AES 128/192/256 (ECB) AVALON®-MM SLAVE

Thomas Ruschival
and opencores.org

www.opencores.org

ruschi@opencores.org

avs_aes_doc (v. 0.5) - 2010/04/02

Contents

1 **Introduction** 3

2 **Interface** 3

2.1 Configuration Generics 3

2.2 Signals 4

3 **Memory Map** 4

3.1 Control Register 5

4 **Protocol Sequence** 6

4.1 Interrupt Behaviour 7

5 **Ressource Usage and Throughput** 7

5.1 Exemplary FPGA implementations 8

6 **Simulation and Software Driver** 9

6.1 Testbench 9

6.1.1 Simulation 10

6.2 Software Driver 10

6.2.1 Configuration 10

6.2.2 Compilation 10

7 **The Inner Core** 10

8 **License and Liability** 13

1 Introduction

The Advanced Encryption Standard (AES)is a symmetric block cypher operating on fixed block sizes of 128 Bit and is specified for key sizes of 128, 192 and 256 Bit designed by Joan Daemen and Vincent Rijmen. The algorithm was standardized by National Institute of Standards and Technology (NIST). For more information on the algorithm see [1].
This component implements an AES encryption decryption datapath in Electronic Codebook (ECB)mode with either 128,192 or 256 Bit keys. The keylength is determined by generics at compile time. Also the decryption datapath can be disabled by generics if it is not needed for the application.
The component provides an Avalon® Memory Mapped (Avalon-MM) slave interface to connect to an Altera® Avalon® switch fabric. The Avalon® interface is implemented in a way that it can also be used to connect to a Whishbone master if the signals are correctly mapped, see [2]. For further information about the Whishbone bus refer to [3].

2 Interface

The AES core is accessed by the interface described in this section. An Avalon® interface was chosen for its simplicity and compatibility with wishbone. Furthermore Avalon® defines interrupt request signals for slaves which would be separate signals in a Wishbone implementation.The component can be used both in polling mode or can provide an interrupt for signalling.
Unfortunately Avalon® is an Altera® proprietary technology. The actual AES core however is a selfcontained entity and can be embedded into other System on Chip (SoC) bus interfaces as well or used indepentently.

2.1 Configuration Generics

The AES core can be configured by generics shown in table 1, consequently they are provided by the Avalon® interface.

Generic name	type	Description
`KEYLENGTH`	NATURAL	Size of initial userkey. Must be 128, 192 or 256 [1] .
`DECRYPTION`	BOOLEAN	Enables the instantiation of the decrypt datapath if true.

Table 1: Component generics

Note: `KEYLENGTH` of 192 fail synthesis with Xilinx ISE ® because of division by 6 in key schedule that cannot be mapped to shift operations (`keyexpansion.vhd`).

[1] All other values raise a compilation failure

2.2 Signals

The Avalon®MM Slave interface is described in [4], the component implements the signals shown in table 2.2. All signals are synchronous, sampled at the rising edge of the clock. The type for all signals is `IEEE1164 std_logic` or `std_logic_vector`. For signals wider that 1 Bit the range is Most Significant Bit (MSB) `downto` Least Significant Bit (LSB)˙
This components has only output signals driven by registers no input signals are directly combinatorially connected to the output signals, thus combinational loops are avoided. All signals are active high. This component does not support burst transfers.

Signal name	Width	In/Out	Description
`clk`	1	in	Avalon® bus clock, also used to drive the core.
`reset`	1	in	*Synchronous* reset signal for Avalon® bus interface. The core itself is designed without need for reset signals.
`writedata`	32	in	Input data to write to location designated by `address`. Bit 31 is most significant Bit.
`address`	5	in	Word offset to the components base address. The memory map of the component for the respective offest is described in 3. Only full 32-Bit words can be addressed no byte addressing is implemented.
`write`[1]	1	in	If asserted enable write of data at `writedata` to location designated by `address`.
`read`[1]	1	in	If asserted output data at location designated by `address` to `readdata`.
`readdata`	32	out	Data output port for reading data at the location defined by `address`. Bit 31 is most significant Bit.
`waitrequest`	1	out	Asserted if writedata was not accepted, this is the case if the keyexpansion is not yet complete and a new is written to the `KEY` address range without previous deassertion of the `KEY_VALID` Bit
`irq`	1	out	If Interrupt behaviour is enabled `IRQ` will be asserted when the operation has terminated. For use of interrupt see 4.1

