

## *Design Solutions*



## AccelChip Introduction

2000

AccelChip's technology is based on research performed at Northwestern University in Chicago, Illinois from 1998 to 2000

2001

AccelChip was founded and granted an exclusive license to commercialize the Northwestern technology

2002

AccelChip introduced its first product, AccelFPGA, the first commercial synthesis tool for the MATLAB language

2003

AccelChip released its second generation product, AccelChip, and AccelWare, libraries of fixed-point DSP models for hardware design

**Headquarters: Milpitas, California**

Dan Ganousis, President and CEO

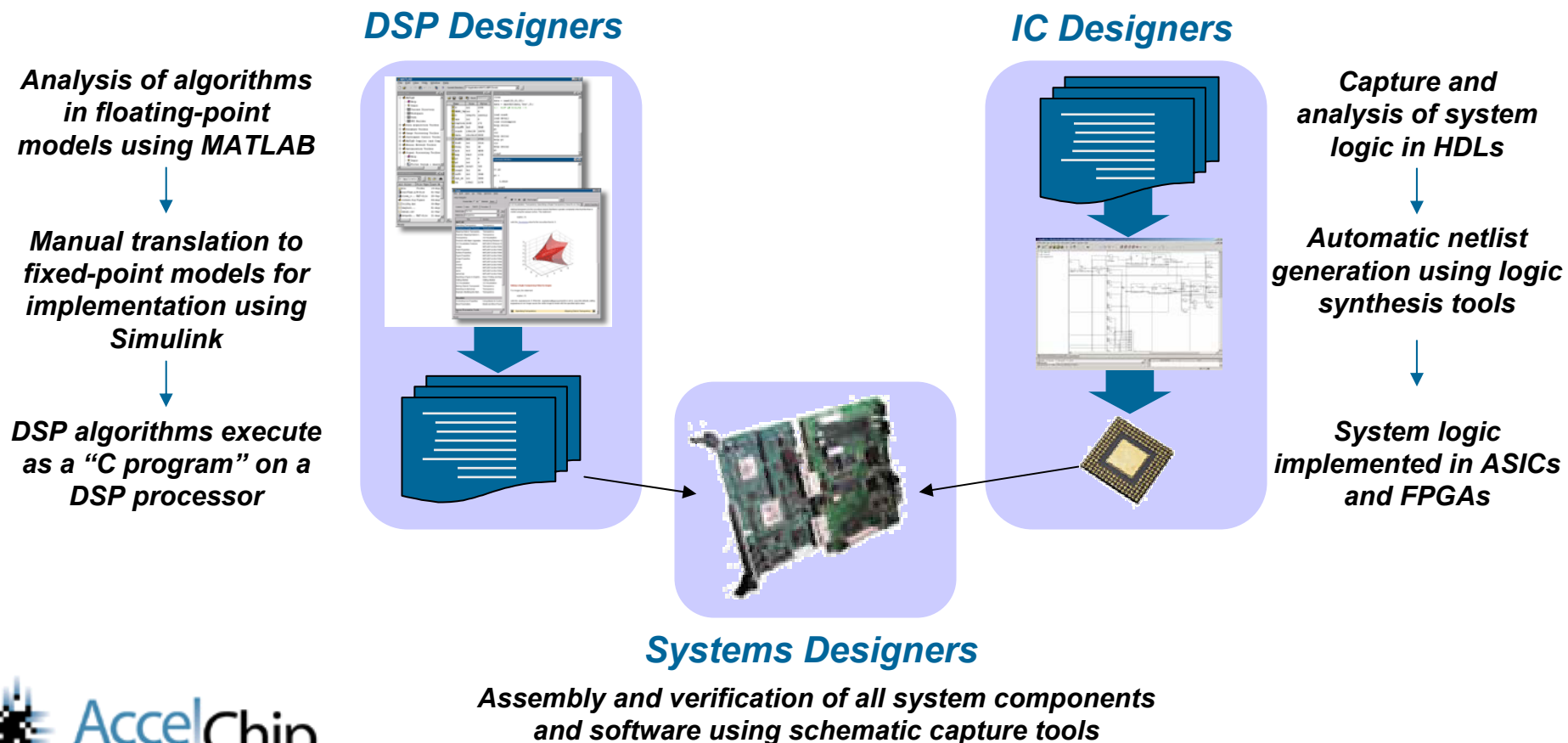
**Product Development: Portland, Oregon**

Michael Bohm, Chief Technical Officer



## DSP Algorithms in Software

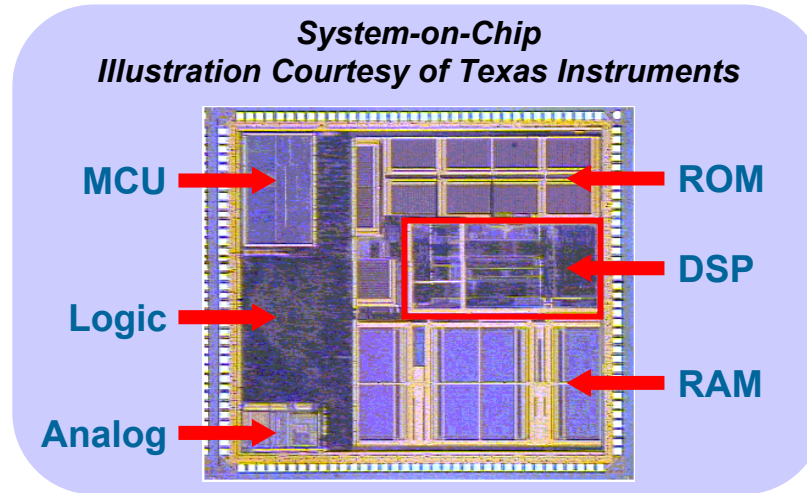
The majority of designers develop and implement DSP algorithms in MATLAB and Simulink, then execute the algorithms on standard DSP processors





## Advantages of DSP Algorithms in Silicon

- Higher performance than DSP processors
- Increased integration of system components
  - Reduced cost, power, and form factor
  - Improved reliability and manufacturability

