AES
AES Encrypt/Decrypt Core

The AES encryption IP core implements Rijndael encoding and decoding in compliance with the NIST Advanced Encryption Standard. It processes 128-bit blocks, and is programmable for 128-, 192-, and 256-bit key lengths.

Two architectural versions are available to suit system requirements. The Standard version (AES-S) is more compact, using a 32-bit datapath and requiring 44/52/60 clock cycles for each data block (128/192/256-bit cipher key, respectively). The Fast version (AES-F) achieves higher throughput, using a 128-bit datapath and requiring 11/13/15 clock cycles for each data block.

Various cipher modes can be supported (CBC, CFB, CTR, ECB, LRW, and OFB). The core works with a pre-expanded key, or with optional key expansion logic.

The AES cores are fully synchronous designs, have been evaluated in a variety of technologies, and are available optimized for ASICs or FPGAs.

Applications

The AES can be utilized for a variety of encryption applications including:

- Protected network routers
- Electronic financial transactions
- Secure wireless communications
- Secure video surveillance systems
- Encrypted data storage

Block Diagram
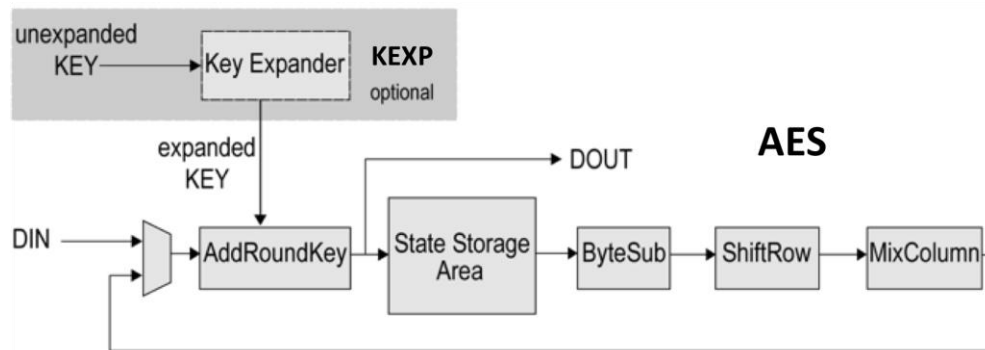
Functional Description

An AES encryption operation transforms a 128-bit block into a block of the same size. The encryption key can be chosen among three different sizes: 128-, 192- or 256-bit. The key is expanded during cryptographic operations.

The AES algorithm consists of a series of steps repeated a number of times (rounds). The number of rounds depends on the size of the key and the data block. The intermediate cipher result is known as state.

	KSIZE = 00	KSIZE = 01	KSIZE = 10
Rounds	10	12	14

Number of rounds as a function of key size

Engineered by

ocean logic

Functional Description (cont.)

Initially, the incoming data and the key are added together in the AddRoundKey module. The result is stored in the State Storage area.

The state information is then retrieved and the ByteSub, Shiftrow, MixColumn, and AddRoundKey functions are performed on it in the specified order. At the end of each round, the new state is stored in the State Storage area. These operations are repeated according to the number of rounds.

The final round is anomalous as the MixColumn step is skipped. The cipher is output after the final round.

Key Expansion

The AES algorithm requires an expanded key for encryption or decryption. The CAST KEXP AES key expander core is available as an AES core option.

During encryption, the key expander can produce the expanded key on the fly while the AES core is consuming it. For decryption, though, the key must be pre-expanded and stored in an appropriate memory before being used by the AES core. This is because the core uses the expanded key backwards during decryption.

In some cases a key expander is not required. This might be the case when the key does not need to be changed (and so it can be stored in its expanded form) or when the key does not change very often (and thus it can be expanded more slowly in software).

Available Versions

The AES core is available in ECB, CFB, CBC, OFB, and CTR modes, and for different datapath sizes. The related KEXP key expander core is available as a standalone core.

Related Products

A smaller and faster AES-E core is also available for encoding-only applications.

AES in CCM, GCM and XTS modes are available as stand-alone cores.

AES-P is a run-time programmable AES core supporting ECB, CBC, CFB, OFB, and CTR modes.

Export Permits

This core implements encryption functions and as such it is subject to export control regulations. Export to your country may or may not require a special export license. Please contact CAST to determine what applies in your specific case.

Support

The core as delivered is warranted against defects for ninety days from purchase. Thirty days of phone and email technical support are included, starting with the first interaction. Additional maintenance and support options are available.

Implementation Results

The AES can be mapped to any ASIC technology or FPGA device (provided sufficient silicon resources are available). The following are sample results for CBC mode with all core I/Os assumed to be routed on-chip, and throughput for a 128-bit key.

AES Standard Core Implementation Results

Family	LUTs	BRAMs	Freq. (MHz)	Throughput (Mbps)
Kintex-7 (-3)	244	2	300	873
Virtex-7 (-3)	244	2	300	873
Kintex UltraScale (-3)	244	2	350	1,018
Kintex UltraScale+ (-3)	243	2	575	1,673
Versal (-2)	219	2	425	1,236

AES Fast Core Implementation Results

Family	LUTs	BRAMs	Freq. (MHz)	Throughput (Mbps)
Kintex-7 (-3)	573	8	250	2,909
Virtex-7 (-3)	566	8	250	2,909
Kintex UltraScale (-3)	580	8	350	4,073
Kintex UltraScale+ (-3)	616	8	475	5,527
Versal (-2)	621	8	375	4,364

The provided figures do not represent the higher speed or smaller area for the core. Please contact CAST to get characterization data for your target configuration and technology.

Verification

The core has been verified through extensive synthesis, place and route and simulation runs. It has also been embedded in several products, and is proven in both ASIC and FPGA technologies.

Deliverables

The core is available in ASIC (RTL) or FPGA (netlist) formats, and includes everything required for successful implementation:

- Sophisticated HDL Testbench (self-checking)
- C Model & test vector generator
- Simulation script, vectors & expected results
- Synthesis script
- User documentation

www.cast-inc.com • info@cast-inc.com
Contents subject to change without notice.
Trademarks are the property of their respective owners.

Engineered by
ocean logic