DesignWare EV61, EV62 and EV64 Processors

**Highlights**

- Optimized for high-performance embedded vision applications
- Based on advanced ARCv2 ISA
- Fast, accurate object detection with a programmable CNN object detection engine
- CNN engine delivers >1000 GOPS/W with 5X better power efficiency than competitive vision processors
- High-performance vision CPU with 512-bit wide SIMD vector DSP and 32-bit scalar CPU
- Single-, dual- and quad-core vision CPU versions
- Supports data- and task-level parallelism
- Runs full range of vision algorithms
- Works with all host processors for vision offload
- High productivity MetaWare programming tools with OpenCV, OpenVX and OpenCL C Compiler

**Target Applications**

- Automotive Driver Assist Systems (ADAS)
- Surveillance
- Cameras
- Gesture recognition
- Object detection and classification
- Face tracking
- Home automation
- Virtual reality
- Games
- Robotics
- Digital signage

The DesignWare® EV61, EV62 and EV64 embedded vision processors are fully programmable and combine the flexibility of software solutions with the high performance and low power consumption of dedicated hardware. The EV6x family features the most integrated vision processor solution available combining a high-performance 32-bit scalar core with a 512-bit vector DSP, an optimized CNN engine, and support for user-defined APEX hardware accelerators. The processors are highly scalable and configurable enabling users to tailor them to their specific application requirements to maximize performance while minimizing power and area. The optional CNN engine is a programmable object detection engine that implements a convolutional neural network (CNN) enabling fast and accurate detection of a wide range of objects such as faces, pedestrians and hand gestures.

The EV61 features a single vision CPU (32-bit ARC® HS scalar CPU with a 512-bit wide SIMD vision digital signal processor), the EV62 features a dual-core vision CPU and the EV64 has a quad-core vision CPU. All three products can be configured with the optional CNN engine.
To speed application software development, the EV6x processor family is supported by a comprehensive software programming environment based on embedded vision standards including OpenCV, OpenVX™, and the OpenCL™ C language, as well as Synopsys’ ARC MetaWare Development Toolkit with an optimizing C/C++ compiler, debugger and instruction set simulator. The combination of high-performance hardware optimized for vision processing and high productivity programming tools makes the EV6x processors an ideal solution for a full range of embedded vision applications.

**ARCv2 ISA**

The DesignWare EV6x processors implement the ARCv2 RISC ISA, which supports advanced processor capabilities and incorporates instructions that improve code density by 18 percent compared to the previous generation of ARC products. The architecture and pipeline are designed to meet the needs of next-generation system-on-chip (SoC) applications and enable the deployment of a full range of 32-bit processors. The ARCv2 ISA enables the implementation of complex, heterogeneous SoCs with processors that are precisely targeted to meet the specific performance and power requirements for each instance on the SoC.

**Features**

- Based on advanced ARCv2 ISA
- High-performance vision CPU with 512-bit SIMD vector DSP and 32-bit scalar CPU
- Single-core (EV61), dual-core (EV62) and quad-core (EV64) vision CPU versions with optional CNN engine and L1 cache coherency
- 4K to 64K instruction and data cache
- 8k to 128K Data Closely Coupled Memory (DCCM)
- Single cycle 64-bit loads and stores
- Radix-4 hardware divider
- APEX custom instruction interface
- Input and output interrupt lines to allow to synchronize with an external host
- ARConnect multicore communication that includes:
  - Interrupt distribution unit
  - Hardware synchronization block
- Multi-bank, low-latency shared memory with concurrent access arbitration
- The optional CNN engine is configurable with up to 880 MACs/cycle and is programmable to run any fixed point CNN graph performing object detection, classification and scene segmentation
- 64-bit or 128-bit AXI bus interface
- JTAG and Compact JTAG (cJTAG) debug interface

**EV Vision CPU**

The EV6x vision CPU features a quad issue super vector architecture with a 32-bit, high-performance scalar pipeline and a 512-bit wide SIMD VDSP that are optimally balanced to achieve excellent performance with low power consumption. The core executes one scalar and three vector instructions (128-bit instruction bundle) per cycle. The pipeline is based on the ARC HS architecture and supports out-of-order retirement, sophisticated branch prediction unit (with early correction of mispredicted branches) and a late stage ALU that improves instruction throughput.

