Communications Research Centre Canada
Centre de recherches sur les communications Canada

An Agency of Industry Canada
Un organisme d'Industrie Canada

SDR WAVEFORM PORTABILITY

Steve Bernier, Claude Bélisle, Charles Auger

IQPC Conference

3 December 2003

Canada

CENTRE DE RECHERCHES SUR LES COMMUNICATIONS RESEARCH CENTRE

Software Defined Radio

- **SDR concept provides for a segregation between hardware providers, software developers, and system integrators**
  - Reduces stovepipe acquisition process
  - Facilitates development and distribution of new applications
    - Make use of third party software

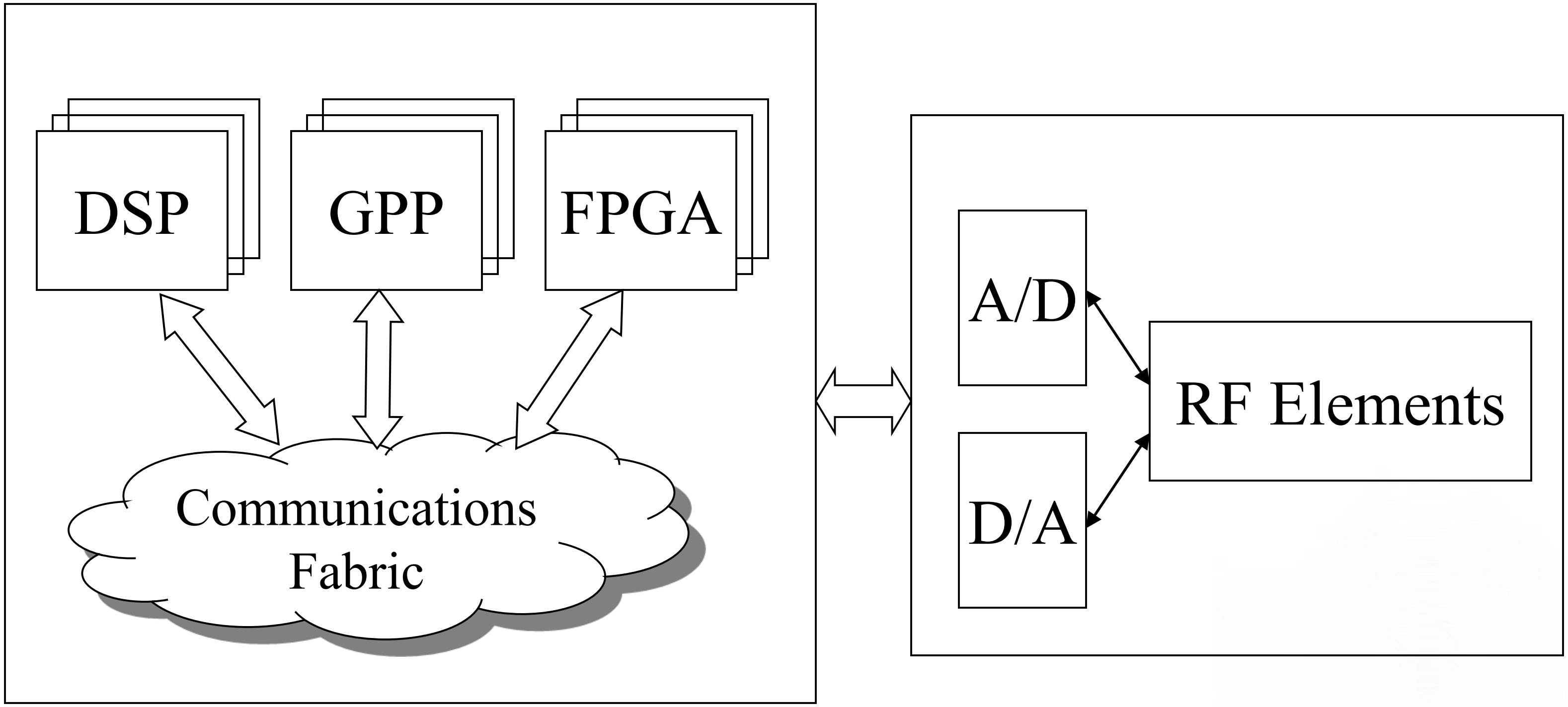
- **Deployment and execution of software on different vendor platforms must be made possible**
  - Software deployment rather than software configuration

