AMD

HOW TO PROGRAM AMD VERSAL™ AI ENGINES IN SIX STEPS

A quick guide for FPGA architects and engineers

AMD Versal™ AI Engines are programmable arrays of vector processors that are ideal for compute-intensive workloads. Programming AI Engines requires a different methodology than traditional FPGAs, but the AMD Vitis™ development platform makes programming easy.

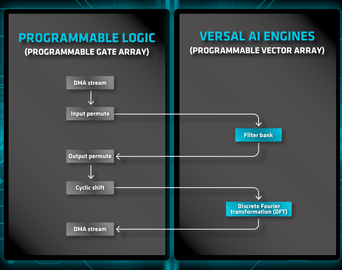
When the AI Engine design is completed within the Vitis platform, the design can be exported and integrated into a larger system using AMD Vivado™ design tools.

Let's see how it works by exploring how to program a polyphase channelizer.

1 Map workloads and functions to programmable logic and Versal AI Engines

Functions that move and reorder data run efficiently in programmable logic.

Filtering and Fourier transformations have inherent parallelism that can be vectorized, so they map efficiently to the Versal AI Engines.



2 Develop and simulate in the Vitis platform

3 Implement and instantiate Versal AI Engine kernels

Build with Vitis C/C++ libraries, APIs, and intrinsics

```
//call filter bank function

polyphase_fir(TT_COEFF (&taps0_1)[NUM_POLY*TAPS_PER_PHASE],
TT_COEFF (&taps1_1)[NUM_POLY*TAPS_PER_PHASE] );


//implement discrete fourier transform

template<class TT_DATA,class TT_COEFF,class TT_ACC,unsigned NSAMP>

dft_1xN_input<TT_DATA,TT_COEFF,TT_ACC,NSAMP>::dft_1xN_input( TT_COEFF
(&coeff0_1)[8], TT_COEFF (&coeff1_1)[8] )

  : coeff0( coeff0_1 ), coeff1( coeff1_1 )

{

  aie::set_rounding(aie::rounding_mode::positive_inf);

  aie::set_saturation(aie::saturation_mode::saturate);

}
```

4 Integrate and interconnect via Graph C code

```
polyphase_fir_graph( std::vector<typename TT_KERNEL::COEFF> tapsA_1,
                     std::vector<typename TT_KERNEL::COEFF> tapsB_1 )

{
k_fir = kernel::create_object<TT_KERNEL>( tapsA_1, tapsB_1 );

source(k_fir) = "polyphase_fir.cpp";
runtime<ratio>(k_fir) = 0.9;

connect<stream,stream>( sig_i[0],     k_fir.in[0] );
connect<stream,stream>( k_fir.out[0], sig_o[0]    );
}
```

5 Compile, simulate, and debug

Compile hybrid design

Simulate and debug

Performing as expected? → NO

YES

6 Export package for use in the AMD Vivado™ Design Suite

Run export to Vivado Design Suite from the Vitis platform

EVALUATE AI ENGINES IN AMD VERSAL ADAPTIVE SoCs

AMD VERSAL

Versal AI Engines are available as options in several Versal adaptive SoCs:

Versal™ AI Core Series | Versal™ AI Edge Series | Versal™ Premium Series

START DESIGNING

Learn more about programming Versal adaptive SoCs, then experiment with shifting DSP algorithms and functions to Versal AI Engines for acceleration of your high-performance signal processing system.

Dive Deeper

© 2025 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, Versal, Vitis, Vivado, and combinations thereof are trademarks of Advanced Micro Devices, Inc. in the United States and other countries. Other product names used in this publication are for identification purposes only and may be trademarks of their respective owners.

AMD
together we advance_