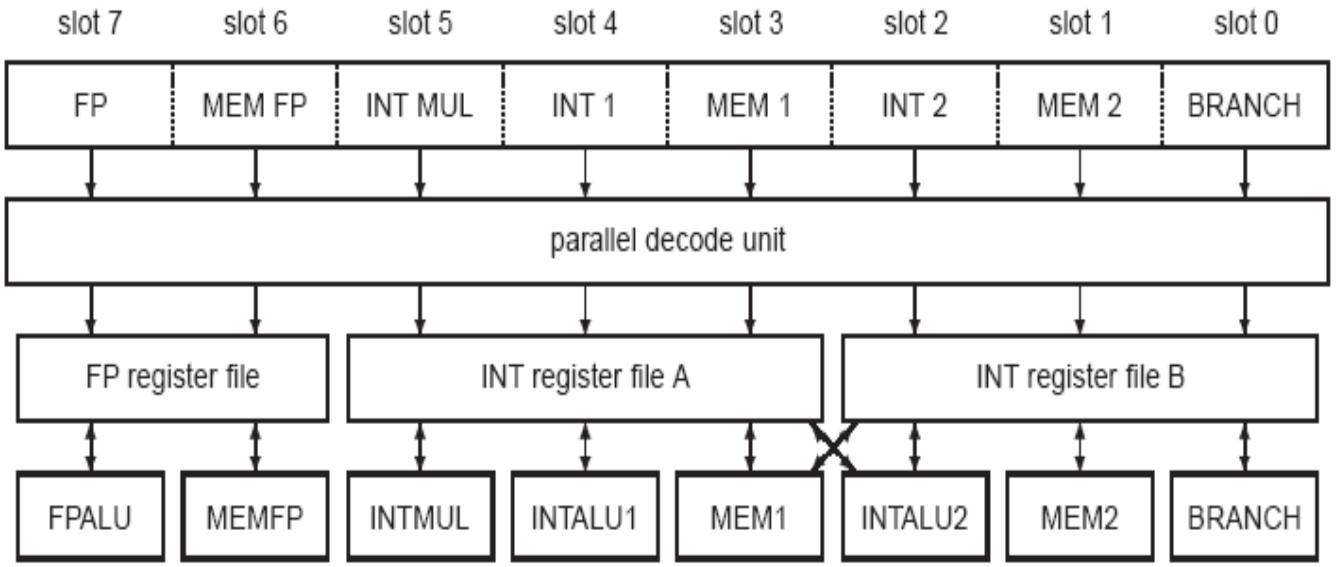


GPP

- EPIC/VLIW Architectures
  - Such a word, once fechted, is decoded in parallel and the single instructions are issued to proper execution units by means of a fixed schema
  - Given such a fixed structure, it is the compiler that has to manage all the scheduling and optimization issues
    - Complex and architecture dependent compilers
      - One-time job!
  - Simple, quite big and quite consuming (between CISC and RISC) architectures
    - Today often used to realize *Digital Signal Processor*

GPP

- EPIC/VLIW Architectures
  - Typical EPIC/VLIW architecture

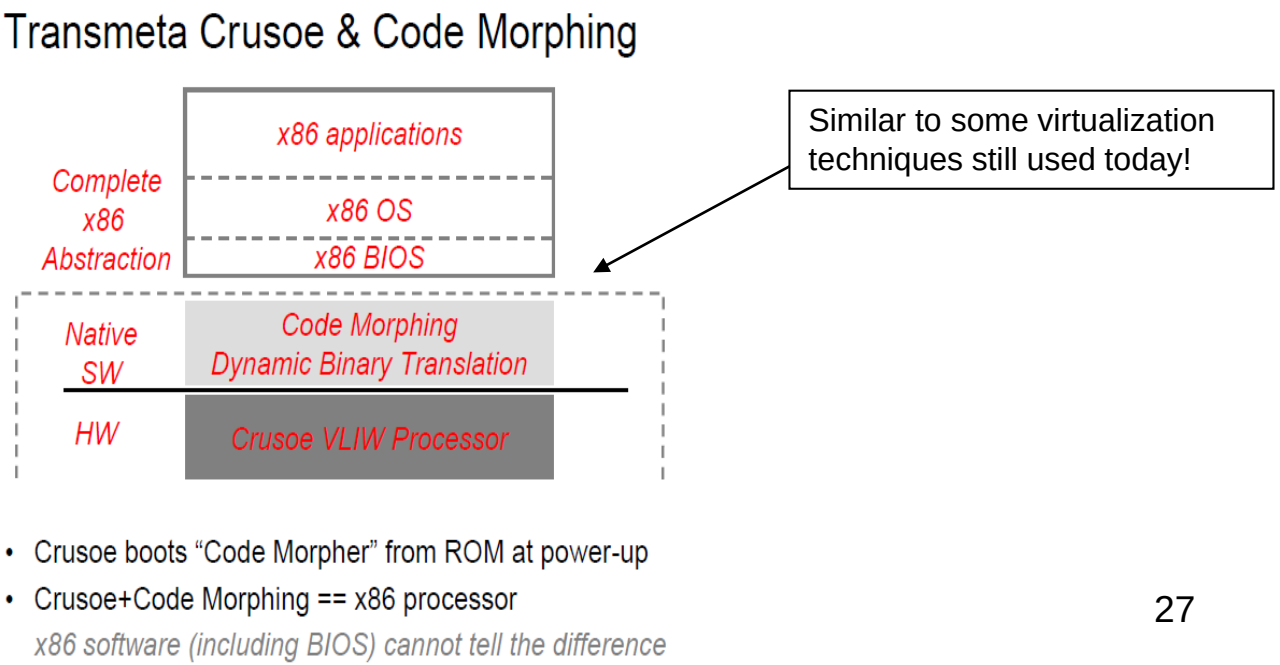


GPP

CISC/VLIW Architectures

GPP

- CISC/VLIW Architectures
  - Similar to CISC/RISC approach but with a VLIW core
    - Famous example at the end of ’90 (maybe the only one?)
      - Transmeta Crusoe (and Efficeon)
        - » Today shoul dbe available only as IP core