- **Application portability becomes essential**
  - With minimum software modifications to minimize cost

Application Portability

- **Current implementations of SDRs do not lend to portability**
  - The three SDR development responsibilities are still tightly integrated
  - Implementation is based on proprietary architectures that uniquely define the roles of hardware providers, software developers and system integrators
  - Limited application expansion possible through COTS software
- **The development of portable applications faces a number of challenges**
  - Heterogeneous digital and RF platforms provided by different vendors
  - Standardization of software development architecture

Platform Configuration



SDR Platform Components

- **SDR platforms are composed of heterogeneous components**
  - **Signal processing components**
    - Digital Signal Processors (DSP)
    - General Purpose Processors (GPP)
    - Field Programmable Gate Arrays (FPGA)
  - **Operating Systems**
    - Multiple vendors
    - Real time vs non-real time
  - **Inter-component communications**
    - Protocols
    - Bus, Star fabric…
  - **RF front end**
    - A/D, D/A, oscillators, filters, antennas

Portability Options

- **Deployment and execution of software on different platforms can be done in a number of ways:**
  - Interpreter with source code (e.g. Postscript)
  - Virtual machine with byte code (e.g. Java)
  - Multiple compile with native code

- **Multiple compile is the only approach that can offer the performance required by modern radio applications**
  - Data rates, modulation formats, error correction, frequency hopping

Portability via Multiple Compile

- **Each application component is compiled for the different platform configurations to be supported**
  - Processing devices
  - Operating systems
- **Provides optimum performance since applications can draw on the full potential of platform components**
  - Not limited to single configuration
- **Software can be ran where it is most efficient, if available. For example:**
  - Synchronization and DDC/DUC on FPGA
  - Filtering and modulation/demodulation on DSPs
  - Error correction and interleaving on GPP

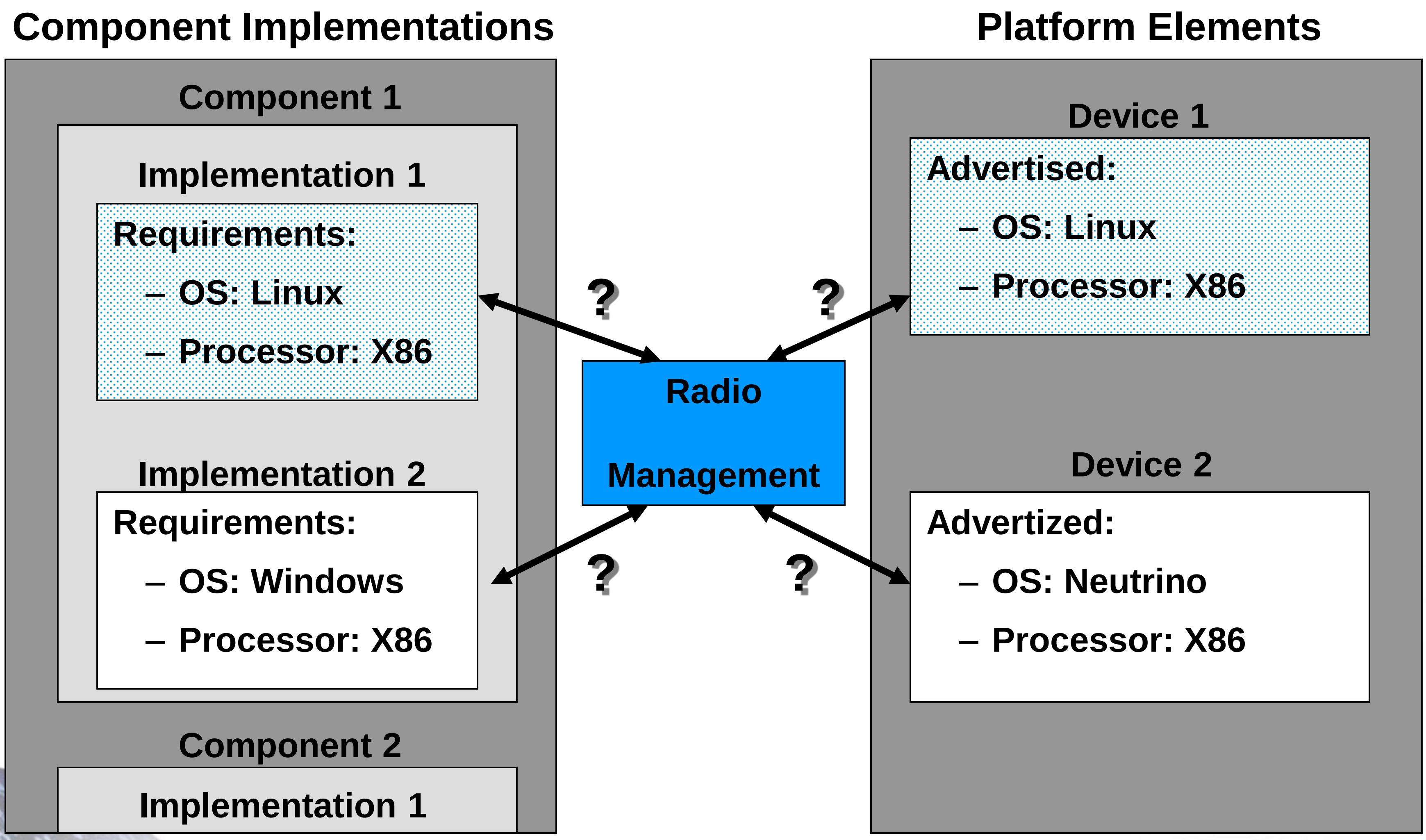
Portability via Multiple Compile (2)

