Software-Based Communications

OBJECT MANAGEMENT GROUP

SW Radio Concepts for Signal Processing Environments

Jimmie Marks
Raytheon
+1-260-429-6422
Jimmie_T_Marks@Raytheon.com

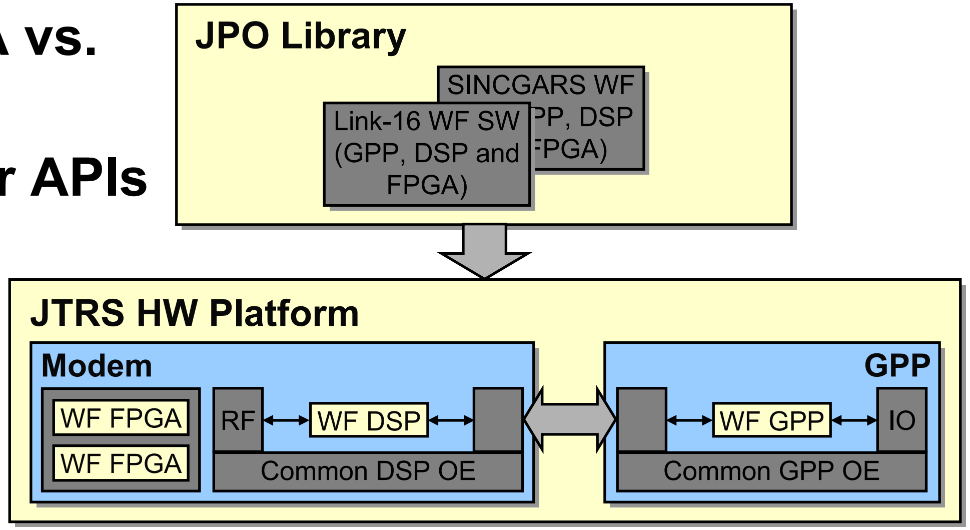
Jerry Bickle
Raytheon
+1-260-429-6280
Gerald_L_Bickle@Raytheon.com

Software Communications Architecture (SCA)