Table 2: Avalon® Bus interface signals

3 Memory Map

The AES core Avalon® slave has an address space of 31 words accessable through the offset described by the signal `address`, see 2.2. This address space is devided into three main sections for the 4-word input data, the 4-word result of the operation and the user key. The actual lenght of the userkey can vary between 4, 6 and 8 words depending on the keysize. For control signals and status information of the component and a control word is provided. The memory

[1]`read` and `write` are mutually exclusive and must not be asserted simultanously.

mapping is descibed in table 3.

Offset	**Name**	**Description**
`0x00-0x07`	`KEY`	Initial user key that will be used for encryption and decryption. The most significant word is written to offset `0x00`. This memory section is *write-only* to the Avalon® interface.
`0x08-0x0B`	`DATA`	Input data, can be either interpreted as cyphertext for decryption or plain text for encryption. The most significant word shall be written to offset `0x08`. This memory section is *write-only* to the Avalon® interface.
`0x10-0x13`	`RESULT`	Result of the operation. The most significant word of the result at offset `0x10`. This memory section is *read-only* to the Avalon® Interface.
`0x14-0x1E`	—	reserved
`0x1F`	`CTRL`	Control and status word of the component can be read and written. Detailed description see 3.1

Table 3: Memory map of the AES core Avalon® slave

3.1 Control Register

The AES Core offers the register `CTRL` to control the function of the core and poll its status. The control register can be accessed in read and write mode. When wrriting to the register reserved Bits shall be assigned a value of `0`. Individual Bits have following functionality decribed in table 3.1.
In case of a Avalon® Bus reset this register is set to `0x00000000` thus invalidating all previously written keys and resetting the AES core.

Offset	Name	Description
31-8	—	reserved
7	KEY_VALID	If asserted key data in the KEY memory range is regarded valid and will be expanded to roundkeys. When deasserted all keys are invalidated and the current operation of the core is aborted. It must be asserted as long as the key shall be used for either encryption or decryption. This bit must be cleared for one clock cylce to load a new key.
6	IRQ_ENA	Enable use of the interrupt request signal. If asserted the component will set IRQ after completing an operation. If not set the component operates in polling mode only.
5-2	—	reserved
1	DEC [1]	If asserted memory content of the DATA range is regarded to be valid and will be *decrypted*. This Bit shall only be deasserted externally if a running AES operation is aborted by deasserting KEY_VALID. 1 It will be set 0 by the core to signal completion of the operation.
0	ENC [1]	If asserted memory content of the DATA range is regarded to be valid and will be *encrypted*. This Bit shall only be deasserted externally if a running AES operation is aborted by deasserting KEY_VALID. It will be set 0 by the core to signal completion of the operation.

Table 4: Bits in the control register

4 Protocol Sequence

The AES component appears as memory mapped peripheral. All writes are fundamental slave write transfers, see [4] and take one clock cycle of the Avalon® bus clock clk. It is not necessary to write all words of a input parameter successively or in one transfer. Bursts are not supported.

Before any AES operation can be started the initial userkey has to be written to KEY segment of the memory map.After the user key is transferred to the component the KEY_VALID Bit must be set to start the key expansion. This Bit can be set simultanously with DEC or ENC Bit of the control register. To invalidate the previous key and use another key the KEY_VALID must be deasserted for at least one Avalon® bus clock cycle During this cycle the new key can already be transferred.

Once a key is passed and marked valid data blocks can be transferred to the DATA segment of the memory map. The AES operation is started by asserting the ENC Bit for encryption or DEC Bit for decryption. While asserting ENC or DEC the KEY_VALID Bit must be kept asserted.
The ENC or DEC Bit respectively is deasserted by the component after completing the requested operation. The result of the operation can be read from the RESULT area of the memory and is not cleared. It will be overwritten by succeeding operations.