- **Will most likely require different software implementations for different platform configurations**
  - E.g. GPP vs FPGA software

- **A deployment architecture is required to <u>automatically select</u> the proper application component implementation compatible with platform configuration**
  - Comparison between platform capabilities and component implementation requirements

- **Allows hot swap capability**
  - If a device becomes inactive, software can be redeployed elsewhere
  - Increase application reliability

Automatic Component Selection



Standardization for Portability

- **To reduce the development cost of the different component implementations, code reuse should be maximized**

- **This can be achieved with a standard development framework that defines:**
  - **A set of Application Programming Interfaces (API)**
    - API for OS
    - API for access to RF equipment
  - **Communications middleware**
    - Between components provided by different developer categories
  - **Deployment Architecture**
    - Component selection,
    - Application load, initialize, execute

Software Communications Architecture

- **The SCA is a radio framework developed to facilitate portability**
  - **Open Architecture**
    - Based on commercial standards
  - **Created by a consortium of companies**
    - Raytheon, BAE System, ITT, Rockwell Collins, Motorola, Harris…
  - **Improved through an open public change proposal process**
    - http://jtrs.army.mil/
  - **An open source reference implementation exists**
    - http://www.crc.ca/scari

Portability with the SCA

- **The SCA addresses the standardization process with:**
  - **Open specification deployment architecture**
    - Based on CORBA Component Model (CCM)
      - XML assembly descriptor defines application component requirements
      - Performs platform capability and capacity verification
      - Component selection based on component requirements
  - **Application Programming Interfaces**
    - POSIX compliancy for OS APIs
    - Device state management ITU X.731 ISO/IEC 10164-2
    - SCA API Supplement
    - Public submission process for new API
      - SDRF and OMG initiative
  - **Communications Middleware**
    - Minimum CORBA