Application Specific Processors

ASP

- In the embedded domain there are often specific problems to be solved
  - Not always a GPP is able to provide required performances but the flexibility provided by SW is still needed
- For this, there are a lot of processor architectures designed and optimized for specific domains
  - The focus could be: ISA, internal architecture, on-chip peripherals, size, processing power, power consumption, …
- The main ones are
  - *Digital Signal Processor*, *Network Processor*, *Microcontroller*, *Graphical Processing Unit*, *Multi-Processor System-on-Chip*

ASP

Digital Signal Processor

ASP

- Digital Signal Processor
  - They are one of the most famous ASP
    - Specialized for *Digital Signal Processing*
      - Specialized ISA and internal architecture
  - They have been designed by analyzing the most common basic operations in typical DSP algorithms

$\mathbf{Z} = \mathbf{X} + \mathbf{Y}$	$z_n = x_n + y_n \quad \forall n$
$\mathbf{Z} = k\mathbf{X}$	$z_n = k \cdot x_n \quad \forall n$
$z = \lvert\mathbf{X}\rvert_1$	$z = \sum_n x_n$
$z = \lvert\mathbf{X}\rvert_2$	$z = \sum_n x_n^2$
$z = \mathbf{X} \times \mathbf{Y}$	$z = \sum_n x_n \cdot y_n$
$\mathbf{Z} = \mathbf{X} * \mathbf{Y}$	$z_n = \sum_k x_k \cdot y_{N-k} \quad \forall n$
$\mathbf{Z} = \mathcal{F}(\mathbf{X})$	$z_n = \sum_k x_k e^{\frac{-2\pi i}{N} nk} \quad \forall n$

ASP

- Digital Signal Processor
  - Dominant operation: $z_{t+1} = z_t + x * y$
    - ***MAC : Multiply Accumulate***
      - A good DSP shall optmize its execution

$$y_i = \sum_{k=0}^{n-1} x_{i-k} * a_k$$

$$y_{i,j} = y_{i,j-1} + x_{i-j} * a_j$$

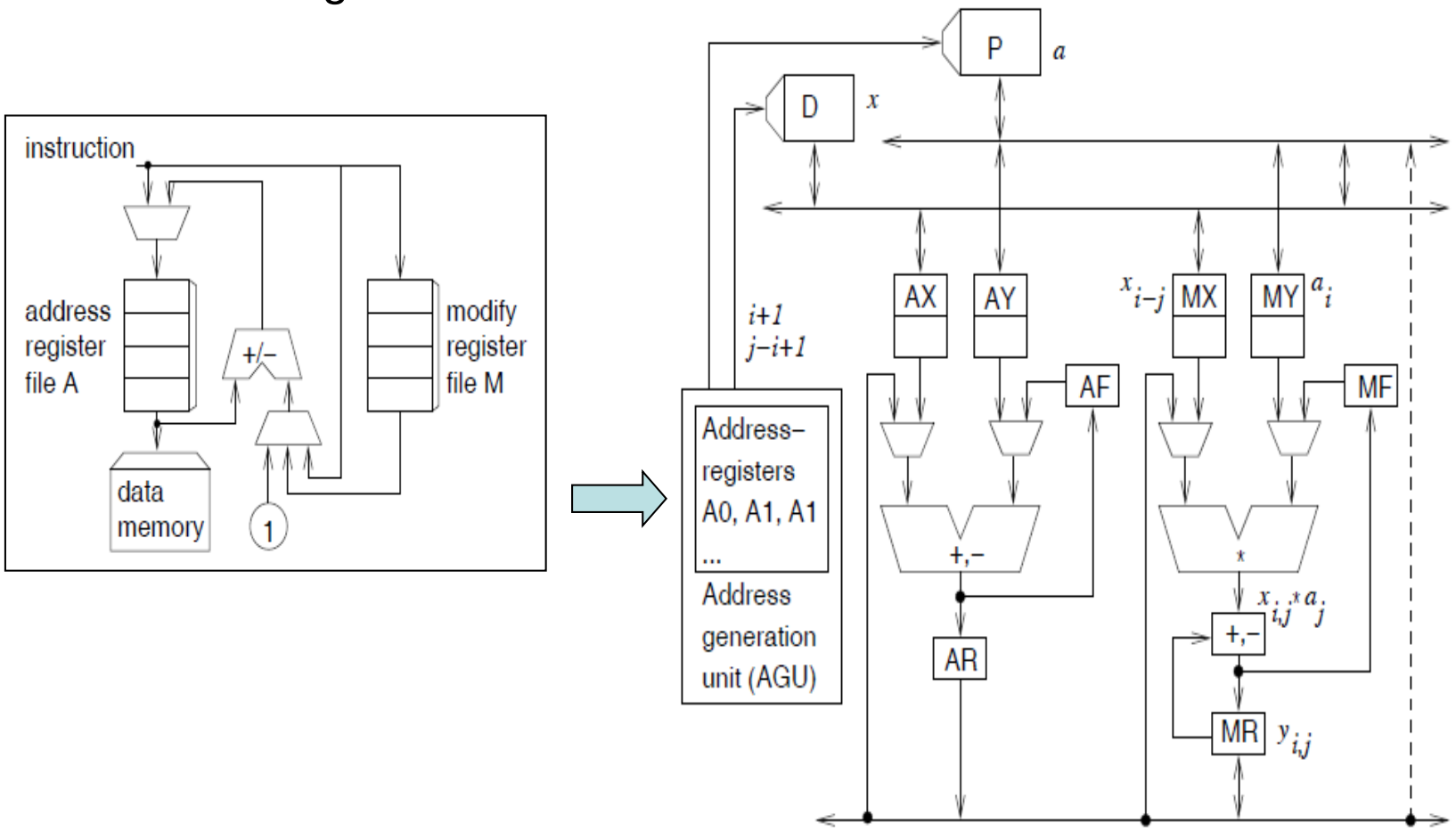
  - Other peculiar features of DSP algorithms
    - A lot of loops with
      - Short body (~10 asm instructions)
      - Loop control variable not modified inside the loop
      - Simple update of the loop control variable
      - Access pattern to the arrays inside the loop quite simple and regular