The VDSP is configurable up to 512-bits wide with vector support for dual 8x8, 16x16, dual 16x16 or 32x32 MAC processing. The DSP SIMD capability can deliver 128 dual 8-bit, 64 dual 16-bit, 32 16-bit, or 16 32-bit MACs per cycle. The VDSP unit includes a vector register file with up to 32 512-bit wide registers. The VDSP supports full predication and has scatter/gather instructions and hardware to maximize performance.

**L1 Cache Memory**

The EV processors feature separate instruction and data L1 cache that can be independently configured for 4K, 8K, 16K, 32K or 64KB size. The I-cache supports 2- and 4-way associativity, and a 32-, 64- and 128-byte line size. The caches can be individually configured to support line locking and invalidate, and to offer debug visibility. The data cache implements the MOESI protocol and supports cache-to-cache transfers.
**Data Closely Coupled Memory**
The EV61, EV62, and EV64 Processors support 8KB to 128KB of data closely coupled memory (DCCM), and 32KB to 256KB of vector memory (VCCM) for the VDSP. The CCM is implemented as a separate memory space and can be accessed every clock cycle.

**Register File and Program Counter**
The EV6x processors' register file has 32 32-bit registers. 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.

**64-Bit Load and Store**
The EV6x processors feature 64-bit load double and store double instructions that can be optionally included in the processor at build time. These are single instructions that load or store 64-bits of data to and from register pairs. There is no additional cycle penalty due to the wider and banked DCCM that support non-aligned loads and stores.

**ARC Processor EXtension (APEX) Interface**
The EV6x processors are designed to be extendable with the addition of custom instructions. These instruction extensions may include more processor and auxiliary registers, new instructions, and additional condition code tests. Custom instructions enable designers to efficiently add their proprietary hardware to the processor to further increase application performance or reduce power consumption.

**Convolution Neural Networks**
CNNs are inspired by the way our brains work to process vision. CNNs are based on a deep-learning algorithm that is trained with many images of an object and then generalized to a graph that can be used by the algorithm to find the object in pictures or video. They perform image analysis using a successive refinement process that uses multiple feature extraction layers. At each layer, the image is matched against the patterns that result from the training phase. Each successive layer extracts increasingly complex features until a final match decision is made.

The CNN algorithm features higher quality and accuracy for object detection versus other vision algorithms. A wide range of visual object types can be detected, including faces, pedestrians, body parts, animals, cars, buildings and many others. It can be used to recognize specific faces, perform facial expression analysis, determine visual attention, and assist with gesture recognition and object tracking. It can also be used for abstract scene recognition and image segmentation by recognizing the sky, mountains, trees, buildings, etc.

**CNN Engine**
The EV6x processors can be configured with an optional programmable embedded deep neural network engine used to run CNN executables. The engine is optimized for object detection, image classification and scene segmentation with excellent performance efficiency. The configurability of the CNN engine enables the flexible mapping of CNN graphs into the engine resources.

CNN graph training is done off-line, typically on a server farm, and the resulting graph is programmed into the object detection engine by the user with the CNN graph mapping tool.

**Cluster Shared Memory**
A low-latency shared data memory is included in the processor to support information passing and coordination between the multiple CPUs and the CNN processing element cores. This memory is used as a software-managed scratch pad and is configurable from 0 to 8MB. To allow for larger sizes, the memory is internally multi-banked, but this is invisible to the software. It includes arbitration to support concurrent access from the CPU cores and/or the CNN processing elements. The shared subsystem data memory is optional.

**Streaming Transfer Unit**
The EV6x processor has a configurable DMA controller to permit efficient transfer of data between the processor and different memory regions on- or off-chip.
**Bus Interface**
The EV6x processor has native support for the ARM® AMBA® AXI™ bus protocol. The AXI bus is 64- or 128-bits wide to improve system throughput.

**Interrupts and Exceptions**
The EV6x processor supports up to four output interrupt pins, and up to three input interrupt lines. These can be used, for example, to synchronize with an external host. The host can also raise an interrupt by writing in a memory-mapped register or by driving an interrupt input pin on the EV6x processor.

**SoC Integration**
The EV6x processors are designed to integrate seamlessly into a SoC. They can be used with any host processor and operate in parallel with the host. The EV6x family includes support for synchronization with the host through message passing and interrupts. In addition, part of the EV6x processor memory map can be made visible to the host. These features enable the host to maintain control while allowing all vision processing to be offloaded to the EV6x processor, reducing power and accelerating vision computation. The EV6x processors can access image data stored in a memory mapped area of the SoC or from off-chip sources independently from the host through the ARM AMBA AXI standard system interface if required.