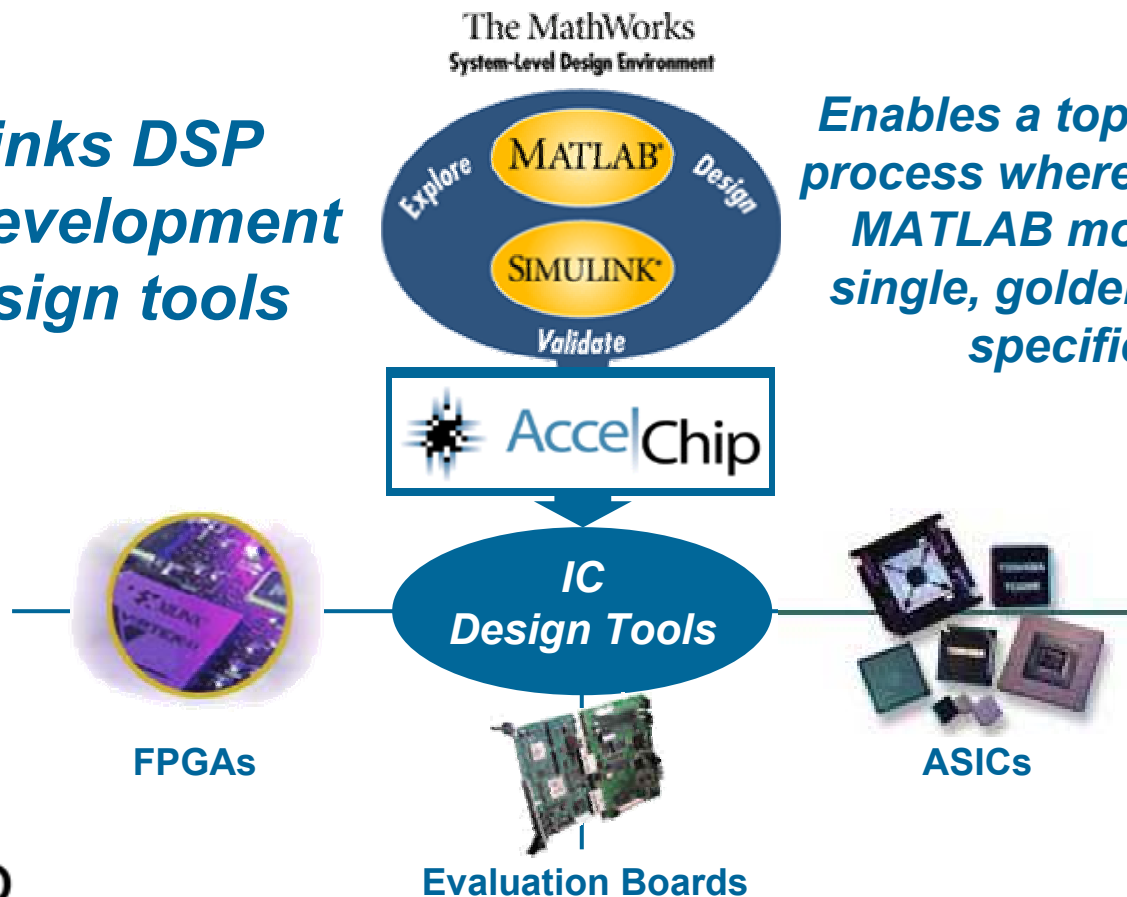




## Enabling DSP Algorithms in Silicon

AccelChip provides innovative design tools and fixed-point DSP models that accelerate the implementation of DSP algorithms in silicon with high quality of results

*Directly links DSP algorithm development with IC design tools*



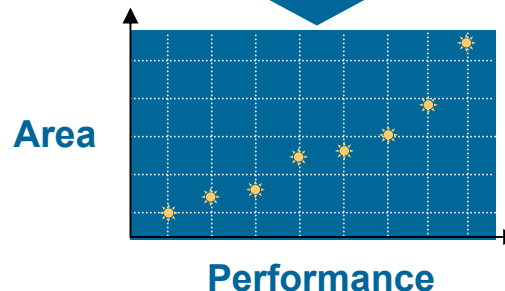
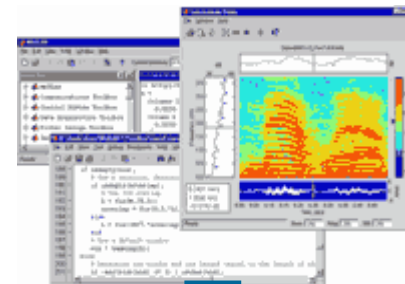
*Enables a top-down design process where floating-point MATLAB models are the single, golden, executable specifications*



## Design Space Exploration

An automated path from MATLAB to IC design tools allows algorithm developers to optimize their models based on accurate estimates of area and performance in different ICs

***Quickly acquire accurate estimates of performance and area in any IC device***



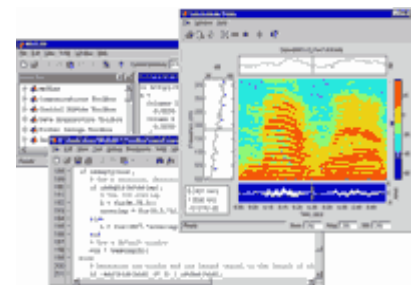
***Enables designers to easily and accurately explore DSP algorithm implementation tradeoffs in different IC devices***



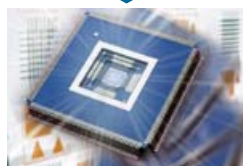
## Optimal Quality of Results

AccelChip's innovative **Technology Aware Automation** produces high quality of results in FPGA implementation

AccelChip's Resource Description Language (RDL) defines the internal architecture and embedded DSP resources of each FPGA vendor's device to enable the AccelChip compiler to make intelligent decisions for FPGA implementation

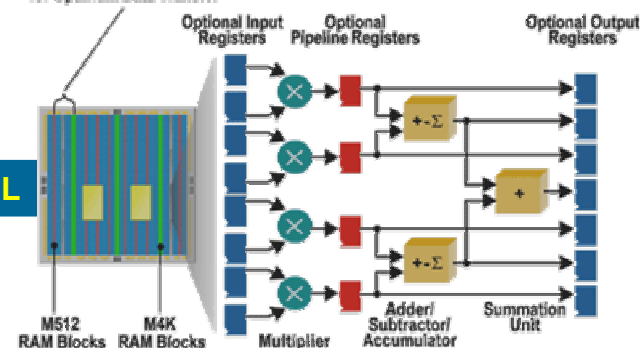


**Structural  
RTL Models**



Optimal IC Implementation

Memory & DSP Blocks Placed for Optimum Data Transfer



*Internal Architecture of  
Altera Stratix devices*





## Enables Transition from Prototype to Production

Vendor independent designs can be automatically retargeted to an ASIC implementation for lower cost production device

*Models are not tied to a specific IC vendor or device allowing retargeting*

*Investment in DSP algorithm development can be leveraged in next generation products*

