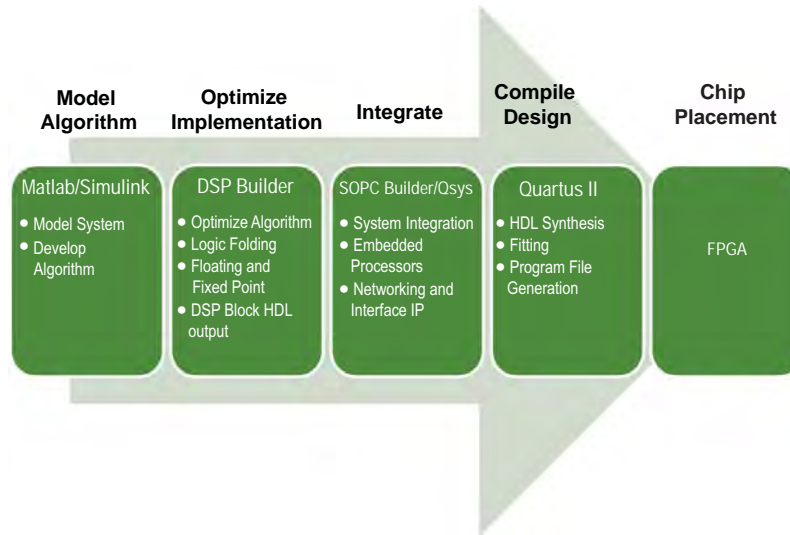


This document describes a recommended design flow that leverages Altera FPGAs' adaptability, variable-precision DSP, and integrated system-level design tools for motor control designs. Designers of industrial motor-driven equipment can take advantage of the performance, integration, and efficiency benefits of this design flow.

Introduction

Industrial motor-driven equipment accounts for more than two-thirds of industrial energy consumption, making their efficient electrical operation a vital component in factory expenses. The replacement of traditional drives with variable speed drives (VSD) in motor-driven systems provides significant efficiencies that can translate to up to 40% in energy savings. Altera's FPGA architectures provide effective platform for VSD systems because of the following flexibility, performance, integration, and design flow advantages illustrated in Figure 1:

Figure 1. Optimized Motor Control FPGA Design Flow



- Performance Scaling—Achieve higher performance and efficiency on different types of motors through parallelism and scalability of functionality.
- Design Integration—Integrate an embedded processor, encoder interfacing, DSP motion control algorithms, and industrial networking in a single device.
- Design Flexibility—Reuse IP and take advantage of variable-precision DSP blocks. Use fixed- or floating-point precision for any part of the control path.

- Deterministic Latency—Implement motor algorithms, and deterministic operations in hardware.
- Powerful streamlined tools—Use of a modeling tool such as Simulink, combined with Altera’s DSP Builder, and a versatile integration tool in Qsys or SOPC Builder, optimizes the full motor system in a low cost FPGA. Although it is common to use off-the-shelf microcontrollers (MCU) or digital signal processors (DSP) to implement processing and control loops that monitor load and adjust position, velocity, and other drive aspects, microcontrollers are limited by their lack of scalability and performance. These deficiencies are most evident in systems of increasingly complex algorithms with high MIPS processing requirements. In addition, writing algorithms in software does not translate easily to hardware-optimized system requirements.

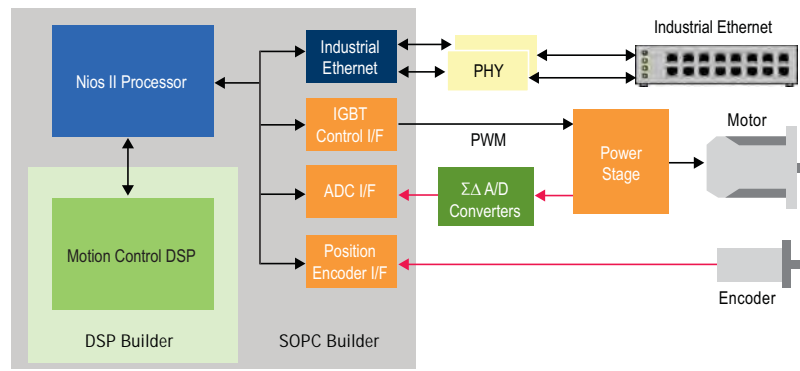
Similarly, while high-end DSPs typically have the power to handle motor control computations, high-end DSPs are not ideal in a system that simultaneously incorporates time-precise operations with task-oriented operations, such as memory interfacing, signal interfacing and filtering, or supporting an Industrial Ethernet protocol standard.

Performance Scaling and Integration Advantages

Many off-the-shelf MCUs or DSPs have the basic elements for general purpose drive operation. However, these devices have fixed memory, narrow analog range, fixed channel of PWMs, and limited support for multi-axis systems. Next generation drives that require more performance and improved motor efficiencies require a platform that provides performance scaling capabilities that correspond with the processing and DSP requirements, while simultaneously providing the flexibility to integrate and optimize the system.