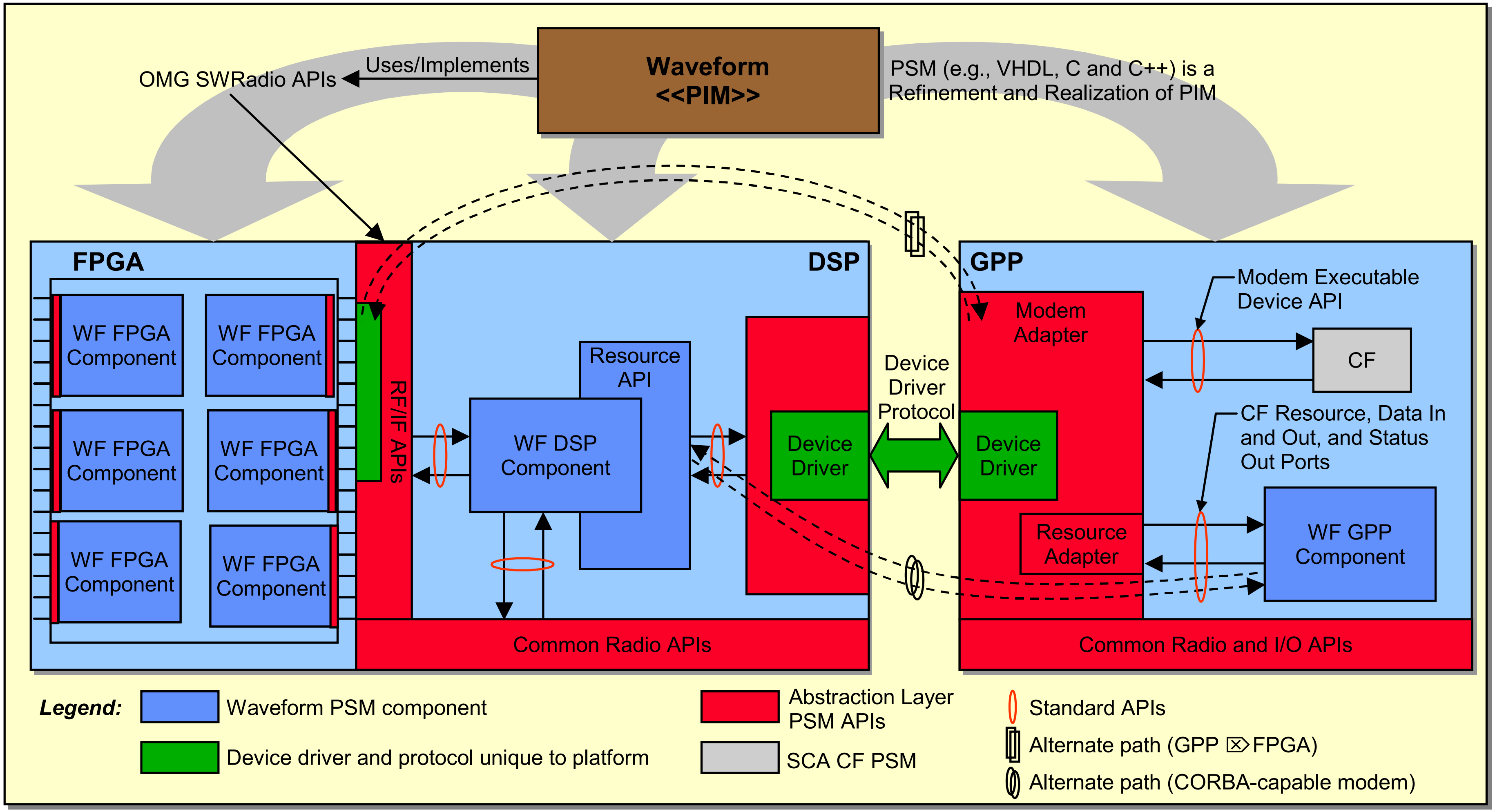
- SCA is a architecture standard to facilitate
  - Common bus technologies
  - Software reuse and portability (Independence of SW from HW)
  - Technology insertion (plug and play of HW and SW)
- Required for all new Gov. radio & communication system procurements
  - SCA v2.2 released Nov 2001
  - SCA v2.2.1 released April 2004
  - SCA v3.0 (TBD) to address Signal Processing SW portability
- SCA is basis of Commercial Standard for Software Defined Radios (SDR)
  - Object Management Group (OMG)
  - Software Defined Radio Forum (SDRF)

Approach to Achieve Portability in Signal Processing

- Overall waveform development process is critical
  - OMG Model Driven Architecture (MDA) approach
  - Platform Specific Model (PSM) as realization of Platform Independent Model (PIM)
- Use Resource Adapters
  - Minimize impact of CORBA vs. non-CORBA environments
- Standardized abstraction layer APIs
  - Supports Portability
- Standardized FPGA design/implementation
  - Produces library for FPGA components

