Algotronix®

130-10 Calton Road
Edinburgh, Scotland
United Kingdom, EH8 8JQ
Phone: +44 131 556 9242
E-mail: cores@algotronix.com
URL: www.algotronix.com

Core Facts

Provided with Core	
Documentation	User Manual
Design File Formats	VHDL or EDIF Netlist
Verification	Test Bench, Test Vectors
Instantiation templates	VHDL
Simulation Tool Used	
ModelSim, Xilinx ISE Simulator	
Support	
Support provided by Algotronix	

Features

- Implementation of AES-GCM with 96 bit IV and 128 bit or 256 bit keys.
- Suitable for MACSEC, IPSEC, OTN and customer specific applications requiring AES-GCM
- 10Gbit/sec performance on Virtex 5
- Overlapped processing allows one packet to start before the previous one completes and delivers 10Gbit/sec on a stream of minimum sized packets
- Pass through functionality for packets not requiring encryption
- Comprehensive self-checking testbench
- Supplied as VHDL source code to allow security review

Applications

- Wired, Optical and Wireless Networking
- 10Gbit Ethernet
- Network Test Equipment

General Description

The Algotronix AES-GCM Core implements the proposed Galois Counter Mode of operation of the AES algorithm. This mode of operation is described in a proposal by Cisco to the National Institute of Standards and Technology (NIST) and has been adopted by the IEEE for the 802.1 MAC standard. NIST has published Special Publication SP800-38D describing GCM mode making it an officially endorsed mode of operation of the AES cipher.

AES-GCM is significantly more complex than the simple standard modes of AES such as ECB and CBC specified in NIST special publication SP800-38A. Unlike simple modes of the cipher which provide only confidentiality, GCM provides both confidentiality and authentication. Authentication is the ability to detect tampering with the encrypted message as it passes between the sender and receiver and in most applications is essential for security. GCM mode also provides a standard method for processing data streams whose length is not a multiple of the 128 bit AES block size. GCM mode is based on the counter (CTR) mode of AES which can be parallelized to achieve very high throughput. Thus GCM is the most suitable of the standard modes of AES to provide both authentication and confidentiality for very high speed networks.

The Algotronix GCM 10G core is an expanded and optimized variant of its AES-GCM offering. The core was originally developed to meet the demanding requirements of a leading manufacturer of test equipment for optical networks. The core delivers 10Gbit/sec worst-case performance on the GCM processing required for IEEE 802.1 MACSEC even on stream of minimum size packets with a key change on each packet. The area of the core has been carefully optimized to allow customers to fit separate Encrypt and Decrypt channels including the MAC security logic into a single FPGA.
The GCM 10G core operates with a fixed 96 bit IV and a 128 bit key as specified in IEEE802.1. Algotronix can configure the core to work with 256 bit keys if required. The core is supplied with a testbench which implements vectors from the GCM proposal. The testbench also includes a set of several thousand random test vectors generated from a software implementation of GCM. These additional vectors have randomly chosen AAD and text (plaintext or ciphertext depending on whether the core is operating as an encryptor or decryptor) lengths and cover several important 'corner' cases not included in the vectors in the GCM proposal. The comprehensive testbench for the G3 core, which implements the entire NIST AESAVS test suite, can be used to verify the AES functionality. The GCM testbench operates in self checking mode and can be used as a regression test to verify any changes you may make to the core source code.
The Algotronix AES-GCM-10G core is supplied as VHDL source code and can be configured using a number of VHDL generic parameters to select only those features which are required in order to conserve area. The core can be configured as Encryptor, Decryptor or Encryptor/Decryptor. The core provides hardware key schedule generation and allows a change of key for each packet while maintaining 10Gbit/sec throughput.
The AES core is an easy to use fully synchronous design with a single clock, the core is designed for stream processing with start and finish signals are used to signal the beginning and end of packets passing through the core. The core has been designed for efficiency in modern FPGAs and makes full use of FPGA specific features such as dual port memory blocks. The memory blocks are the only FPGA specific components used and the core could be implemented on ASIC by replacing the FPGA memory blocks with memories from an ASIC library.

