WP445 (v1.0) January 20, 2014

Enabling High-Speed Radio Designs with Xilinx All Programmable FPGAs and SoCs

By: Michel Pecot

This white paper describes the capabilities of Xilinx® 7 series All Programmable FPGAs and SoCs to implement high-clock-rate signal processing functionality typically used by the datapath of digital radio applications.

A clock rate higher than 500 MHz can be supported on a mid-speed grade 7 series device with almost 100% of the logic slices, more than 90% of the DSP48 slices, and 70% of the block RAMs utilized. This requires the designer to follow some rather simple design rules that cover both algorithmic and implementation aspects. These rules are reviewed in the paper.

Although almost 100% of the slices can be used at 500 MHz for pure datapath designs, only 75% to 80% of the logic resources (LUTs) are effectively occupied, which essentially relates to routing and control set resource constraints. A higher LUT utilization ratio is achievable with the next-generation 20 nm UltraScale™ architecture, which provides significantly increased capabilities.

© Copyright 2014 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. AMBA, AMBA Designer, ARM, ARM1176JZ-S, CoreSight, Cortex, and PrimeCell are trademarks of ARM in the EU and other countries. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. All other trademarks are the property of their respective owners.

Introduction

Radio units are the most numerous components of a typical wireless network deployment. Consequently, they contribute greatly to the total cost and power consumption of the system. The contribution becomes even more significant with the introduction of small cells and cloud Radio Access Networks (RANs), together with the requirements for multi-mode and multi-Radio Access Technology (RAT) support over larger bandwidths (up to 100 MHz and more), and a greater number of antennas.

To respond to these ever demanding requirements from network operators, radio equipment vendors are continually innovating to improve the capabilities and efficiency of their systems. New technologies have emerged over the past several years for power amplifiers (e.g., digital Doherty, switched mode, and envelope tracking techniques) and antenna systems (e.g., active antennas, multi-band solutions, full-duplex networks), which are starting to be extensively deployed in the field.

These advanced techniques significantly increase the computational complexity of digital signal processing algorithms, especially for functions like crest factor reduction (CFR) or digital pre-distortion (DPD), but also require much higher levels of flexibility, programmability, and integration. The Xilinx All Programmable FPGA and SoC technology, which is based on a configurable fabric, is ideally suited to meet these challenges. It is also very attractive because it offers a number of high-speed transceivers and flexible I/Os, which can provide the physical interfaces necessary for any digital radio solution (interfaces to baseband, DAC/ADC, and analog components), while the FPGA logic can implement virtually any interfacing solution (CPRI, JESD204B, etc.), as well as potential complementary switching functionality.

Built upon the 28 nm High-Performance Low-Power process technology (HPL), the Xilinx Kintex®-7 FPGAs and Zynq®-7000 All Programmable SoCs (AP SoCs) [Ref 1][Ref 2][Ref 3] deliver the required level of signal processing performance to integrate complex digital radio applications. These devices also offer a significant power advantage over: previous technology nodes, competing solutions using the 28 nm HP process, and even the 28 nm LP process. Power is an important performance metric for such applications since it allows better thermal dissipation while also providing cost savings on power supply components.

Zynq-7000 AP SoCs[Ref 3] are ideal for the integration of digital radio systems. Based on the same programmable logic (PL) as the Kintex-7 FPGAs, the Zynq-7000 AP SoC also integrates a complete processor system (PS) built around a dual-core ARM® Cortex™-A9 CPU tightly coupled with the PL. This architecture enables the implementation of custom logic in the PL and custom software in the PS, facilitating the realization of unique, differentiated system functions. The levels of performance provided by this SoC implementation simply cannot be matched by any two-device solution due to limited I/O bandwidth, higher latency, and/or higher power budget.

The integration of PL with a complete PS is of particular interest to those designing radio applications. One ARM Cortex-A9 processor can support the OA&M and board-level control functionality, running an RTOS such as Linux; the second core, running bare metal, can execute DSP software, such as CFR/DPD coefficient estimation (potentially complemented with some hardware acceleration in the PL), while the PL implements the signal processing datapath.

To provide cost-effective solutions, it is necessary to fully exploit the capabilities of the FPGA fabric, and architect the required functionality for the highest possible clock rate. This indeed minimizes the application footprint, but also brings some benefit to system power consumption.

The purpose of this white paper is to demonstrate that such high-speed radio designs can readily be built on 7 series FPGA fabric, provided that some simple design rules are respected. Xilinx has built a design example representative of a typical radio datapath which demonstrates that a clock rate higher than 500 MHz can be supported with almost 100% of the slice resource utilized on a mid-speed grade (-2) device. This cannot be achieved with the slowest speed grade (-1), but in most applications, mid-speed grade devices are required in any case to support 10G transceiver speed. Xilinx's recently announced 20 nm UltraScale family[Ref 4][Ref 5] can support such performance on its slowest speed grade.

Although almost 100% of the slices can be used at 500 MHz for a pure datapath design,[1] only 75% to 80% of the LUTs are effectively occupied (each slice contains four LUTs). This relates both to silicon and tools aspects, as described later in this white paper.

The white paper is organized in four main sections:

- The first section provides an overview of the digital radio architectures used in wireless networks, with some comments on how this is likely to evolve to meet future system requirements. The principal dimensioning parameters are reviewed, and a brief description of the signal processing functionalities implemented is also provided.
- The second section summarizes Xilinx solutions for digital radio applications.
- The third section describes guidelines for building efficient radio designs. Algorithmic and implementation aspects are covered, as well as tips and tricks to support high clock rates.
- The fourth section demonstrates the high-speed capabilities of the Kintex-7 FPGA fabric on a radio datapath design example.

Architecture and Functionality of Digital Radio Systems

A high-level block diagram of a typical radio system deployed in wireless networks is shown below in Figure 1. It is comprised of six principal functional blocks:

- The baseband interfacing is based on the CPRI (or OBSAI) protocol, although 10GE is likely to be considered in the future. Each 20 MHz input/output stream requires 983.04 Mb/s (assuming 16-bit I/Q data), and high-speed serial links are used with up to 10.14 Gb/s throughput per link (which might evolve to 19.6 Gb/s). This block includes the mapping between the serial data and antenna I/Q samples, potentially coupled with some switching functionality.
- The digital front end (DFE) integrates the transmitter and receiver digital datapaths, and essentially implements the signal processing functionality. This white paper focuses on this DFE block to demonstrate the high-speed capabilities of Xilinx 7 series FPGAs.
- The controller is used for OA&M and board-level control, and for some signal processing software functionality, such as DPD, coefficient estimation and update (possibly complemented with some hardware acceleration implemented within the DFE block). It also integrates the various control and memory interfaces (DDR, Ethernet, UART, GPIO, I2C, SPI, flash, etc.).

1. Such designs are not fully representative of real radio applications, which also integrate interfacing and control functionality, but the intent is to explore the limits of the fabric capabilities for signal processing functionality.

- The DAC/ADCs interfacing can be based on LVDS or serial (JESD204B) technology depending on the selected devices. Apart from the transmit antenna paths, additional DAC interfaces can be required to support specific PA technologies, e.g., envelope tracking. Similarly, in addition to the receiver antenna paths, one or several complementary ADC interfaces are used to feed back the transmitter output(s) for calibration and/or DPD coefficient estimation purpose.
- The analog front end includes the DAC/ADC devices and the analog processing: modulators, local oscillators, power amplifiers, low noise amplifiers, analog filters, duplexers, etc.
- A DDR memory is commonly used to store the controller software code and/or measurement and calibration data.

