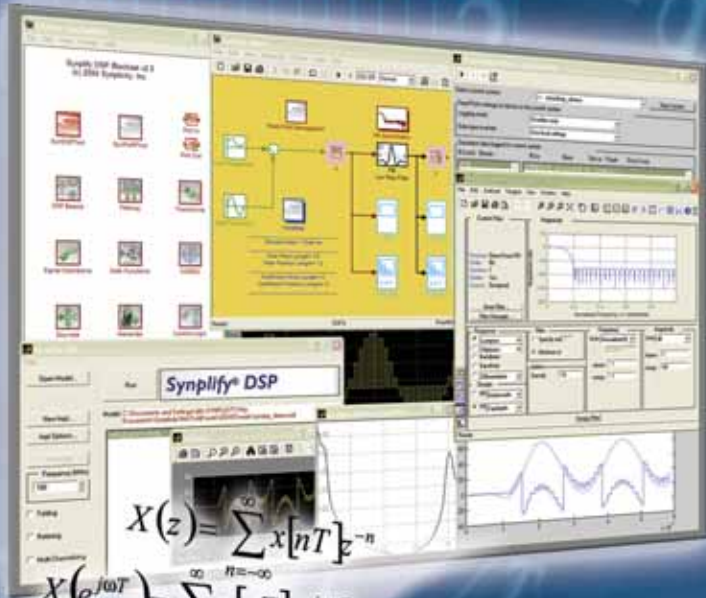


# Synplify® DSP

## A Breakthrough in DSP Design



Synplify's Synplify DSP synthesis solution offers a fast, efficient way to implement DSP algorithms into silicon. By automating architectural optimizations like pipelining, resource sharing, and multi-channelization, engineers can save months of RTL coding, simplify design capture, speed verification, and create technology-independent IP.

### Synplify DSP Software Uniquely Offers:

- Technology-independent DSP modeling library
- Comprehensive multi-rate and vector math support
- Fixed-point quantization and analysis tools
- Powerful DSP synthesis engine
- Architecturally optimized Verilog and VHDL implementation
- Target the latest devices from Xilinx including Virtex-5

$$X(z) = \sum_{n=-\infty}^{\infty} x[nT] z^{-n}$$
$$X(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} x[nT] e^{-j\omega n T}$$
$$= R(e^{j\omega T}) + jI(e^{j\omega T})$$
$$= A(e^{j\omega T}) e^{j\phi(e^{j\omega T})}$$
$$x[nT] = \frac{1}{2\pi j} \oint_C X(z) z^{nT-1} dz$$
$$x[nT] = \frac{1}{2\pi} \int_{-\pi/T}^{\pi/T} X(e^{j\omega T}) e^{-j\omega n T} d\omega$$
$$y[nT] = \sum_{k=0}^N b_k x[(n-k)T] + \sum_{k=1}^M a_k y[(n-k)T]$$
$$H(z) = \frac{\sum_{k=0}^N b_k z^{-k}}{\sum_{k=1}^M a_k z^{-k}} = \frac{b_0 \prod_{i=1}^N (z - z_i)}{\prod_{j=1}^M (z - p_j)} z^{M-N}$$

For more information on Synplify's Synplify DSP solution and all of Synplify's offerings, please visit our website at [www.synplify.com](http://www.synplify.com) or contact [info@synplify.com](mailto:info@synplify.com)



Synplify®

Simply Better Results