ASP

- Digital Signal Processor
  - For all this the internal architecture is designed to have
    - Datapath efficient for MAC
    - Dedicated circular buffer to store loop asm instructions
    - Dedicated register to manage the loop control variable
    - Dedicated adder/subtractor to increment/decrement the loop control variable without involving the main ALU
  - Moreover…
    - High-performance memory access
      - High-performance ultra-wide bus (e.g. 256 bit)
      - Enhanced cache hierarchy
        - » e.g. SHARC: 3 L0 cache for data, instructions and constants
    - Often based on VLIW architecture due to high intrinsic parallelism and simple control flow of involved algorithms

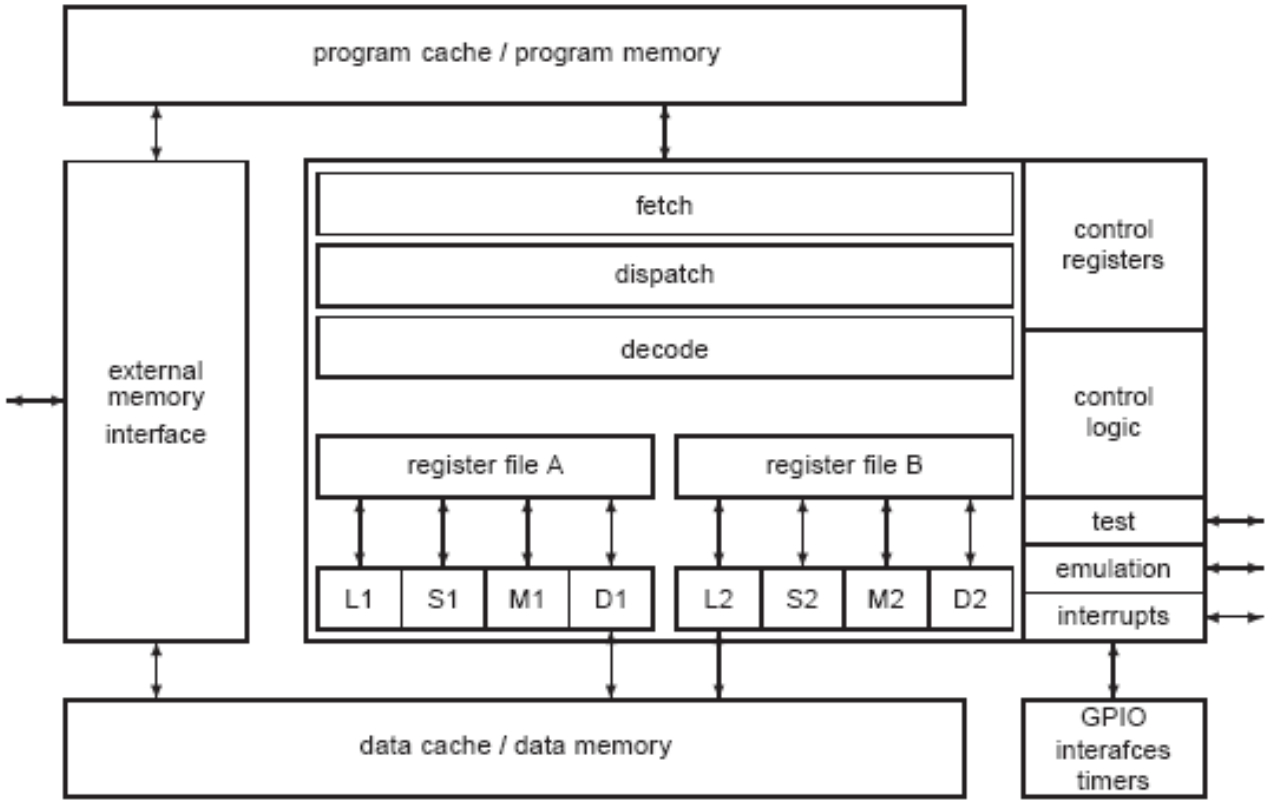
ASP

- Digital Signal Processor
  - Analog Devices ADSP 2100



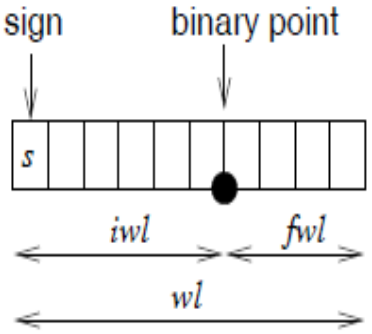
ASP

- Digital Signal Processor
  - Texas Instruments VLIW DSP



ASP

- Digital Signal Processor
  - Other specific DSP features
    - Fixed-point arithmetic