FPGAs can easily scale performance based on the application requirements. Designers can embed multiple processors or use the flexible DSP capabilities in the FPGA, and then leverage additional logic, custom instructions, or one of the many supported industrial networking protocols. Altera FPGA’s enable designers to implement multiple embedded processors to control each subsystem independently. The parallel nature of Altera FPGAs supports integration of most motor control system building blocks. For example, Altera’s Nios II embedded processor (32-bit RISC soft processor) can control all of the various interfaces and sensors and encoders. Designers can then use variable-precision, floating-point DSP blocks to perform field oriented control (FOC) or other math-intensive algorithms.

Figure 2 illustrates the variety of elements that can be integrated in the FPGA to create a single “drive-on-a-chip” system. Integrated IP functions can run in parallel, ensuring that there are no bottlenecks in either sequential or time-delayed operations

Figure 2. FOC Model Including Complex Math Algorithms

This design flow supports integration of useful IP, including the following:

- **Position Feedback**—Encoders with high precision position feedback, such as EnDAT, Hiperface, and BiSS allow 10x faster speed and position data.
- **IGBT Control**—Use Insulated Gate Bipolar Transistors (IGBTs) to switch high voltages required to drive AC motors. Using Space Vector Modulation (SVM) techniques in the FPGA to Pulse Width Modulate (PWM) the gate input of the IGBTs to generate the sinusoidal voltage wave necessary to drive the motor. The IGBTs can be of 2-level or 3-level varieties.
- **ADC Interface**—Interfacing to an external analogue-to-digital converter (ADC) to measure current feedback from the motor. Sigma-delta ADCs are easier to opto-isolate from high drive voltages, have lower noise, and support sampling of their outputs by the FPGA to give fast and accurate readings.
- **Networking Interface**—Implement real-time protocols in the FPGA to accommodate the Industrial Ethernet protocol standards required for the application, such as Ethernet/IP, PROFINET IO/IRT, and EtherCAT. Industrial Ethernet is becoming a more common feature in industrial drives.

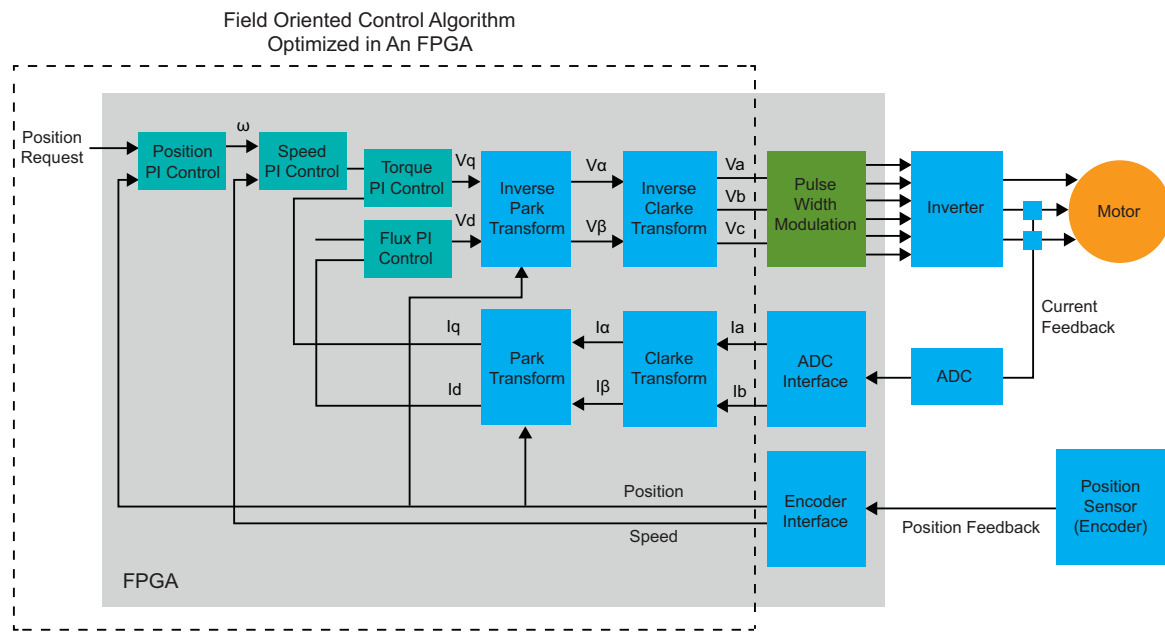
The proliferation of these DSP-based motor control functions, communications, and interface standards make FPGAs an ideal platform for industrial motor drives.

Handling Complex Math Algorithms

Drive technologies vary widely, depending on the motor type and the application. For example, a drive that controls pumps and fans has different requirements and feedback mechanisms from those that control CNC machines or packaging equipment. The data gathered from these encoders and sensors is fed back to the control system for use in math algorithms to determine the correct voltage level for the target system load and torque requirements.

For example, the commonly used Permanent Magnet Synchronous Motor (PMSM) uses a math intensive Field Oriented Control (FOC), also known as vector control, as part of the control loop algorithm. FOC is also useful in industrial servo motors that require precise torque control. FOC techniques help to reduce motor size, cost, and power consumption. FOC provides improved speed and torque control by metering precise voltage levels and corresponding motor speed to provide a constant torque even with varying load. FOC also reduces torque ripple and electromagnetic interference. However, as shown in Figure 3, this math model is fairly complex, and running this algorithm at very high speed requires significant computing power.