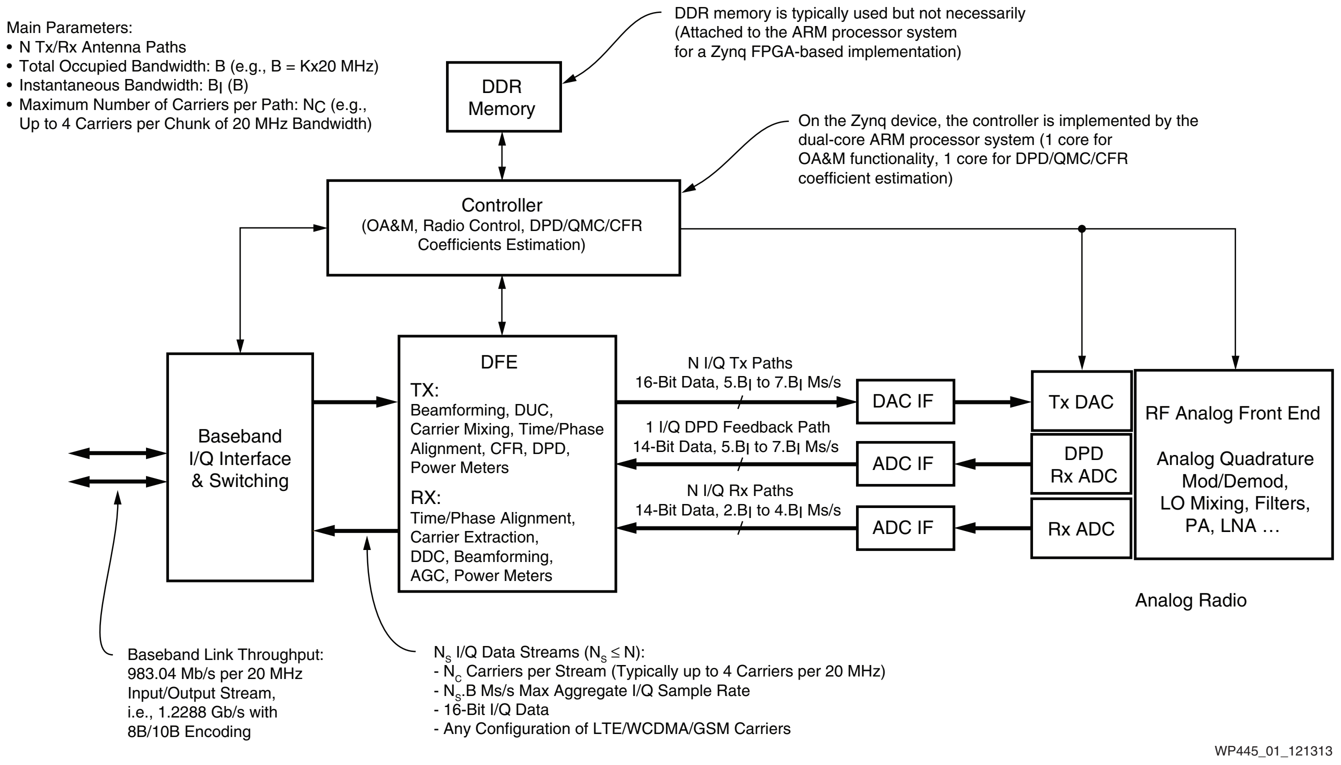

Main Parameters:
• N Tx/Rx Antenna Paths
• Total Occupied Bandwidth: B (e.g., B = Kx20 MHz)
• Instantaneous Bandwidth: BI (B)
• Maximum Number of Carriers per Path: NC (e.g., Up to 4 Carriers per Chunk of 20 MHz Bandwidth)
DDR Memory
DDR memory is typically used but not necessarily (Attached to the ARM processor system for a Zynq FPGA-based implementation)
On the Zynq device, the controller is implemented by the dual-core ARM processor system (1 core for OA&M functionality, 1 core for DPD/QMC/CFR coefficient estimation)
Controller (OA&M, Radio Control, DPD/QMC/CFR Coefficients Estimation)
Baseband I/Q Interface & Switching
DFE
TX: Beamforming, DUC, Carrier Mixing, Time/Phase Alignment, CFR, DPD, Power Meters
RX: Time/Phase Alignment, Carrier Extraction, DDC, Beamforming, AGC, Power Meters
N I/Q Tx Paths 16-Bit Data, 5.BI to 7.BI Ms/s
1 I/Q DPD Feedback Path 14-Bit Data, 5.BI to 7.BI Ms/s
N I/Q Rx Paths 14-Bit Data, 2.BI to 4.BI Ms/s
DAC IF
ADC IF
ADC IF
Tx DAC
DPD Rx ADC
Rx ADC
RF Analog Front End
Analog Quadrature Mod/Demod, LO Mixing, Filters, PA, LNA …
Analog Radio
Baseband Link Throughput: 983.04 Mb/s per 20 MHz Input/Output Stream, i.e., 1.2288 Gb/s with 8B/10B Encoding
NS I/Q Data Streams (NS ≤ N):
- NC Carriers per Stream (Typically up to 4 Carriers per 20 MHz)
- NS.B Ms/s Max Aggregate I/Q Sample Rate
- 16-Bit I/Q Data
- Any Configuration of LTE/WCDMA/GSM Carriers
WP445_01_121313

Figure 1: **High-Level Architecture of Typical Radio Applications**

The Zynq-7000 architecture can implement all this functionality, with the exception of the analog front end and DDR memory. Three system parameters essentially dictate the implementation complexity, and define the appropriate target device(s):

- The number of TX/RX antenna paths
- The instantaneous bandwidth
- The number and type of input/output carrier streams (single or multi-RAT, GSM support or not)

Multi-mode systems, supporting both LTE and WCDMA, and potentially GSM (with frequency hopping) are more and more often deployed. The ability to dynamically or semi-statically change the number of active carriers and their type and bandwidth is also a common requirement, which allows switching between different network configurations without having to develop and maintain multiple versions of the application software.

As an example, there are N_S I/Q data streams at the transmitter input, each one carrying up to N_C carriers. For traditional remote radio units (RRUs), there is a one-to-one mapping between the input streams and the N antenna paths ($N=N_S$), while for beam-forming applications ($N \geq N_S$), each input stream is mapped to the N antenna paths through some matrix operation, to form the beams. The same applies at the receiver side, with a number of antenna paths that can be doubled to support diversity combining.

The instantaneous bandwidth (B_I), defined as the total bandwidth spanned by the carriers (not necessarily contiguous in frequency), determines the sampling rate of the different processing steps:

- Transmit carrier mixing is implemented at around 1.5 B_I to limit the complexity of the composite signal interpolation.
- When applied to the composite signal, CFR works at 3 to 4 times B_I to avoid peak regrowth with subsequent interpolation stages. Solutions can, however, be designed to operate directly after carrier mixing.
- DPD is implemented at 5 to 7 times B_I, which also dictates the DAC sampling rate and feedback ADC rate.
- The receive ADC rate is typically chosen in the range of 1.5 to 2.5 times B_I to reduce analog filtering complexity.
- Finally, similarly to the transmitter side, the receive carrier extraction works at around 1.5 B_I.

A more detailed view of the functionality that can be integrated within Xilinx technology is shown in Figure 2.