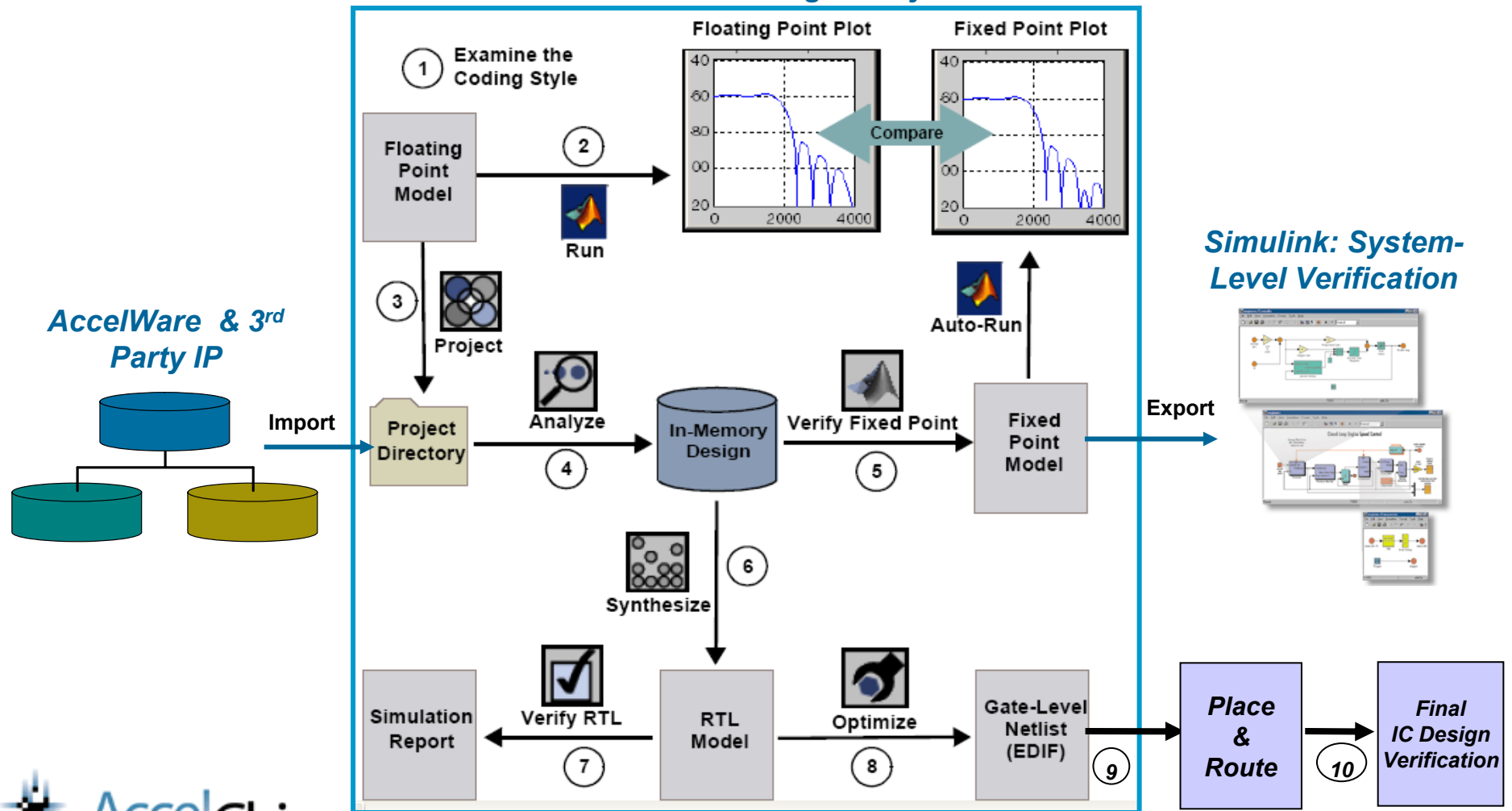






# AccelChip Design Process

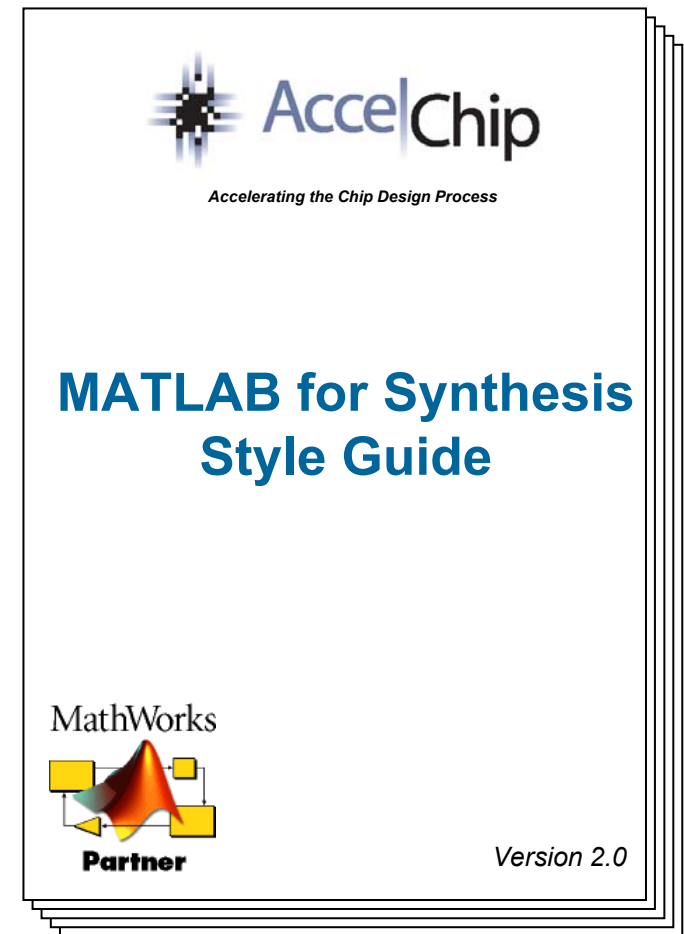
*MATLAB: Design Entry*





## Synthesizable MATLAB Style Guide

- Modeling Standards
  - Enables management to establish corporate standards for synthesizable and reusable MATLAB models
- Topics Included
  - Structure of models and scripts for use with high-level synthesis tools
  - Coding style guidelines of MATLAB models for optimal IC implementation
  - Reference guide for synthesizable MATLAB constructs and syntax
  - Design examples to illustrate top-down DSP design flow