Figure 3. Field Oriented Control (FOC) Model



FOC involves controlling the motor's sinusoidal 3-phase currents in real time to create a smoothly rotating magnetic flux pattern, where the frequency of rotation corresponds to the frequency of the sine waves. The technique controls the amplitude of the current vector to maintain its position at 90 degrees with respect to the rotor magnet flux axis ("quadrature" current). This allows designers to control torque while keeping the "direct" current component (0 degrees) at zero. The algorithm involves the following steps:

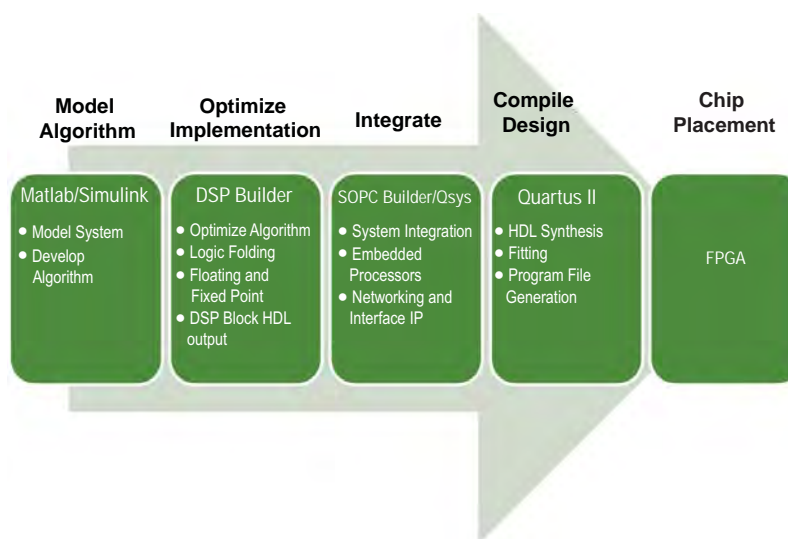
1. Convert the 3-phase feedback current inputs and the rotor position from the encoder into "quadrature" and "direct" current components using Clarke and Park transforms.
2. Use these current components as the inputs to two proportional and integral (PI) controllers running in parallel to limit the "direct" current to zero and the "quadrature" current to the desired torque.
3. Convert the "direct" and "quadrature" current outputs from the PI controllers back to 3-phase currents with inverse Clarke and Park transforms.

Altera FPGAs, with the industry's first variable-precision DSP block, provide the flexibility to choose the precision level that exactly matches the requirements, and also supports single- or double-precision floating-point types. These factors make the DSP block an ideal choice for implementing FOC control loops and other complex math algorithms. The integrated DSP block—a feature in many of Altera's 28-nm FPGA architectures—allows configuration of each block at compile time in either 18-bit or high-precision mode.

Leverage Powerful Development Tools

Optimizing motor control designs requires versatile tools (and a practical tool flow) to help model and simulate the system, implement complex algorithms with low latency, and have the ability to integrate the system together and fine tune the performance to the exact needs of the motor drive. Using the integrated tool flow shown in [Figure 4](#) allows designers to take advantage of features that reduce development time, and provide a more flexible, powerful model that is scalable for different types of drive systems.

Figure 4. Optimized Motor Control FPGA Design Flow



Altera provides embedded industrial designers with powerful and easy to use development tools—such as the Quartus® II design software, MegaCore® IP library. Altera also provides system integration tools, such as Qsys or SOPC Builder for task-oriented operations, and DSP Builder for DSP optimization, as illustrated in [Figure 4](#). In addition, Altera offers the Eclipse-based Nios® II Embedded Design Suite (EDS)—that complement the FPGA hardware to streamline your design flow.

Nios II Embedded Design Suite

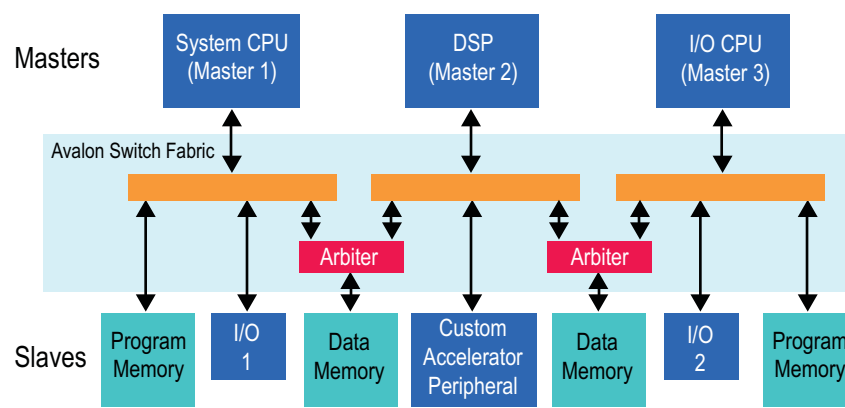
Altera provides powerful and easy to use embedded development tools, such as the Eclipse-based Nios II EDS, the Nios II embedded processor, and support for standard operating system (OS) and real-time operating system (RTOS) from a number of popular vendors. The Nios II EDS supports instantiation of multiple, general purpose 32-bit RISC soft processors. These processors are capable of performance up to 340 MIPS (Dhrystones 2.1), and can run independently with its own custom instruction set, data path, and address space.