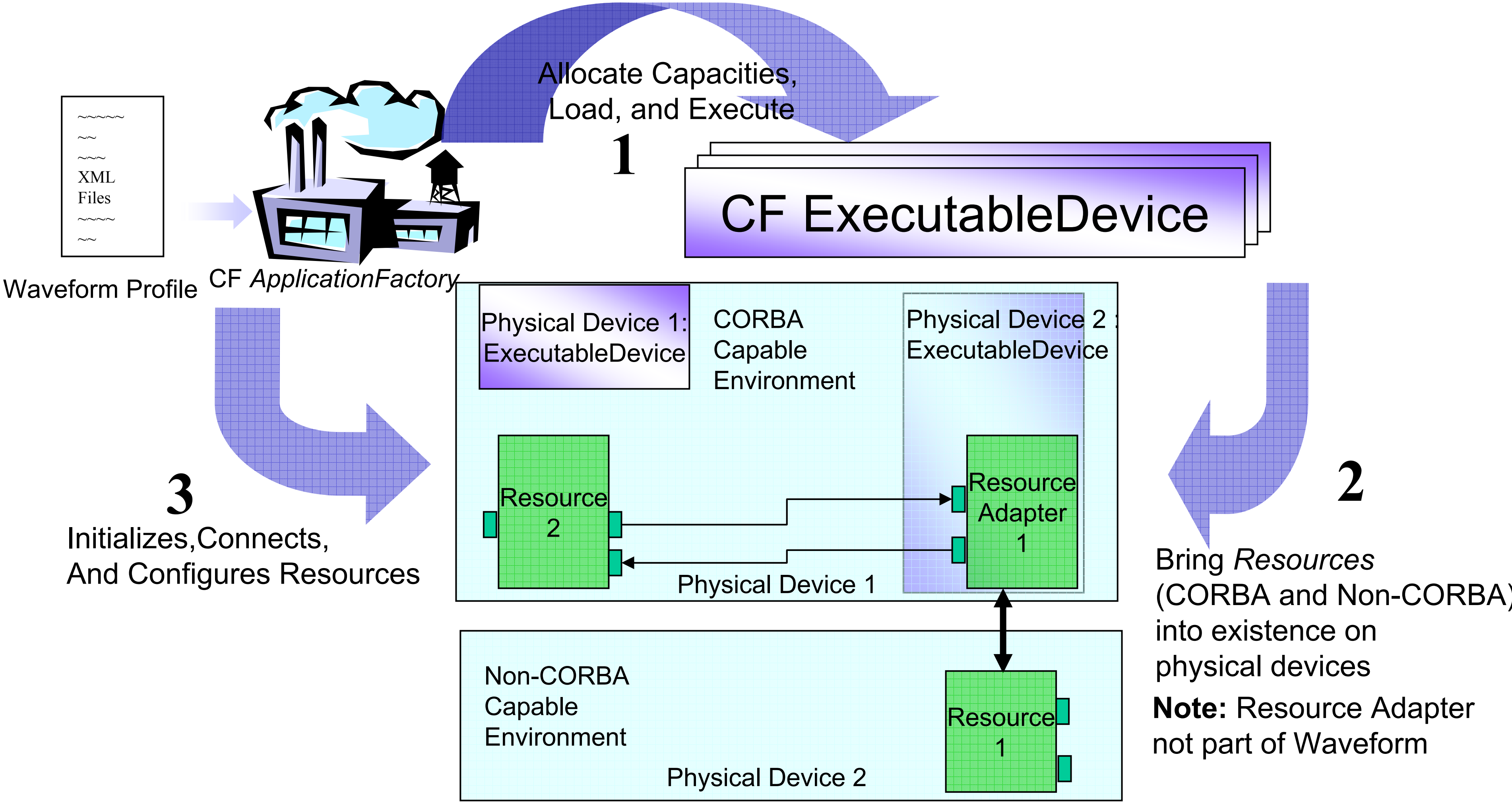


Approach to Achieve Portability in Signal Processing



Standardize critical APIs – allow for architecture changes

Application Deployment with Resource Adapter



Allocate Capacities, Load, and Execute

1

CF ExecutableDevice

Waveform Profile

CF *ApplicationFactory*

~~~~~
~~
~~~
XML
Files
~~~~
~~
Physical Device 1: ExecutableDevice
CORBA Capable Environment
Physical Device 2 : ExecutableDevice
Resource 2
Resource Adapter 1
Physical Device 1
Non-CORBA Capable Environment
Resource 1
Physical Device 2

3

Initializes,Connects, And Configures Resources

2

Bring *Resources* (CORBA and Non-CORBA) into existence on physical devices

**Note:** Resource Adapter not part of Waveform
~~~~~