    - Saturating Arithmetic

	0111
+	1001
Standard *wrap-around* arithmetic	10000
saturating arithmetic	1111

ASP

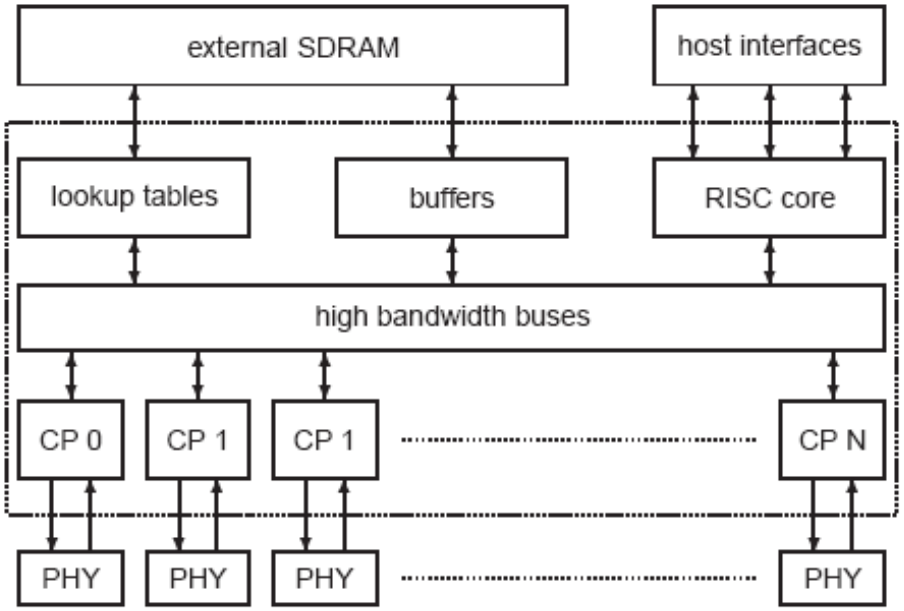
Network Processor

ASP

- Network Processor
  - Architectures dedicated to *packet processing*
    - They are normally complex SoC dedicated to network applications
      - e.g. high-performance routers
  - A typical *packet processing* application is composed of several simple routines that manage a great number of packets
    - Typical operations
      - Management of multiple independent physical channels
        - » Buffering of I/O packets
        - » Data-link headers management
        - » Fields search in IP headers
        - » Address search in look-up tables
        - » Evaluation of CRC codes
        - » Sending and removing packets

ASP

- Network Processor
  - Composed of one or more RISC cores for supervision and management, and several dedicated HW components
    - PP (*Packet Processor*) or CP (*Channel Processor*)
      - I/O interfaces, queues management, fast memories, crypto-cores
        - » Independent (concurrent) physical channels are managed by dedicated hardware units

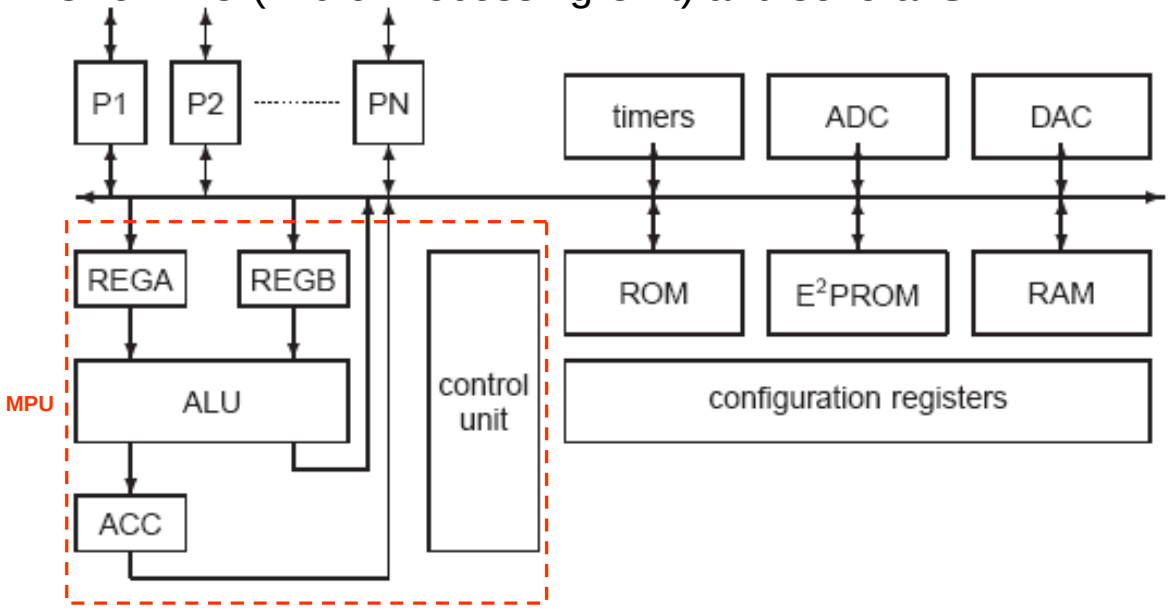


ASP

Microcontrollers

ASP

- Microcontrollers
  - The term *microcontroller* (MCU – *Micro Controller Unit*) refers to a class of processors that integrates a lot of peripherals and interfaces on a single chip
    - They are SoC that allow to solve problems that require a limited computational power but with a lot of NF requirements
      - One MPU (Micro Processing Unit) and several *SPP*



ASP

- Microcontrollers
  - Main features
    - Normally it is missing an interface to a external RAM
      - Short programs and limited pin-out for very simple and cheap PCB
    - To provide a lot of peripheral by keeping a low pin-out it is often used I/O ports multiplexing
      - Configuration registers to map pins with SPP
        - » I2C, SPI, CAN, JTAG, PWM, UART/USART, watchdog, timer, ADC/DAC, comparators, etc.

ASP

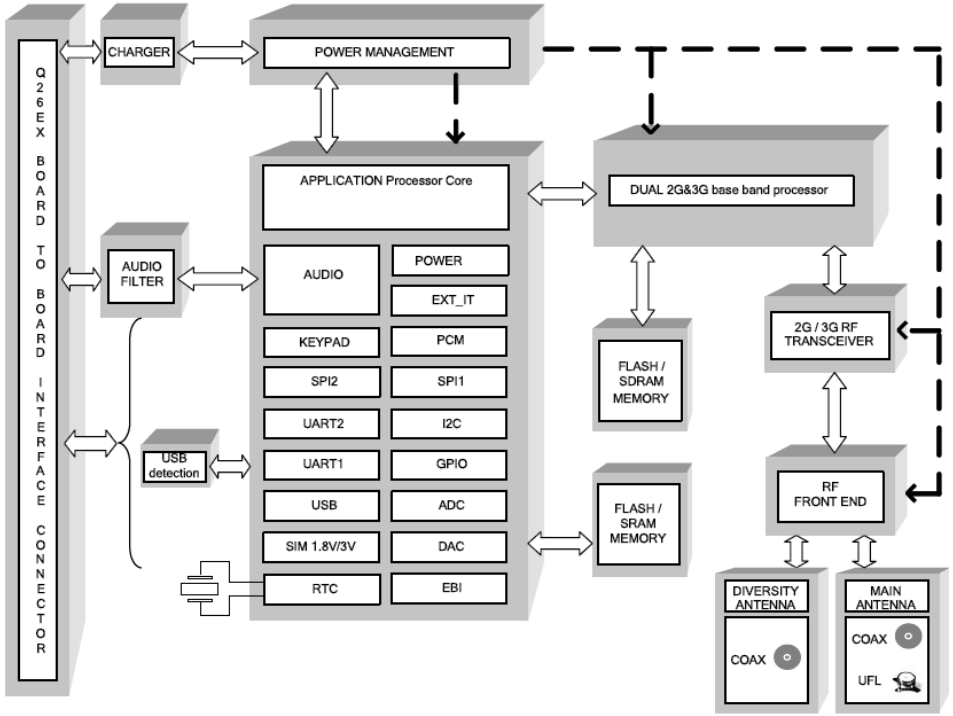
- Microcontrollers
  - “Hystorical” MPU cores
    - Atmel AVR (ISA)
    - TI MSP430 (ISA)
    - Intel 8051 (ISA)
  - ARM MPU cores
  - x86 MPU cores
    - Intel Quark (IA-32)