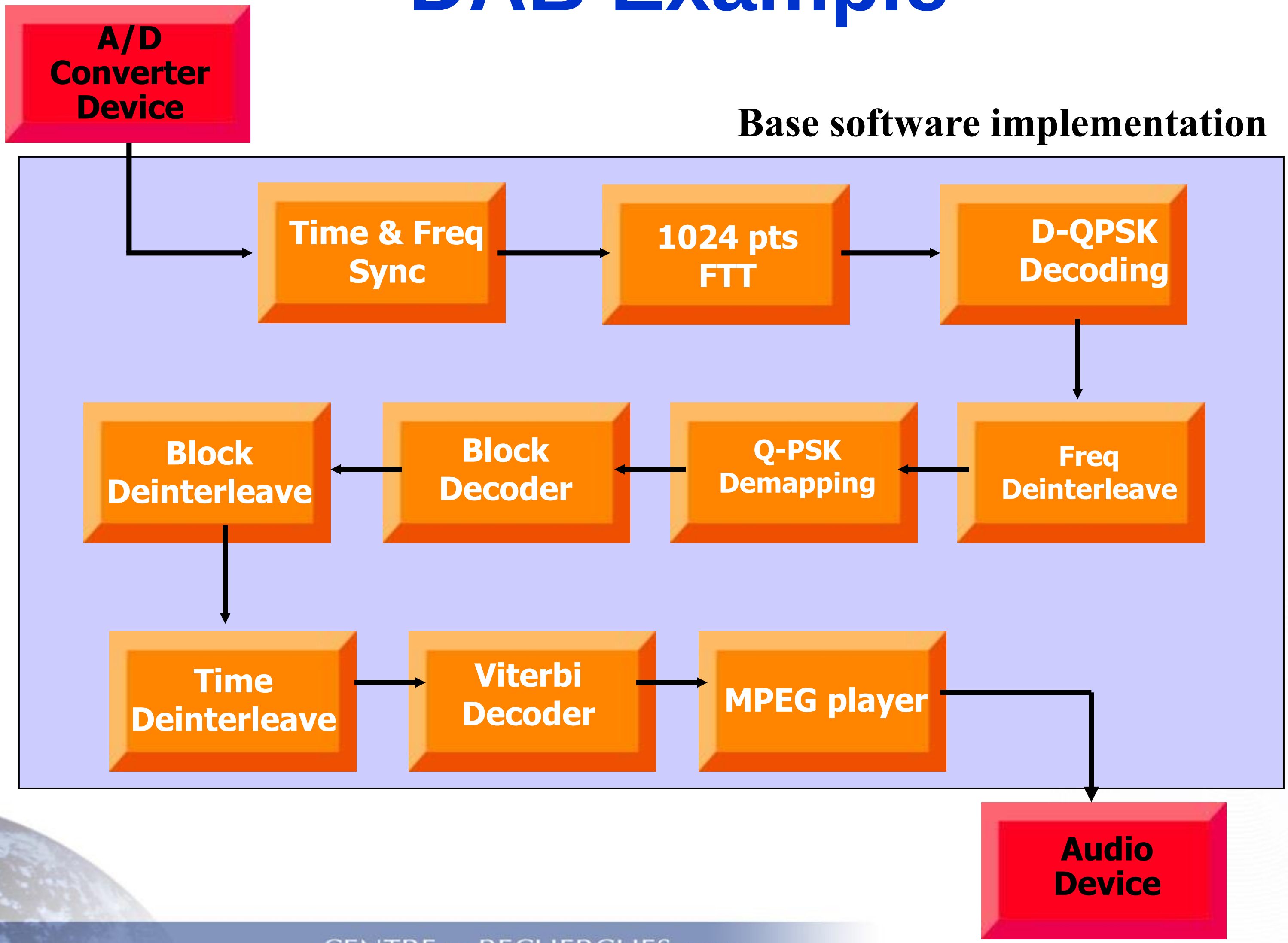
Component Implementation Granularity

- **For ultimate portability, each component should be recompiled for every possible platform element configuration**
  - Various combinations of processors, OS, and middleware !!!
  - Deployment manager selects proper combination