Implementation Statistics

Example implementation statistics for an Encrypt only design are provided below. The core can target all recent families of Xilinx chips and has a variety of configuration options, Algotronix will run the core through FPGA vendor design tools and provide figures for devices and options not listed in the tables below and on request. A clock frequency of 156.25MHz is used to process 10Gbit ethernet traffic, the tables below show the maximum achievable clock frequency.

Table 1a: Example Implementation Statistics for AES-GCM, 128 bit Key, 'Push Button' flow with Fmax specified by clock constraint. AES Sboxes implemented in logic.

Family	Example Device	Fmax (MHz)	Slices[1]	IOB[2]	GCLK[2]	BRAM	MULT/ DSP48	DCM	Throughput (GBit/sec)	Design Tools
Virtex-7	XCV330T-3	202	4042	673	1	0	0	0	12.9	Vivado 2014.1
Virtex-7 Ultrascale	XCKU075-3	266.6	2225 (CLBs)	673	1	0	0	0	17	Vivado 2014.1

Table 1b: Example Implementation Statistics for AES-GCM, 128 bit Key, 'Push Button' flow with Fmax specified by clock constraint. AES Sboxes implemented in block RAM.

Family	Example Device	Fmax (MHz)	Slices[1]	IOB[2]	GCLK[2]	BRAM	MULT/ DSP48	DCM	Throughput (GBit/sec)	Design Tools
Virtex-7	XCV330T-3	190.4	2987	673	1	22	0	0	12.16	Vivado 2014.1

Table 1c: Example Implementation Statistics for AES-GCM, 256 bit Key, 'Push Button' flow with Fmax specified by clock constraint. AES Sboxes implemented in block RAM.

Family	Example Device	Fmax (MHz)	Slices[1]	IOB[2]	GCLK[2]	BRAM	MULT/ DSP48	DCM	Throughput (GBit/sec)	Design Tools
Virtex-7	XCV330T-3	206	3290	673	1	30	0	0	13.1	Vivado 2014.1

Notes:

1) Actual slice count dependent on percentage of unrelated logic – see Mapping Report File for details

2) IOB count when all core I/Os and clocks are routed off-chip, **which is not the intended usage**. Data will normally be input through fast serial I/O and processed by user logic before being supplied to the AES-GCM core. GClk signal is normally shared with user's design. Example devices are chosen to provide sufficient I/O to route out all signals, smaller devices would normally be used.

Functional Description

The main functional blocks are as shown in Figure 1, and explained below. The various I/O signals shown on the diagram are defined in Table 2.

AES

This block provides a pipelined implementation of the ECB mode of the AES Algorithm for use by GCM. The AES implementation is based on Algotronix' proven AES-G3 core.

GF Multiply

The GCM algorithm requires a 128 bit x 128 bit Galois Field (GF) multiply operation for each AES encryption. The Algotronix AES-GCM core provides a highly configurable GF multiplier with parameterisable data path width and digit size. Data path width is selected to match that chosen for the AES unit and the digit size is chosen to match the Galois Field multiply throughput with the throughput of the AES core. Wide digit sizes require more area but compute the multiply in fewer clock cycles. The Algotronix implementation of the GF multiplier was chosen to minimize latency following a change of cryptographic key so that the core can efficiently support minimum sized packets with a potential key change on each packet.

GCM Mode Logic

This block contains the datapaths required to route data between the various computational units within the core under the control of the GCM-Control block.