**Complete Suite of Development Tools**
To facilitate rapid development of EV6x processor-based solutions, the processors are supported by a complete suite of development tools. This includes the MetaWare Development Toolkit, which includes a C/C++ Compiler that generates highly efficient scalar code, a source-level debugger and the ARC nSIM instruction-set simulator. Also available is the MetaWare EV SDK Option, which consists of the OpenCV Library, OpenVX Runtime framework & kernels and an OpenCL C language compiler. The OpenCL C compiler is a vectorizing compiler for programming the VDSP. OpenCL C is an open standard programming language (developed by the Khronos Group) that supports vectorization and is used to ease the programming of the 512-bit wide VDSP. OpenCL C is a C-like language and is used with the EV6x processors to develop kernels that are executed in the OpenVX graph.

<table>
<thead>
<tr>
<th><strong>MetaWare Development Toolkit</strong></th>
<th>Includes the MetaWare C/C++ Compiler, MetaWare Debugger and nSIM Instruction Set Simulator</th>
<th>• Develop highly optimized code with an efficient C/C++ compiler for scalar processors</th>
<th>• Debug code with a comprehensive source-level debugger and profiler</th>
<th>• Use a simulator to develop and debug software before hardware is available</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MetaWare EV SDK Option</strong></td>
<td>Includes the OpenCV Library, OpenVX Runtime &amp; Kernels and OpenCL C Compiler</td>
<td>• Use a standard library of open source functions for common vision applications with the OpenCV library</td>
<td>• Develop high-performance vision applications with the OpenVX runtime framework and kernels</td>
<td>• Use OpenCL C language to develop kernels for use in the OpenVX environment</td>
</tr>
<tr>
<td><strong>ARC nSIM Pro</strong></td>
<td>Includes fast nSIM Instruction Set Simulator and EV Virtualizer Development Kit (VDK)</td>
<td>• Use a fast simulator to develop and debug software before hardware is available</td>
<td>• Use a virtual prototype of an EV Processor system to start early software development</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. DesignWare ARC software and development tools

The EV SDK also provides an implementation of the standard OpenCV Library. OpenCV (an Open Source Computer Vision library) is a software library of more than 2500 algorithms that can be used with the MetaWare tools and provides a software infrastructure for embedded vision applications. OpenCV can be used to detect and recognize objects, and a full range of machine vision capabilities.
To enable the development of high-performance vision applications, the OpenVX framework including the 43 standard kernels is included in the EV SDK. OpenVX is a Khronos standard for acceleration of embedded vision algorithms. Client-defined OpenVX functions are supported, with the kernels described in standard C/C++ or in OpenCL. OpenVX graphs are automatically mapped, tiled (where possible), and executed on the EV6x processor and object detection engine. OpenCV and OpenVX are complementary and used together in vision applications.

**Documentation**

The following documentation is available for the DesignWare EV6x Processors:

- EV6x Processor Release Notes
- EV6x Processor Getting Started guide
- ARCv2 Programmers Reference
- EV6x Processor Databook

Testing, Compliance and Quality Verification of the EV6x processors follows 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 ARCv2 ISA compliance as well as state- and control- specific coverage points that have been exercised using constrained pseudo-random environments and a random instruction sequence generator.

**Deliverables**

The DesignWare EV61, EV62 and EV64 Processors are delivered as Verilog HDL in the ARChitect IP Library. The HDL is configured and output from the ARChitect IP Configurator tool. To test that the product performs as expected, a basic testbench of Customer Confidence Tests (CCT) is included.

**About DesignWare IP**

Synopsys is a leading provider of high-quality, silicon-proven IP solutions for SoC designs. The broad DesignWare 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 Virtual Development Kits 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 DesignWare IP, visit [http://www.synopsys.com/designware](http://www.synopsys.com/designware). Follow us on Twitter at [http://twitter.com/designware_ip](http://twitter.com/designware_ip).

[1] Product is based on a published Khronos Specification and is expected to pass the Khronos Conformance Process. Current conformance status can be found at [www.khronos.org/conformance](http://www.khronos.org/conformance).