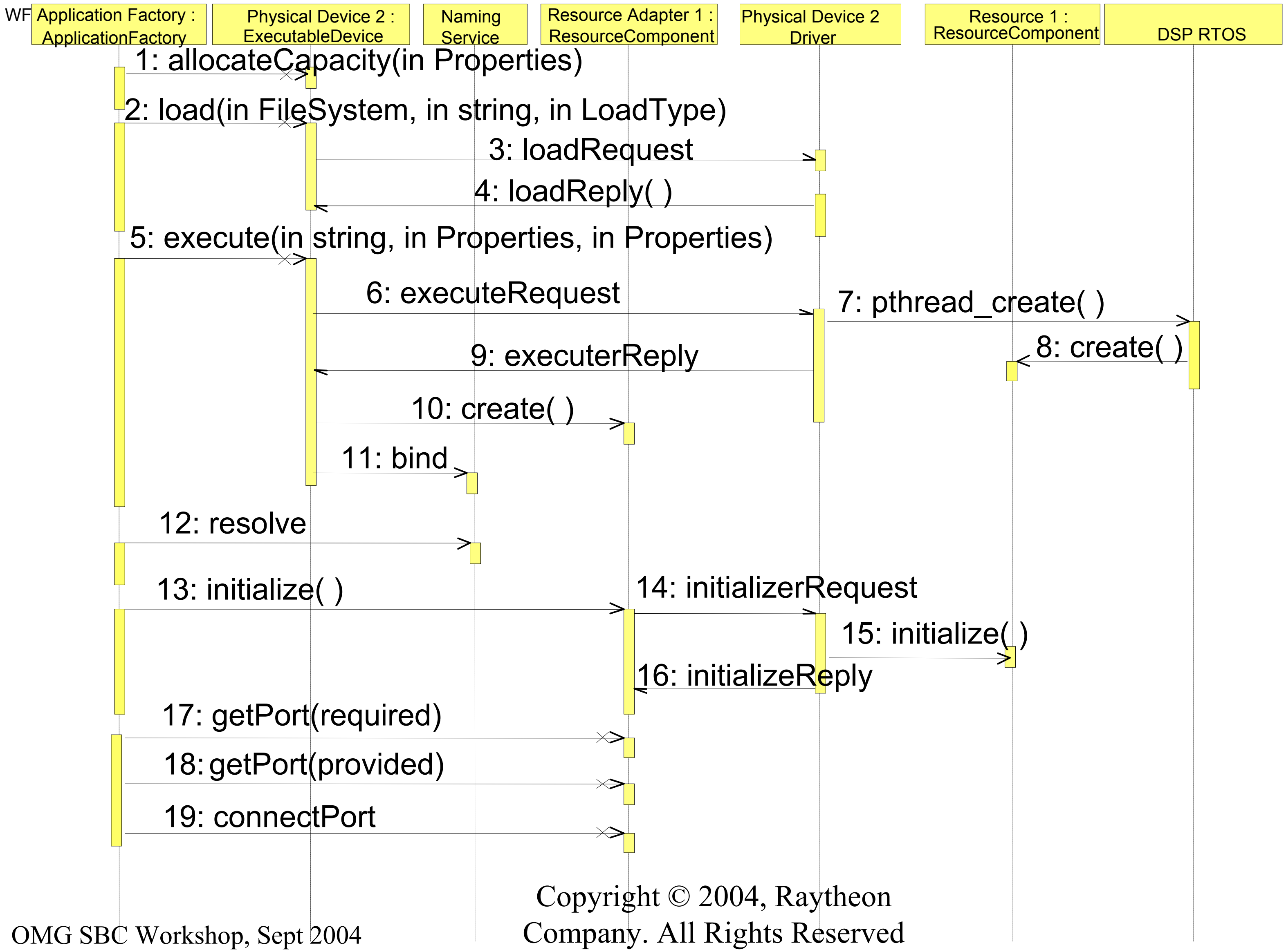
Resource Adapter's ExecutableDevice Characteristics

