# ARC EM5D and EM7D DSP-Enhanced Processors

## Highlights

- Combined RISC + DSP processors
- Easy DSP programming support with Metaware C/C++ Compiler
- Feature-rich DSP software library for easy algorithm programming
- ITU-T base operations for leveraging ITU-T C code base
- ARCv2DSP ISA adds over 150
- DSP instructions
- Fixed point, vector/SIMD DSP processing support
- Power-efficient unified 32x32
- MUL/MAC unit
- Highly configurable DSP and processor features for optimal design
- 1.81 DMIPS/MHz and 4.18 CoreMark/MHz
- Support for I$ And D$ (EM7D)
- Configurable hardware divider and ECC/ parity support
- Supports acceleration for APEX Processor Extensions
- JTAG debug interface

## Target Applications

- Voice/speech processing, including voice activation
- Always-on sensor processing
- Baseband control for wireless devices
- Embedded audio processing (offload host)
- Automotive sensors
- Portable and wearable IoT devices

## 32-Bit Processors for Low-Power Embedded DSP Applications

The Synopsys ARC® EM DSP Family, which includes the EM5D, EM7D, EM9D and EM11D processors, is optimized for use in low-power embedded applications where DSP performance and low-power consumption is a requirement. Wearable devices in the IoT market need this combination of features to enable optimum device performance and extended battery life. The ARC EM5D supports up to 2MB each of instruction and data closely coupled memory (CCM). The EM7D adds support for up to 64KB of instruction and data cache.

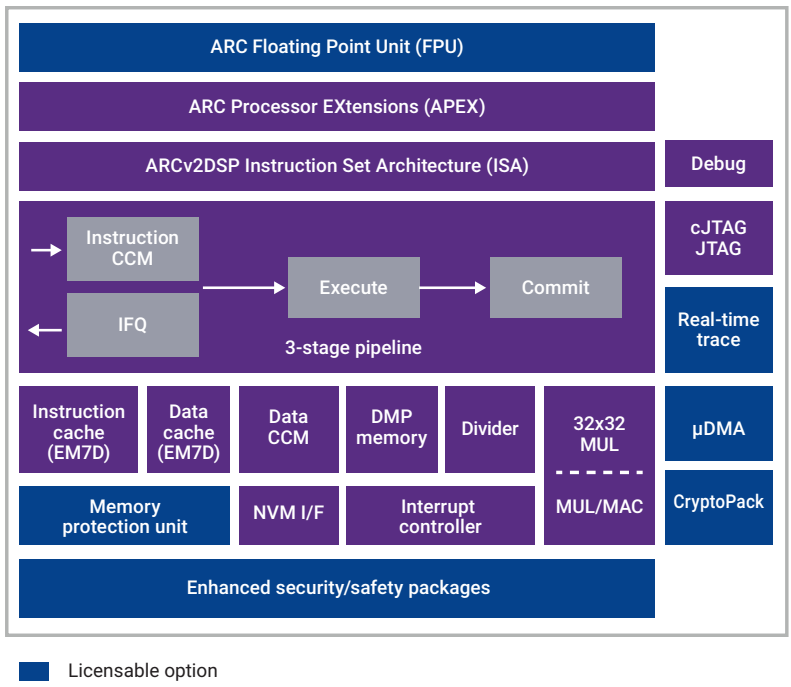


Figure 1: EM5D and EM7D Block Diagram

The processors are based on the area, power and code-efficient ARCv2 Instruction Set Architecture (ISA) and feature a balanced 3-stage Harvard architecture pipeline that provides efficient throughput, delivering up to 1.81 DMIPS/MHz and 4.18 CoreMark/MHz. These DSP-enhanced cores include the ARCv2 DSP ISA, which adds over 150 optimized DSP instructions. The EM5D and EM7D cores feature a power-efficient unified 32x32 MUL/MAC unit and support for fixed-point DSP datatypes and vector/SIMD (single instruction

multiple data) operations. To enable easy DSP software development the ARC MetaWare Development Toolkit features a rich DSP software library and the included C/C++ Compiler supports commonly used DSP datatypes for easy algorithm programming. The EM5D and EM7D hardware DSP capabilities are configurable to allow tailoring of the DSP features to match the application's DSP requirements more effectively.