ASP

- Microcontrollers
  - “Complex” microcontrollers”
    - They integrate also SPP that act as radio transceivers
      - IEEE 802.15.4
        - » TI CC2430/2450, Atmel ZigBit
      - There are also SoC/SoM with BT and/or Wi-Fi
    - Often the MPU is reserved, at least partially, to execute SW related to the protocol stack
      - Sometimes there is a MPU dedicated to this

ASP

- Microcontrollers
  - “Complex” microcontrollers”
    - GPP + SPP + 3G/4G SPP
      - e.g. Sierra Wireless Q26 Extreme



ASP

Graphical Processing Unit

ASP

- Graphical Processing Unit
  - They are ASP dedicated to the management of basic and advanced graphic operations
    - Normally used as graphic co-processor for a GPP but in the last years thare are multi-core GPU that could also used to perform different kind of functions
    - **GPGPU** (**General Purpose Computing on GPU)**
      - » http://gpgpu.org/
      - In GPGPU, GPU is exploited also to perform parallel processing (tipically SIMD) toghether with the main GPP one
        - » ATI: CTM (Close to Metal)
        - » nVidia: CUDA (Compute Unified Device Architecture)
        - » OpenCL (Open Computing Language)

ASP

Multi-Processor SoC (MPSoC)

ASP

- Multi-Processor SoC
  - Further increase of clock rates of processors has recently come to a standstill: the large energy consumption of processors using multi-gigahertz clock speeds is a key reason for this
    - In order to still improve the overall processing power, several processors (in particular more than one GPP/ASP, other than several SPP) must be employed
  - This led to the design of chips comprising multiple (GPP/ASP) processors as well as additional components such as SPP, peripheral devices and memories
    - Systems implemented in that way are called **MPSoC**

ASP

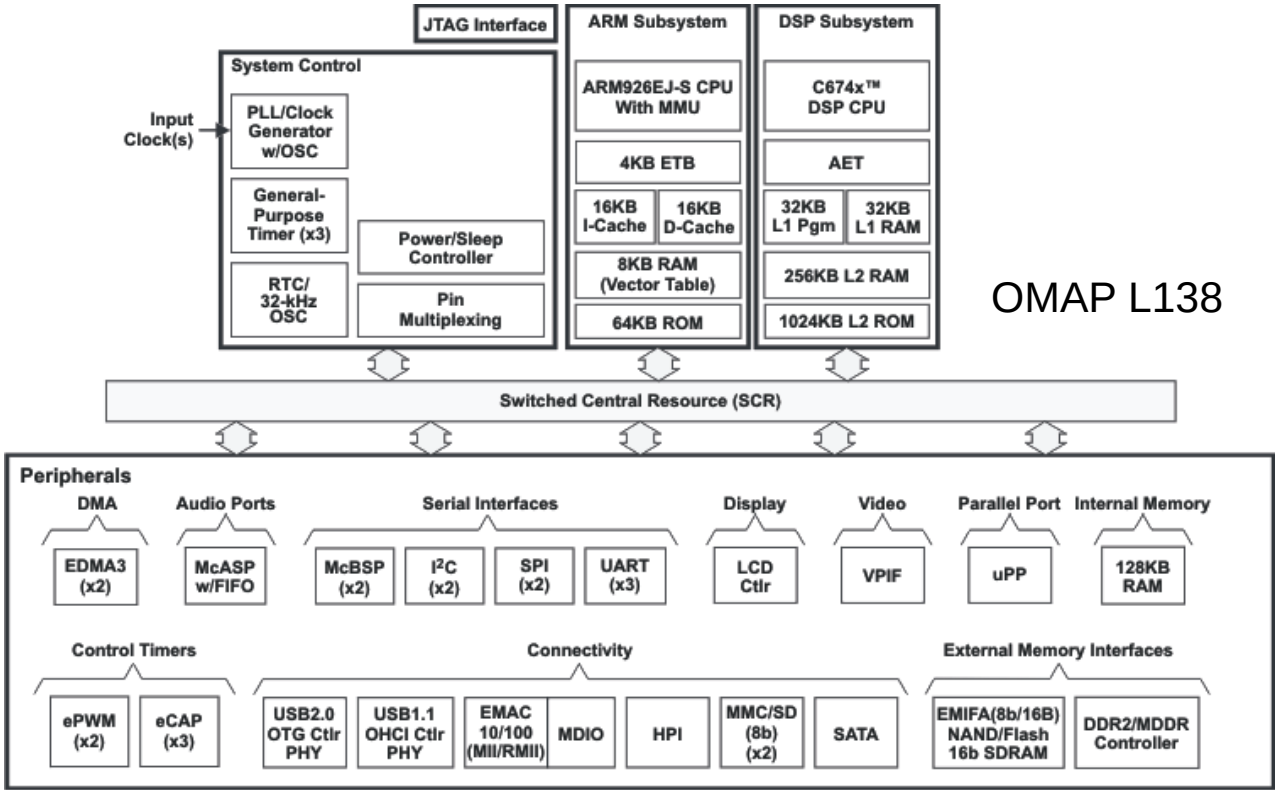
- Multi-Processor SoC
  - For general purpose computing and desktop PCs, multi-processor systems are typically homogeneous: *multi-core*
  - For embedded systems, energy efficiency has top priority and it is typically obtained with highly specialized processors
    - In order to save energy, unused areas are typically powered-down
  - However, exploit such multiprocessor-based systems with applications written in a sequential language is a challenge
    - Mapping techniques and programming support are of key importance
      - *e.g. OpenCL, OpenMP/OpenAMP*
        - » Frameworks for writing programs that execute across homogeneous and/or heterogeneous platforms consisting of GPP, GPU, DSP and other processors (also SPP on FPGA)