Qsys and SOPC Builder System Integration Tools

Altera's Quartus II development software includes the latest Qsys (and legacy SOPC Builder) system integration tools. These tools help designers to define and generate a complete system on a chip (SoC) by automating the task of integrating hardware components, as shown in Figure 7. Rather than using traditional design methods to define and connect HDL modules manually, Qsys or SOPC Builder help you define system components in a GUI and then generates the interconnect logic automatically. These tools generate HDL files that define all components of the system, and a top-level HDL file that connects all the components together. These tools generate either Verilog HDL or VHDL.

These system integration tools use an Avalon interface to connect any logical device (either on-chip or off-chip). In motor systems, Avalon interfaces connect the soft processor with the other elements of the drive system. This reduces the complexity in the system integration and provides a more cohesive and intuitive system for optimization. The interconnect fabric manages these connections by allowing for simultaneous multimastering through slave-side arbitration. These tools insert arbitration modules in front of each slave port that manage requests from the different masters, and abstracts the system's interconnect details from master and slave ports alike. Figure 5 shows the interconnect fabric connecting multiple slaves and masters in a system.

Figure 5. Manage System Interconnect with Qsys or SOPC Builder



Altera's system integration and embedded development tools help designers quickly build the interfaces that connect a processor to a hardware accelerated motor control algorithm designed in DSP Builder.

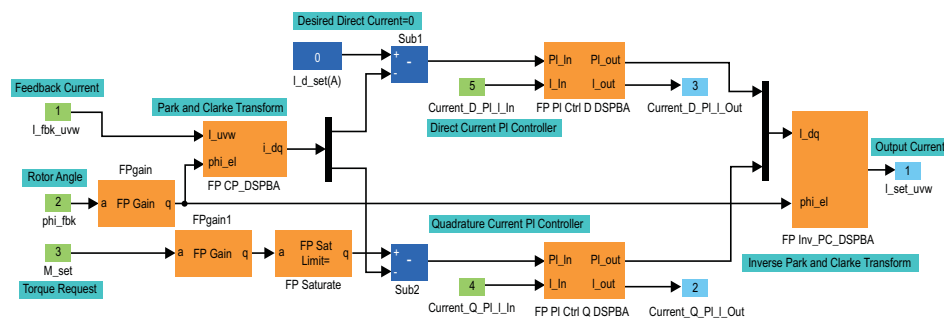
DSP Builder and Matlab/Simulink

Motor control system designers can take advantage of DSP capabilities in FPGAs for high speed, math intensive motor control algorithms. Altera provides DSP Builder to shorten DSP design cycles by helping designers create a hardware representation of a DSP design in an algorithm-friendly development environment. DSP Builder integrates the algorithm development, simulation, and verification capabilities of The MathWorks MATLAB® and Simulink® system-level design tools with the Altera Quartus II software and third-party synthesis and simulation tools. You can combine Simulink blocks with DSP Builder blocks and IP blocks to verify system level specifications and perform simulation.

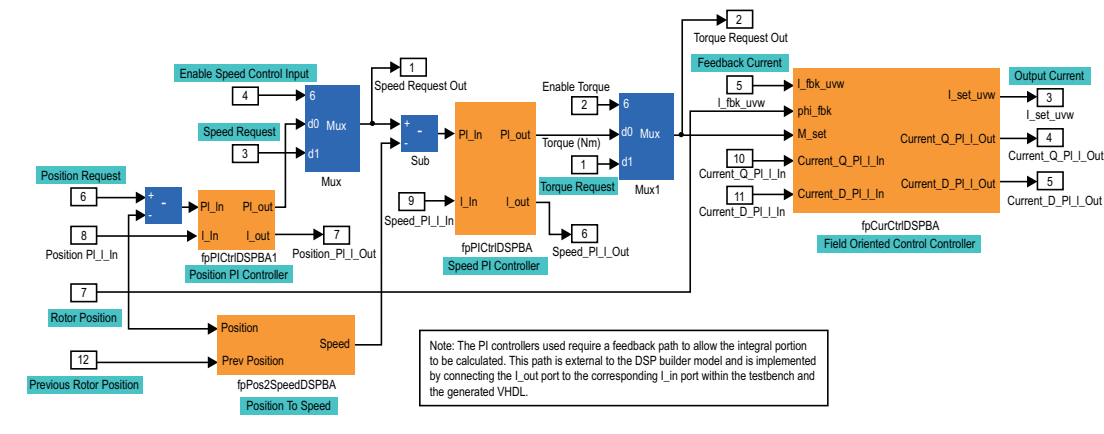
Modeling FOC in an FPGA

Altera's integrated DSP Builder tool allows designers to model the FOC algorithm directly in a Matlab/Simulink environment by constructing a block diagram that connects primitive blocks that represent your system, as shown in Figure 6. The primitive blocks used in this example are ADD, SUB, MULTIPLY, CONSTANT, COSINE, and SINE. Simulink allows you to run bit-accurate mathematical simulations of the behavior of the algorithm against a model or system. When you have finished developing your algorithm, DSP builder automatically generates pipelined RTL that is targeted and optimized for your chosen Altera FPGA device.