The ARC EM5D and EM7D processors maintain the high code density capability that is synonymous with the ARCv2 ISA and the efficient MetaWare C Compiler. The cores offer excellent real-time control and DSP performance within a very small footprint that can be as small as 0.03mm$^2$ (40nm LP).

## ARCv2DSP ISA

The Synopsys ARC EM5D and EM7D processor cores implement the DSP enhanced ARCv2DSP 16-/32-bit RISC ISA, which supports advanced processor capabilities and incorporates 150+ DSP instructions that improve DSP performance (Table 1). The architecture and pipeline are designed to meet the needs of next-generation system-on-chip (SoC) applications and enable the development of a full range of 32-bit processor cores, from low-end, extremely power-efficient embedded DSP cores to high-performance solutions that are binary compatible and designed with common pipeline elements. The ARCv2DSP ISA enables the implementation of complex heterogeneous SoCs with processor cores precisely targeted to meet the specific performance and power requirements for each instance on an SoC, while offering the same software programmer's model to simplify program development and task partitioning.

### Instruction Set Summary

The ARCv2DSP ISA supports the following classes of DSP instructions/ operations: basic saturating arithmetic, vector unpacking, accumulator operations, vector 16x16 MAC, dual 16x16 MAC, dual 16x8 MAC, 32x16 MAC, 32x32 MAC and complex (16+16)x(16+16) MAC.

| Operation | Mnemonics | Description |
|---|---|---|
| **Arithmetic** | | |
| Absolute value | ABSS, ABSSH, VABS2H, VABSS2H | (Non-)saturating absolute value |
| Add | ADDS, ADCS, VADD2H, VADDS2H, VADD4B | (Non-)saturating add |
| Add-subtract | VADDSUB2H, VADDSUBS2H | (Non-)saturating add-subtract |
| Min/max | VMIN2H, VMAX2H | Minimum/maximum |
| Negate | NEGS, NEGSH, VNEG2H, VNEGS2H | (Non-)saturating negative |
| Normalize | VNORM2H | Vector normalize |
| Subtract | SUBS, SBCS, VSUB2H, VSUBS2H, VSUB4B | (Non-)saturating subtract |
| Subtract-add | VSUBADD2H, VSUBADDS2H | (Non-)saturating subtract-add |
| **MUL/MAC operations** | | |
| 16X16 integer | MPYW, MPYW_S, MPYUW, MPYUW_S | Signed/unsigned 16x16 MUL |
| 32x16 fractional | MPYWHFM, MPYWHFMR, MACWHFM, MACWHFMR, MSUBWHFM, MSUBWHFMR, MPYWHFL, MPYWHFLR, MACWHFL, MACWHFLR, MSUBWHFL, MSUBWHFLR | 32x16 fractional MUL/MAC/MSUB, saturating, (non-)rounding |
| 32x16 integer | MPYWHL, MACWHL, MPYWHUL, MACWHUL, MPYWHKL, MACWHKL, MPYWHKUL, MACWHKUL | Signed/unsigned 32x16 MUL/MAC |
| 32x32 fractional | MPYF, MPYFR, MACF, MACFR, MSUBF, MSUBFR, MPYDF, MACDF, MSUBDF | 32x32 fractional MUL/MAC/MSUB, saturating, (non-)rounding |
| 32x32 integer | MPY, MPY_S, MPYU, MAC, MACU, MPYM, MPYMU, MPYD, MPYDU, MACD, MACDU | Signed/unsigned 32x32 MUL/MAC |
| Dual 16x8 | DMPYHBL, DMPYHBM, DMACHBL, DMACHBM | Dual 16x8 signed MUL/MAC, inner-product style |
| Vector 16x16 fractional | VMPY2HF, VMPY2HFR, VMAC2HF, VMAC2HFR, VMPY2HWF, VMAC2HNFR, VMSUB2HF, VMSUB2HFR, VMSUB2HNFR | Vector/SIMD 16x16 fractional MUL/MAC/ MSUB, saturating, (non-) rounding |
| Vector 16x16 integer | VMPY2H, VMPY2HU, VMAC2H, VMAC2HU | Vector/SIMD 16x16 integer MUL/MAC, signed/unsigned |