ASP

- Multi-Processor SoC
  - DSC (*Digital Signal Controller*)
    - uC + DSP
  - Homogeneous multi-core (GPP)
    - Standard Desktop PC & Workstation
    - General-Purpose/Application-Specific Accelerators
      - Intel Xeon PHY
      - Ramon Chips RC64 (see Extra folder)
  - Heterogeneous multi-core
    - APU (*Advanced/Accelerated Processing Unit*)
      - GPP+DSP/GPU+SPP
      - GPP+SPP+FPGA
      - GPP+SPP+FPGA+AnalogComponets

ASP

- Multi-Processor SoC
  - GPP+DSP+SPP
    - Texas Instruments OMAP Series



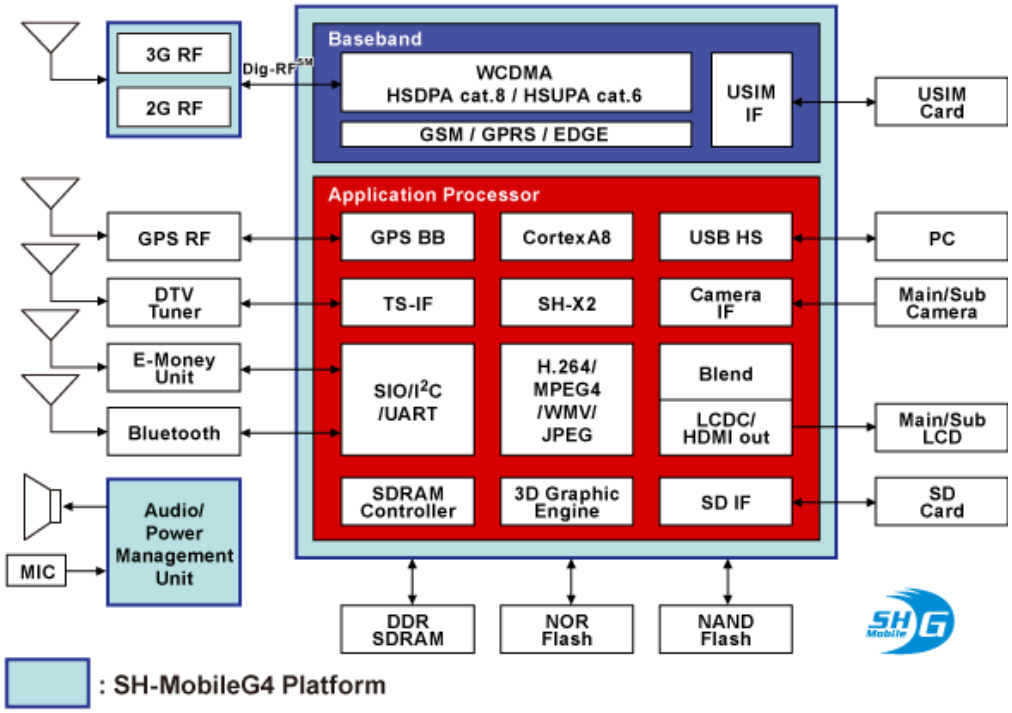
OMAP L138

ASP

- Multi-Processor SoC
  - GPP+GPU+SPP
    - *Single-Board Computer*
      - *Raspberry*
      - *BeagleBoard*
      - *WAND Board*
    - *Netbook, Tablet & SmartPhone*
      - *Often based on ARM o ATOM cores*
        - » Apple/Samsung

ASP

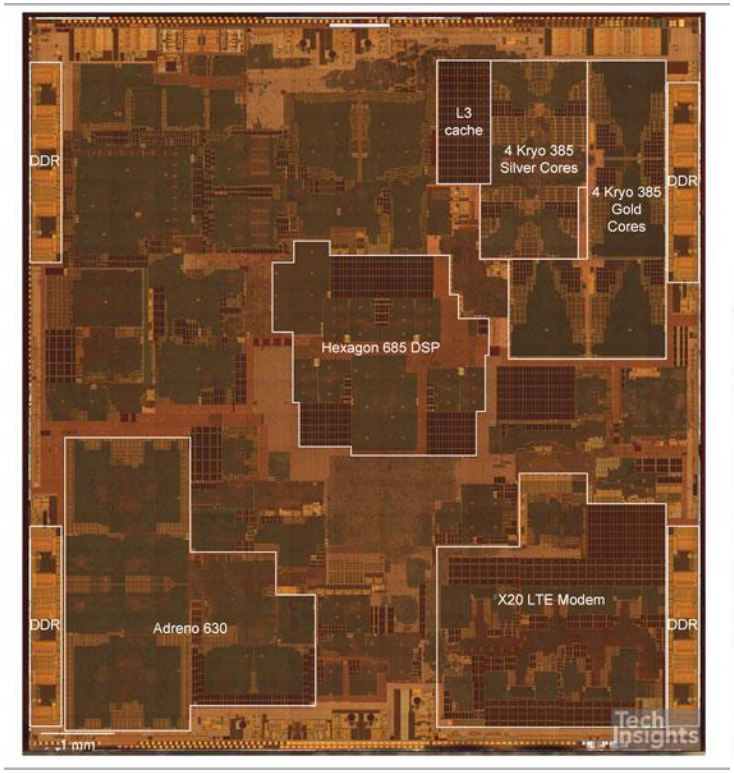
- Multi-Processor SoC
  - GPP+GPU+3G+SPP
    - Renesas SH-Mobile Gx Series