Figure 6. Example FOC for Permanent Magnet Synchronous Machine



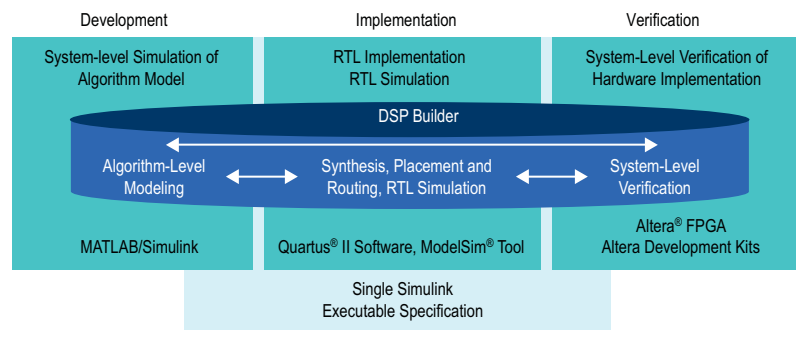
This design example also includes position and speed control loops, which allow the control of rotor speed and angle, as shown in Figure 7. A typical motor control IP system includes a Space Vector PWM, current and torque control loops, and speed and position control loops. Depending on the FPGA and CPU resource utilization, designers can partition these elements between a hardware and software implementation.

Figure 7. Position, Speed, FOC Controller for Permanent Magnet Synchronous Machine


A typical legacy design flow includes creating a model of the electrical and motor system, developing the algorithm through simulation, and then writing the 'C' code that implements the algorithm to run on the DSP. This design flow has the following important drawbacks.

- Algorithm modeling is often carried out with floating point and subsequently translated to fixed point for implementation on the DSP. The floating-to-fixed conversion was previously a manual process in which scaling and overflow protection are complicated.
- The 'C' code implementation must be re-verified against the model.
- Increasing the algorithm run-time performance requires the following additional steps:
 - a. Hand optimize the 'C' code for better performance.
 - b. Upgrade to a faster more expensive DSP processor.
 - c. Run the algorithm in parallel (if possible) on more than one DSP device.

Designers can optimize an FPGA-based DSP system design with DSP Builder. DSP Builder performs optimizations such as pipelining and resource sharing to produce an efficient RTL representation. You can combine existing MATLAB functions and Simulink blocks with Altera DSP Builder blocks and other IP cores to link system-level design and implementation with DSP algorithm development, as shown in [Figure 8](#). DSP Builder allows system, algorithm, and hardware designers to share a common development platform.

Figure 8. DSP Builder System Level Design

The DSP Builder advanced blockset natively supports algorithm modeling with fixed-point, or single- or double-precision floating-point types. Designers can initially model the algorithm in Simulink using a higher precision than necessary, and then scale the precision within the tool for final implementation. DSP Builder provides the following specific advantages:

- Push-button FPGA implementation of the algorithms with the advanced blockset. No manual conversion steps are necessary.
- Directly observe run-time latency, data throughput, and algorithm usage results in Simulink before running in hardware. Perform design space exploration with Simulink to choose the most suitable implementation.
- Optimize and fix primitive operators at generation time, including functions such as SQRT & trigonometric, which are typically slow with a variable runtime when implemented in software. This technique provides a predictable algorithm runtime and significant acceleration to some operators.

 For more information about DSP Builder, including the standard and advanced blocksets, visit the [DSP Handbook](#) website.

Folding Theory Performance Improvements

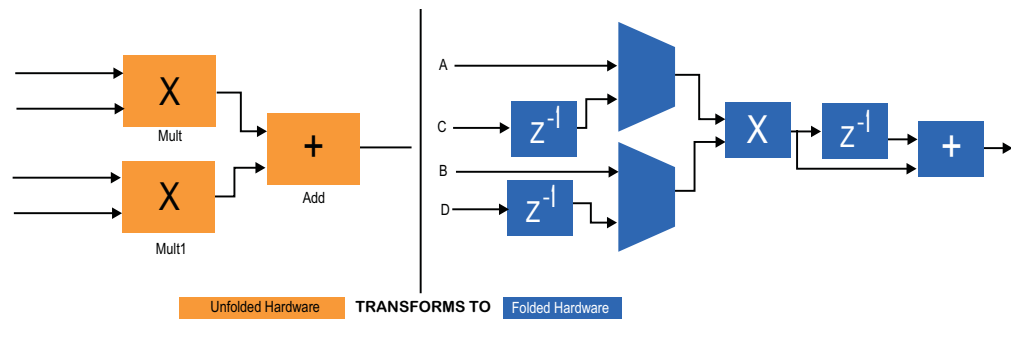
Folding theory, which is closely related to Time Division Multiplexing (TDM), can greatly improve DSP motor control designs. When the system clock rate is faster than the data or sample rate, a single hardware component (such as a multiplier) can potentially process multiple data points. Folding theory allows multiple channels to access system resources such as multipliers and adders in a similar way to the TDM factor, thus resulting in resource savings.