Table 1: ARCv2DSP ISA DSP instructions

| Operation | Mnemonics | Description |
|---|---|---|
| Dual 16x16 fractional | DMPYHF, DMPYHFR, DMACHF, DMACHFR, DMPYHWF | Dual 16x16 fractional MUL/MAC, inner-product style, saturating, (non-) rounding |
| Dual 16x16 integer | DMPYH, DMPYHU, DMACH, DMACHU | Dual 16x16 integer MUL/MAC, inner-product style, signed/unsigned |
| Complex (16+16)x(16+16) | CMPYHFMR, CMPYHNFR, CMPYHFR, CMPYCHNFR, CMPYCHFR, CMACHNFR, CMACHFR, CMACCHNFR, CMACCHFR | (16+16)x(16+16) complex MUL/MAC, rounding |
| FFT butterfly (16+16)x(16+16) | CBFLYHF0R, CBFLYHF1R | First half and second half of 2-cycle FFT butterfly |
| **Saturation and rounding** | | |
| Saturate | SATH, SATF | Saturate 32b to 16b/32b |
| Round | RNDH | Round and saturate 32b to 16b |
| **Shifting** | | |
| Saturating shifts | ASLS, ASRS, VASLS2H, VASRS2H | Saturating left/right shift |
| Shifts with rounding | ASRSR, VASRSR2H | Right shift with rounding |
| Non-saturating shifts | VASL2H, VASR2H, VLSR2H | Non-saturating left/right shift |
| **Vector unpacking/packing** | | |
| 16b to 32b packing | VALGN2H, VPACK2HL, VPACK2HM | Pack two 16b into 2x16 vector |
| Replicate | VREP2HL, VREP2HM | Replicate 16b in 2x16 vector |
| 8b to 16b unpacking | VEXT2BHL, VEXT2BHM, VSEXT2BHL, VSEXT2BHM, VEXT2BHLF, VEXT2BHMF | Expand two bytes to 2x16 vector |
| 16b to 8b packing | VPACK2HBL, VPACK2HBM, VPACK2HBLF, VPACK2HBMF | Pack two 16b values into two upper/lower bytes |
| Permute | VPERM | Permute bytes and sign extend |
| **Divide and square-root** | | |
| Divide | DIV, DIVU, DIVF | Integer/fractional divide |
| Remainder | REM, REMU | Signed/unsigned integer remainder |
| Square-root | SQRT, SQRTF | Integer/fractional square root |
| **Accumulator operations** | | |
| Get/set | GETACC, SETACC | Get/set accumulator value |
| Flag | FLAGACC | Copy accumulator status flags |
| Left-shift | ASLACC, ASLSACC | (Non-)saturating shift left |
| Normalize | NORMACC | Normalize accumulator |
| **Auxiliary operations** | | |
| XY address pointer update | MODIF | Apply modifier |
| Bitstream parsing | BSPEEK, BSPOP, BSPUSH | Access head/tail of bitstream |

Table 1: ARCv2DSP ISA DSP instructions (continued)

## Features

- 3-stage Harvard architecture pipeline
- Fractional data type support
- Multiple rounding modes
- Supports MUL/MAC operations
- FFT butterfly, square root and divide acceleration
- Single and dual 32b/40b and single 64b/72b accumulators
- Single cycle access closely coupled memory
- Up to 64 Kbyte of instruction and data cache (EM7D)
- User-configurable program counter

- Configurable zero overhead loop counter
- Architectural clock gating and enhanced sleep modes and instructions
- 32-bit instruction and data busses
- ARM® AMBA® AHB™ and AHB-Lite™ native bus interfaces
- Integrated watchdog Timer
- ECC and parity support

## Pipeline

The EM DSP processors have a low latency three-stage pipeline that is optimally balanced to achieve very low power consumption with excellent embedded real-time and DSP performance. The pipeline is designed to give longer access to memory, allowing maximum clock speeds comparable to typical 5-stage pipeline-based cores at all process nodes. The pipeline is a Harvard architecture with separate instruction and data memory storage that can be simultaneously accessed. The ARC EM5D and EM7D pipeline supports precise exceptions with a commit point after the second stage.