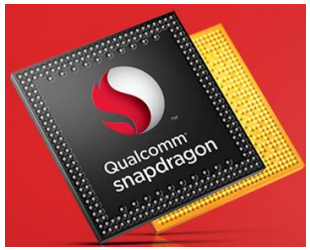


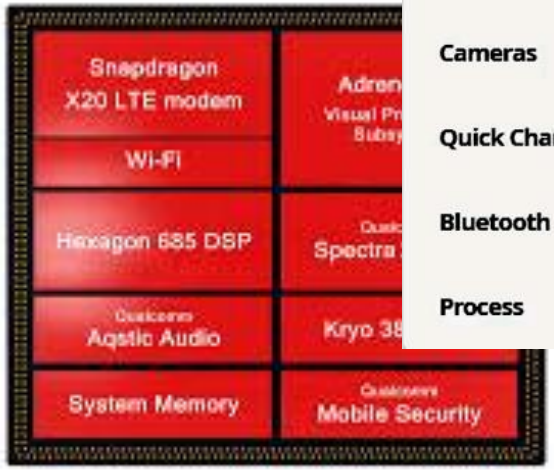
RF: Radio Frequency, MIC: Microphone, GPS: Global Positioning System, DTV: Digital Television, TS: Transport Stream, IF: Interface, SIO: Serial Input, I^2C: Inter-Integrated Circuit, UART: Universal Asynchronous Receiver Transmitter, SDRAM: Synchronous DRAM, DDR: Double Data Rate, MPEG: Moving Picture Experts Group, JPEG: Joint Photographic Experts Group, USIM: Universal Subscriber Identity Module, USB HS: Universal Serial Bus High Speed, LCDC: Liquid Crystal Display Controller, HDMI: High-Definition Multimedia Interface, SD: Secure Digital

ASP

- Multi-Processor SoC
  - GPP+DSP+GPU+SPP
    - Qualcomm Snapdragon 845



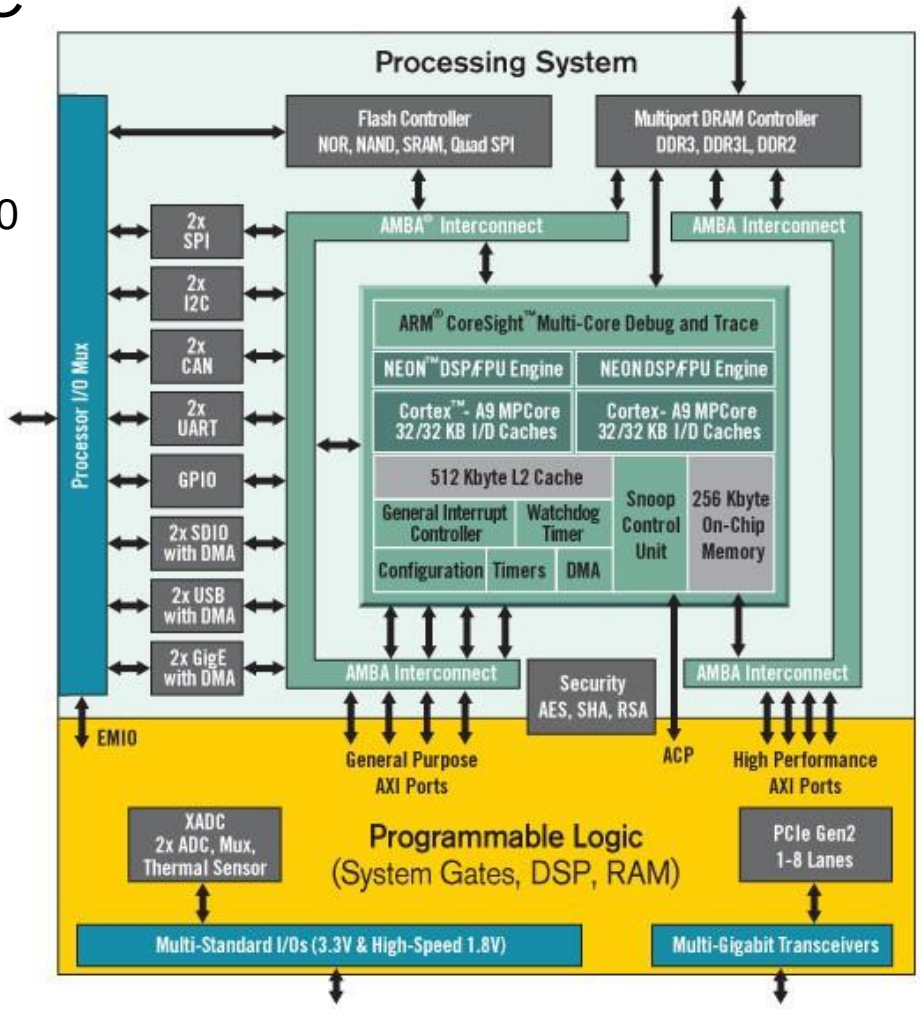




	Snapdragon 845
CPU	4x 2.8GHz Kryo 385 (Cortex-A75) 4x 1.7GHz Kryo 385 (Cortex-A55)
GPU	Adreno 630
RAM	LPDDR4X
DSP	Hexagon 685 with HVX
Modem	X20 LTE 1200Mbps down 150Mbps up
Cameras	32MP single or 16MP dual
Quick Charge	4+
Bluetooth	5.0
Process	10nm LPP FinFET