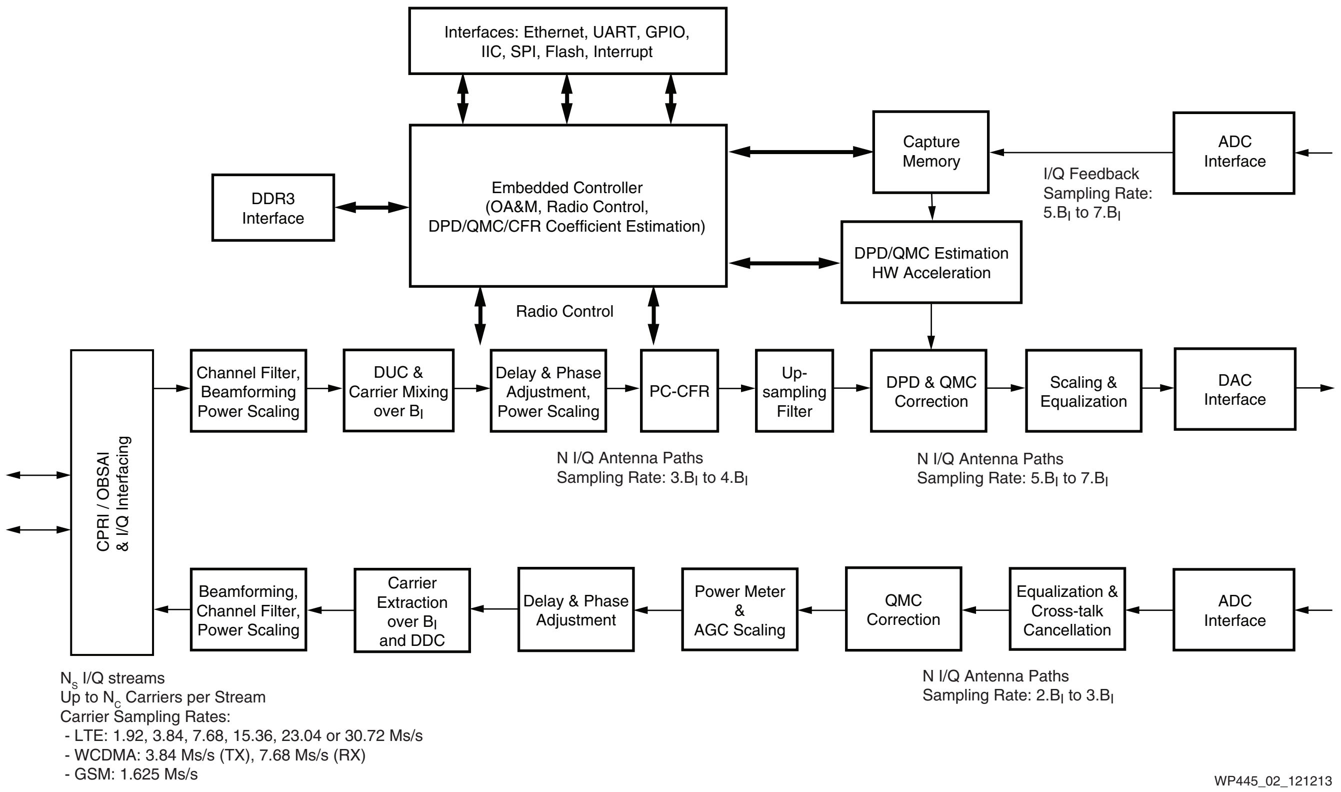

Interfaces: Ethernet, UART, GPIO, IIC, SPI, Flash, Interrupt
DDR3 Interface
Embedded Controller (OA&M, Radio Control, DPD/QMC/CFR Coefficient Estimation)
Capture Memory
ADC Interface
I/Q Feedback Sampling Rate: 5.B$_I$ to 7.B$_I$
DPD/QMC Estimation HW Acceleration
Radio Control
CPRI / OBSAI & I/Q Interfacing
Channel Filter, Beamforming Power Scaling
DUC & Carrier Mixing over B$_I$
Delay & Phase Adjustment, Power Scaling
PC-CFR
Up-sampling Filter
DPD & QMC Correction
Scaling & Equalization
DAC Interface
N I/Q Antenna Paths Sampling Rate: 3.B$_I$ to 4.B$_I$
N I/Q Antenna Paths Sampling Rate: 5.B$_I$ to 7.B$_I$
Beamforming, Channel Filter, Power Scaling
Carrier Extraction over B$_I$ and DDC
Delay & Phase Adjustment
Power Meter & AGC Scaling
QMC Correction
Equalization & Cross-talk Cancellation
ADC Interface
N$_S$ I/Q streams
Up to N$_C$ Carriers per Stream
Carrier Sampling Rates:
- LTE: 1.92, 3.84, 7.68, 15.36, 23.04 or 30.72 Ms/s
- WCDMA: 3.84 Ms/s (TX), 7.68 Ms/s (RX)
- GSM: 1.625 Ms/s
N I/Q Antenna Paths Sampling Rate: 2.B$_I$ to 3.B$_I$
WP445_02_121213

Figure 2: **Detailed Functionalities of Digital Radio Systems**

When targeting the Zynq-7000 AP SoC, the embedded controller is implemented by the PS, which includes a rich set of dedicated communication peripherals such as UARTs, SPI, I2C, gigabit Ethernet, and memory controllers. One of the ARM® Cortex™-A9 cores is used for OA&M and board-level control, while the second is running the DSP software (e.g., DPD coefficient estimation). If a 7 series FPGA is being targeted, one or multiple instances of the MicroBlaze™ soft processor core can be built into the fabric to implement the controller functionality while the control interfaces must be implemented in fabric.

The principal signal processing functions commonly required on the transmit side include the following:

- Channel filtering is applied to the carriers of each input data stream.
- For beam-forming applications, a complex matrix multiplication is applied to the N_S data streams (also referred to as *beams*) to generate the N antenna streams. The computation of the matrix coefficients can be performed in software by the embedded controller (with potential support of fabric-based acceleration).
- The carriers of the N antenna streams are up-converted to around 1.5 times the instantaneous bandwidth, and mixed together to generate the N composite signals.
- The N antenna paths usually need to be accurately aligned in time and phase. This operation basically consists of a resampling filter, which can be implemented with a Farrow structure.
- CFR is applied to each antenna stream. Various solutions can be considered, but the pulse cancellation technique (PC-CFR), with a pulse matched to the carrier configuration, is viewed to provide the best performance.
- DPD is applied to each antenna stream at a rate of 5 to 7 times the instantaneous bandwidth. This function mainly consists of a non-linear filter implementing a model of the inverse of the power amplifier time/frequency response, associated with a least square system solver estimating the model coefficients.
- Correction of the offset, gain, and phase imbalance impairments of analog quadrature modulation (QMC) and equalization can finally be applied to each transmit path, and potentially also to the DPD feedback path(s).
- The CFR, DPD, QMC, and equalization operations involve the estimation and update of a set of coefficients, which is usually performed in software and shared by multiple antenna paths. Some fabric hardware acceleration can be used for applications requiring a fast update rate (for DPD especially). The functionality, typically built into hardware, includes correlation computation and/or complex MAC engines.
- Power measurement and scaling engines are also applied at various transmit stages to the individual carriers and composite signals.

The signal processing functionality integrated on the receive side is much simpler and usually includes the following steps:

- QMC and equalization can be applied to each antenna path to correct the analog impairments within the instantaneous bandwidth, together with some possible cross-talk cancellation, removing any interference generated by the transmit path.
- Like the transmit side, the antenna paths are usually required to be time and phase aligned.
- The N_C carriers of each antenna stream are extracted and further down-converted to the baseband rate.
- For adaptive antenna systems, the beam-forming matrix multiplication generates the N_S beam streams from the N antenna streams.
- Channel filtering is finally applied to the carriers embedded in each data stream.
- Similar to the transmitter, power measurement and gain control are performed at various stages of the receiver.

Calibration is another function that is quite critical for active antenna systems. This is usually implemented as a mix of hardware and software, and is easily enabled by the flexibility offered by FPGA technology.

The fundamental requirement driving future development of cellular systems is to provide increased capacity and throughput with improved coverage at lower system cost. This, of course, impacts the radio units that are required to support wider bandwidths, a higher number of multi-RAT carriers (with flexible configuration capabilities), and many more antennas. From an FPGA implementation perspective, a few points are worth mentioning. Although doubling the instantaneous bandwidth doubles the carrier mixing (and extraction) rate, adds an up-sampling (and down-sampling) stage for all carriers, and also doubles the rate of the composite signal processing, it increases the FPGA resource requirement by a factor of less than two (this factor gets closer and closer to two as the bandwidth increases). Indeed, resource sharing can be applied (e.g., control logic) and multipath data multiplexing for low data rates allows higher implementation efficiency. For similar reasons, the same applies when doubling the number of TX/RX paths, particularly because some functionality can be shared by the antenna paths (e.g., the hardware accelerators used for DPD coefficient estimation). Again, this is true for up to six to eight TX/RX paths (depending on the supported bandwidth), while for larger configurations, the resource requirement becomes much more closely proportional to the number of paths. The effect of increasing the number of carriers (with a constant instantaneous bandwidth) is more difficult to assess because it only affects the number of filtering paths of the digital up-converter (DUC) and digital down-converter (DDC) without impacting the composite signal processing (although a more complex DPD model might be required).

Overview of Xilinx Radio Solutions

Xilinx has long recognized the importance of radio units in wireless network design and has introduced silicon and IP products specifically dedicated to this application. The FPGA technology offers full flexibility, and allows Xilinx to provide optimized solutions for virtually any feature set, in terms of the number of TX/RX paths, instantaneous bandwidth, and number/type of carriers. Xilinx products have reached a point where equipment manufacturers can implement all the digital radio hardware and software into a single device. Of particular interest is the Zynq-7000 AP SoC[Ref 3], which can be complemented with some Kintex-7 FPGAs[Ref 2] for high-end applications requiring compute power levels not available in a single device.

The Zynq-7000 family provides full hardware and software programmability along with a suite of peripherals hardened for low cost and power, such as memory controllers, gigabit Ethernet, UARTs, SPI, and I2C. The PS, with its dual-core ARM® Cortex™-A9 CPU is capable of up to 2,000 DMIPs per core. A double-precision floating point unit is associated with each core together with the NEON® media accelerator. The PS supports both symmetric multi-processing (SMP) and asymmetric multi-processing (AMP). One of the cores supports the board-level control functions running a real-time operating system such as Linux, while the other is used to implement some signal processing algorithms, which do not warrant a full hardware solution (e.g., DPD coefficient estimation). The PL is connected to the PS through multiple low-latency, high-speed wide buses and shared memory interfaces, which provide a bandwidth that would be impractical for a discrete solution with an external general-purpose or DSP processor.