## Efficient User Design Environment

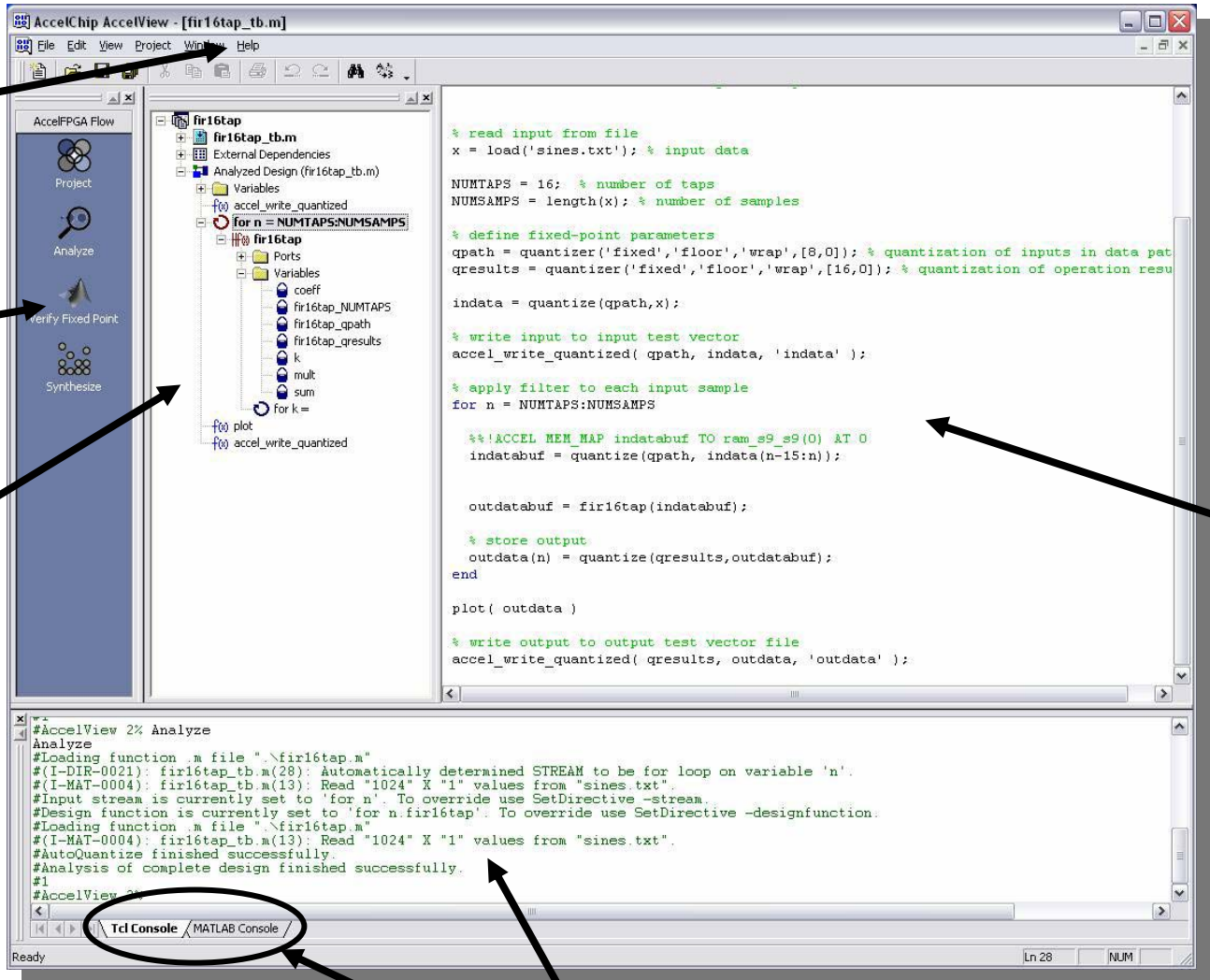
Enhanced  
Online Help  
Facility

Design  
Flow  
Manager

Hierarchical  
Project  
Navigator

MATLAB  
and RTL  
Source  
Code  
Windows

Integrated Command Consoles





## Design Flow Manager

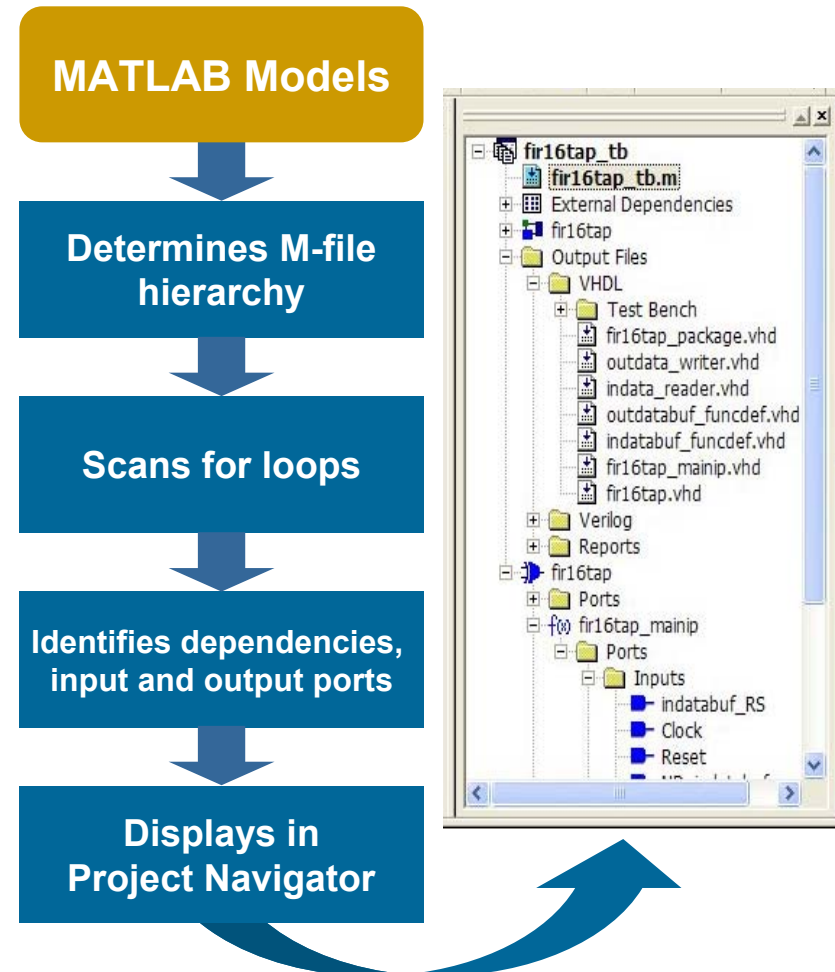
- Ease of use for algorithm developers
  - Icons are progressively disclosed to efficiently guide the designer through the IC design process
  - Algorithm developers with no prior IC design experience can easily go from MATLAB to IC design for accurate design space exploration
- Tool integration
  - AccelChip Design Flow Manager provides direct interfaces to 3<sup>rd</sup> party tools to provide a single, integrated DSP design flow
    - Logic synthesis and simulation tool integration
    - FPGA and ASIC vendor's tool integration
    - AccelChip contains a standard interface for integrating external tools





## Project Navigator

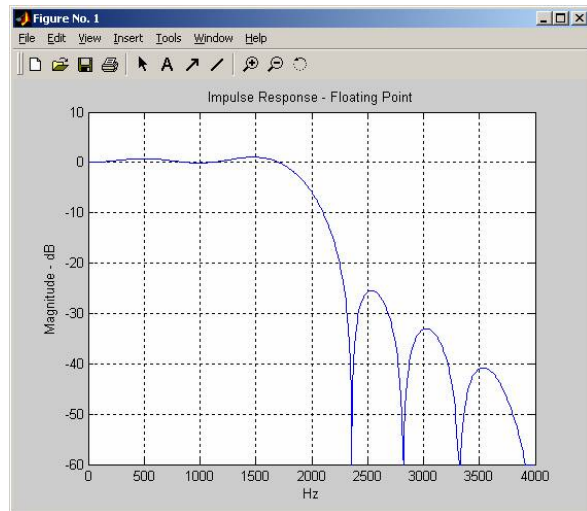
- Design analyzer
  - AccelChip automatically analyzes MATLAB models and displays the design hierarchy in the Project Navigator
- Language parser
  - Identifies non-synthesizable constructs and modeling styles
  - Assures adherence to MATLAB modeling standards
- Synthesis control
  - Allows IC design engineers to efficiently assign constraints and compiler commands





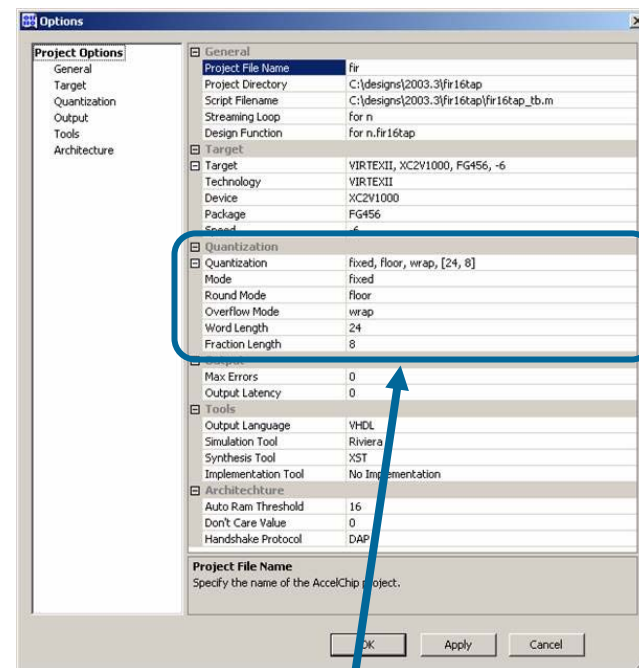
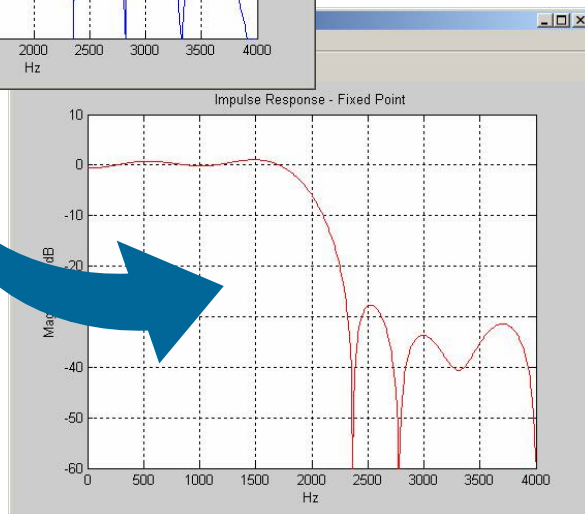
## Floating to Fixed-Point Conversion