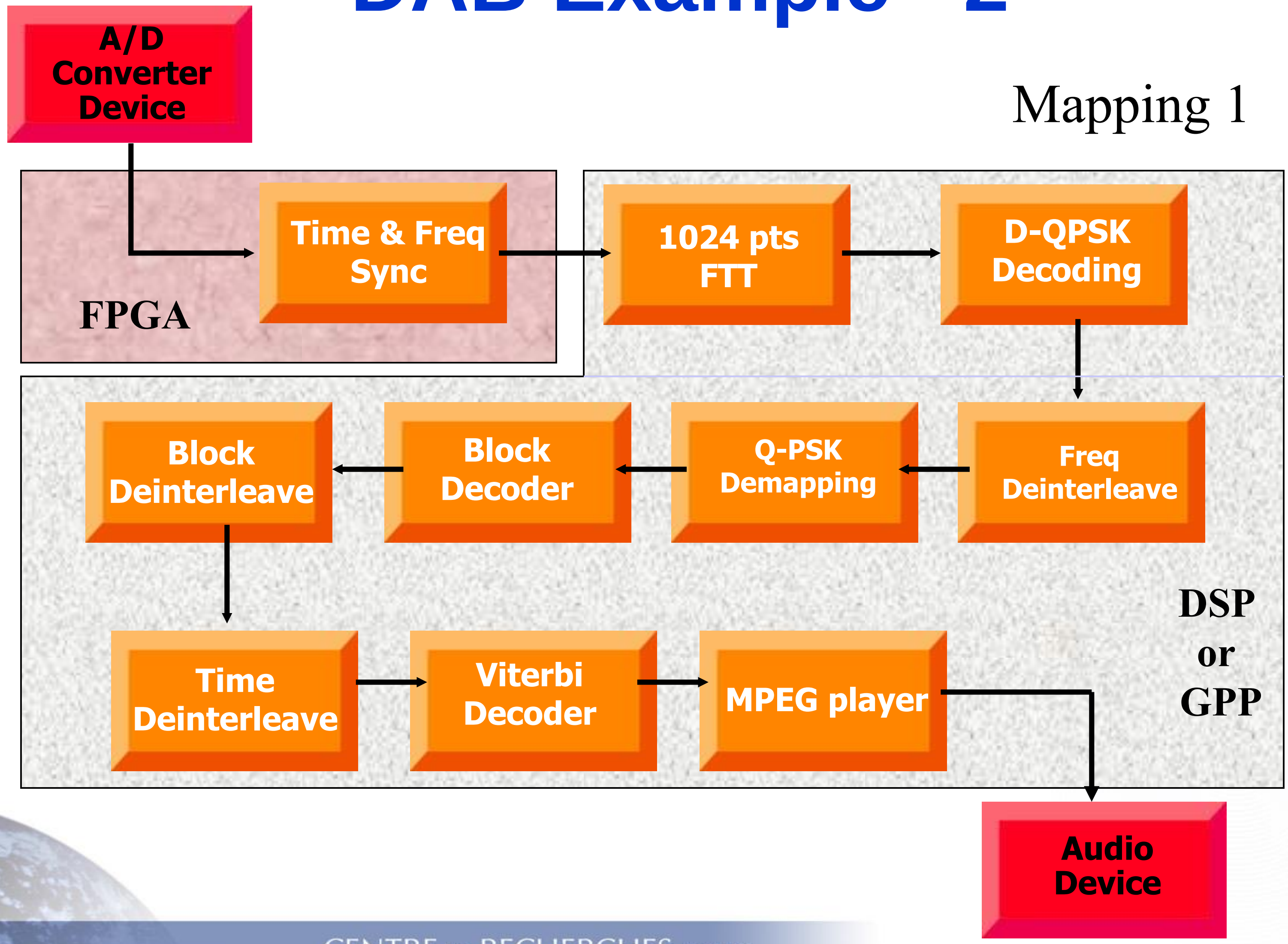
- **When FPGAs are used, a certain level of component aggregation is required**
  - No Dynamic Loader available for FPGAs
  - Components must be combined into a single loadable image
    - otherwise one component per FPGA

- **Implementation granularity depends on FPGA capabilities and radio reconfiguration flexibility required**
  - FPGA image can be composed of many application components providing increasing application performance but decreasing reconfiguration flexibility and increasing development cost