## Configurable Options

The EM5D and EM7D processor cores support a broad range of configurable options, enabling optimization for a specific application's performance, power and size requirements. The included ARChitect configuration tool allows selection and configuration of processor features via an easy-to-use graphical user interface. The tool produces verified RTL and synthesis scripts that are compatible with industry-standard design flows. With ARChitect, designers can add or remove features that improve the efficiency or performance of the core for their application, including options such as multipliers, hardware divide, memory configuration, program counter width, address bus width, timers, interrupts, register file structure and user-defined hardware and instructions.

## Closely Coupled Memory

The ARC EM5D and EM7D processors support 512B to 2MB of closely coupled memory (CCM) for both instruction and data memory. The CCM is implemented as separate memory spaces for the Instruction Closely Coupled Memory (ICCM) and Data Closely Coupled Memory (DCCM). Both memory spaces can be accessed every clock cycle. Both ICCM and DCCM can be read and written to from outside the core through AHB-Lite target interfaces. They can also both be separately configured for 512B, 1KB, 2KB, 4KB, 8KB, 16KB, 32KB, 64KB, 128KB, 256KB, 512KB or 1MB. The EM5D and EM7D share the same features, but the EM7D adds up to 64KB of instruction and data caches.

## Register File and Program Counter

ARC EM5D and EM7D register files are configurable, with a default size of 16 x 32-bit registers that can be increased to 32 x 32-bit registers at build time. The register file can be constructed from fast, single-cycle access memory or flip-flops, and supports one or two write ports (one is the default) and two read ports.

The general-purpose registers can be used for any purpose by the programmer. Some of the core registers have defined special purposes like stack pointers, link registers and loop counters. The register file can be expanded to up to a total of 60 registers. The ARCv2 16-bit instructions have limited direct access to register r0–r15 and can indirectly access the remaining registers' various instructions. If an instruction references an unimplemented core register, an Instruction Error exception will be raised.

The program counter is read only and located at r63 in the register file. The program counter is a 32-bit word aligned register for use as a source operand in all instructions supporting PC-relative addressing. The default size for the program counter is 16-bit, and this is user configurable at build time to 20-, 24- or 32-bits.

## Bus Interfaces

The EM5D and EM7D cores have native support for the ARM AMBA AHB and AHBLite bus protocols. This is a build-time option with the ARM AMBA AHB interface as the default selection. These enable the solutions to be easily connected to the SoC infrastructure in most chips without incurring any delay or complication in the bus interface.

## Error Protection

ARC EM cores provides support for error protection and caches where present. The different protection schemes may be combined to achieve several levels of protection against malicious or misbehaving code in critical applications.

ARC EM supports two error detection techniques: single-bit error correction, double-bit error detection (SECDED) and parity which detects single-bit errors. Both odd and even parities are supported. Data-only protection or data and address protection are both supported for ECC.

## Interrupts and Exceptions

EM5D and EM7D processors support up to 32 interrupts and exceptions, with the default setting of three exceptions and no interrupts. Designers can select any additional number of interrupts up to 29 (32 total interrupts and exceptions) at build time. The interrupts support multiple user-defined priority levels. The Exceptions Reset, Memory Error and Instruction Error have a higher priority than the highest priority interrupts. The interrupts are serviced through a jump table that allows higher flexibility in the location and implementation of the interrupt vectors. Interrupts can be triggered by hardware or software.

## ARC Processor EXtension (APEX) Technology

The EM5D and EM7D processors are designed to be extendable with the addition of user-defined instructions.These extensions may include more core and auxiliary registers, new user-defined instructions, and additional condition code tests. The extensions of the core enable efficient addition of user-defined hardware and other capabilities that are tightly coupled to the core and can greatly increase system performance and save power consumption.