AccelChip includes utilities that automate the conversion of floating-point MATLAB models to fixed-point representation for IC design



Example:  
16 Tap FIR Filter

Floating Point  
to  
Fixed Point  
conversion  
*automatically*



Default parameters can be  
easily modified to allow fast  
iteration and analysis of fixed-  
point precision





## AccelWare

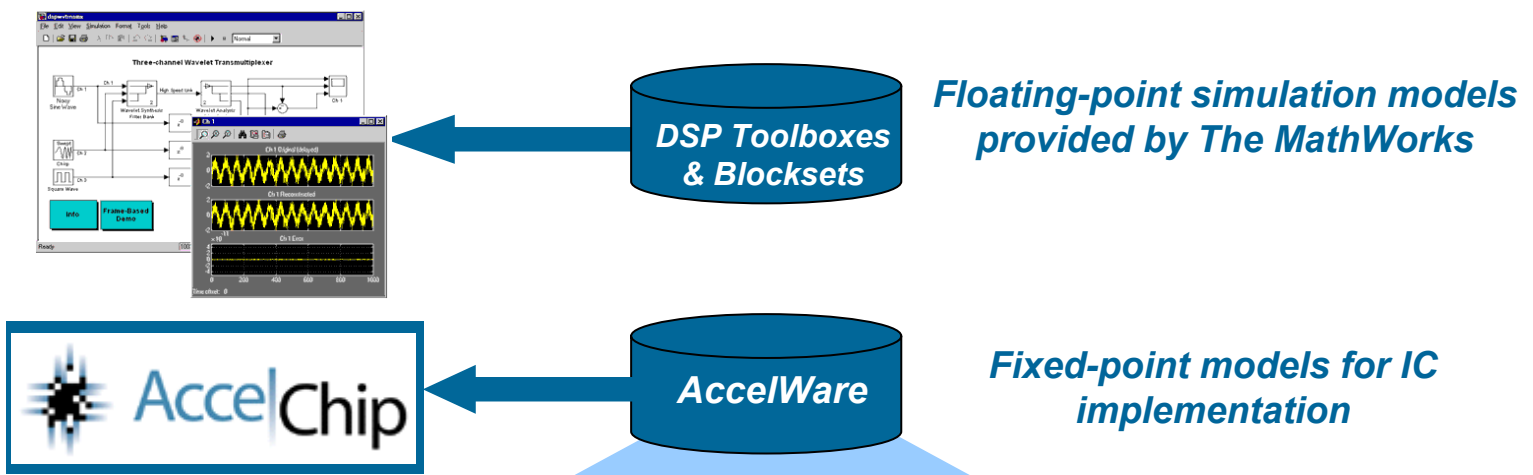
Pre-designed DSP model libraries to improve productivity and quality of results in IC implementation

- Fixed-point models for IC implementation
- Bit-true and cycle-accurate
- Automatically implemented from MATLAB models
- Parameterizable
  - Permits models to be tailored for multiple applications
- Synthesizable
  - Permits models to be targeted to multiple technologies

