AES Control State Machine

The AES encryption process is controlled by a finite state machine (FSM) that sequences the execution of all cryptographic transformations and manages key selection, round progression, and synchronization between modules. The FSM follows the standard AES round structure while enabling an efficient iterative hardware implementation.

Specifically, the control unit coordinates the execution of the initial round, the intermediate rounds, and the final round, enabling the correct application of the AES transformations at each stage. This structured control is essential for maintaining correctness while achieving efficient and high-performance operation on FPGA platforms.

Algorithm Reference

This implementation follows the AES (Rijndael) cipher, the algorithm standardized as the Advanced Encryption Standard (AES).

The AES algorithm is a symmetric block cipher that operates on 128-bit data blocks and applies a series of transformations organized into multiple rounds. It consists of:

- key expansion
- byte substitution (SubBytes)
- row shifting (ShiftRows)
- column mixing (MixColumns)
- key addition (AddRoundKey)

all coordinated through a control unit across multiple encryption rounds.

State Descriptions

KeySchedule (KS)

This state initiates the key expansion process. The round keys required for encryption are generated, and the FSM remains in this state until the key schedule process is completed. Once the signal Finish_Key is asserted, the FSM transitions to the initial AddRoundKey state.

AddRound0

This state performs the initial AddRoundKey operation, where the plaintext is XORed with the initial round key. This step marks the beginning of the AES encryption rounds.

SubBytes_loop

In this state, the SubBytes transformation is applied using the AES S-box. This operation introduces non-linearity and is executed iteratively during the intermediate rounds.

ShiftRows_loop

The ShiftRows transformation is performed, cyclically shifting the rows of the state matrix to enhance diffusion across the data block.

MixColumns_loop

This state applies the MixColumns transformation, performing matrix multiplication over GF(2^8) to further increase diffusion. This operation is executed only during the intermediate rounds.

AddRoundKey_loop

The AddRoundKey operation is applied for the current round using the corresponding round key. The round counter is updated in this state. If the round count is less than the maximum number of rounds, the FSM transitions back to SubBytes_loop; otherwise, it proceeds to the final round states.

SubBytes_final

This state applies the SubBytes transformation for the final round. Unlike intermediate rounds, no MixColumns operation follows.

ShiftRows_final

The ShiftRows transformation is executed for the final round, completing the diffusion stage before the final key addition.

AddRoundKey_final

The final AddRoundKey operation is performed, producing the ciphertext output. Once completed, the FSM transitions to the Finish state.

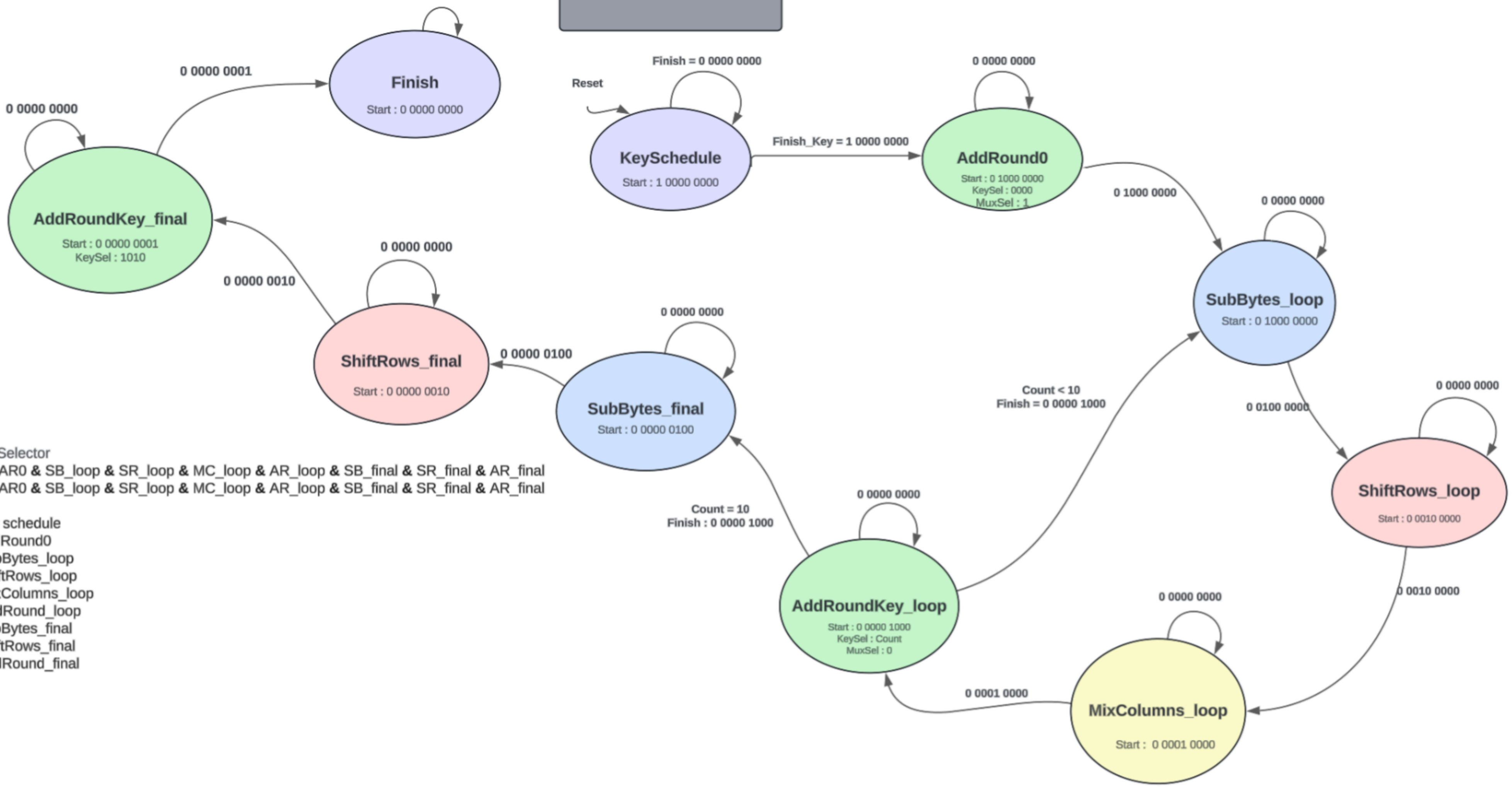
Finish

This terminal state indicates that the encryption process has completed successfully. The output ciphertext is stable, and the system waits for a new start or reset signal.

State Machine Diagram

The following figure presents the finite state machine (FSM) used to control the AES implementation. It illustrates the main states of the algorithm and highlights the functional modules associated with each stage of the encryption process.

State Machine



KeySel = Key Selector
Start = KS **&** AR0 **&** SB_loop **&** SR_loop **&** MC_loop **&** AR_loop **&** SB_final **&** SR_final **&** AR_final
Finish = KS **&** AR0 **&** SB_loop **&** SR_loop **&** MC_loop **&** AR_loop **&** SB_final **&** SR_final **&** AR_final

KS : Key schedule
AR0 : AddRound0
SB_loop : SubBytes_loop
SR_loop : ShiftRows_loop
MC_loop : MixColumns_loop
AR_loop : AddRound_loop
SB_final : SubBytes_final
SR_final : ShiftRows_final
AR_final : AddRound_final