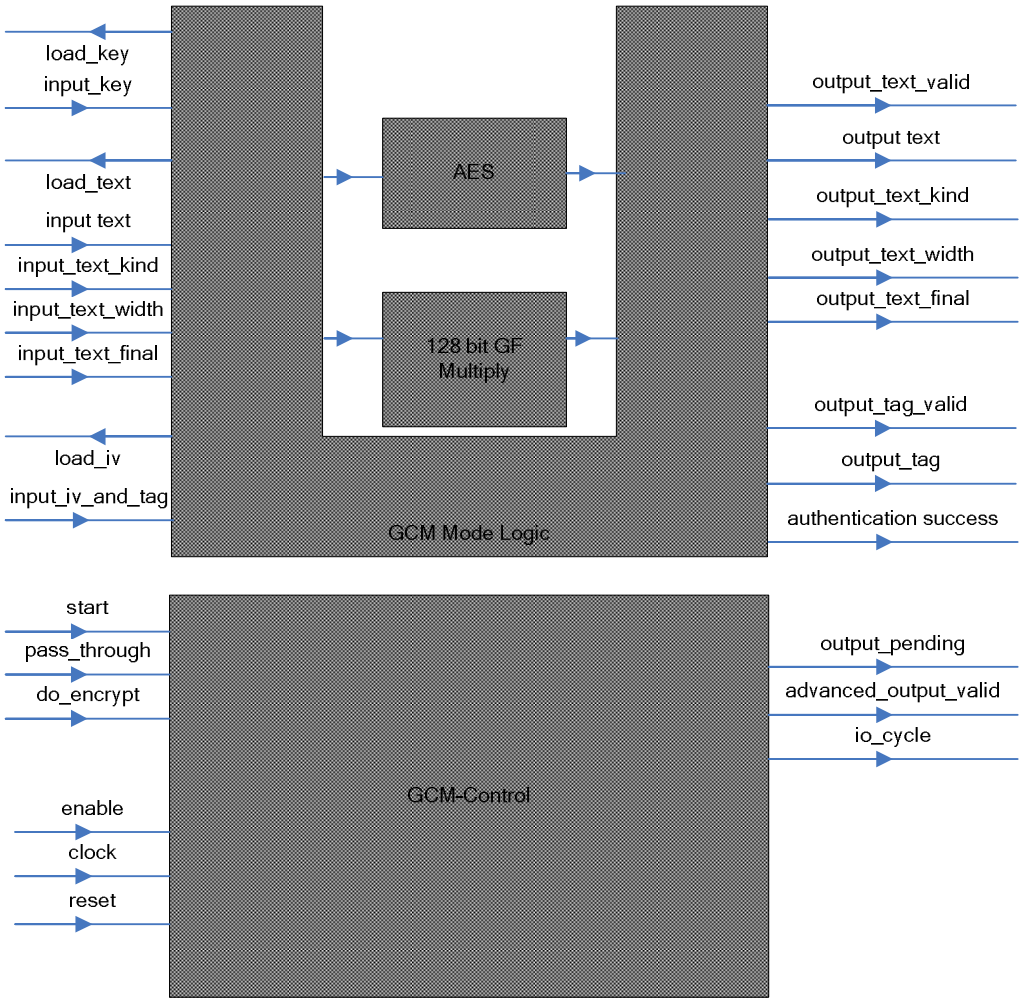

Figure 1, AES-GCM-10G Core Block Diagram

Compilation Options

The core can be configured easily using a set of VHDL generic parameters. Normally, it is unnecessary for users to modify the design source code although the code is supplied and they are free to do so if they wish. Algotronix can also customise the core as a service for users with particular requirements which are not met by the standard product.

- **cipher_function** - specifies whether an Encryptor, Decryptor or Encryptor/Decryptor is required. Most applications of AES-GCM-10G have a separate transmit and receive channel using ENCRYPT and DECRYPT configurations respectively.
- **output_ciphertext_tag_on_decrypt –** On decrypt the core compares the value of the tag supplied with the ciphertext with the internally calculated 'correct' value to determine whether the packet is authentic and sets the authentication_success signal accordingly. The core also outputs the value of the tag on the output_text bus following the packet AAD and decrypted plaintext. This generic parameter specifies whether the tag value output should be the 'correct' tag calculated by the core or the possibly incorrect tag supplied with the ciphertext. Outputting the ciphertext tag provides more information if the system wishes to log authentication errors.
- **force_output_low_until_valid** – When true the AES core will hold the output_text bus low at all times when valid output data is not present. When this signal is false the circuitry to hold the output zero will be omitted, saving some area. In this case the core output 'output_text' will show the values at intermediate rounds of the cipher as well as the final round. This data is not fully encrypted and, if available to an attacker, could compromise security of both the key and data. Therefore, this parameter should only be set to false if the user design which contains the core can guarantee that an attacker will not be able to monitor the core output directly. The circuits which provide this function are on the critical timing path so in the 10Gbit application the surrounding circuitry would normally be designed so that this parameter can safely be set to false.
- **target_device** – This edition of the core can target all families of Xilinx FPGAs. Alternative editions target other FPGA manufacturers as well as a 'platinum' edition which can target all leading FPGAs or ASIC. To achieve 10Gbit/sec performance the Xilinx Virtex 5 or later high performance FPGA families are recommended. The core can be compiled to target lower performance Xilinx FPGA families but the achievable clcok frequency may not be sufficient for 10G throughput.