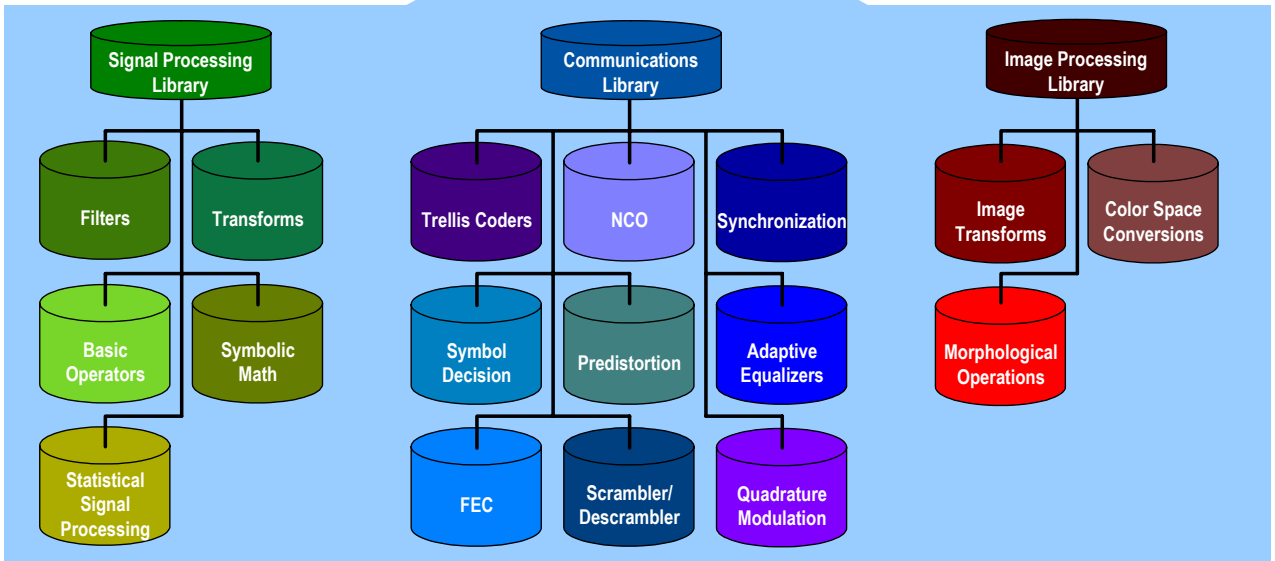




# AcceWare Model Libraries



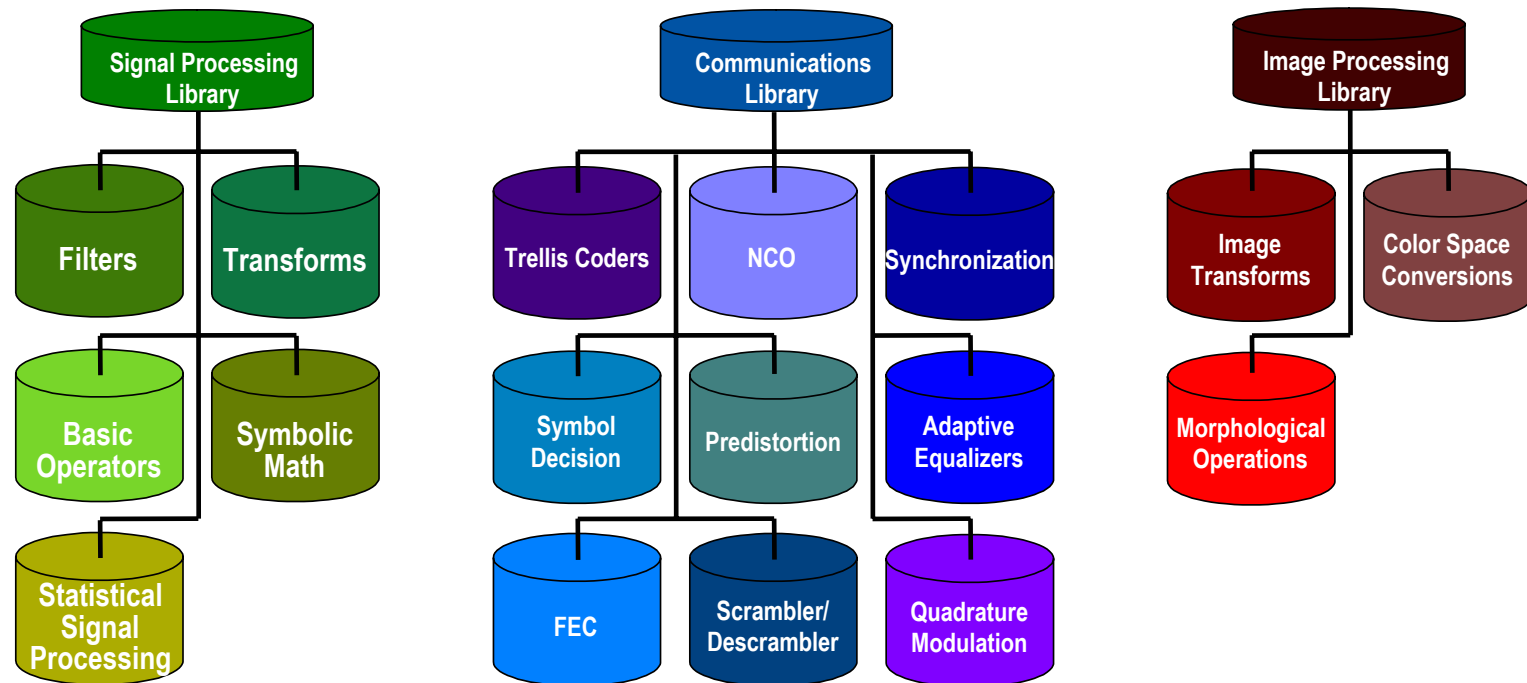
Over 150 DSP models available that provide high quality of results





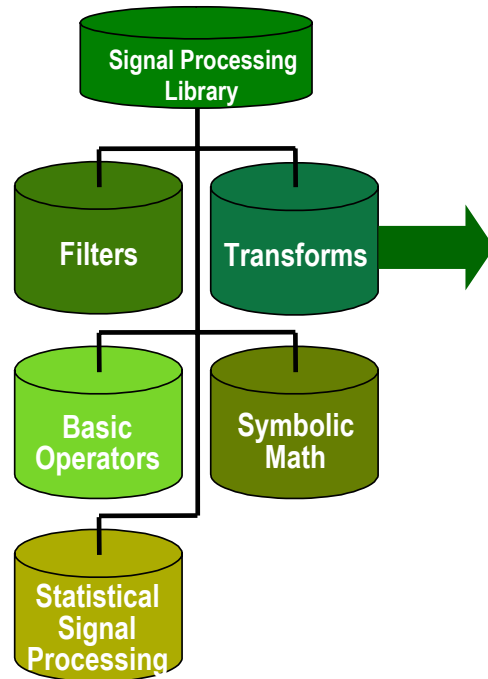
## AccelWare Model Libraries

- Target specific design applications
  - Signal Processing Library
  - Communications Library
  - Image Processing Library





## AccelWare Model Library Example



**AccelChip adds these enhanced models for hardware design**

**The MathWorks provides these floating-point models**

### Transform Blockset

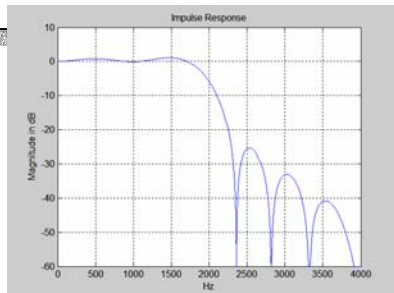
fft	Programmable length fast Fourier transform
ifft	Programmable length inverse fast Fourier transform
fft2	2D programmable length fast Fourier transform
ifft2	2D programmable length inverse fast Fourier transform
dct	Programmable discrete cosine transform
idct	Programmable inverse discrete cosine transform
hilbert	Hilbert transform
conv	Convolution and polynomial multiplication
fft_rad4	Programmable length radix-4 fast Fourier transform
ifft_rad4	Programmable length radix-4 inverse fast Fourier transform
dct8x8	Fixed 8x8 discrete cosine transform
idct8x8	Fixed 8x8 inverse discrete cosine transform
fastconv	Fast convolution (frequency domain filter)



## How AccelWare Works

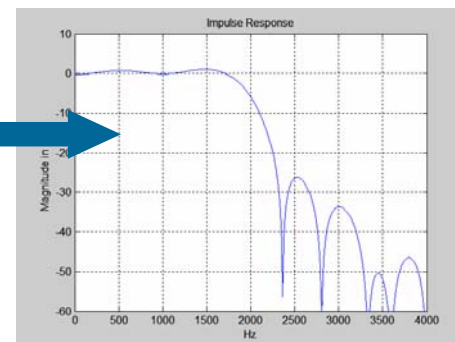
### Floating Point Model

```
% create FIR filter
b = fir1(48,[0.35
0.65]);
...
% filter data series
y=filter(b,1,data)
...
```

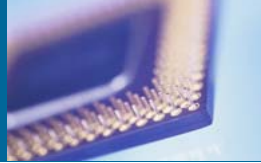


### AccelWare form for FIR filter implementation

### Fixed-Point Equivalent Model



- AccelChip reads the M-file into project directory
- AccelChip scans the M-file and identifies MATLAB functions corresponding to AccelWare components
- User specifies implementation properties for AccelWare components via forms or parameters in M-model
- AccelChip generates fixed-point model for synthesis



## Industry Leading Performance

<i>2048 Point FFT</i>	<i>Fmax</i>	<i># cycles</i>	<i>Throughput (μsec)</i>
AccelWare	117	352	3
Xilinx	194	2048	10.56
<i>Reed-Solomon Encoder</i>	<i>Logic Cells</i>	<i># Registers</i>	<i>Fmax</i>
AccelWare	360	275	197
Altera	1479	288	130



## AccelWare Benefits

- Superior quality of results
  - AccelWare was created by experienced IC designers
  - Optimal algorithm to IC design implementation
  - Silicon proven to eliminate risk
- Increased algorithm development productivity
  - AccelWare models provided for built-in MATLAB functions
  - Allows use of legacy MATLAB models without modification
- Extends MATLAB models with hardware design implementation parameters
  - Examples: internal word widths, resource sharing, radix (2 or 4)



## Consulting Services

AccelChip offers a variety of services to ensure our customers are successful in adopting a new design methodology