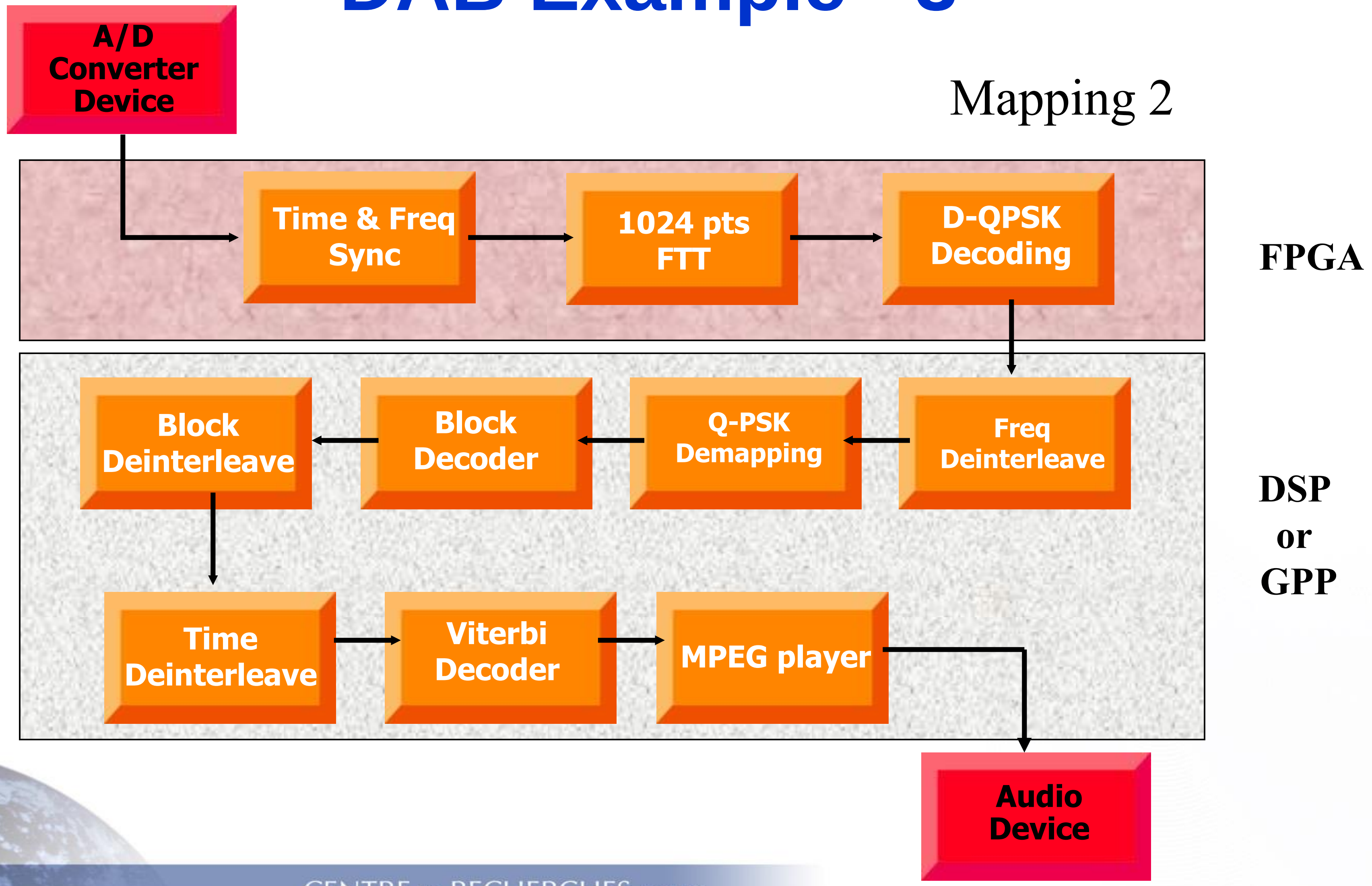
Component Implementation
DAB Example



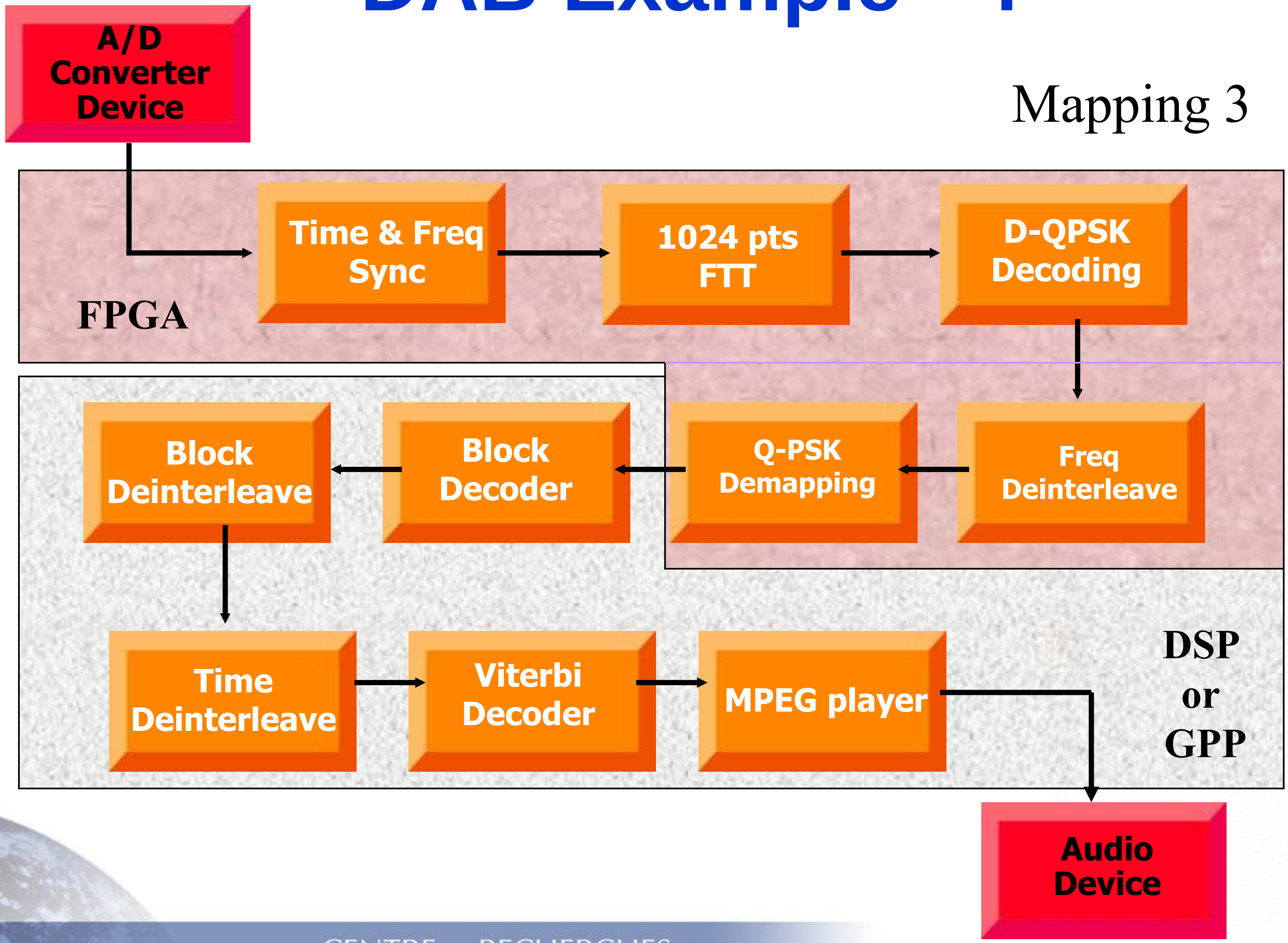
Component Implementation DAB Example - 2



Component Implementation
DAB Example - 3



Component Implementation
DAB Example - 4



Quality of Service

- **In some instances, the platform configuration could support multiple implementations of a same component**
  - Java or C++
  - FPGA or GPP code
- **SCAv2.2 does not offer a QoS mechanism to select best implementation**
  - SCA loads components according to assembly descriptor file
- **Modifications to the SCA is needed**
  - QoS requirements to be included in SAD
- **Tools such as the CRC Waveform Application Builder (WAB), Component Editor and Waveform Optimizer could be used to address QoS requirements**

Software Accelerators

- **While FPGA offer increased performance (processing speed and lower power consumption) over DSP and GPP, current use limits portability**
  - Development cost is increased since FPGA programming is platform specific
  - Optimum granularity level is difficult to estimate

- **A better use of FPGA would be to consider them as a bank of selectable signal processing functions**
  - Similar to math coprocessor, DirectX, MMX

- **Deployment manager compares application component list with Software Accelerator functions provided by the FPGA**
  - When a match is made, FPGA component is used instead of loading DSP or GPP component

Software Accelerators – 2

- Software accelerator concept requires certain modifications to current SDR implementations
- FPGA implementations require the use of an internal data bus to individually address each function and connect them as defined in the application description
- A standard component descriptor is required to identify functions provided by the FPGA

Conclusion

- **Application Portability is an essential element for SDR technology**
  - It is the mean by which true segregation of development roles wil be acheived

- **Multiple compile is most suitable approach for heterogeneous platforms**
  - One implementation per platform element configuration
    - Processor + OS

- **Portability requires a certain level of standardization, offered by the SCA.**
  - Open specification Deployment Architecture
  - Application Programming Interfaces (*)
  - CORBA middleware

- **The concept of Software Accelerator in FPGA should be explored to provide higher application performance without reducing portability**