## Licensable Options

- FPU Floating Point Unit offers single- and double-precision math support
- The enhanced security package enables designers to create a secure, tamper resistant environment that protects their systems and software from evolving security threats
- CryptoPack provides APEX acceleration for software encryption algorithms, including AES, 3DES, ECC, SHA-256, CRC, RSA
- The Memory Protection Unit (MPU) gives the software running on the processor the ability to control access rights to the memory
- ARC Trace provides instruction and data real-time trace features to the Synopsys ARC processors
- Tightly coupled microDMA engine allows system resources and peripherals to access memory resources independent of the processor
- ARConnect module facilitates multicore implementations with hardware for intercore message passing, interrupt handling, semaphores and debug

## Complete Suite of Development Tools

To facilitate rapid development of software for EM5D and EM7D, they are supported by a complete suite of development tools. This includes the MetaWare Development Toolkit that generates highly efficient code ideal for deeply embedded applications and has an LLVM-based C/C++ DSP Compiler, ARC xCAM and nSIM simulators, and the ARChitect core configuration tool (Table 2).

| Function | Product | Benefit |
|---|---|---|
| Tools | MetaWare development toolkit | · Develop efficient code with an optimizing C/C++ compiler, debugger and instruction set simulator<br>· Rich DSP software library supporting commonly used DSP functions<br>· ITU-T base operations library for easy voice codec porting |
| | JTAG debuggers | · Debug programs quickly with JTAG probes and cables from Ashling, Digilent and Lauterbach |
| Simulation models | nSIM Pro | · Build fast Virtual Prototypes to debug software before hardware is available |
| | xCAM | · Use a 100% Cycle Accurate Model to aid in hardware verification and software optimization |
| Operating Systems | MQX RTOS | · Use a small, fast, real-time operating system optimized for running on ARC processors |
| | 3rd Party | · Use open-source RTOSes such as FreeRTOS and Contiki, and 3rd party RTOSes such as ThreadX and OPENRTOS |
| Middleware | 3rd Party | · Choose middleware provided by ARC Access Program members and ported to ARC processors |

Table 2: Synopsys ARC EM5D and EM7D software and development tools

## embARC

Synopsys' embARC Open Software Platform is an easily accessible and productive solution for developing ARC processor-based embedded software. It gives software developers online access to a comprehensive suite of free and open-source software that eases the development of code for IoT and other embedded applications. Device drivers, operating systems and middleware ported to and optimized for ARC processors are available for download free of cost from the embARC.org website. The website also provides access to software development tools and documentation as well as user forums to facilitate the sharing of information and expertise among the ARC-based design community.

## Documentation

The following documentation is available for the ARC EM 5D and 7D processors:

- ARC EM5D and EM7D Databook
- ARC EM5D and EM7D Release Notes
- ARC EM5D and EM7D Getting Started
- ARCv2DSP Programmers Reference
- ARC EM Integration Guide

## Testing, Compliance, and Quality

Verification of the ARC EM5D and EM7D processors follow a bottom-up verification methodology from block-level through system-level. Each functional block within the product follows a functional coverage-driven test plan. The plan includes testing for ARCv2DSP ISA compliance as well as state- and control-specific coverage points that have been exercised using constrained pseudorandom environments and a random instruction sequence generator.

## Deliverables

The ARC EM5D and EM7D processors are delivered to system designers as Verilog HDL in the ARChitect IP Library. The HDL is configured and output from the ARChitect IP Configurator tool. To ensure that the product performs as expected, a basic testbench of Customer Confidence Tests (CCT) is delivered with the processor cores.

The MetaWare Development Toolkit comes complete with a new DSP C/ C++ Compiler specifically supporting all the DSP instructions in the ARCv2DSP ISA. The MetaWare Toolkit also features a DSP software library supporting a full set of commonly used DSP functions (Table 3).