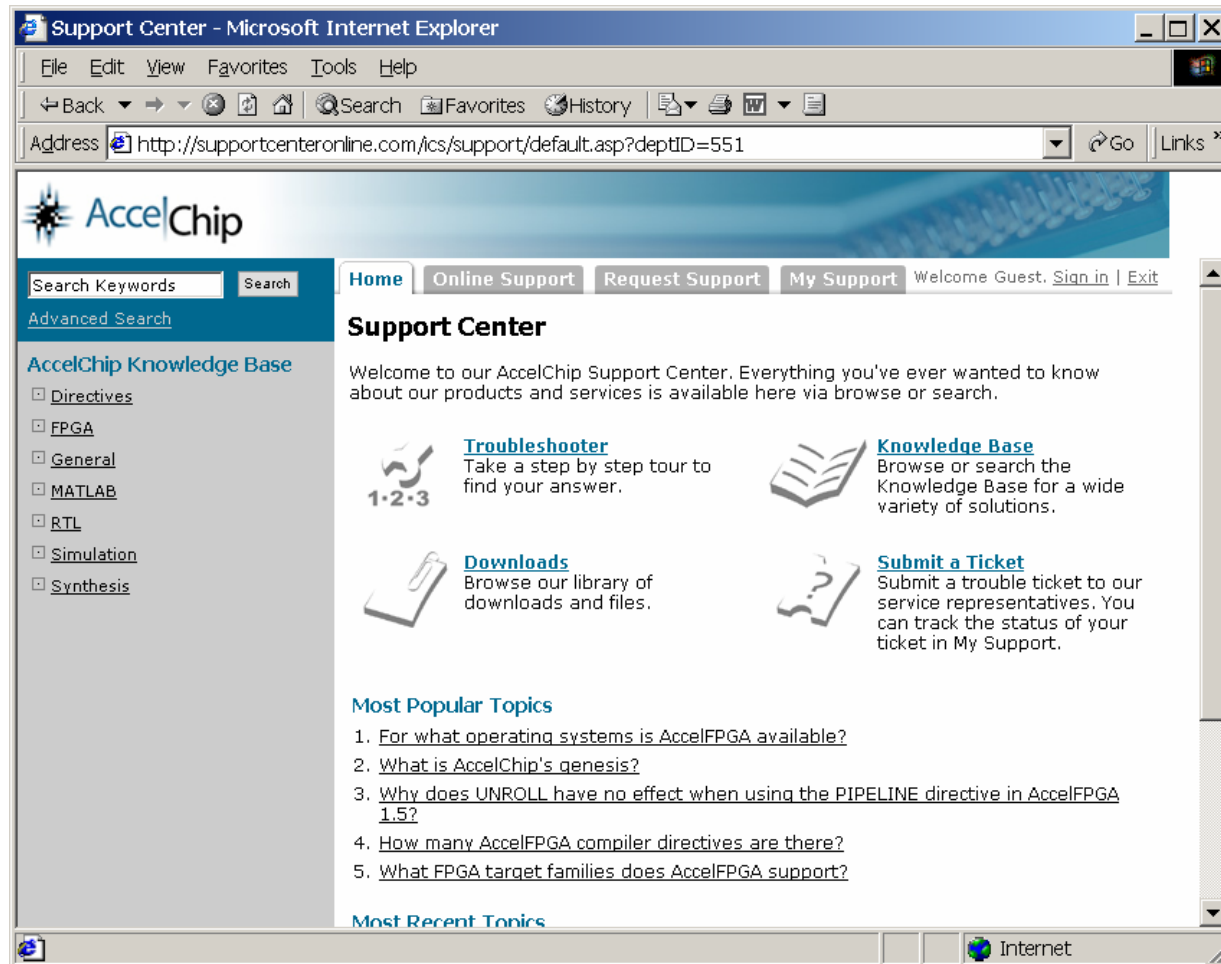
- Educational services
  - Product knowledge when you need it, where you need it
  - QuickStart project assistance
- Consulting services
  - Methodology and design process optimization
  - Scripting and custom user interface development
- Design services
  - Turnkey product design
  - Onsite design assistance
- German distributor – Trias Mikroelektronik GmbH
  - Provides superior local technical support and design assistance







## On-Line Customer Support Center





## AccelChip Benefits

- **Highest productivity**
  - ✓ High-level synthesis of floating-point MATLAB models eliminates the need to translate to fixed-point models
- **Vendor independence**
  - ✓ Allows designers to quickly and easily compare DSP algorithm implementation in different FPGA devices and to transition to ASICs for lower cost production devices
- **Reduces risk**
  - ✓ Only one representation of the DSP algorithm is required so there is no chance of revision control mistakes
- ✓ **Produces an optimal IC implementation**
  - ✓ Enables accurate design exploration at the algorithm level that improves floating-point model development
  - ✓ AccelWare models produce high quality of results in IC implementation



## Summary

AccelChip enables a top-down DSP design process that creates an automated process for implementing algorithms in silicon with high quality of results, eliminating many man months from the DSP design cycle



A Modulation and Encoding Technology for the Last Mile

**“With AccelChip we were able to explore algorithms in MATLAB and automatically synthesize Verilog for our designs. As a result, we were able to reduce development time by 75% and automatically verify our Verilog implementation.”**

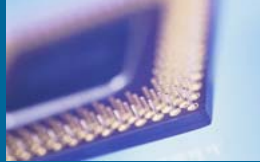
Marc Landry  
President and CEO  
XtendWave, Inc.

**(( ( LNX )) )**

Linking People and Technology

**“AccelChip made it easy for us to successfully implement our DSP design in a Xilinx Virtex-II FPGA. Using AccelChip, we were able to convert our customized, mixed radix FFT design from MATLAB to VHDL in days rather than weeks.”**

Michael J. Groden  
Director, Digital & Mixed-Signal Systems  
LNX Corporation



# Additional Information



## Signal Processing Library

- Basic Operators
  - `aw_sum` - Integrate and dump function
  - `aw_cumsum` - Cumulative sum
  - `aw_vco` - Voltage controlled oscillator
  - `aw_min` - Minimum value
  - `aw_max` - Maximum value
  - `aw_mean` - Average value
- Statistical Signal Processing
  - `aw_corrcoef` - Cross-correlation coefficients
  - `aw_xcorr` - Cross-correlation function
  - `aw_cov` - Covariance function



## Signal Processing Library (cont'd)

- Symbolic Math
  - `aw_sqrt` – Square root (estimate)
  - `aw_exp` - Complex exponential
  - `aw_conj` - Complex conjugation
  - `aw_round` - Round towards closest integer
  - `aw_angle` - Phase angle of a complex number
  - `aw_tan` – Tangent
  - `aw_imag` - Imaginary part of a complex number
  - `aw_polyval` - Polynomial evaluation
  - `aw_dot` - Vector dot product
  - `aw_sin` - Sine
  - `aw_cos` - Cosine
  - `aw_svd` - Singular values and singular vectors
  - `aw_eig` - Eigenvalues and eigenvectors



## Signal Processing Library (Cont'd)

- Transforms
  - `aw_fft_fx` - Fixed length fast Fourier transform
  - `aw_fft` - Programmable length fast Fourier transform
  - `aw_ifft_fx` - Fixed length inverse fast Fourier transform
  - `aw_ifft` - Programmable length inverse fast Fourier transform
  - `aw_fft_rad4` - Programmable length radix-4 fast Fourier transform
  - `aw_ifft_rad4` - Programmable length radix-4 inverse fast Fourier transform
  - `aw_dct` – Programmable discrete cosine transform
  - `aw_idct` - Programmable inverse discrete cosine transform





## Signal Processing Library (Cont'd)

- Transforms
  - `aw_dct8x8` - Fixed 8x8 discrete cosine transform
  - `aw_idct8x8` - Fixed 8x8 inverse discrete cosine transform
  - `aw_fft2_fx` - 2D fixed length fast Fourier transform
  - `aw_fft2` - 2D programmable length fast Fourier transform
  - `aw_ifft2_fx` - 2D fixed length inverse fast Fourier transform
  - `aw_ifft2` - 2D programmable length inverse fast Fourier transform
  - `aw_conv` - Convolution and polynomial multiplication
  - `aw_fastconv` - Fast convolution (frequency domain filter)
  - `aw_hilbert` - Hilbert transform