The underlying AES core uses the Finite State Machine (FSM) shown in 1 for processing of the data. The signals data_stable and key_stable are accessible over the control status word CTRL 3.1. key_ready is a signal driven by the keygenerator when all keys are expanded. The signal

[1] ENC and DEC are mutually exclusive and must not be asserted simultanously.

round_index is the counter for the rounds and the address to select a roundkey.
NO_ROUNDS is the total number of rounds the processing takes, a constant defined by the generic KEYLENGTH 2.1. The AES standard in[1] defines 10 rounds for 128 Bit key, 12 rounds for a 192 Bit key and 14 rounds for a 265 Bit key.
Thus depending on the keylength the processing of a datablock needs at maximum 15 clockcycles from data_stable=1 to completion, if the key is already expanded.

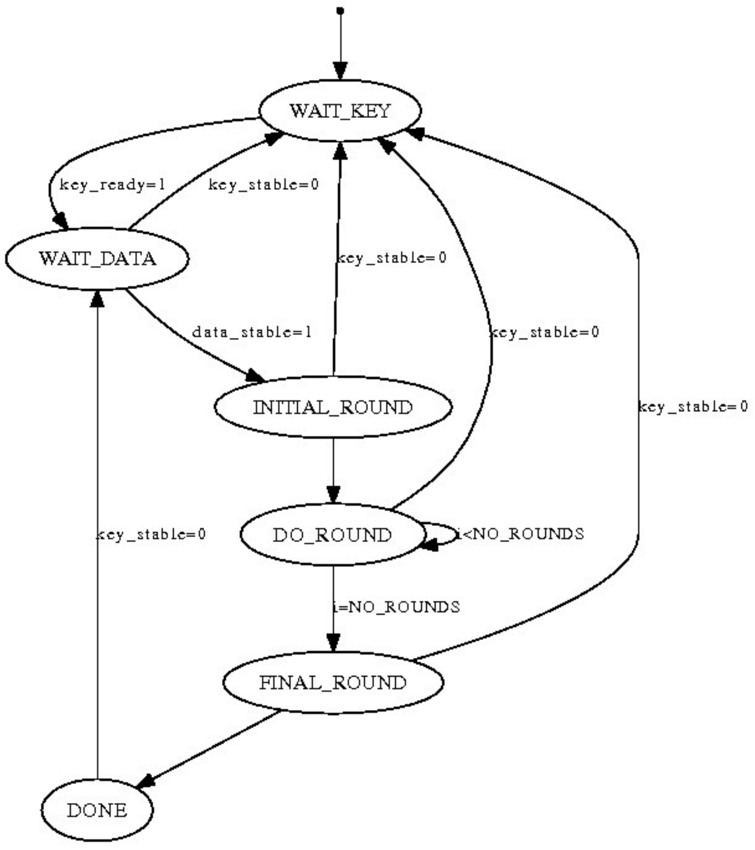

Figure 1: Finite State Machine of encryption and decryption process

4.1 Interrupt Behaviour

By setting IRQ_ENA in the control register 3.1 the component is configured to issue interrupt requests. If IRQ_ENA is asserted the interrupt request IRQ 2.2 will be set when the computation has completed in addition to clearing the ENC or DEC Bit. The IRQ 2.2 signal will remain set until clearing IRQ_ENA or a read operation on the RESULT area of the components address range.

5 Ressource Usage and Throughput

The Avalon® interface communicates a 32-Bit DWORD per clock cycle. Therefore a key is transmitted in 4 to 8 cyles plus one cyle to activate keyexpansion with the control word 3.1. A payload datablock or the result consist always of 4 DWORDs, thus it takes 4 cyles to send data to the core, one cycle to activate the computation with the control register 3.1 and 4 cycles to retrieve the data.

The keyexpansion component computes one column of a roundkey each clock cylce. AES takes, depending on the keylength, 10, 12 or 14 roundkeys with each 4 columns, see [1]. The keyexpansion therefore takes 40, 48 or 56 cycles until the encryption or decryption can start. The roundkeys are stored until invalidated, see 4 thus this step is is only needed once after power-up until the key changes.

The AES-core computes one iteration (round) of the Rijndael-Algorithm each clock cycle, thus a 128 Bit datablock is encrypted or decrypted in 10, 12 or 14 cylces plus an initial round.

The maximum throughput $T_{max}[Bits]$ depends on the maximum operation frequency f_{max} and the keylength which influences the number of rounds $N_{rnd}\epsilon\{10, 12, 14\}$.

$$T_{max} = \frac{(1 + N_{rnd}) \cdot 128Bit}{f_{max}} \tag{1}$$

Note: Equation 1 assumes that the roundkeys are already generated and does not include the constant of 4+1+4 Avalon® bus cylces for transmission of data, activation and result retrieval.

