Computer Systems, Inc.
MERCURY

Design Considerations

FM3TR Waveform Reference Implementation

SDR Forum Contract

March 23, 2007

Revision 1.0

Table of Contents

1 Introduction

This document describes the important design decisions and architectural choices made in the implementation of the FM3TR waveform.

2 Waveform Components

2.1 CF Resource Base

"CF Resource Base" library implements the functionality common to all waveform components. Using such an approach increases code reusability and reduces development time. For example, when writing new components, waveform developers can choose to use the generic property set implementation in this library, or implement their own. If a waveform component inherits CF__Resource_impl class, this default library is used. If you do not want to use this library, simply inherit POA_CF::Resource instead.

2.2 Base Assembly Controller

The Assembly Controller is responsible for the control and configuration of all waveform components (Resource) in an SCA compliant waveform. The Base Assembly Controller component implements a set of generic functionality that can be used in multiple waveforms. Both audio and data waveforms use the same assembly controller component.

SCA Requirement 3.1.3.2.1.5 states that the Application should delegate runTest(), start(), stop(), configure() and query() operations to the assembly controller. Therefore, the assembly controller needs to have the references to all waveform components. This information is provided to the assembly controller by defining connections between each component and the assembly controller in the SAD.

SCA Requirement 3.1.3.2.2.5.1.3 states that the Application Factory's create() operation will only configure the application's assembly controller component. Therefore, the assembly controller needs to know each configurable property of each waveform component. Additionally, for each configurable property of each waveform component, a configurable property should also be defined for the assembly controller and a mapping between the waveform component property and the assembly controller property should be created. The BaseAssemblyControllerPropMappingDB.h defines this mapping.

Therefore, if you want to add a new configurable property to one of the waveform components, you will need to add a configurable property to the assembly controller as well and update the mapping definition file to show the link between the two properties. After that, the assembly controller can route configure and query requests to the appropriate components.

Due to the same reasons, SAD level property settings should only be done by overriding the assembly controller's properties. An example of this can be seen in the SAD for the data mode.

2.3 Audio Resource versus Audio Device

The implementation of the audio input and output functionality is hardware specific due to the differences in various sound cards and operating systems. Additionally, the SCA Application Environment Profile does not define functions to access sound cards. Therefore, it is preferable to implement such functionality as an Audio Device. However, the FM3TR waveform implementation team chose to implement this functionality as a waveform component (Resource) to be able to share the source code with the user community. Users can easily turn this implementation into a device implementation if required.

3 Component Deployment

3.1 Resource Loader

The SCA specification allows waveform developers to implement waveform components as executables or shared libraries. The FM3TR waveform reference implementation uses an approach that shows the benefits and costs of both approaches. The waveform components are implemented as shared libraries (i.e. dll files). A program called the Resource Loader loads these dll files and instantiates the components using the code entry points. In other words, the Resource Loader acts as a wrapper to turn shared library waveform components into executables.

Even though the Resource Loader is not a part of the waveform, the source code is provided to allow developers understand and modify the functionality. Users can choose to include this functionality in their own executable devices or merge it with the waveform components to create executable components. Using the Resource Loader, the FM3TR waveform components can be deployed on any executable device.

3.2 Pentium Device

The FM3TR project contains the binaries for an executable device implementation called the Pentium Device. Users can also choose to use their own executable devices to run the FM3TR waveform components. When the provided startup script is executed, the Pentium Device finds the SCARI-Open Domain Manager from the Naming Service, obtains a list of Device Managers, and registers itself to the first available Device Manager. After registration, the SCARI-Open Application Factory can use the Pentium Device to deploy the waveform components.

If you would like to connect the Pentium Device to the Event Channel, add the following text into the SCARI-Open Node1 DCD file. Alternatively, you can replace the existing SCARI-Open DCD file with the one provided in the xml/PentiumDevice folder. After the connection is added to the DCD file, the Pentium Device should report that a connection to the Event Channel is established during boot-up. You can also view the incoming and outgoing events using the SDR Control Room tool.

This is an optional connection that is not required for the Pentium Device to function properly.

```
      <connectinterface id="DCE:e30aa656-468f-4230-a90d-28188fd23116">
        <usesport>
           <usesidentifier>PentiumDevice_IDM_Event_Consumer</usesidentifier>
           <componentinstantiationref refid="DCE:b349a150-0dae-4119-81a8-
045903e33b15"/>
        </usesport>
        <findby>
           <domainfinder type="eventchannel" name="IDM_Channel"/>
        </findby>
     </connectinterface>
```

3.3 Code Entry Point

The SCA specification does not provide a standardized way to communicate the value of the entry point information in SPD (unlike stack size and priority which are passed with the IDs PRIORITY_ID and STACKSIZE_ID). Therefore, the FM3TR waveforms use an executable type property to pass this information.

The SCARI-Open source code can be modified to pass along this information. If you would like to try this, add the following code to the execute() method of the DeployedComponentLauncher.java (before the invocation to execute at line# 817):

```
any = org.omg.CORBA.ORB.init().create_any();

any.insert_string(implementation.getEntryPoint());

options.add(new SCA.CF.DataType("DLL_ENTRY_POINT", any));
```

The above code is provided by Communications Research Center of Canada (CRC) – the developers of the SCARI-Open.

4 Operating System Abstraction (SCEOS)

The SCEOS library provides an operating system abstraction layer to improve portability. The library provided as a part of this project contains the Windows implementations of common OS functions. Users can choose to add additional implementations of the same functions to support other operating system types.

5 Log Service

Since the SCA specification adopted OMG's Lightweight Log Service in version 2.2.1, the Log Service used in SCARI-Open (version 2.2) is a legacy interface. As a result of this transition, only one component of the FM3TR waveform (CVSD Encoder) uses the SCA 2.2 log service to serve as an example on how to implement a log producer and define log connections in the domain profile.