With the Zynq-7000 and Kintex-7 families, Xilinx offers a comprehensive range of high-performance PL fabric, capable of supporting higher than 500 MHz performance, ideally suited to radio applications. The key attributes are:

- Up to 2,020 DSP48 slices[Ref 6], integrating a 48-bit MAC (18×25 multiplier), with pre-adder and rounding logic.
- Up to 955 true dual-port 36K block RAMs[Ref 7], supporting 1-bit to 72-bit width configurations.
- Up to 75K logic slices[Ref 8], each containing four dual output 6-bit LUTs (also configurable as distributed memory or shift register) and eight flip-flops.
- Up to 32 transceivers[Ref 9], supporting 12.5 Gb/s, and configurable to support the wide range of physical interfaces adopted within digital radio units.
- Up to 500 high-performance I/Os to implement lower-speed interfaces.

The recently announced 20 nm UltraScale architecture[Ref 4][Ref 5] brings even more capabilities, with some improvements of the DSP48 architecture, increased routing resources, significant power reduction, and 500 MHz performance achievable on the lowest-speed grade devices.

Xilinx has developed a portfolio of IP dedicated to the signal processing functionality of digital radio applications. The FIR and DDS compilers, the PC-CFR, and the DPD IP cores are of particular interest[Ref 10][Ref 11][Ref 12][Ref 13]. For low-level operators like complex MAC, IPs are also available, but it is usually more flexible to directly build the corresponding function either in VHDL or with other tools provided by Xilinx, like System Generator. It is beyond the scope of this document to provide a comprehensive description of these solutions, and the reader is referred to Xilinx product documentation for more details. However, the critical point is that each of these cores can support a clock rate higher than 500 MHz on mid-speed grade 7 series devices.

Xilinx has performed a comprehensive device mapping analysis for various system requirements, ranging from 2×2 20 MHz to 8×8 100 MHz (assuming two to four carriers per 20 MHz of bandwidth). Based on the usage of Xilinx IP, this analysis shows that the largest Zynq-7000 device can support up to four antenna paths with 80–100 MHz instantaneous bandwidth and fully flexible multi-RAT carrier configuration. For eight antenna paths, the bandwidth must be reduced to 20–40 MHz. For larger configurations, a dual-chip solution is required with a complementary Kintex-7 device.

High Clock Rate Design Rules and Benefits

Depending on the system configuration, the datapath of a digital radio application implemented on Xilinx technology represents about 60% to 80% of the total design logic resource, while control and interfacing use the remaining 20% to 40%. With respect to the other fabric resource, almost all the DSP48 slices and 50% to 60% of the block RAMs are commonly utilized by the datapath. Effectively, this functionality can benefit the most from a higher clock rate implementation because data multiplexing and operator folding is usually not applicable for control and interfacing designs. Logic and block RAM resource savings is, however, possible for I/Q data switching, when configured to operate at a higher clock rate, although significant savings are only achievable for complex applications, e.g., 16 ports or more.

The purpose of the remaining two sections of this white paper is to demonstrate that radio signal processing functionality can be implemented within a mid-speed grade 7 series FPGA with high utilization ratios (i.e., with 60% to 80% of the LUT resource utilized, about 100% of the DSP48 slices, and 50% to 70% of the block RAMs), with all resources being clocked at more than 500 MHz. Essentially, because of the routing and control set(1) resource limitations, it is generally not possible to use up to 100% of the LUTs at such a speed. Nevertheless, part of the remaining logic can be used for control and interfacing, running at low speed. These utilization ratios can be improved with the 20 nm UltraScale architecture, which provides increased routing and control set resource. The tool capabilities also impact these aspects, especially the backend tools (placer and router), which govern the effective LUT packing into slices. Xilinx's Vivado® Design Suite is continuously seeking improvements on this front.

When aiming to minimize the resource utilization for radio datapath functionality, and consequently optimize cost and power dissipation, two complementary problems must be considered.

First, the overall architecture and individual functional blocks must be optimized from an algorithmic perspective to exploit the system specificities, while still meeting the required performance levels. For instance, efficient DUC/DDC architectures can be derived, based on multi-stage carrier mixing/extraction, whenever groups of carriers can be identified, which are constrained to be located within specific sub-bands of the instantaneous bandwidth.

After the appropriate system architecture has been defined, the hardware implementation must be specified for the highest possible clock rate. The purpose is to minimize the hardware footprint through data multiplexing (sharing a given functionality between multiple data streams), or alternatively through operator folding (exploiting the clock cycles available per data sample). In addition, low-level

1. The control set resource refers to the clock, clock enable, and reset signals, which are shared by all the storage elements within a slice, hence, the constraints when trying to use all the LUTs within a device.

operators need to be built to exploit as much as possible from the various fabric features (DSP48 functionalities, block RAM configurations, shift register, and distributed memory modes of the LUTs).

Before describing these aspects in more detail, a simple example is used to illustrate some general trends and highlight the benefits of designing for high clock speed. The example architecture is a single antenna path DUC, supporting any configuration of LTE and WCDMA carrier mix, with up to four active carriers occupying a total bandwidth of 20 MHz, and located anywhere within a 100 MHz instantaneous bandwidth. Support for 5, 10, 15, and 20 MHz LTE bandwidths is assumed; dynamic switching of the type, bandwidth, and frequency location of each carrier is enabled.

The DUC functionality has been architected for three different clock rates: 245.76, 368.64, and 491.52 MHz. The design is built with the Xilinx System Generator tool,(1) generating a mix of IP (FIR compiler), structural (e.g., instantiation of DSP48 primitives), and inference VHDL code. The same options have been selected for the synthesis and backend tools when implementing these designs. The resource utilization results are summarized in Table 1. The LUT-to-slice and FF-to-slice ratios are similar in all cases, indicating that the tool packing performances are equivalent for all three clock rates. Therefore, the results can be fairly compared.

Table 1: **Implementation Results of a Simple DUC Design at Various Clock Rates**

Clock Rate	Slices	LUT	FF	DSP48 Slices	Block RAMs (18K)	Normalized Dynamic Power
245.76 MHz	2,440	6,130	8,830	108	2	1
368.64 MHz	1,950	4,620	7,470	79	2	1.11
491.52 MHz	1,550	3,450	5,210	57	1	1.11

The first point to notice is that the DSP48 slice count scales as expected, being approximately inversely proportional to the clock rate. The block RAM resource also decreases with the clock rate, but rather as a step function. This is common for radio signal processing designs, where block RAMs are essentially used to store large sets of coefficients for functionalities operating at a relatively high sampling rate — for example, sine/cosine values of a DDS, coefficients of a CFR peak cancellation pulse generator, sampling of a non-linear function in a DPD model, etc. The resource required for any functional block only changes when the clock rate crosses some thresholds, multiple of the signal sampling rate, hence, this stepping behavior. The same applies to the DSP48 slice resource, but with a much finer grain for the crossing points. This makes the variation look much more linear as a function of the clock rate.

The analysis is not as straightforward for the logic resource. When moving from 368.64 MHz to 491.52 MHz (1.33 clock ratio), the LUT and FF counts are divided by 1.34 and 1.44 respectively, while doubling the clock rate from 245.76 MHz to 491.52 MHz only reduces these numbers by a factor of 1.8 and 1.7. This non-linear behavior is essentially due to the logic controlling the pure signal processing functionality, which does not scale linearly. The signal sampling rate also has some impact. For instance, a filter working on a 25 MSPS signal needs slightly less than twice the logic resource when implemented at 250 MHz compared to 500 MHz, while

1. The System Generator tool is running on top of the Matlab/Simulink software, based on a specific blockset integrating both high-level IPs (FIR compiler, FFT, etc.) and low-level functionalities (adder, delay line, logic operator, register, memory, DSP48 primitive, etc.), which allows accessing all the FPGA features. It is especially adapted to develop and build signal processing designs, since it can benefit from the rich simulation capabilities of the Matlab software.

for 500 MSPS operation, a polyphase implementation at 250 MHz requires more than twice the logic resource than a direct implementation at 500 MHz.

A rule of thumb to derive a first-order estimate of the logic resource evolution is that increasing the clock rate by a factor of x yields a decrease of the logic utilization by a factor of about 85% to 110% of x. This is applicable provided that the absolute clock change is larger than a few times the base signal rate (30.72 MHz for LTE applications).