5.1 Exemplary FPGA implementations

The component has only be implemented and tested on an Altera® CycloneII EP2C35 FPGA. For this setup a Makefile is provided in `./sys/AlteraQuartus9.1`. All other values in the table are only results of synthesis[0] and are not verified on actual hardware.

The design is kept mostly vendor independent in generic VHDL. For Altera® chips the AES SubByte component is specially designed using M4K Blockrams as dual-port ROM. For non-Altera® FPGAs a second VHDL architecture exists also trying to make use of ROM functions of the target chips however the success varies on RTL compiler capabilities.

[0]Synthesized with Altera® QuartusII® Web edition Version 9.1 or Xilinx® ISE 9.1 Webpack

Configuration	Target FPGA[1]	LE / Slices	HW RAM	f_{max}[Mhz]
256 Bit Key, encrypt + decrypt	Xilinx® Spartan3A XC3S1400A-5FG484	- / 1609	18 RAMB16BWE	91
	Xilinx® Virtex5 XC5VLX30-3FF324	- / 297	18 18k-Blocks 4 36k-Blocks	224
	Altera® CylconeII EP2C35F484C8	1937 / -	39912 Bits in 22 M4K-Blocks	65
	Altera® StratixII EP2S30F484C5	585 / -	39912 Bits in 22 M4K-Blocks	103
128 Bit Key, encrypt + decrypt	Xilinx® Spartan3A XC3S1400A-5FG484	- / 1523	18 RAMB16BWE	91
	Altera® CylconeII EP2C35F484C8	1776 / -	39912 Bits in 22 M4K-Blocks	65
256 Bit Key, encrypt	Xilinx® Spartan3A XC3S1400A-5FG484	- / 680	14 RAMB16BWE	159
	Xilinx® Virtex5 XC5VLX30-3FF324	- / 297	10 18k-Blocks 4 36k-Blocks	268
	Altera® CylconeII EP2C35F484C8	969 / -	22528 Bits in 14 M4K	97
	Altera® StratixII EP2S30F484C5	524 / -	22528 Bits in 14 M4K	145
128 Bit Key, encrypt	Xilinx® Spartan3A XC3S1400A-5FG484	- / 594	14 RAMB16BWE	159
	Altera® CylconeII EP2C35F484C8	797 / -	22528 Bits in 14 M4K	95

Table 5: ressource usage on different targets and configuration

All of the above configurations in table 5.1 use hardware key expansion. Downloading of software generated roundkeys is not yet supported. The decryption and encryption datapaths share a common keyexpansion block, mulitplexing the address signals is one of the main reasons for regression of the maximum frequency f_{max} of the configuration compared to encryption only versions.

6 Simulation and Software Driver

6.1 Testbench

In `./bench/VHDL/` a “selfchecking testbench” is provided which runs tests for a default `TESTKEYSIZE` is 256 Bit . For different key lengths the constant `TESTKEYSIZE` has to be changed appropriately. Expected results for all testcases and key lengths are included. The expected results were generated by AES Calculator applet, written by Lawrie Brown from ADFA, Canberra Australia [7]. The testbench consists of a sequence of 5 test cases:

[1]This table is not meant to be a benchmark between FPGAs of different vendors, it is only a rough estimation for the user of the core. The FPGA families cannot be compared easily, see also [5] and [6]for further details.

1. load key1, load data1, encrypt : (basic encryption test)
2. key1, data1, decrypt: (basic decryption test)
3. key1, data1, encrypt: (test if internal state was changed)
4. key1, data2, encrypt: (encryption test with new data)
5. key2, data2, encrypt: (encryption test with new key)

6.1.1 Simulation

The component library is “`avs_aes_lib`”. All files are expected to be compiled into this library as all files depend at least on the package `avs_aes_lib.avs_aes_pkg`.
A Makefile for Mentor Graphics® Modelsim® is given in `./sim/`. The default make target `simaes` will create the library “`avs_aes_lib`” and a “`work`” library, compile all files and run a testbench.

6.2 Software Driver

This AES Core Avalon® slave was also tested on a NiosII® processor. To use it in software a simple driver is provided in `./sw/` among with an example program of the basic usage. Find more detailed description in the doxygen documentation in `./doc/sw/html`
The driver consits of the two files `avs_aes.c` and `avs_aes.h`.

