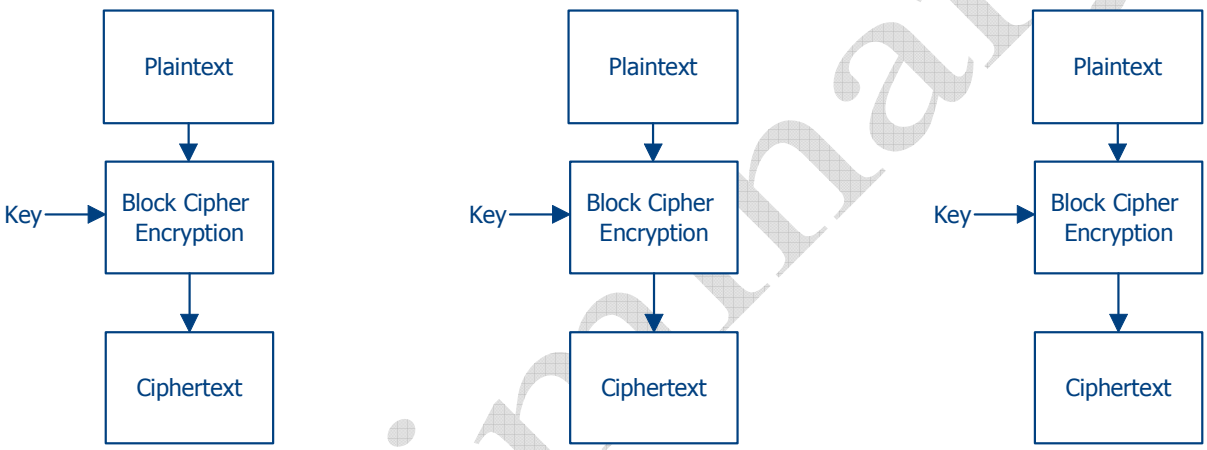
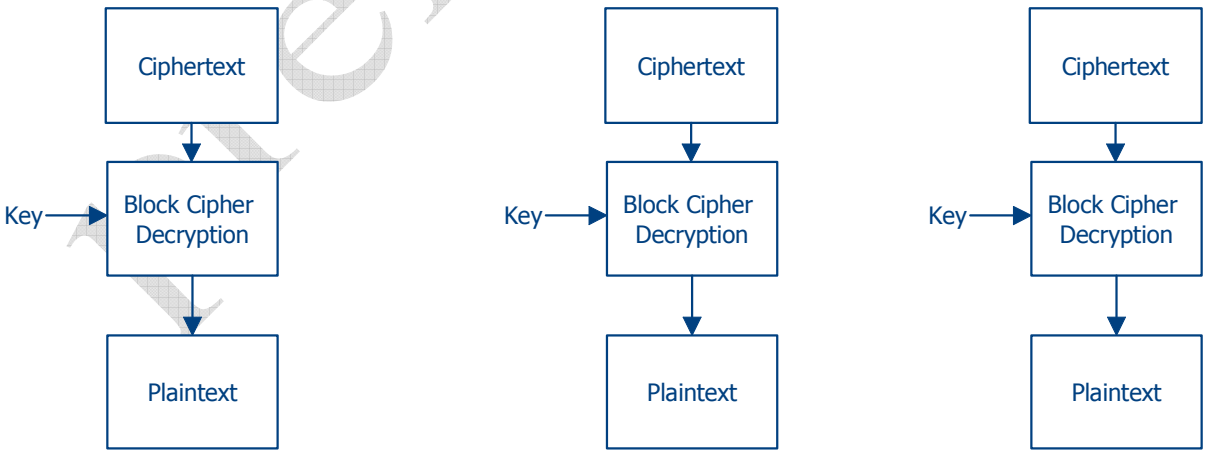
AixSolve *AES Megafunction*

The AixSolve Advanced Encryption Standard (AES) Rijndael Core is a fully NIST Advanced Encryption Standard implementation. The core supports 128, 192 and 256 block- and key width either in electronic codebook or cipher-block mode.

Electronic Codebook (ECB) mode encryption:



Electronic Codebook (ECB) mode decryption:

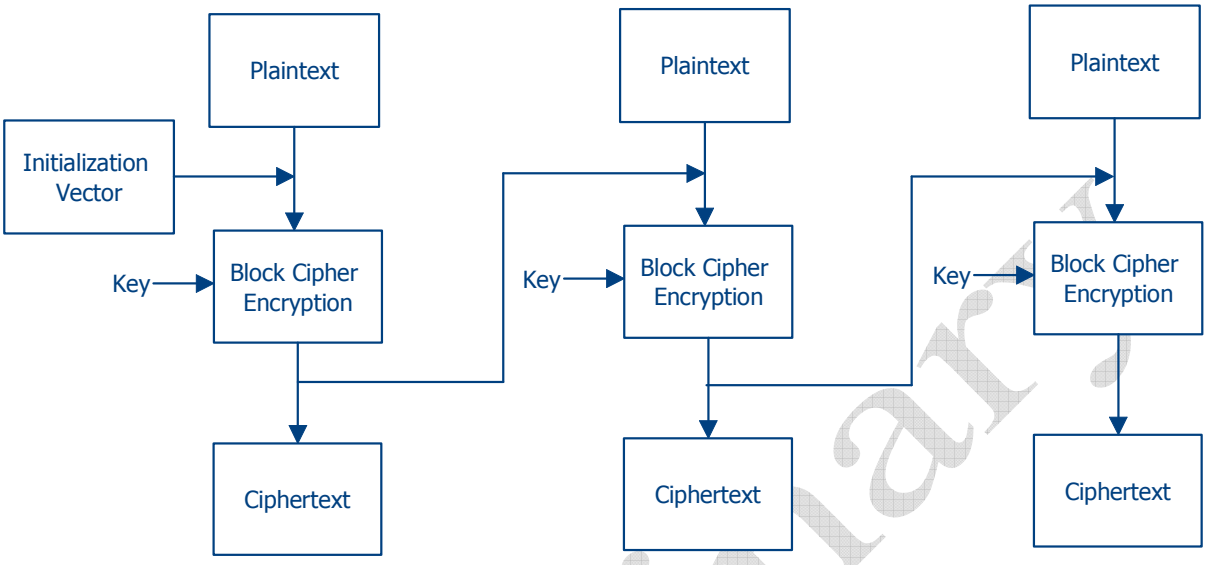


AixSolve GmbH
Kaiserstrasse 100
52134 Herzogenrath

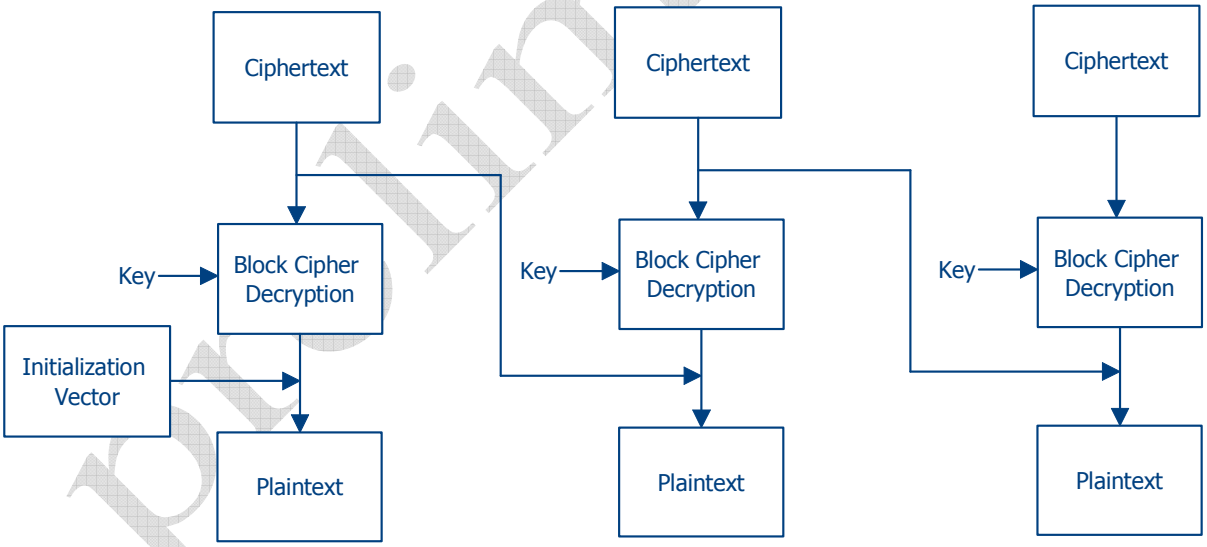
ver.10/01/2012 14:58:00

Tel.: +49 2407 5739 0
Fax.: +49 2407 5739 11
Mail: sales_aixsolve@aixsolve.de

Cipher-block chaining (CBC) mode encryption:



Cipher-block chaining (CBC) mode decryption:



AixSolve GmbH
Kaiserstrasse 100
52134 Herzogenrath

Tel.: +49 2407 5739 0
Fax.: +49 2407 5739 11
Mail: sales_aixsolve@aixsolve.de

Features:	• Encipher and/or decipher • Variable configurable block width: 128 (AES), 192 and 256 • Variable configurable key width (128, 192, 256) • Software key expansion (not part of the AES core)
Modes:	• Electronic codebook (ECB) • Cipher-block chaining (CBC)
Signals:	• clk (in); clock • rst (in); reset • data[BLOCK_WIDTH] (in); input data or round key • nr[4](in); number of rounds fix, depends on key width • opcode[2] (in); load-key/cipher/decipher • start (in) • busy (out) • q[BLOCK_WIDTH] (out); output data

Resources & Parameters:

testet on Altera Cyclone II PCI Testkit

Block-Size=128(AES)	**AES-128**	**AES-256**	**Block-Size=256 KEY=256**
t-Cipher [clks]	11+2	15+2	15+2
Logic Cells	8246	8245	16436
Dedic Logic Registers	393	265	777
Memory Bits	2048	2048	4096
M4Ks	8	8	15
LUT only LCs	7774	7884	15499
Register-Only LCs	129	128	260
LUT/Register LCs	343	233	727

AixSolve GmbH
Kaiserstrasse 100
52134 Herzogenrath

Tel.: +49 2407 5739 0
Fax.: +49 2407 5739 11
Mail: sales_aixsolve@aixsolve.de

Entity declaration:

```
entity aes_core is
  generic (
    CIPHER_CAPS : std_logic_vector(1 downto 0) := "11";
                  -- 1x = decipher, x1 = cipher
    BLOCK_WIDTH : natural := 128
  );
  port (
    clk       : in std_logic;
    rst       : in std_logic;

    nr        : in std_logic_vector(3 downto 0);

    opcode    : in std_logic_vector(1 downto 0);
    data      : in std_logic_vector(BLOCK_WIDTH-1 downto 0);
    start     : in std_logic;

    q         : out std_logic_vector(BLOCK_WIDTH-1 downto 0);
    busy      : out std_logic
  );
end aes_core;
```