Table 1 also provides an estimate of the normalized dynamic power for the three clock rates tested, derived in the same conditions (process, junction temperature, toggle rate, etc.), and excluding the I/O and device static power. The results are in line with the resource figures, and demonstrate that increasing the clock frequency does not generally affect dynamic power consumption, which is a linear function of the clock frequency and the resources used (including the routing resource). However, minimizing resource utilization through a higher clock rate can allow a smaller device to be targeted for the complete system, with significant reduction of static power.

When looking to build a resource-efficient radio application, the first step is to optimize the system-level architecture. This point is illustrated with the previous DUC design. Mixing of the active carriers is performed at 122.88 MSPS, and two different solutions can be built:

- **Solution 1:** Lower-bandwidth carriers (5, 10, 15 MHz) are first up-sampled to the highest carrier rate, 30.72 MSPS; all carriers are then up-sampled to 122.88 MSPS.
- **Solution 2:** All carriers are first up-sampled by 4, which brings the highest bandwidth carriers to 122.88 MSPS (a single 20 MHz carrier in this case), and the lower bandwidth carriers are further up-sampled to 122.88 MSPS.

The second approach is preferred because it is much more efficient from a resource perspective. In both cases, the up-sampling by 4 applied to all carriers needs to be able to process a 20 MHz carrier at 30.72 MSPS on input, hence, the same filter complexity is required (e.g., cascading of 23-tap and 15-tap half-band filters). However, the hardware must support four complex inputs at 30.72 MSPS in the first case, but only a single one in the latter approach — although with advanced multichannel capabilities to support the different carrier configurations. This subsystem is, therefore, almost four times less complex with the second architecture: the DSP48 requirements are truly divided by four, while the logic resource reduction is slightly less, due to the overhead for supporting the advanced multichannel mode. The other subsystem (up-sampling of the 5, 10, and 15 MHz carriers) is also significantly less complex: although the same filtering architecture is required in both cases, shorter filters can be used with the second approach since the input carriers have already been up-sampled.

The second example, a DUC supporting twelve 5 MHz LTE carriers within 100 MHz of instantaneous bandwidth, highlights the benefits of multi-stage carrier mixing. The assumption for this example is that three groups of four carriers can be isolated, each one spanning a 20 MHz sub-band. Two different architectures can be derived:

- A direct implementation, not taking profit of the sub-band splitting constraint, where the 12 carriers are independently up-sampled from 7.68 MSPS to 122.88 MSPS, and then mixed together to generate the output composite signal.
- A multi-stage implementation, where the 12 carriers are first up-sampled to 30.72 MSPS. At this stage, the four carriers in each sub-band are mixed together, and the three composite signals are further up-sampled to 122.88 MSPS and mixed together to generate the output signal.

The difference between the solutions lies in the up-sampling process from 30.72 to 122.88 MSPS. To get the same output spectral quality in each case, higher performance interpolation filters are required for the multi-stage approach, since these are operating on 20 MHz signals, instead of 5 MHz for the direct implementation. A cascade of two 7-tap half-band filters is used for the latter case, while the multi-stage solution requires these filters to be extended to 27 and 11 taps respectively. Additionally, assuming a 491.52 MHz clock rate, the direct approach requires the implementation of 3 DDS (each one supporting the mixing of four carriers at 122.88 MSPS), while 2 DDS are sufficient for the alternative solution, one for each mixing stage. The resource utilization counts are summarized in Table 2, showing that the multi-stage approach provides a clear advantage, with half the DSP48 slices required, 33% reduction of the block RAM count, and about 45% of the logic saved. The actual resource savings depends on the number of carriers to be processed and the number of sub-bands that can be extracted, but in practice, the multi-stage architecture always brings some benefit.

Table 2: **Resource Utilization of Direct and Multi-Stage Mixing Architectures**

Design	LUT	FF	DSP48 Slices	Block RAMs (18K)
Direct Mixing	1,850	2,150	33	6
Two-stage Mixing	1,050	1,400	16	4

After the system architecture has been optimized, the hardware must be carefully built to support the highest possible clock frequency. This involves multiple related aspects, the most relevant being described below. Good quality HDL coding is essential to enable an optimal usage of the fabric features, since the backend tools (placer and router) are not able to modify the logic generated by the synthesis. Direct instantiation of hardware primitives, therefore, needs to be considered when inference does not work properly, since this adversely affects the quality of the final design, not only from a timing perspective, but also in terms of power consumption. It is, however, beyond the scope of this white paper to provide a complete overview of the Vivado tool design methodology, and the reader is referred to the *Design Methodology Guide for the Vivado Design Suite*[Ref 14] for more information.

Appropriate pipelining is indisputably the key enabler when designing for high speed. Fortunately, the basic operators of a radio datapath application can be pipelined while still meeting the latency requirements. The hardware design must, therefore, exploit the abundant flip-flop resources available within the FPGA fabric. Recommendations for the different hardware elements are summarized below (speed performance figures are for mid-speed grade Kintex-7 devices):

- The DSP48 architecture offers four levels of internal registers[Ref 6]: two on the inputs (the second one being also used by the pre-adder output), one for the multiplier (MREG), and the last one at the accumulator output (PREG). When the pre-adder is used, it is necessary to enable all four levels of pipelining to reach more than 500 MHz, while three of them are enough otherwise (MREG, PREG, and one of the input registers). When used, the pattern detector (which is very useful to round the DSP48 output) induces a speed penalty, but up to 549MHz can still be supported.
- Block RAMs[Ref 7] are the weak resource from a timing perspective. Two registers are integrated on output to mitigate this limitation, and both must be used when targeting higher than 500 MHz operation. Additionally, out of three write modes available, READ_FIRST cannot be used, because it is limited to 477 MHz; WRITE_FIRST or NO_CHANGE, which support up to 543 MHz, should be selected instead. NO_CHANGE is preferred whenever possible because it also minimizes power consumption. When building memories requiring more than

one block RAM, it is advisable to choose the configuration that minimizes data multiplexing to optimize speed capabilities and resource utilization. For instance, a 16K memory storing 16-bit data is better built with block RAMs configured as 16K×1-bit, rather than as 1K×16-bit. Unfortunately, this is not the most *power-effective* solution, so intermediate architectures might be preferred when power is critical.

- The slice logic can intrinsically support very high clock rates; rather, it is the number of logic levels as well as the data routing paths that limit speed. It is recommended to insert a register every one or two LUT levels when building high-speed designs.

When using a high clock frequency to implement a functionality operating on a signal with a relatively low sampling rate, two extreme alternatives might be envisaged:

- The first one exploits the multiple cycles available per input or output sample, to minimize the size of the operator (multicycle or folded architecture).
- The second approach consists of time-multiplexing as many signals as possible on the input of the function, which is built with a parallel architecture, assuming a single cycle per sample.

Whenever possible, the latter solution should be given preference, since the logic requirements can be significantly reduced. This is illustrated with the following example, considering the filtering of 16 data streams sampled at 30.72 MSPS, with a single-rate symmetric filter. The two extreme architectures are:

- **Multichannel**, implementing a 16-channel filter running at 491.52 MHz
- **Multicycle**, where 16 filters running at 491.52 MHz are built (with 16 clock cycles available per input sample)

The results are provided in Table 3 for filter lengths of 95 and 81 taps, and demonstrate the superiority of the multichannel approach.

Table 3: **Multichannel versus Multicycle Filter Implementation**

Filter Design	Filter Characteristics	Slices	LUT	FF	DSP48
16-channel parallel filter	95-tap symmetric	590	1,660	2,355	48
	81-tap symmetric	450	1,300	2,015	41
16x single channel multicycle filter	95-tap symmetric	650	1,760	3,610	64
	81-tap symmetric	650	1,760	3,610	64

The advantage over the multicycle architecture is even more important when folding is not perfect — in other words, when the number of multiplications required per sample is not a multiple of the clock-to-sample-rate ratio. In effect, the 95-tap filter requires 48 multiplications per sample, which is a multiple of 16, while 41 multiplications are needed for the 81-tap filter.

LUT/SRL compression is another fabric feature that can be exploited to generate better-packed designs, optimizing resource utilization as well as dynamic power consumption. The principle is to use the two outputs of the LUT6 to pack two different functions into a single LUT, as shown in Figure 3.