Different data points use the TDM factor to access the shared hardware resource. Similarly, in a system with multiple parallel data sources or data channels, instead of duplicating hardware for each channel or data source, designers can use one datapath to process multiple data channels.

By default, the hardware that DSP Builder generates for a primitive subsystem can receive and process new data every clock cycle. However, some designs may not require a computation every clock cycle. For those designs with a sample rate lower than the clock rate, DSP Builder advanced blockset's folding functionality can take advantage of the disparity between both rates to optimize the use of generated

hardware. Designers can implement the core algorithm in the most intuitive way, as if there was no folding or TDM factor. With folding theory there is no requirement to explicitly implement signal multiplexing and data buffering schemes, which is normally required in manually folded designs. Folding can reduce hardware in a combination of blocks that are not used every cycle, as illustrated in Figure 9.

Figure 9. Unfolded and Folded Hardware Examples



Use the following terminology guidelines when making performance comparisons between DSP builder and a processor to ensure accurate latency and throughput measurements:

- A single core processor takes a latency of x clock cycles to process one calculation, and cannot start a new calculation until the first completes. The throughput is therefore one calculation per x clock cycles.
- A DSP builder system takes a latency of y clock cycles to process one calculation, but can start a new calculation every “folding factor” clock cycles. The throughput is therefore one calculation per “folding factor” clock cycles.

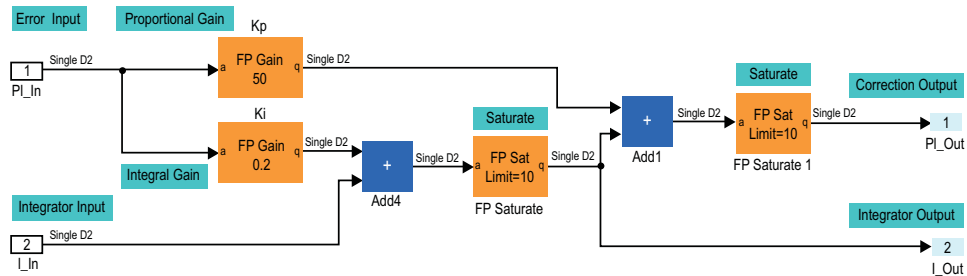
By tuning the folding factor you can trade-off the throughput, resource usage, and latency of the generated logic without requiring redesign. The next section presents an example system and testbench that demonstrates the impact of folding factor. In this context, the “folding factor” clock cycles is smaller than x clock cycles.

Benchmarking the FOC Algorithm

To highlight the advantage of the various features and functions described above, Altera developed a benchmarking exercise to simulate the FOC algorithm and compare the results between a standard “non-folded” and “folded” implementation. The FOC algorithm consists of Park/Clarke transforms and PI control blocks that require add/subtract, Multiply, Sin, Cos operators, along with some constant values

and saturation logic for implementation, as shown in Figure 10 and Figure 11. In a typical FOC controller the inputs are sampled at 10-100 ksp/s, a rate easily handled by the 100 MHz FPGA clock rate. At 100 ksp/s, a new sample must be processed every 10us. Keeping this processing latency constant and to a minimum help the performance of the control algorithm.

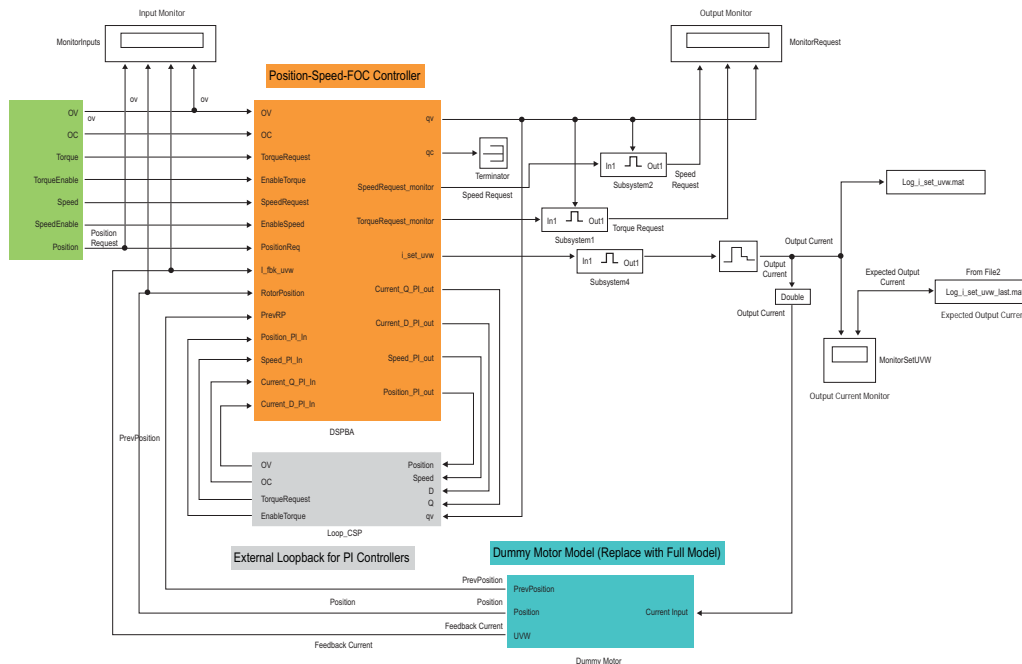
Figure 10. Simple Proportional and Integral (PI) Controller