6.2.1 Configuration

To be adapted to different address mappings and key sizes two macros are use in `avs_aes.h`:

define	default	Description
KEYWORDS	8	Key size in 32 Bit words
AES_BASEADDR	0x40000	Base address at which the AES Core is mapped to the Avalon® switchfabric

Table 6: user changeable macros in header

6.2.2 Compilation

Yes indeed very difficult to compile two files and one header..... try this one:

```
GCCNIOS -I. -o test aes_tester.c avs_aes.c
```

Of course you need a cross compiler, or just use the Nios2 IDE.

7 The Inner Core

The algorithmic core is devided into two seperate datapaths one for encryption and a second for decryption operation. The two datapaths are independent, however they share the keyexpansion component which provides decrypt and encrypt keys (which are the same only in opposite order).

Each datapath is controlled by its own FSMİf configured by the generic `DECRYPTION` 2.1 the decryption datapath is included and some multiplexers are generated for the shared signals, e.g. `result` or `roundkey_index`.
For reference the encryption data path of `aes_core.vhd` is given in figure 2. The decryption datapath is left for the reader or any other author of this document.

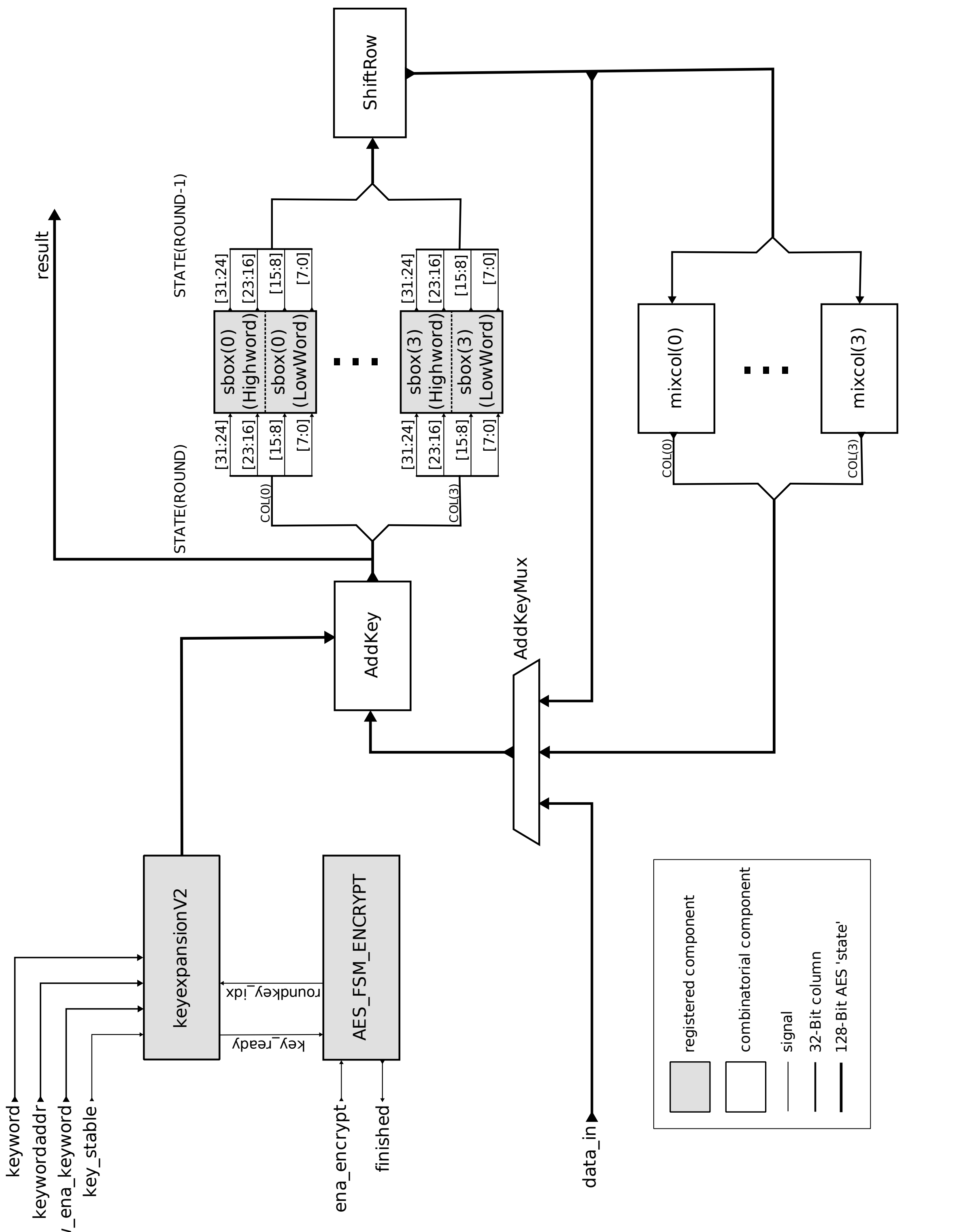

Figure 2: Encrypt datapath of the AES core as implemented in aes_core.vhd

8 License and Liability

The “AES 128/192/256 (ECB) Avalon®-MM Slave” component, all its subcomponents and documentation (like this document you are reading) are published under following license:

Note: The term “SOFTWARE” in the above licence applies in this case not only to software as executable code but also to documentation, hardware description or compiled netlists for actual target hardware. As Chips generally don’t just reproduce “the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution” the datasheet of the product must also contain it.

List of Acronyms

Advanced Encryption Standard (AES)
NIST approved symmetric block cypher

Electronic Codebook (ECB)
application of a cypher algorithm without further processing of the blocks

Finite State Machine (FSM)
Behavioural Model with finite number of states and transitions

Least Significant Bit (LSB)
least value bit in a vector

Most Significant Bit (MSB)
highest value bit in a vector

National Institute of Standards and Technology (NIST)