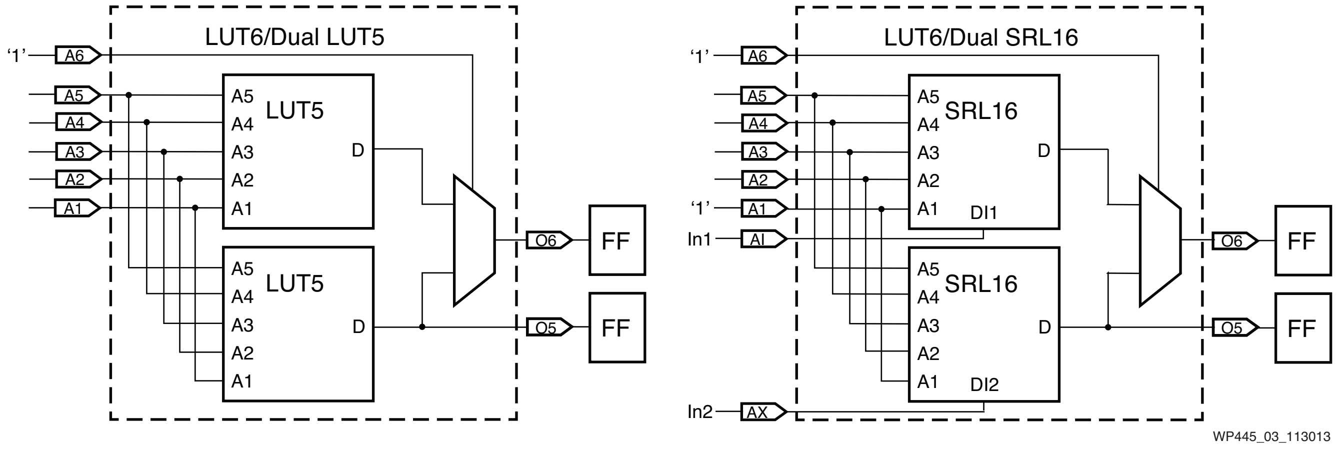


Figure 3: **LUT and SRL Compression**

Two LUT5s implementing a logic function or a memory can, therefore, be packed into a single LUT6 provided that they share the same inputs, or the same read/write address for a memory. When two 2-input LUTs are packed into a single LUT6, they do not necessarily need to share the same inputs. Similarly, two SRL16s can be packed into a single LUT6.

This is useful for digital radio designs, which commonly integrate many small memories sharing the same address (e.g., ROMs storing filter coefficients), and many short delay lines used for latency alignment of different signal paths. Data multiplexing functionalities, especially 2-to-1 multiplexers, are also widely used and can benefit from this LUT-packing capability.

LUT/SRL compression must, however, be used carefully when targeting high clock rates. First, the two flip-flops connected to the O6/O5 LUT outputs within the slice must absolutely be used to avoid any timing issues. For the same reasons, it is recommended that this capability is applied to related logic only, which further limits routing congestion.

When building a complex application, it is also important to define a good hierarchy, which logically partitions the design into appropriate functional blocks. This provides a convenient way to register the outputs of hierarchical boundaries and contain critical paths within specific modules. It is easier to analyze and repair timing paths that are localized within a single module. In effect, when targeting very high clock speeds, it can be worth using multiple register stages at some levels of the hierarchy, to optimize timing and give more freedom to the backend tools.

A good design hierarchy should gather related logic together, which makes area grouping and logic compression more efficient. Defining appropriate area group constraints (PBLOCKs) is actually essential for high-speed designs because it allows the minimization of routing delays, achieving better slice packing, therefore, reducing overall resource utilization and power consumption. It is usually more efficient to define a few "large" PBLOCKs, close to the top level of the hierarchy, rather than constraining a design with too many "small" PBLOCKs. It is also advisable to define each PBLOCK to lie within a single clock region to avoid clock skew issues, although this is not strictly necessary. Actually, having the different PBLOCKs slightly overlapping each other can be beneficial to give more freedom to the tools, in which case they are likely to extend outside of a single clock region.

From a development perspective, building an adequate hierarchy is very helpful to get repeatable results when a module is used more than once. Similarly, applying implementation attributes at the module level can keep the code simpler and more scalable, as the attribute is propagated to all signals declared in that module.

Other aspects related to clocks and control sets must also be carefully addressed. These issues are pertinent to any FPGA design and are extensively described in the Xilinx technical documentation[Ref 14], but they are much more critical at high clock rates. A good methodology for clock management and distribution is essential, and it is particularly important to reduce the number of independent primary clocks as much as possible. The PLL and MMCM resource can be used to generate the additional clocks eventually required, though for the datapath of a radio application, a single clock is usually enough (and is recommended). Placing the clocking elements towards the top level of the design hierarchy is also desirable since it allows easier clock-sharing between multiple modules. This might result in the need for fewer clocking resources, improved timing performance, and lowered resource and power utilization. Finally, appropriate resynchronization techniques must be used between unrelated clock domains.

Similarly, an adequate reset strategy must be defined. It is generally not necessary to have a reset on any single register, and doing so is actually not recommended because it creates very high fanout nets that impact timing performance and increase routing complexity. Resets must, therefore, be used only where absolutely required. Global resets should be avoided. Synchronous resets must be preferred and are actually mandatory for DSP48s and block RAMs.

It is also desirable to limit the usage of clock-enable signals, although this is more difficult to achieve in practice. Indeed, they are typically required to assess the validity of data samples or operator outputs in multicycle implementations, and useful as well to implement efficient power reduction techniques. In any case, clock-enable signals must be appropriately registered to eliminate high-fanout nets.

This also applies to reset nets, in the event that some cannot be avoided.

500 MHz Design on a 7 Series Device

The purpose of this section is to demonstrate that radio signal processing functionality clocked at more than 500 MHz can be implemented within a mid-speed grade 7 series FPGA with high utilization ratios, reflecting the requirements of practical designs. More precisely, the intent is to reach around 75% of the LUT resource, 95% of the DSP48 slices, and 70% of the block RAMs. Part of the remaining logic can further be used for control and interfacing, running at “low” speed. Although not included in this analysis, it is deemed unrealistic to try using more than 85% to 90% of the LUTs with 75% running at 500 MHz, due to the routing and control set constraints.

A test design has been built that integrates signal-processing functionality, representative of radio applications. A high-level block diagram of the proposed subsystem is shown in Figure 4.

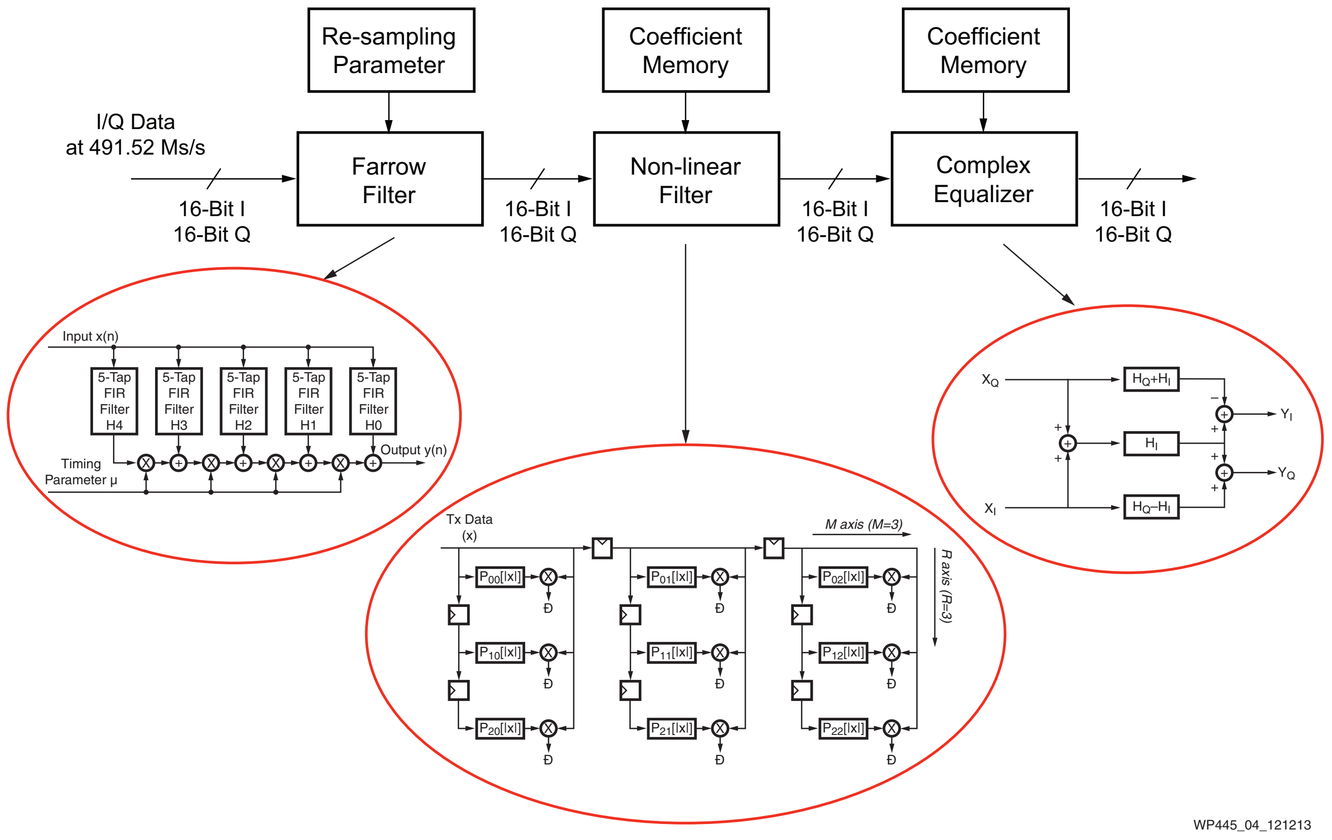


Figure 4: **Radio Subsystem Test Design**

Part of the processing commonly applied at the transmitter side to the composite signal is integrated:

- A resampling function, used for the time alignment of the different antenna paths
- A non-linear filter, representing the datapath of a typical DPD solution
- A complex equalizer, usually applied to correct the linear impairments generated in the signal bandwidth by the analog front end.

A 16-bit datapath is considered, and the signal sampling rate is assumed to be in the range of 500 MHz (e.g., 491.52 MHz), corresponding to that which can be used for a 100 MHz instantaneous bandwidth system. The resampling function is a 4th-order Lagrange interpolator. It is built with a Farrow structure around five 5-tap filters, combined with a cascade of multipliers. The non-linear filter implements a LUT-based Volterra model:

$$y(n) = \sum_{m=0}^{M-1} x(n-m) \sum_{r=0}^{R-1} F_{m,r}(|x(n-m-r)|)$$

where $M = R = 4$.

The non-linear functions $F_{m,r}()$ are based on block RAMs or distributed memories, and the signal modulus is computed with a CORDIC algorithm. Finally, a cascade of two 15-tap complex equalizers is integrated, each one being built with three real filters and coefficient reloading capabilities.

The design was developed with the System Generator tool, which produces a mix of IP (FIR compiler), structural, and inference VHDL code. The target device is a mid-speed grade Kintex-7 XC7K160T device, which comprises 25,350 slices (101,400 LUTs and 202,800 FFs), 600 DSP48 slices, and 650 block RAMs (18K).

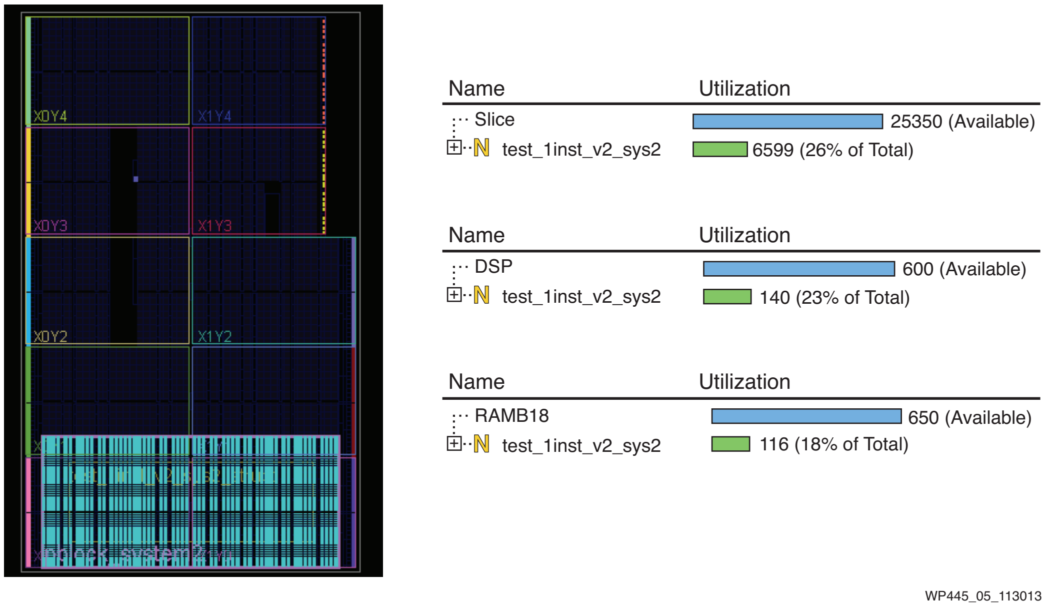
The implementation of this functionality requires 140 DSP48 slices. Hence, up to four instances can be integrated within the XC7K160T device. However, the amount of logic used by a single instance is too low to reach the LUT utilization target with four instances; a device with 10 to 12 DSP48 slices per 1K LUT is needed, while this ratio is only 6 for the XC7K160T. The same holds for the block RAMs. Some logic functionality has, therefore, been added to achieve appropriate utilization ratios.

Since the non-linear filter is the only function using block RAMs, the choice was to add a number of instances of this block, with all the multipliers replaced with adders. To increase the logic content even further, the CORDIC block has been duplicated (applying it to delayed versions of the input instead of simply delaying the output).

The implementation results presented below have been obtained with Vivado Design Suite 2013.2, using a simple push-button flow with a timing constraint set to 1.9 ns (i.e., 526 MHz) for the clock.

A single instance of the design is now considered, including seven additional "logic versions" of the non-linear filter — three using block RAMs for the functions $F_{m,r}()$, the four others using distributed memories to increase the logic and block RAM content to a level compatible with the device resource mix. A PBLOCK is specified containing the number of DSP48 slices just necessary to implement the functionality with minimal logic content. The PBLOCK contains 6,600 slices, 144 DSP48 slices, and 168 block RAMs 18K.

The device floorplan after implementation is shown in Figure 5, demonstrating the effectiveness of the area group constraint. Timing results indicate that a clock rate of up to 530 MHz can be supported. Specifying an area group constraint is not strictly necessary with a single instance of the design, and the same timing performance can be achieved without a PBLOCK. However, the main goal of this first analysis is to evaluate the packing capabilities. In effect, removing the PBLOCK constraint yields the same results for the LUT and FF utilization, but the number of occupied slices increases by 48% to 9,750.



Name	Utilization
Slice	25350 (Available)
test_1inst_v2_sys2	6599 (26% of Total)

Name	Utilization
DSP	600 (Available)
test_1inst_v2_sys2	140 (23% of Total)

Name	Utilization
RAMB18	650 (Available)
test_1inst_v2_sys2	116 (18% of Total)

Figure 5: **Implementation Results for a Single Instance of the Test Design**

The resource utilization results are summarized in Table 4, showing that high occupancy percentages are achieved. In effect, the PBLOCK slices are fully utilized, with 75% of the LUTs and 80% of the FFs effectively occupied. In addition, 97% of the DSP48 slices and 69% of the block RAMs available in the PBLOCK are used.