Core I/O Signals

Descriptions of all I/O signals are provided in Table 2. In most cases these signals will connect to signals in the surrounding user design, not directly to I/O pins on the FPGA.

Signal	Signal Direction	Description
clock	input	156.25MHz Clock – active on rising edge
io_cycle	output	The AES-GCM core inputs and outputs data every second clock cycle. This signal allows external circuits to align with the input/output cycles of the AES-GCM core. This signal continues in gaps between packets whereas the load_text signal is only active when data is actually being transferred.
reset	input	Asynchronous reset – active high. For Xilinx FPGA implementation, unless security considerations mandate an asynchronous reset, it is recommended to specify that the reset signal is implemented synchronously in the synthesis tools. This can result in reduced area and improved performance.
enable	input	Module clock enable – 0: module is inactive, 1: module runs. This signal can be used for flow control to pause the core when the system is not ready to supply or to accept data.
do_encrypt	input	Specifies whether the core should operate in Encrypt or Decrypt mode. This input is only significant if the compilation option cipher_function is set to ENCRYPT_DECRYPT i.e. hardware for both encryption and decryption has been included.
pass_through	input	Specifies that the following packet is to be passed through the core without any security processing. This is a convenience feature so that the user does not have to provide a separate path around the GCM core for non-MACSEC packets.
start	input	Starts a new packet to be processed. The control signals do_encrypt and pass_through are sampled and these parameters are fixed for the next block of operations. The key is assumed to have changed and is reloaded and the hash key H required by the GCM algorithm is recalculated.
input_text_final	input	Set by external circuits to mark the final block of input text in a packet.
input_text_kind[1:0]	input	Specifies whether the current 128 bit block of input text is additional authenticated data (TEXT_AAD) or plaintext/ciphertext for encryption or decryption (TEXT_TEXT) .
input_text_width[4:0]	input	Specifies the number of bytes in the current 128 bit block which are actually used. In GCM more significant bits are filled first – so on a partial block it is the least significant bits which will be empty. Unlike the simpler modes of AES, GCM can deal with data streams which are not a multiple of 128 bits. All 128 bits will be used on every block except the last block in a stream of AAD or TEXT blocks to be processed
input_text [127:0]	input	Data input. A complete 128 bit AES block is transferred in a single clock cycle.
load_text	output	Load flag – high when the core is ready to load an IV over the input_iv_and_tag bus. This happens simultaneously with loading the first block of text over input_text.
input_iv_and_tag [127:0]	input	Used to transfer the 96 bit IV on bits [127:32]. On decrypt the bus is also used to transfer the 128 bit expected ICV value simultaneously with loading the final block of text through input_text.
load_text	output	Load flag – high when input_text is being loaded. The user may bring enable low during the clock cycle when load_text = 1 if the external system is not ready to provide more input text.
load_key	output	Load flag – high when the key is being loaded.
input_key [127 : 0]	input	Bus to input the 128 bit key for the AES encryptor. Two clock cycles are required for a 256 bit key.
output_text_valid	output	Valid flag – high when output_text is valid.
advanced_output_text_ valid	output	High on the clock cycle before output_text_valid. Can be used as advanced warning of a data transfer by flow control circuitry.