## Signal Processing Library (Cont'd)

- General Purpose FIR Filters
  - `aw_fir` - Programmable coefficient FIR filter
  - `aw_fir_cmplx` - Complex, programmable coefficient FIR filter
  - `aw_fir_fx` - Fixed coefficient FIR filter
  - `aw_fir_cmplx_fx` - Complex, fixed coefficient FIR filter
- Linear Phase FIR Filters
  - `aw_lpfir` - Programmable coefficient linear phase FIR filter
  - `aw_lpfir_cmplx` - Complex, programmable coefficient linear phase FIR filter
  - `aw_lpfir_fx` - Fixed coefficient linear phase FIR filter
  - `aw_lpfir_cmplx_fx` - Complex, fixed coefficient linear phase FIR filter
- Adaptive FIR Filters
  - `aw_lmsfir` - Real valued adaptive FIR filter



## Signal Processing Library (Cont'd)

- Fractional Sample Delay Filters
  - `aw_fsd` - Fractional sample delay filter
- Multi-Rate Filters
  - `aw_rrsdec` - Programmable accumulate and dump (sinc) decimation filter
  - `aw_dec2x` - Fixed 2x decimation filter with even/odd output
  - `aw_rrsdec_fx` - Fixed accumulate and dump (sinc) decimation filter
  - `aw_cicdecimate` - Cascaded Integrator-Comb (CIC) decimation filter
  - `aw_cicinterpolate` - Cascaded Integrator-Comb (CIC) interpolation filter
  - `aw_firhalfband` - Half-band FIR filter with parameterizable 2x decimation
  - `aw_interp2x` - Fixed 2x interpolation filter with even/odd output
  - `aw_nonint_interp` - Programmable, non-integer interpolation filter
  - `aw_nonint_dec` - Programmable, non-integer decimation filter



## Signal Processing Library (Cont'd)

- Polyphase Filters
  - `aw_fir2x` - Programmable 2-phase (even/odd) FIR filter
  - `aw_fir2x_fx` - Fixed coefficient 2-phase (even/odd) FIR filter
  - `aw_polyfir` - Programmable polyphase FIR filter
  - `aw_polyfir_fx` - Fixed coefficient polyphase FIR filter
- IIR Filters
  - `aw_iir_biquad` - Programmable coefficient IIR 2<sup>nd</sup> order section
  - `aw_iir_biquad_fx` - Fixed coefficient IIR 2<sup>nd</sup> order section
  - `aw_iir_lpf` - Multiplierless 1<sup>st</sup> order IIR low pass filter section



## Signal Processing Library (Cont'd)

- Complex Pulse Shaping Filters
  - `aw_pulse_shaper` - Programmable coefficient pulse shaping with programmable interpolation
  - `aw_pulse_shaper2x` - Programmable coefficient pulse shaping with fixed 2x interpolation
  - `aw_pulse_shaper_fx` - Fixed coefficient pulse shaping with programmable interpolation
  - `aw_pulse_shaper2x_fx` - Fixed coefficient pulse shaping with fixed 2x interpolation



## Communications Library

- Encoders/Decoders
  - `aw_rsenc` - Fixed code parameter  $\{N, K, t\}$  Reed-Solomon/BCH encoder
  - `aw_rsdec` - Fixed code parameter  $\{N, K, t\}$  Reed-Solomon/BCH decoder
  - `aw_convenc` - Convolutional encoder
  - `aw_vitdec` - Viterbi decoder for punctured or non-punctured codes
  - `aw_rsenc_prog` - Programmable code parameter  $\{N, K, t\}$  Reed-Solomon/BCH encoder
  - `aw_rsdec_prog` - Programmable code parameter  $\{N, K, t\}$  Reed-Solomon/BCH decoder



## Communications Library (Cont'd)

- Interleavers/Deinterleavers
  - `aw_convintlv` - Fixed depth convolutional interleaver
  - `aw_convintlv_prog` - Programmable depth convolutional interleaver
  - `aw_convdeintlv` - Fixed depth convolutional deinterleaver
  - `aw_convdeintlv_prog` - Programmable depth convolutional deinterleaver
- Numerically Controlled Oscillator
  - `aw_nco` - Numerically controlled oscillator – complex exponential
  - `aw_nco2x` - Numerically controlled oscillator outputting 2 samples per clock





## Communications Library (Cont'd)

- Synchronization
  - [aw\\_carrier\\_recovery](#) - Digital phase detector and complex phase rotator
  - [aw\\_clk\\_recovery](#) - Spectral line algorithm for digital clock recovery
- Symbol Decision
  - [aw\\_slicer](#) - Programmable QPSK-256QAM complex slicer
- Adaptive Equalizers
  - [aw\\_fse](#) - Fractionally spaced, complex adaptive equalizer



## Communications Library (Cont'd)

- Quadrature Modulation
  - [aw\\_phase\\_splitter](#) - Real to I-Q complex quadrature demodulator with no decimation
  - [aw\\_quadmod2x](#) - Programmable IF passband quadrature modulator
  - [aw\\_phase\\_splitter2x](#) - Real to I-Q complex quadrature demodulator and fixed 2x decimation
- Scramblers/Descramblers
  - [aw\\_iess310](#) - IESS-310 compliant bit serial scrambler/de-scrambler
  - [aw\\_iess310\\_byte](#) - Enhanced IESS-310 compliant byte-wide scrambler/de-scrambler
  - [aw\\_scram16](#) - Custom 16-bit wide scrambler/de-scrambler
  - [aw\\_mcns\\_byte](#) - MCNS/J.83 compliant byte-wide scrambler/de-scrambler



## Communications Library (Cont'd)

- Predistortion
  - [aw\\_polyeval](#) - Programmable order complex polynomial evaluator
  - [aw\\_curvefit](#) - Programmable order complex polynomial curve fit
  - [aw\\_linearity\\_error](#) - Compute complex linearity error metric
  - [aw\\_cmplx\\_mag](#) - Compute magnitude of complex number
- Trellis Coders
  - [aw\\_tcm\\_encode](#) - Programmable rate  $(k-1)/k$  or  $(2k-1)/2k$  trellis encoder
  - [aw\\_tcm\\_decode](#) - Programmable rate  $(k-1)/k$  or  $(2k-1)/2k$  trellis decoder
  - [aw\\_tcm\\_decode\\_j83b](#) - J.83 annex B compliant trellis decoder



## DSP Platform IP

- Digital Modem
  - [aw\\_qam\\_mod](#) - QPSK-256QAM digital modulator with Forward Error Correction (FEC)
  - [aw\\_qam\\_mod\\_dsp](#) - QAM symbol modulation with pulse shaping
  - [aw\\_qam\\_mod\\_fec](#) - QAM symbol mapping with scrambling & FEC
  - [aw\\_qam\\_demod](#) - QPSK-256QAM digital demodulator with Forward Error Correction (FEC)
  - [aw\\_qam\\_demod\\_dsp](#) - Quad down conversion with equalization, matched filtering, carrier recovery, and clock recovery of QAM symbols
  - [aw\\_qam\\_demod\\_fec](#) - QAM symbol demapping with FEC, descrambling and framing
  - [aw\\_mcns\\_cablemodem](#) - MCNS/DAVIC compliant subscriber-end digital 64/256QAM cable modem
  - [aw\\_ofdm](#) - IFFT-based OFDM 2k/8k carrier digital modem