| Function | Data types | | | |
|---|---|---|---|---|
| | FP32 | Q31 | Q15 | Q7 |
| **Scalar math functions** | | | | |
| Cosine | ✓ | ✓ | ✓ | |
| Sine | ✓ | ✓ | ✓ | |
| Tangent | | ✓ | ✓ | |
| Arccosine | | ✓ | ✓ | |
| Arcsine | | ✓ | ✓ | |
| Arctangent | | ✓ | ✓ | |
| Two argument arctangent (atan2) | | ✓ | ✓ | |
| Square root | ✓ | ✓ | ✓ | |
| Exponential 2X | | ✓ | ✓ | |
| Base 2 logarithmic (log2) | | ✓ | ✓ | |
| Reciprocal 1/x | | ✓ | ✓ | |
| Fractional division | | ✓ | ✓ | |
| **Vector math functions** | | | | |
| Addition | ✓ | ✓ | ✓ | ✓ |
| Subtraction | ✓ | ✓ | ✓ | ✓ |
| Offset—addition with scalar | ✓ | ✓ | ✓ | ✓ |
| Absolute value | ✓ | ✓ | ✓ | ✓ |
| Negation | ✓ | ✓ | ✓ | ✓ |
| Shift | | ✓ | ✓ | ✓ |
| Multiplication | ✓ | ✓ | ✓ | ✓ |
| Scale—multiplication by a scalar and shift | ✓ | ✓ | ✓ | ✓ |
| Dot product | ✓ | ✓ | ✓ | ✓ |
| Minimum | ✓ | ✓ | ✓ | ✓ |
| Maximum | ✓ | ✓ | ✓ | ✓ |
| **Complex math functions** | | | | |
| Conjugate | ✓ | ✓ | ✓ | |
| Dot product | ✓ | ✓ | ✓ | |
| Magnitude | ✓ | ✓ | ✓ | |
| Magnitude squared | ✓ | ✓ | ✓ | |
| Complex multiplication by complex | ✓ | ✓ | ✓ | |
| Complex multiplication by real | ✓ | ✓ | ✓ | |
| **Matrix functions** | | | | |
| Addition | ✓ | ✓ | ✓ | |
| Subtraction | ✓ | ✓ | ✓ | |
| Multiplication | ✓ | ✓ | ✓ | |
| Transpose | ✓ | ✓ | ✓ | |
| Scaling | ✓ | ✓ | ✓ | |
| Inversion | ✓ | | | |
| **Transforms** | | | | |
| Complex FFT | | ✓ | ✓ | |
| Complex iFFT | | ✓ | ✓ | |
| Real FFT | | ✓ | ✓ | |
| Real iFFT | | ✓ | ✓ | |
| DCT type II | | ✓ | ✓ | |
| DCT type III | | ✓ | ✓ | |
| DCT type IV | | ✓ | ✓ | |
| MDCT | | ✓ | ✓ | |
| iMDCT | | ✓ | ✓ | |

Table 3: Summary of DSP software library functions

| Function | Data types | | | |
|---|---|---|---|---|
| | FP32 | Q31 | Q15 | Q7 |
| **IIR filters** | | | | |
| Biquad cascade IIR direct form 1 | ✓ | ✓ | ✓ | |
| Biquad cascade IIR direct form 2 | ✓ | | | |
| IIR lattice | ✓ | ✓ | ✓ | |
| **FIR filters** | | | | |
| Convolution | ✓ | ✓ | ✓ | ✓ |
| Correlation | ✓ | ✓ | ✓ | ✓ |
| Real FIR | ✓ | ✓ | ✓ | ✓ |
| Complex FIR | | ✓ | ✓ | |
| FIR decimator | ✓ | ✓ | ✓ | |
| FIR interpolator | ✓ | ✓ | ✓ | |
| Adaptive LMS FIR | ✓ | ✓ | ✓ | |
| **Interpolation** | | | | |
| Linear interpolation | ✓ | ✓ | ✓ | ✓ |
| Bilinear interpolation | ✓ | ✓ | ✓ | ✓ |
| Polynomial evaluation | ✓ | ✓ | ✓ | ✓ |

Table 3: Summary of DSP software library functions (continued)

## About Synopsys IP

Synopsys is a leading provider of high-quality, silicon-proven IP solutions for SoC designs. The broad Synopsys IP portfolio includes logic libraries, embedded memories, embedded test, analog IP, wired interface IP, wireless interface IP, security IP, embedded processors, and subsystems. To accelerate prototyping, software development and integration of IP into SoCs, Synopsys' IP Accelerated initiative offers IP Prototyping Kits, IP Virtualizer Development Kit and IP subsystems. Synopsys' extensive investment in IP quality, comprehensive technical support and robust IP development methodology enables designers to reduce integration risk and accelerate time-to-market.

**For more information on Synopsys IP, visit synopsys.com/ip.**

SYNOPSYS®
Silicon to Software™


07/10/23.CS12086_ARC_EM5D-EM7D_DS.