TECHNIQUES FOR COMMERCIAL SDR WAVEFORM DEVELOPMENT

Anna Squires
Etherstack Inc.
145 W 27th Street
New York NY 10001
917 661 4110
anna.squires@etherstack.com

ABSTRACT

Software Defined Radio (SDR) hardware platforms have been used in commercial defence and Land Mobile Radio (LMR) communications for several years now. However the real value of SDR - software reusability, upgradeability and portability – is still often not achieved because the software itself fails to exploit the full potential of these platforms.

This paper introduces SDR software development practices successfully used in commercial radio projects to maximise software portability, maintainability and performance (PMP), and to minimise software cost. These involve the application of specialist expertise, tools and procedures at each of the specification, design, implementation, integration and maintenance phases.

The practices described have been developed to address real problems found in industry, such as managing the deployment of a waveform across multiple disparate hardware platforms that may include heterogeneous devices or use the Software Communications Architecture (SCA).

1. INTRODUCTION

Wireless voice communications can be broadly divided into three industries: cellular, defence and Land Mobile Radio (LMR). The emergence of large-scale commercial cellular communications set a precedent in all three industries for bigger networks with more users and features, which coincided with a need for improved spectrum efficiency and utilization. These factors demand more intelligent radios, and intelligent radios require more software. Accordingly, all three industries began to shift from traditional analogue to digital technology.

In the commercial cellular industry, this shift from analogue (AMPS/ETACS/NMT450) to digital happened swiftly, helped by high sales volumes that meant the newly essential software intelligence was available to manufacturers already incorporated into dedicated function integrated chipsets. Unfortunately such a solution was unavailable in the lower-volume defence and LMR industries, which instead built software capable platforms out of affordable technology developed for the first generation of digital cellular devices: low-cost, low-power and high-peripheral count generic solid-state processors such as GPPs, DSPs and FPGAs.

As this shows, the commercial use of SDR is not new. Rather SDR platforms – that is, those that use generic processing nodes such as GPPs, DSPs and FPGAs to execute radio function in software – have been commercially available since the first digital Land Mobile Radio (LMR) mobile and base stations went to market approximately eight years ago.

Out of the first SDR hardware platforms an entire field has grown. However research in SDR software – or SDR waveform – development has lagged behind progress in SDR hardware and platform technology. Most waveform developments still involve cobbling together software design techniques and tools from other fields and various different vendors, rather than offering an integrated approach geared for SDR.

As more radio function is implemented within the waveform and the value of the waveform therefore grows, it is worth considering how the wireless communications industries might improve the execution of waveform development in order to protect this value. The dedicated SDR tools and techniques introduced in this paper have been developed by a commercial SDR waveform company in order to optimise waveform portability, maintainability and performance and thus ensure best return on investment in waveform development.

2. WHAT IS A COMMERCIAL SDR WAVEFORM?

According the SDR Forum's draft nomenclature [1] a waveform is:

a) The set of transformations applied to information to be transmitted and the corresponding set of transformations to convert received signals back to their information content.

b) Representation of a signal in space

c) The representation of transmitted RF signal plus optional additional radio functions up to and including all network layers.

In practice, the term 'waveform' is applied to a range of different software implementations, from a single signal processing algorithm or group of algorithms to a commercial SDR waveform product. However in terms of structure and complexity, these are entirely different.

Firstly, the signal-in-space component itself can vary widely in intricacy. A production-level commercial SDR waveform deployed on a radio used within a real communications network is substantially more complicated than, for example, the operation of just the Physical Layer of that waveform in demonstration in a controlled environment. It requires a great deal of additional complexity at the Network Layer, and must also co-ordinate the behaviour of many algorithms and higher control functions across all layers of the waveform.

Secondly, and more significantly, in addition to meeting complex signal-in-space and co-ordination requirements, a commercial SDR waveform must **manage the relationship between hardware and software** in order to optimise:

2.1. Portability

The waveform should be developed in a manner that ensures it will be compatible with and therefore reusable across as many current and future hardware platforms as possible. It should also be developed in a manner that minimises the work required to integrate it to any one potential hardware platform.

An important aspect of portability is deployability: the ability to divide the waveform into arbitrary processor and tasking entities so that during deployment it can be split in different ways over a combination of platform processing resources. If you design correctly for optimal portability you also get reusability – the ability to reuse modules of waveform code in other developments – for free.

Optimal portability allows the maximum value to be derived from the waveform relative to the cost of its development, as the waveform can be reused on multiple platforms and repeat development due to redesign of a hardware platform is avoided. Also, an optimally portable waveform prevents waveform considerations from constraining hardware design decisions.