output_text_kind	output	Specifies whether the current 128 bit block of output text is additional authenticated data (TEXT_AAD) or plaintext/ciphertext for encryption or decryption (TEXT_TEXT).
output_text_width[4:0]	output	Specifies the number of bytes in the current 128 bit block which are actually used. Unlike the simpler modes of AES, GCM can deal with data streams which are not a multiple of 128 bits. All 128 bits will be used on every block except the last block in a stream of AAD or TEXT blocks.
output_text [127 : 0]	output	Data output: in the current version of the core this bus is 128 bits wide and the 128 bit block of text is transmitted in a single clock cycle.
authentication_success	output	On decrypt indicates whether the hash function has successfully authenticated the data stream. This signal is valid on the clock cycle following the cycle with output_tag_valid = '1'
output_tag_valid	output	Valid flag – high when output_tag is valid.
output_tag [127 : 0]	output	Output bus for the ICV computed from the ciphertext. On decrypt this is compared with the expected ICV input over input_iv_and_tag and if they are equal authentication_success is asserted.
output_pending	output	Indicates that the core is currently processing input data and there will be further output_valid cycles.
output_side_final	output	Indicates that the block currently being output is the final AAD or text in the packet. This signal is useful in pass through mode where it is impossible to determine where one packet finishes and the next one starts by looking at the output_text_kind signal.

Table 2: Core I/O Signals.

Description of Operation

In the description below 'block' is used to refer to an AES block of 128 bits of data, 'packet' is used to refer to an AES-GCM processing unit consisting of IV, AAD, plaintext/ciphertext and tag with an associated key. In the 802.1 MACSEC application the packet must have two blocks of AAD and may have zero or more blocks of plaintext/ciphertext.

Processing a packet of GCM data is initiated by pulsing the start signal high for one clock cycle. The control signals do_encrypt and pass_through signals which set the operating mode of the core for this packet must be valid at this time. These signals are latched internally during start and changes during period while the core is processing data will have no effect.

In this document the sequence of activity is described but the number of clock cycles between phases of activity is not specified. Ideally, the user circuit should synchronize to the GCM core using the load_text, load_key and output_valid signals rather than by assuming a set number of clock cycles between various operations. Future updates to the core may have slightly different delays as a consequence of changes to improve latency and/or throughput.

The AES-GCM_10G core implements the AES portion of the algorithm using an implementation with a 128 bit datapath and a 5 stage pipeline for 128 bit keys (7 stage pipeline for 256 bit keys which require more rounds of AES processing). This AES unit can encrypt or decrypt a 128 bit block of text with a latency of 10 clock cycles (14 clock cycles for 256 bit keys) and a throughput of one block every two clock cycles. This timing follows directly from the iterative nature of the AES algorithm which requires ten cycles of inner loop processing when the key length is 128 bits.

Each ‘packet’ or sequence of blocks to be processed by AES-GCM requires two ‘overhead’ encryptions as well as the encryptions necessary to process the plaintext or ciphertext. One overhead encryption is needed to calculate the hash key ‘H’ and another to encrypt the result of the GF hash algorithm to produce the tag. Thus, maintaining 10Gbit/sec throughput is harder for small packets since the percentage overhead is higher.

Since the key may potentially change for each packet it is also vital to minimize the latency associated with keyschedule calculation. This is achieved by operating the G3 AES implementation in ‘ONLINE’ mode where the keyschedule is generated on the fly as required rather than pre-calculated and stored in a buffer (more details are provided in the AES G3 product documentation). The AES-GCM-10G core also provides two keyschedule units to allow overlapped operation of the Encryptor where processing the next packet can be started before the final operations are completed on the previous packet. In this case some of the pipelined encryptors will be operating from the new keyschedule while others are operating with the old one so two keyschedule units are necessary. Overlapped operation is vital to maintain throughput and keep the encryption pipeline full when dealing with minimum sized packets.

Although the AES encryption operation in AES GCM can be pipelined to increase throughput parallelizing the GHASH operation is more problematic because GHASH is an iterative algorithm with a feedback loop: the input value for one stage involves XOR’ing the new data with the GHASH value from the previous stage. The implementation of the Galois Field multiplier required to calculate GHASH is based on the Kurutsaba algorithm.