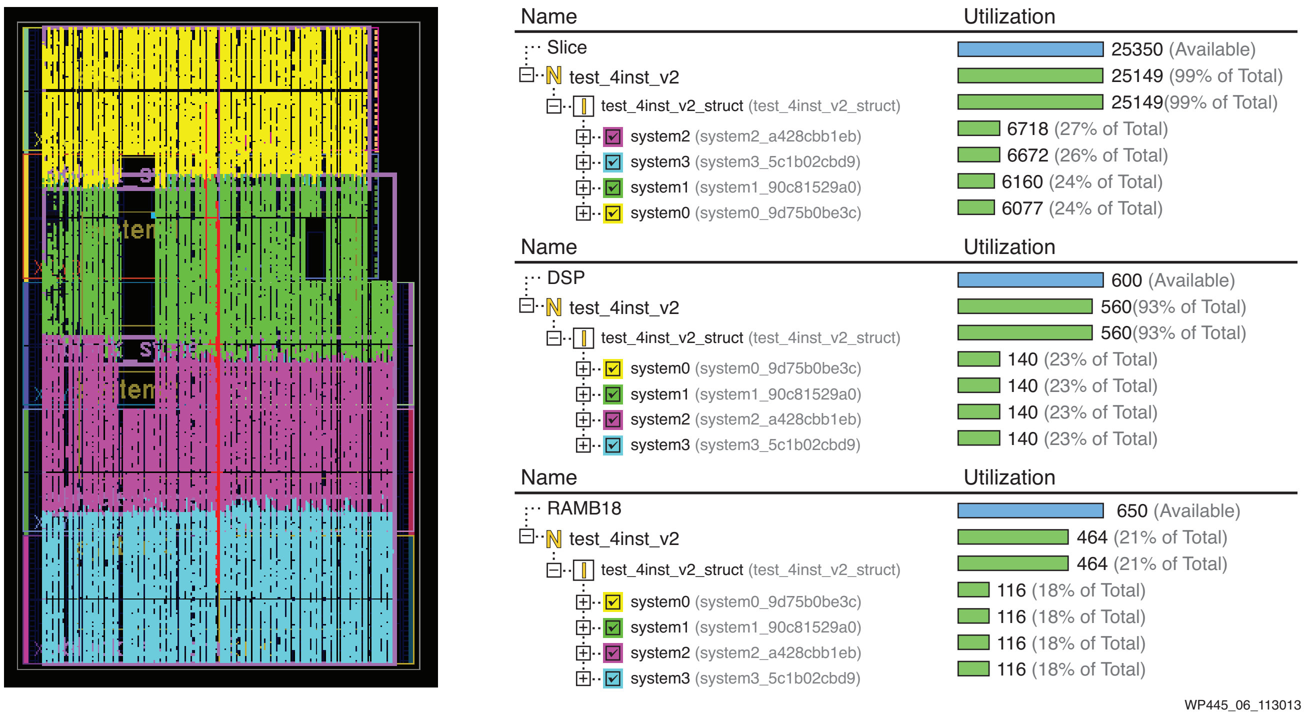
Table 4: **Resource Utilization Summary for a Single Instance of the Test Design**

Resource	Slice	LUT	FF	DSP48	Block RAM (18K)
One-Instance Design	6,599	19,740	42,421	140	116
PBLOCK	6,600	26,400	52,800	144	168
% of PBLOCK Utilization	100%	75%	80%	97%	69%

To demonstrate the ability to implement 500 MHz functionality with a high device utilization ratio, four instances of the previous design are now considered. Two of these instances are exactly the same as used in the previous analysis, including seven "logic builds" of the non-linear filter (three using block RAMs, four using distributed memories), while for the two others, only six "logic builds" of this filter are integrated (three with block RAMs, three with distributed memories).

A PBLOCK has also been specified for each instance, with some slight overlap between each one.

The device floorplan after implementation is shown in Figure 6, together with the resource partition between the four subsystems.



Name	Utilization
Slice	25350 (Available)
test_4inst_v2	25149 (99% of Total)
test_4inst_v2_struct (test_4inst_v2_struct)	25149(99% of Total)
system2 (system2_a428cbb1eb)	6718 (27% of Total)
system3 (system3_5c1b02cbd9)	6672 (26% of Total)
system1 (system1_90c81529a0)	6160 (24% of Total)
system0 (system0_9d75b0be3c)	6077 (24% of Total)

Name	Utilization
DSP	600 (Available)
test_4inst_v2	560(93% of Total)
test_4inst_v2_struct (test_4inst_v2_struct)	560(93% of Total)
system0 (system0_9d75b0be3c)	140 (23% of Total)
system1 (system1_90c81529a0)	140 (23% of Total)
system2 (system2_a428cbb1eb)	140 (23% of Total)
system3 (system3_5c1b02cbd9)	140 (23% of Total)

Name	Utilization
RAMB18	650 (Available)
test_4inst_v2	464 (21% of Total)
test_4inst_v2_struct (test_4inst_v2_struct)	464 (21% of Total)
system0 (system0_9d75b0be3c)	116 (18% of Total)
system1 (system1_90c81529a0)	116 (18% of Total)
system2 (system2_a428cbb1eb)	116 (18% of Total)
system3 (system3_5c1b02cbd9)	116 (18% of Total)

WP445_06_113013

Figure 6: **Implementation Results for Four Instances of the Test Design**

The overall resource utilization results are summarized in Table 5, showing that the occupancy targets are achieved: 99% of the device slices are utilized, with 73% of the LUTs and 79% of the FFs effectively used. In addition, 93% of the DSP48 slices and 71% of the block RAMs available in the device are occupied, with all resources being clocked at more than 500 MHz. The timing results actually indicate that a clock rate of up to 515 MHz can be supported. The usage of area group constraints is mandatory in this case, and without PBLOCKs, the timing performance would drop down to around 400 MHz.

Table 5: **Resource Utilization Summary for Four Instances of the Test Design**

Resource	Slice	LUT	FF	DSP48 Slices	Block RAM (18K)
Four-Instances Design	25,149	73,682	161,207	560	464
XC7K160T Device	25,350	101,400	202,800	600	650
% of Device Utilization	99%	73%	79%	93%	71%

A small number of large PBLOCKs have been used for this analysis, one per subsystem, each containing within 6,350 to 7,200 slices. Defining more PBLOCKs with finer granularity can potentially help to reach higher LUT utilization ratios, but this has not been investigated in the context of this study.

Conclusion

This white paper has demonstrated the capabilities of 7 series All Programmable FPGAs and SoCs to support high clock rate designs, enabling more efficient implementations of digital radio applications. The analysis has focused on the datapath of such applications, integrating the signal processing functions that work directly on the antenna data streams. It is effectively this functionality that can benefit mostly from high clock rate implementation, by exploiting resource sharing resulting from data multiplexing and operator folding. This is usually not applicable for control and interfacing.

The signal datapath essentially consists of linear and non-linear filtering operators (such as the DUC, DDC, and DPD models), correlation engines (DPD update, power estimation, and cross-talk cancellation), complex MAC operators (carrier mixing/extraction, beam-forming, and PC-CFR), and waveforms generators (NCO/DDS and cancellation pulse generators). Depending on the system configuration, it represents about 60% to 80% of the total design logic resource when implemented on Xilinx technology, while control and interfacing use the remaining 20% to 40%. With respect to the other fabric resource, almost all the DSP48 slices and 50% to 60% of the block RAMs are commonly used by the datapath.

A design example, representative of such functionality, has been built using the Vivado Design Suite, and it has shown that a clock rate higher than 500 MHz can be supported on a mid-speed grade 7 series device with very high resource utilization ratios. In effect, almost 100% of the slice resource, more than 90% of the DSP48 slices and 70% of the block RAMs can be utilized, provided that simple design rules are followed. These rules, which encompass both algorithmic and implementation aspects, are described in this white paper.

Although almost 100% of the slices can be used at 500 MHz for pure datapath designs, only 75% to 80% of the LUTs are effectively occupied. This mainly relates to the routing and control set resource constraints. Tool performance also impacts these aspects, especially the backend tools (placer and router), which govern the effective LUT-packing into slices, and the Vivado Design Suite is continuously improving on this front.

The datapath design considered in this analysis is not fully representative of real radio applications, which also integrate interfacing and control functionality. The intent of this exercise, however, is merely to explore the limits of the fabric capabilities for signal processing functionality.

References

1. Backgrounder Dec. 2012, *Moving a Generation Ahead with All Programmable FPGAs, SoCs and 3D ICs*
2. DS180, *7 Series FPGAs Overview*, Xilinx Data Sheet
3. DS190, *Zynq-7000 All Programmable SoC Overview*, Xilinx Data Sheet
4. Backgrounder Jul. 2013, *Introducing UltraScale Architecture: Industry's First ASIC-Class All Programmable Architecture*
5. WP435, *Xilinx UltraScale: The Next Generation Architecture for Your Next Generation Architecture*, Xilinx White Paper
6. UG479, *7 Series DSP48E1 Slice*, Xilinx User Guide
7. UG473, *7 Series FPGAs Memory Resources*, Xilinx User Guide
8. UG474, *7 Series FPGAs Configurable Logic Block*, Xilinx User Guide
9. UG476, *7 Series FPGAs GTX/GTH Transceivers*, Xilinx User Guide
10. PG149, *LogiCORE IP FIR Compiler*, Xilinx Product Guide
11. PG141, *LogiCORE IP DDS Compiler*, Xilinx Product Guide
12. PG097, *LogiCORE IP Peak Cancellation Crest Factor Reduction*, Xilinx Product Guide (registration required to obtain this document)
13. PG076, *LogiCORE IP Digital Pre-Distortion*, Xilinx Product Guide
14. UG949, *Design Methodology Guide for the Vivado Design Suite*, Xilinx User Guide

Revision History

The following table shows the revision history for this document:

Date	Version	Description of Revisions
01/20/14	1.0	Initial Xilinx release.

Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at http://www.xilinx.com/ legal.htm#tos.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.