2.2. Maintainability

The waveform should be developed in a manner that allows it to be maintained. This ensures the waveform can be upgraded with fixes and new features, and therefore has a far longer lifespan. Portability and testability are important pre-requisites for maintainability.

2.3. Real-time Embedded Performance

The waveform should be designed and implemented in a manner that will allow best real-time embedded performance to be achieved across a range of potential platforms unknown at the time of development.

Note that the aim of SDR waveform development is to *optimise* these three factors. Some trade-off between them may be required (although not necessarily; often they reinforce one another) and as long as innovations in hardware continue, waveform design will always be chasing a moving target. Best-practice development is about accommodating these practicalities to derive maximum value from the waveform relative to the cost of its development.

3. INDUSTRY STATUS OF SDR WAVEFORM DEVELOPMENT

Designing and implementing a commercial SDR Waveform that both meets the signal-in-space requirements of the corresponding communications standard or specification and correctly manages the relationship between the hardware and the software is extremely complex.

Despite this, most focus in SDR research and industry is on SDR hardware and on standardisation of the interfaces between hardware and software. Of roughly 120 papers that were submitted to the 2007 SDR Forum SDR Technical Conference, only a handful considered aspects of SDR waveform design and none looked at the problem in entirety.

Many initiatives – including the JTRS program and the resultant Software Communications Architecture (SCA) – have the outcome, if not the stated intent, of providing guidance on waveform design. However again, the benefits the SCA can offer in this regard are limited to the interfaces between the waveform and the SDR/SCA platform, whereas the majority of gains in Portability, Maintainability and Performance (PMP) are achieved during design and implementation of the waveform itself. Building an SCA-compliant waveform does not therefore guarantee these have been optimised.

The most likely explanation for this 'hardware-up' approach lies in wireless communications' heritage in analogue technology. As this has been the norm for many decades, more industry expertise is found in hardware design.

As a result, the complexity of SDR waveform software is often poorly understood and SDR waveform development poorly executed, leading to spiralling development times, spiralling costs, and inferior products. For example, the commercial development of non-SDR, fixed platform complex radio software such as GSM, TETRA or APCO P25 is generally measured in tens of millions of US dollars.

This figure is for software that can only be deployed on a single platform and does not factor in the added complexity required of a useful, generic SDR waveform implementation. To see more widespread adoption of SDR waveform development best-practice, and an attendant improvement in waveform products and reduction in waveform cost, the wireless communications industries must mature on two fronts.

Firstly, SDR Waveform specialisation must be cultivated. Secondly, the true and complete separation of SDR hardware and SDR waveform development must occur.

3.1. SDR Waveform Specialisation

Just as hardware design is executed by hardware experts, so commercial SDR waveform design should be executed by waveform experts.

SDR Waveform Engineers are familiar with a wide variety of different wireless communications standards and specifications. They also specialise in wireless technology and embedded software design best-practice for SDR hardware platforms.

A well defined wireless communications standard or specification is the starting point of any waveform. This is fundamental for interoperability; without it, no two independently developed waveforms or radios will communicate correctly or completely. Many problems stem from poorly defined standards, and thus attention during development of the standard or specification should be paid to exhaustively describing the air interface, rather than detailing how the waveform should be implemented.

That job belongs to the SDR Waveform Engineer. Due to their expertise, waveform engineers are efficient at taking a communications standard and:

a) Interpreting it.

b) Identifying deficiencies in the standard and compensating for these.

c) Combining the standard and additional proprietary customer features into a unified waveform requirements specification.

c) Deriving from this a design that is modular and optimises PMP.

d) Ensuring that the waveform will suit embedded platforms and not compromise embedded performance or resource use.

e) Designing and using appropriate SDR test and other tools to develop, verify and maintain the waveform.

f) Writing documentation to accompany the waveform.

These are each specialist skills that an SDR Waveform Engineer refines over many years and multiple waveform developments and deliveries.

3.2. Separation of Waveform and Platform Development

Optimal waveform design is achieved in complete isolation from a target or reference hardware platform.

In most waveform programs, the target or reference hardware platform is known during development. This can only degrade the quality of the waveform, because any design decisions made in order to accommodate the platform will immediately compromise the waveform’s portability. Once this occurs, the waveform is married to the hardware and will not have a life beyond it.

Optimal waveform design is achieved when the waveform is developed in complete isolation from the platform, by planning from the specification or standard down rather than from the platform up.

This does not mean that the hardware is disregarded – an SDR Waveform Engineer must consider the particular characteristics of embedded platforms at every design decision. Rather, it means that embedded best performance for SDR platforms is planned into the design from the outset, in conjunction with portability and maintainability.