As well as the ‘start’ signal which signals the start of a new packet the AES-GCM10G core has a ‘final’ input signal which goes high on the last 128 bit block of AAD or text in a packet. When there is no gap between packets these signals may go high on the same clock cycle so that processing of the next packet starts immediately. When there is a gap between packets there may be a considerable delay between the final signal and the start signal for the next packet. When the gap between packets is small, so the AES unit is still processing the previous packet when the new packet is started, then the gap between the start signal for the new packet and the final signal from the previous packet must be a multiple of two clock cycles. This constraint allows all the stages of the AES pipeline to transition on the same clock cycle. The constraint is not usually a problem because the external user circuitry is normally assembling a 128 bit block of text for the AES unit from multiple smaller words and has this property naturally. If the inter packet gap is long enough so the previous packet has cleared the output side of the AES-GCM unit then there is no alignment constraint and start may occur on any clock cycle.

There are many possible timing scenarios depending on inter packet gap and whether packets are pass-through or processed by GCM. For reasons of space and convenience detailed timing charts for the different scenarios are not provided in the product description but are produced as required using the testbench and supplied separately.

Verification Methods

The testbench includes a behavioral model of AES-GCM and a self-checking configuration of the top level entity in the VHDL design which uses the behavioral model to check the results from the synthesisable implementation code. This means that, except for known-answer-tests the testbench only generates stimulus for the hardware, checking of the response from the synthesisable implementation is done by the self-checking code.

The AES_GCM_10G testbench can operate in three modes:
a. As a Known Answer Test reading in files of test vectors which contain the expected results and checking the output of the hardware against the expected values in the vector file. Test vectors from the original AES_GCM proposal and samples from standards bodies are provided as well as longer vector files created by Algotronix from C language implementations of AES GCM. Known Answer Testing also verifies the behavioral model within the self test code.
b. As a Qualification test reading vector files provided for qualfiication testing of AES-GCM and writing a response file for submission to the NIST approved test house.
c. As a Random test generating a user specified number of random test vectors on the fly.

The self-checking configuration of the AES_GCM _10G core can also be instantiated within the user's own simulations. This makes it easy to verify the core operates properly when connected to the user circuitry surrounding the core. The assertions within the self checking code will detect and report most situations where the user design is not driving the core correctly making integrating the core within the larger user design easier.

Recommended Design Experience

It is recommended that the user is familiar with the VHDL language and with the Xilinx design flow and simulation tools. The core can also be instantiated inside a wrapper to allow use with a Verilog design flow.

It is also recommended that the user has a background in data security or takes appropriate advice when considering how to implement AES-GCM-10G in a larger system.

Ordering Information

This product is available directly from Algotronix under the terms of the SignOnce IP License. Please contact Algotronix for pricing and additional information about this product using the contact information on the front page of this datasheet. To learn more about the SignOnce IP License program, contact Algotronix or visit the web:

Email: commonlicense@xilinx.com
URL: www.xilinx.com/ipcenter/signonce

Export Control

Strong encryption technology such as AES is the subject of international export regulations. Algotronix is located in the United Kingdom and export of this core is regulated by the UK government.

The core is freely available within the European Union and in addition can be supplied immediately to the following countries: United States, Australia, New Zealand, Canada, Norway, Switzerland, Japan.

Export to other countries requires an export licence. The UK Department of Business, Enterprise and Regulatory Reform publishes information on their website (www.berr.gov.uk) which gives an indication of average export licence processing times for various countries and the percentage of licence requests which are granted. For many countries obtaining an export licence can be done relatively quickly and with only a small amount of additional paperwork.

It is the the responsibility of the customer to comply with all applicable requirements with respect to re-export of products containing the AES technology.

Related Information

Industry Information

The AES standard documents FIPS197, SP800-38A and AESAVS, the original GCM proposal to NIST ("The Galois/Counter Mode of Operation (GCM)" by David McGrew and John Viega) and the NIST special publication SP800-38D document describing GCM mode are available from the National Institute of Standards and Technology, Computer Security Resouce Center website (www.csrc.nist.gov).

Xilinx Programmable Logic

For information on Xilinx programmable logic or development system software, contact your local Xilinx sales office, or:

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-559-7114
URL: www.xilinx.com

URL: www.algotronix.com

Version Control Information	
Subversion Revision Number	**51**
Date	2014/06/14 17:39:04
Document	Aes Gcm 10g Data Sheet, Xilinx Edition
Status (blank field indicates OK/no warnings)	
	(Table auto-updates, do not edit field values by hand)