The testbench includes the following elements shown in Figure 11:

- Input stimuli (green)—provides control inputs (position request)
- Motor model (cyan)—models PMSM motor
- DSP Builder Position-Speed-FOC Controller (yellow)—models control algorithm
- External loopback model (grey)—loops the integrator feedback output from PI controllers back to input

Figure 11. Position-Speed-Field Oriented Control Controller for Permanent Magnet Synchronous Machine



Design Tuning with DSP Builder

Designers can control key system parameters from Matlab workspace variables, allowing tuning of the design in the following scripted ways:

- Folding factor—Allows sweep of latency, throughput, and resource usage trade-offs to find the implementation sweet spot
- Fixed-point arithmetic precision—Observe the effect of tuning precision at different stages in the algorithm, for algorithm performance and resource usage
- Algorithm tuning—Simulate the actual algorithm against a physical model of the plant (motor) and tune parameters for PID controllers, filters, and observers at the modeling stage

Benchmark Results

The following section details the algorithm benchmarking results achieved through modeling in Simulink using single-precision floating-point and fixed-point types implemented in a Cyclone IV device. The results indicate that the design example meets the required 100 MHz clock rate, resource usage, and algorithm latency requirements.



After a successful compilation in the Quartus II software, designers can obtain accurate resource information by clicking on the Quartus block link in the Simulink diagram.



Typically a design that does not require the high dynamic range afforded by floating point is implemented in fixed point. However, floating point avoids arithmetic overflow during algorithm development and tuning.

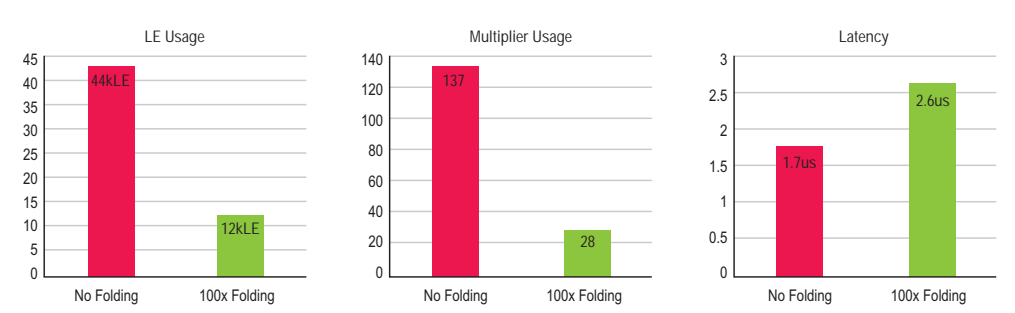
By default, DSP builder creates a fully pipelined VHDL representation that can accept new input values every clock cycle. The result obtained for this “unfolded” configuration was then compared to a “fully folded” configuration.

The results in [Table 1](#) illustrate the significant reduction in the operator count as a result of the folding factor, which allows use of a lower density Cyclone IV device. In addition, the latency increase remains acceptable for the algorithm. The speed of the control loop is the algorithm latency, plus the settling time. At 5 μ s, the results are 200k loops (or pwm outputs) a second, well within the required specification.

Table 1. Folding Factor Advantage

Specification	No Folding	Folding Factor 100x
Addsub blocks	22	1
Multiplier blocks	22	1
Sin blocks	4	1
Maximum throughput	100 Msps	1 Msps

Figure 12. System Resources and Latency



DSP Builder allows both fixed- and floating-point implementations. Table 2 presents a comparison of the resources required for “fully folded” fixed- and floating-point implementations. The fixed-point precision is controlled using Matlab workspace variables that allow designers to perform simple “what if” experiments with the design.

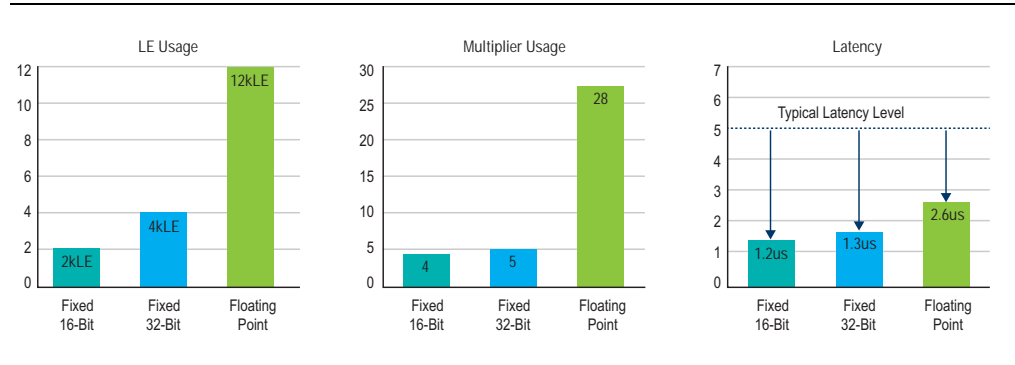
Table 2. Floating-Point vs. Fixed-Point Comparison