4. COMMERCIAL SDR WAVEFORM DESIGN AT ETHERSTACK

Etherstack is an independent, specialist commercial SDR waveform company that has been developing waveforms for radio manufacturers and defence clients internationally since the outset of commercial SDR.

As an independent waveform supplier, Etherstack is driven by commercial imperative to design SDR waveforms that are as flexible, portable, reusable and maintainable as possible – and that can also be optimised for best performance on small form factor embedded radio platforms. Etherstack also needs to be able to execute waveform development efficiently in order to minimise waveform cost.

Etherstack has independently built and then refined over many waveform deliveries the tools and development methodologies necessary to successfully optimise PMP. These are introduced below.

4.1. Specification and Design

As mentioned earlier, it is not the role of a communications standard or specification to describe how a waveform should be implemented. Usually too a customer will have proprietary features that require incorporation into the waveform design. Developing a new waveform therefore involves deriving specific waveform requirements from the combined requirements of the standard and the client, and then interpreting these in a design.

At Etherstack, the first step in this process of interpretation is to represent the design as functional layers that accord with those in the OSI model, if and where practical. These layers are then divided into many communicating modules by applying a disciplined and structured process of decomposition into consistent design entity types and collaborating patterns of entities. This process continues iteratively until the functional complexity of each entity is minimised. At this stage radio functions that are potentially susceptible to platform dependency have already been identified and isolated.

From there, the information flow between each communicating entity in the waveform is represented as a group of thoroughly defined interfaces, in order to implement the scenarios or use cases identified in the original waveform requirements. This process is illustrated in Figure 1.

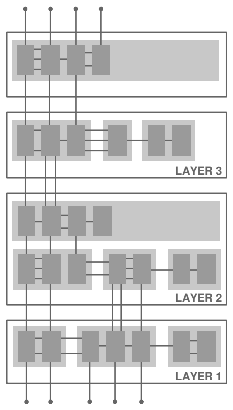

Figure 1: Functional Waveform Decomposition

4.2. Implementation

Once the design is complete, it is implemented as a Base Waveform.

The Base Waveform is the golden code from which each subsequent Target Waveform is derived during deployment on a new hardware platform. This approach is key for maximising the lifetime of the waveform: if a client updates or redesigns their hardware, the Base Waveform is deployed afresh on the new platform, rather than trying to move software designed solely for a legacy platform across to it. The Base Waveform also acts as a central repository for all waveform maintenance, and provides a reference for regression testing and Target Waveform integration verification.

The Base Waveform is built by mapping the design entities into well defined implementation entity types provided by Etherstack's Core Services. These Core Services are a set of sophisticated structural blocks and mechanisms, and combine to allow almost complete abstraction of the waveform from the underlying operating system and hardware platform. They also facilitate the isolation of each functional entity in the waveform to maximise deployment flexibility, and introduce to the waveform intrinsic diagnostic and test support that is compatible with Etherstack's Development, Simulation and Automated Test Environments.

The Base Waveform is implemented entirely in a general purpose programming language such as ANSI C to allow for efficient test and development cycles. As illustrated in Figure 2, it executes within Etherstack's Development and Simulation Environment on a laptop or desktop computer in concurrent operation with Etherstack's Automated Test Environment.

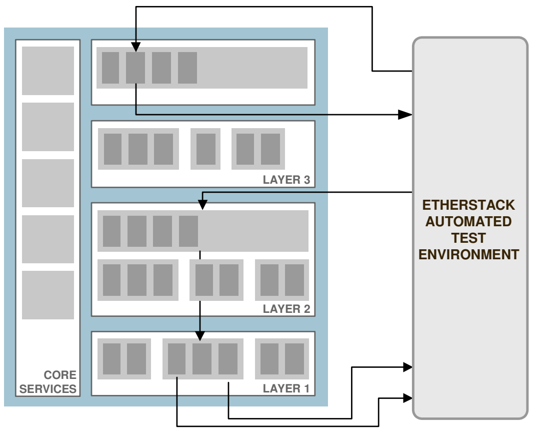

Figure 2: Testing of Implemented Base Waveform

4.3. Integration

It is at the integration stage that the advantages of the PMP design methodologies applied during the design and implementation phases are reaped.

The correct division of the waveform into isolated communicating entities of minimum functional complexity allows the waveform engineer complete flexibility to position different waveform entities over the different processing nodes on a heterogeneous platform in a manner dictated by the requirements of the platform itself. This is illustrated in Figures 3 to 6.