- Execute, load, terminate, unload, runtest operations delegated to Non-CORBA environment
  - Execute –
    - Creates waveform process/thread
      - Stack Size
      - Priority
      - User Defined Executable Parameters as ID/Value string pairs
    - Creates Resource Adapter
  - Load - loads waveform code
  - Terminate
    - Destroys waveform process/thread
    - Destroys Resource Adapter
  - Unload – unloads waveform code

Resource Adapter
Entry Point Characteristics

- Non-CORBA Environment Implementation Considerations
  - Entry Point Executable parameters
    - Argv format
  - Static Symbol Environment
    - Waveform Process/Thread entry point name is the same for all waveforms to ensure portability of waveforms
  - Dynamic Symbol Environment
    - No restriction on entry point name

Resource Adapter UML Sequence Diagram Illustration



Resource Adapter Characteristics

- Resource Adapter implementation works the same for all waveform applications.
  - Allows for specific waveform properties and payload control properties to be used.
  - Waveform Developers are not involved with specific message definitions and transport
- All Resource Adapter operations are delegated to Resource (e.g., Resource 1) on the non-CORBA Environment except for port operations
- PropertySet Interface (config & query ops)
  - Only specific types are supported
    - Boolean, string, integer (16, 32), unsigned integer (16, 32), octet
  - Property Identifiers are integer strings to enable efficient property processing in signal processing environment.

Resource Adapter Characteristics, cont’d

- Resource Adapter Ports
  - Data Ports
    - Data In Port and Data Out Port
    - Both based upon the SCA Packet Building Block
      - Payload is a Octet Sequence Type
      - Control type is Properties type
        - Sequence of ID/value pairs
        - Types for values are restricted to Boolean, string, integer (16, 32), unsigned integer (16, 32), octet
  - Status Out Port
    - Based upon SCA SignalError Building Block
      - Error Type is integer 16 Type

Resource APIs for non-CORBA “C” PSM

- Resource PIM Operations to Resource “C” PSM
  - initialize(): {raises = (InitializeError)}
    - int initialize(void)   -- returns int InitializeError
  - releaseObject(): {raises = (ReleaseError)}
    - int releaseObject(void) -- returns int ReleaseError
  - start(): {raises = (StartError)}
    - int start(void)   -- returns int StartError
  - stop(): {raises = (StopError)}
    - int stop(void)   -- returns int StopError
  - configure(in configProperties: Properties) : {raises=(InvalidConfiguration, PartialConfiguration)}
    - int configure(Properties configProperties,   int *nbconfigProp);

Resource APIs for non-CORBA “C” PSM, cont’d

- Resource PIM Operations to Resource “C” PSM
  - query(inout configProperties: Properties): {raises = (UnknownProperties)}
    - int query(Properties configProperties,
int *nbconfigProp);
  - runTest(in testId:unsigned long, inout testValues:Properties):{raises=(UnknownTest, UnknownProperties)}
    - int runTest(unsigned long testId,
Properties testValues,
int *nbtestValues);
      - Return integer status indicates type of exception

Resource Port APIs for non-CORBA "C" PSM

- **Data Ports PIM to C language PSM mapping**
  - **PIM Operation**
    - **pushPacket (in control : Properties, in payload : OctetSequence)**
  - **Data In Port**
    - **int MWF_pushPacket(Properties control,
int nbctrl, unsigned long size, Uchar * buffer);**
  - **Data Out Port**
    - **int MDM_pushPacket (Properties control,
int nbctrl, unsigned long size, Uchar * buffer);**
  - **Status Port**
    - **void signalError (short errorDetails);**