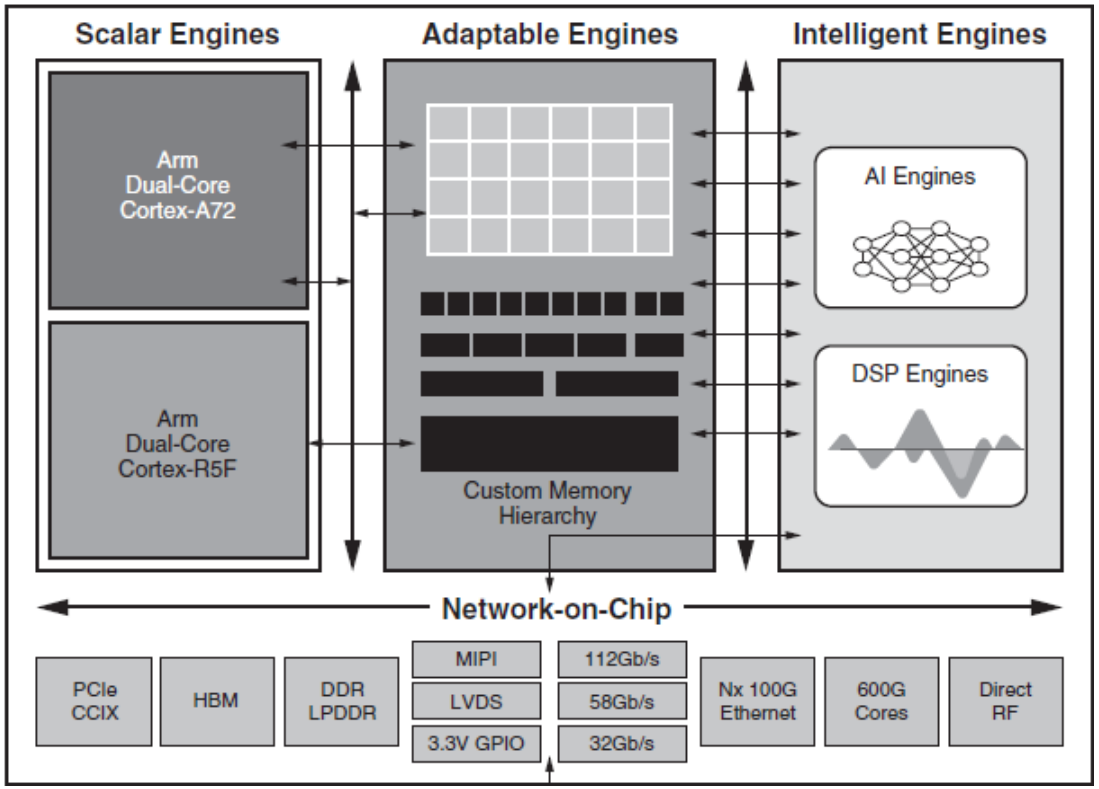
ASP

- Multi-Processor SoC
  - GPP+SPP+FPGA
    - ZedBoard
      - Xilinx Zynq-7000



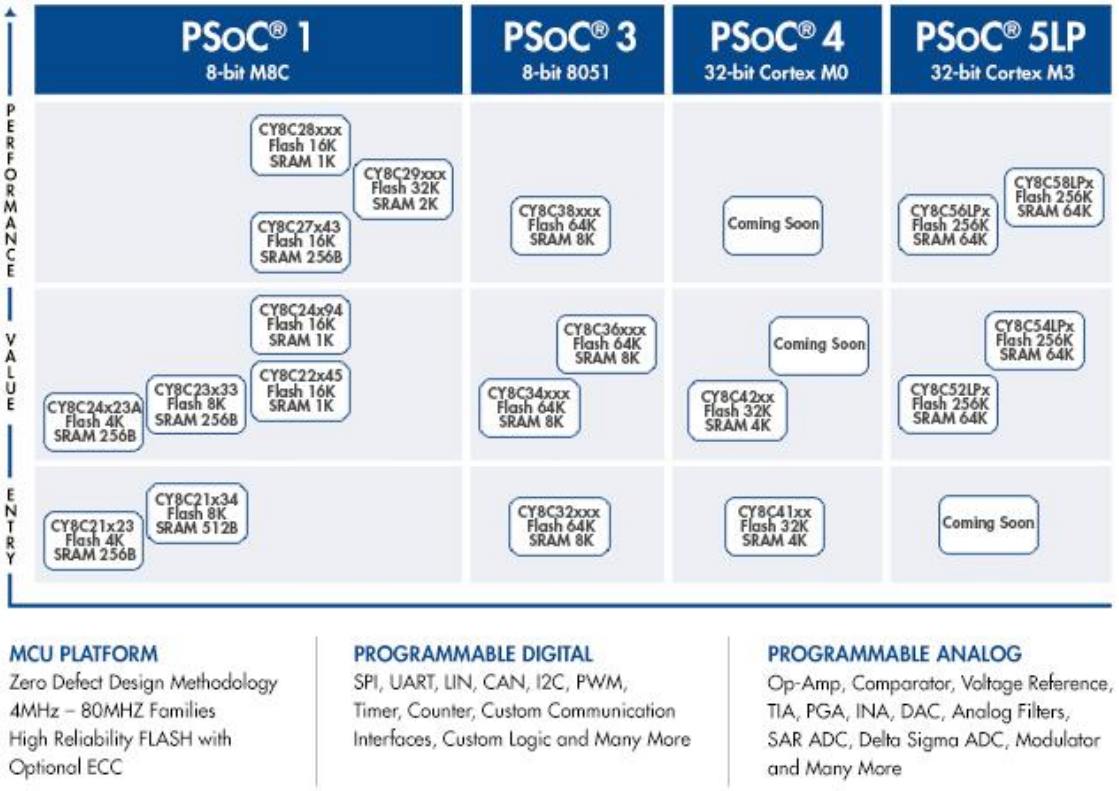
ASP

- Multi-Processor SoC
  - ACAP (*Adaptive Compute Acceleration Platform*)
    - GPP+SPP+FPGA
      - » Xilinx VERSAL (see Extra folder)



ASP

- Multi-Processor SoC
  - GPP+SPP+FPGA+Analog
    - Cypress PSOC family



Conclusions

Conclusions

- The topic is very wide and it is not possible to provide a detailed and complete dissertation
  - The goal has been to provide the basic concepts needed to make choices and comparisons with respect to different requirements