Integrating the Base Waveform to a target platform (and thus deriving the Target Waveform) therefore involves firstly deciding which waveform entities will execute on which radio devices according to resource availability and compatibility between particular entity functions and available processing nodes. The next step is to ensure the Transceiver, Audio, Data, Security, Database and Application waveform-to-platform interfaces are compatible. Lastly, the waveform entities are integrated to each relevant processing node on the platform, and optimised for best performance on that node if necessary.

The integrated Target Waveform is verified on the hardware platform via automated testing, using Etherstack's Automated Test Environment and identical test scripts to those applied to the Base Waveform. The thorough definition of interfaces between waveform entities during the design and implementation phases allows each to be tested in isolation, as well as for the waveform to be tested as a whole. This is illustrated in Figure 7, and testing is described further in Section 4.4.

It is worth recalling that due to Etherstack's Core Services, the operating system used on each processing node is immaterial; the waveform is almost entirely operating system and hardware platform independent.

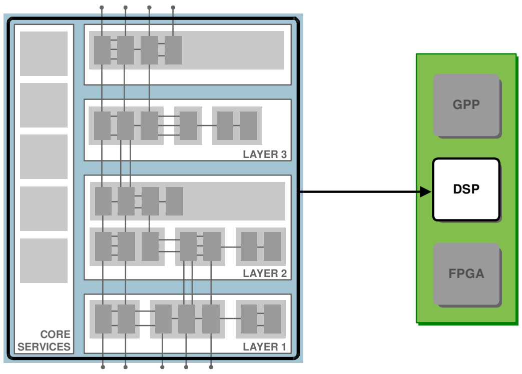

Figure 3: Integration of Entire Waveform to DSP

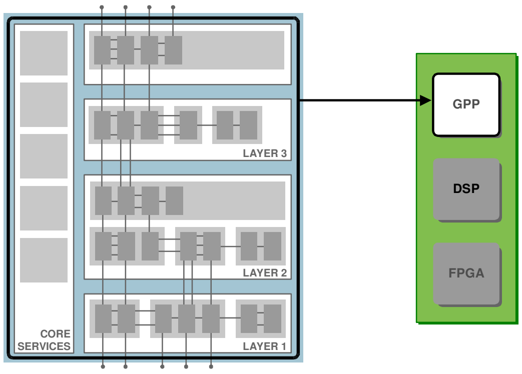

Figure 4: Integration of Entire Waveform to GPP

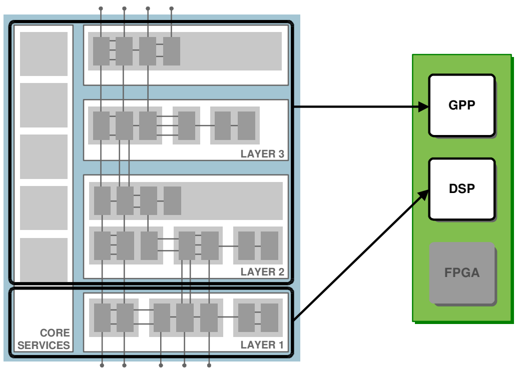

Figure 5: Integration of Waveform to GPP and DSP

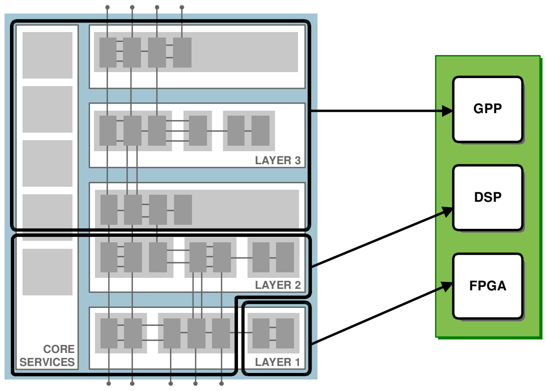

Figure 6: Integration of Waveform to GPP, DSP and FPGA

4.4. Maintenance

Any benefits gained due to the application of correct SDR waveform development methodology rely on a test environment built on the same principles. This must be compatible with both the simulation environment and all potential target platforms in order to verify the waveform during development, prove its correct implementation as a Base Waveform, prove the correct deployment of Target Waveforms and provide regression testing for ongoing maintenance of Base and Target Waveforms.