US standardisation office

System on Chip (SoC)
System of seperate functional interacting together implemented on a single chip

Glossary

Bit
Binary Digit, atomary information unit

Byte
String of Bits - nowadays mostly a string of 8 Bits, also called oktett

Master
Entity initiating and controlling communication.

memory mapped
Method of addressing peripheral components like Avalon Slaves via the same address bus as main memory

Slave
Entity responding to communication requests by a Master.

switch fabric
Interconnect between IP-Cores providing arbiration and glue logic. Altera® Avalon® term

References

[1] "Fips-197 announcing the advanced encryption standard (aes)," National Institute of Standards and Technology (NIST), 100 Bureau Drive, Stop 1070, Gaithersburg, MD, US, Nov. 2001. [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[2] "Wishbone - computer bus," wikipedia.org. [Online]. Available: http://en.wikipedia.org/wiki/Wishbone_computer_bus)#Comparisons

[3] R. Herveille, "Wishbone soc architecture specification, revision b.3," Opencores Organization, Sept. 2002. [Online]. Available: http://www.opencores.org/downloads/wbspec_b3.pdf

[4] "Avalon interface specification," Altera Corporation, 101 Innovation Drive, San Jose, CA, US, 2005. [Online]. Available: http://www.altera.com/literature/manual/mnl_avalon_spec.pdf

[5] A. Percey, "Advantages of the virtex-5 fpga 6-input lut architecture," Xilinx Inc., 2100 Logic Drive, San Jose CA USA, Dec. 2007. [Online]. Available: http://www.xilinx.com/support/documentation/white_papers/wp284.pdf

[6] "Stratix iii fpgas vs. xilinx virtex-5 devices: Architecture and performance comparison," Altera Corporation, 101 Innovation Drive, San Jose CA USA, Oct. 2007. [Online]. Available: http://www.altera.com/literature/wp/wp-01007.pdf

[7] L. Brown, "Aes calculator," ADFA, Canberra, Australia, 2005. [Online]. Available: http://www.unsw.adfa.edu.au/~lpb/src/AEScalc/index.html

Change History

Rev.	Chapter	Description	Date	Reviewer
0.1	all	initial document	2009/02/01	T. Ruschival
0.2	all	added interrupt	2009/03/25	T. Ruschival
0.3	all	added generics	2009/04/20	T. Ruschival
0.4	all	cleanup for opencores.org	2009/05/20	T. Ruschival
0.5	all	final release	2010/03/07	T. Ruschival
0.6	3,6	fixed memory map, added test-bench description	2010/04/02	T. Ruschival