Abstraction Layer DSP OS APIs

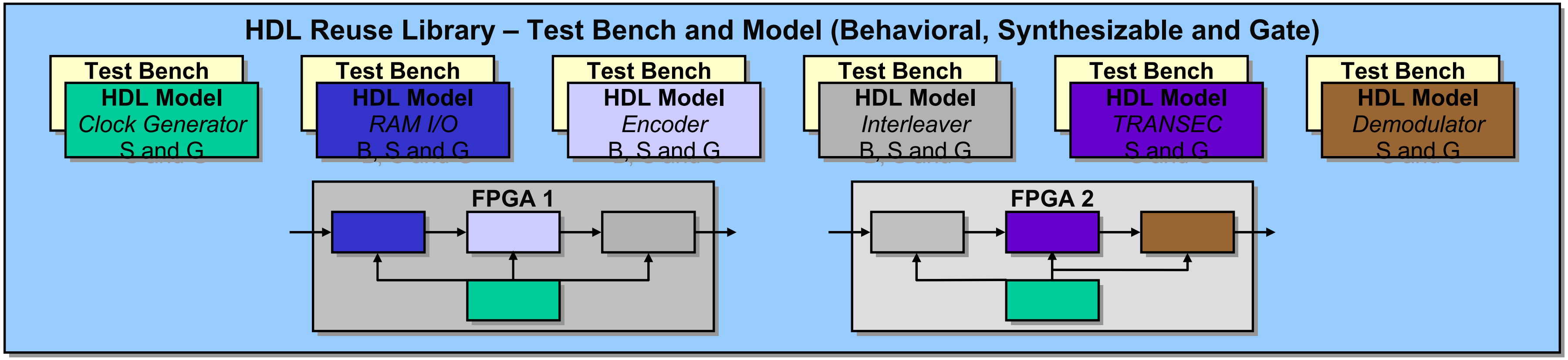
- DSP Application Environment Profile (AEP) defines required Operation System (OS) APIs for DSP environments
  - Work in progress that is currently being evaluated for standardization through the SCA
  - Standards used in whole or partially
    - DSP AEP C Standard (ISO/IEC 9899:1990 - Partial
    - POSIX.1 (ISO/IEC 9945 -1):1997 - Partial
    - POSIX.1b (ISO/IEC 9945 -1):1997 - Partial
    - POSIX.1c (ISO/IEC 9945 -1):1997 – Partial
- Current AEP major units of functionality include
  - Semaphores
  - Timer support
  - Pthreads

Abstraction Layer SW Radio APIs

- Common radio APIs
  - Provides common service definitions that are applicable for all applications
  - File services, OMG lightweight services (log, event, naming, etc.)
  - Operating system APIs (RT POSIX subset)
- Common layer APIs
  - Provides interfaces that cross-cut through facilities that correlate to layers
  - Protocol data unit, error control, flow control, measurement, QOS and stream facilities
- Physical layer APIs
  - Modem APIs include all digital signal processing elements required to convert bits into symbols and vice versa
  - RF/IF APIs provide configuration and control of basic devices (e.g., antenna, amplifier) of physical channel
  - I/O APIs Defines configuration properties for audio and serial facilities

FPGA Portability

- Portability is keyed to the development methodology employed
  - All design components must be designed to be portable
  - Capability & potential demonstrated by IP-Cores for FPGAs
- Portability enhanced by tool & technology independence
  - Design languages allow for it (e.g., VHDL)
- Portability enhanced by levels of design abstraction
  - Abstract behavioral, synthesizable, and silicon-tied gate level
  - Fully parameterized models, protocols, interfaces (API) are required
  - Allows for free interchange, re-use of modules, quick reconfiguration, plug & play design, and rapid prototyping



References

- SCA
  - **http://jtrs.army.mil/sections/technicalinformation/fset_technical_sca.html**
- OMG SWRadio Submission
  - **http://www.omg.org/docs/dtc/04-05-04.pdf**
- SDRF
  - **http://www.sdrforum.org/index.html**