As with Etherstack's Core Services, Etherstack's Automated Test Environment took tens of engineer years to develop. During its development, the following features were identified as essential in order to support optimal waveform PMP:

a) A library of human-readable test scripts for each waveform that comprehensively covers the behaviour captured in the corresponding waveform requirements specifications. Each of Etherstack's test scripts contains many hundreds of test vectors that are applied to both internal and external interfaces on the waveform under test, in the process deriving human-readable test outputs.

b) Automation. To be exhaustive, waveform testing should be automated just as hardware verification is automated. This involves the automatic execution of each test script in a library in sequence, and of each test vector in a test script – accompanied by automated verification of the results.

c) Identical automated testing of the Base Waveform in simulation and a Target Waveform deployed on an embedded hardware platform. Etherstack builds intrinsic support for hardware agnostic testing into every waveform via the Core Services. Support for testing on the target hardware is similarly built into the Automated Test Environment. This allows the same test scripts to be executed via the Automated Test Environment over both the

Base Waveform in simulation on a PC, and over each Target Waveform on each target hardware platform. This is key for verifying not only that the Base Waveform meets the specifications, but that the Target Waveform does also – and that the behaviour of the two is entirely consistent.

d) The ability to replicate a problem scenario discovered during a Target Waveform's operation on an embedded platform in the Base Waveform, by executing a live log derived from the target hardware platform in the Base Waveform simulation environment.

e) Every internal interface must be testable. This ensures that the resolution of the software under test can be varied from the entire waveform down to a single entity – key if entities or groups of entities may be deployed on different processing nodes in a platform.

f) Graphic tools for visual representation of all signalling and state activity output during testing. These are valuable for diagnostics and as an aid to help clients understand the operation of the waveform. In addition to interface activity, Etherstack's test environment outputs diagnostic information about the waveform such as internal entity state, registered variable state, timer expiry and so on – all of which can be viewed graphically.

It is also worth noting that in addition to automated testing, successful maintenance of the Base Waveform and each of its Target Waveforms requires highly controlled source code management and versioning.

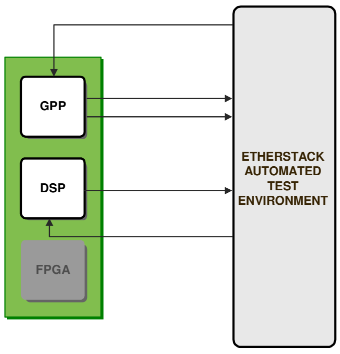

Figure 7: Testing of Waveform on Target Hardware

5. PORTING TO THE SOFTWARE COMMUNICATIONS ARCHITECTURE (SCA)

If a waveform has been designed correctly, porting it to the SCA in order to make it SCA compliant is uncomplicated. At Etherstack, this is achieved by merely identifying how the waveform will be divided into SCA Resources and Devices, and then applying "SCA Wrappers" that convert the identified modules into the relevant Resource or Device. This is illustrated in Figure 8. Using this approach, manufacturers can deploy the save waveform, with exactly the same features, on both SCA and non-SCA radios.

Etherstack's Automated Test Environment is also capable of executing the test script suite over the SCA ports of the ensuing SCA waveform, so the full benefits of a non-SCA deployment are retained.

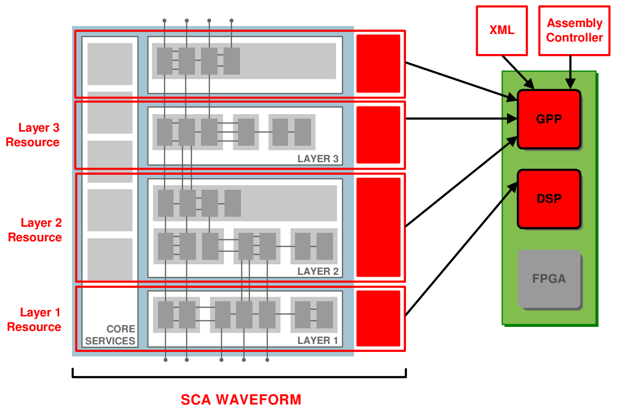

Figure 8: Porting an SDR Waveform to the SCA

6. CONCLUSION

Commercial SDR Waveforms involve complex signal-in-space and co-ordination functions, and a carefully managed relationship between hardware and software that aims to optimise waveform PMP - Portability, Maintainability and Performance.

To improve the quality and longevity of SDR waveforms in light of these requirements, this paper advocates the cultivation of SDR waveform development as an independent, specialist enterprise within the wireless communications industries.

It also recommends specific practices to apply at each phase of waveform development through specification, design, implementation, integration and maintenance, in order to optimise PMP and therefore capitalise on the value of SDR.

7. REFERENCES

[1] SDR Forum Cognitive Radio WG, *Cognitive Radio Definitions and Nomenclature - DRAFT*, SDRF-06-P-009-V0.5.0, 30 May 2008.