Specification	Fixed 16-Bit	Fixed 32-Bit	Floating Point
LEs	2K	4K	12K
18-bit multipliers	4	5	28
Latency	1.21 μ S	1.36 μ S	2.65 μ S

Table 2 Notes:

- (1) The floating point result uses a floating-point sine implementation. Using an alternative fixed-point implementation reduces the LE usage by 4K LEs, and multiplier usage by 16

Figure 13. System Resources and Latency



Results Summary

The benchmarking experiments illustrate the following conclusions:

- Using the FOC model and floating-point precision does slightly impact LE resource utilization. However, latency increases only slightly (still under 5 us), providing an acceptable latency level for the operation.
- Lowering the precision to 16-bit also lowers resource usage because of the narrower data path.
- The folding factor provides optimized hardware resources while still allowing 1 Msps throughput. This allows real-time processing of up to 10 channels of the 100 ksps FOC algorithm.

Conclusion

Today's modern MCUs and DSPs are being pushed beyond their performance ranges in next generation motor control systems. Designers require the flexibility to fine tune motor control algorithms to reduce cost and power. Off the shelf DSP solutions have limited fixed-point or floating-point capabilities that do not accommodate other components required to drive the system.

Conversely, Altera FPGAs allow integration of components, such as a processor that can manage the overall operation, flexible interfacing to easily connect to customized subsystems, and an optimized design flow that simplifies the complex motor control loops and algorithms in parallel. A motor system is a combination of various fast control loops, timed output pulse frequencies, as well as multi-sensor interfacing and filtering required. Altera FPGAs inherent parallel processing capabilities and high performance variable-precision DSP blocks, reduce bottlenecks and provide an optimal solution for motor control systems.

In addition to these inherent FPGA advantages, Altera provides the optimum design methodology. Using The Mathworks Simulink/Matlab tools for modeling, Altera's DSP Builder for motor algorithm optimization, Qsys or SOPC Builder for system integration, and the Quartus II software for design synthesis and fitting, represents a comprehensive and integrated design methodology that can tackle the most complex drive systems.

Further Information

- Altera Industrial Website
www.altera.com/end-markets/industrial/ind-index.html
- White Paper: *Lowering the Total Cost of Ownership In Industrial Applications*
www.altera.com/literature/wp/wp-01122-tco-industrial.pdf
- White Paper: *A Flexible Solution For Industrial Ethernet*
www.altera.com/literature/wp/wp-01037.pdf
- White Paper: *Developing Functionally Safe Systems With TuV-Qualified FPGAs*
www.altera.com/literature/wp/wp-01123-functional-safety.pdf
- Webcast: *Achieve Lower Total Cost of Ownership for Industrial Designs*
www.altera.com/education/webcasts/all/wc-2010-lower-tco-for-industrial-designs.html

- Video: *3 Ways to Quickly Adapt to Changing Ethernet Protocols*
www.altera.com/education/webcasts/videos/videos-adapt-to-changing-ethernet-protocols.html
- More Industrial Videos and Webcasts:
www.altera.com/servlets/webcasts/search?endmarket=industrial

Acknowledgements

- Kevin Smith, Sr. Member of Technical Staff, Altera Corporation
- Wil Florentino, Sr. Technical Marketing Manager, Altera Corporation
- Jason Chiang, Sr. Technical Marketing Manager, Altera Corporation
- Stefano J. Zammattio, Product Manager, Altera Corporation

About Altera

As the programmable logic pioneer, Altera delivers innovative technologies that system designers can count on to rapidly and cost effectively innovate, differentiate, and win in their markets. With our fabless business model, we can focus on developing technologically advanced FPGAs, CPLDs, and HardCopy® ASICs.

Using an Altera industrial-grade FPGA as a coprocessor or SoC brings flexibility to industrial applications. Providing a single, highly integrated platform for multiple industrial products, an Altera FPGA can substantially reduce development time and risk. Altera FPGAs offer the following advantages:

- Design integration through hard IP blocks, embedded processors, transceivers, and other functions-to increase application functionality and lower total costs
- Reprogrammability, even in the field, to support evolving Industrial Ethernet protocols and changing design requirements
- Performance scaling via embedded processors, custom instructions, and DSP blocks
- Obsolescence protection, plus a migration path to future FPGA families, which supports the long life cycles of industrial equipment
- Familiar tools, use familiar, powerful, and integrated tools to simplify design and software development, IP integration, and debugging.

Document Revision History

Table 3 shows the revision history for this document.

Table 3. Document Revision History

Date	Version	Changes
July 2011	1.1	■ Minor text edits.
April 2011	1.0	Initial release.