AES-128 Hardware Architecture: RTL Implementation on FPGA

Hardware Implementation
Register-Transfer Level Design

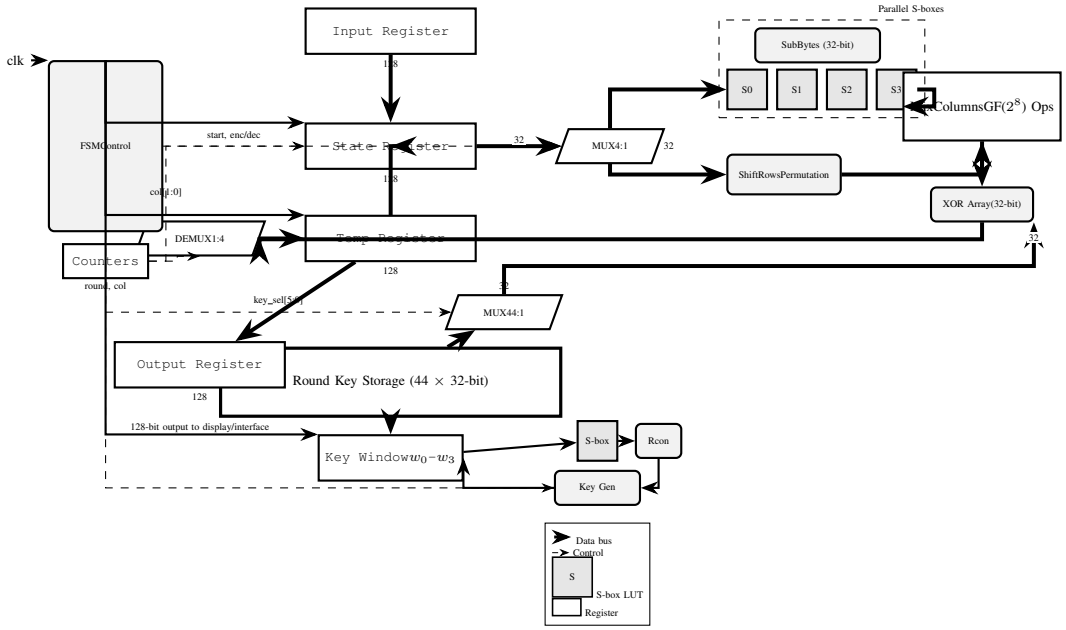

Fig. 1: Complete AES datapath architecture showing register-transfer level implementation. The design uses column-wise processing with a 32-bit datapath, four parallel S-boxes, and explicit register staging. Control signals manage column/round iteration and key selection.

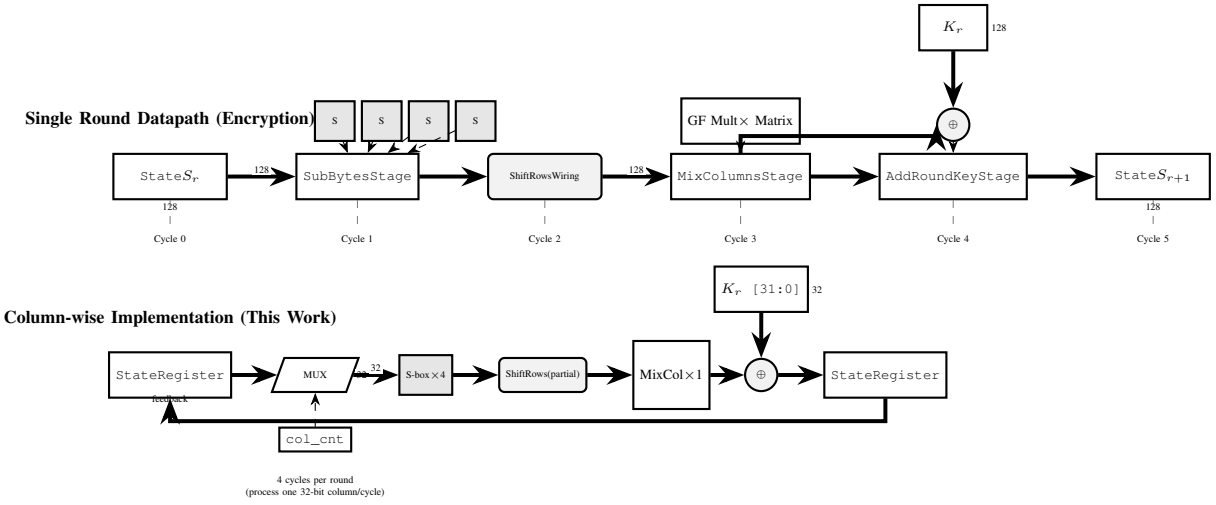

Fig. 2: Pipeline structure comparison: (top) Traditional round pipeline processes entire 128-bit state per cycle with high resource usage. (bottom) Column-wise implementation processes 32-bit columns sequentially, reducing hardware by 75% with 4× cycle count.

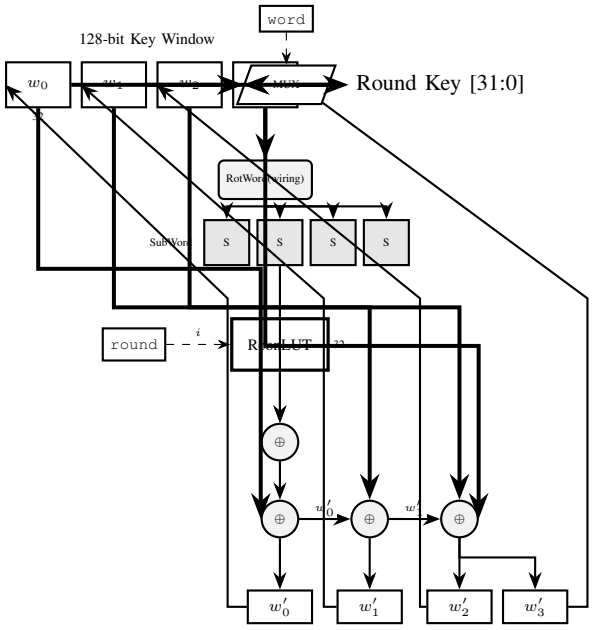

Fig. 3: On-the-fly key expansion hardware. Only four 32-bit word registers (w_0–w_3) are stored, updated each round using SubWord, RotWord, and Rcon operations. This achieves 90% memory reduction versus storing all 44 words.

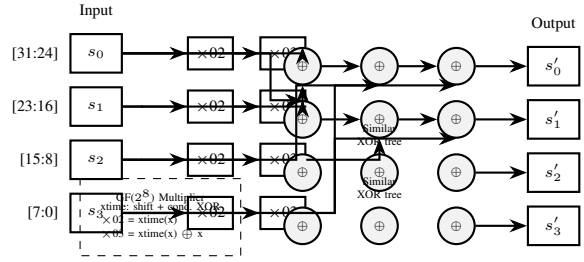

Fig. 5: MixColumns hardware implementing matrix multiplication in GF(2^8). Each output byte uses dedicated GF multipliers ($\times 02$, $\times 03$) and XOR trees. The $\times 02$ operation (xtime) is a shift with conditional XOR by 0x1B.

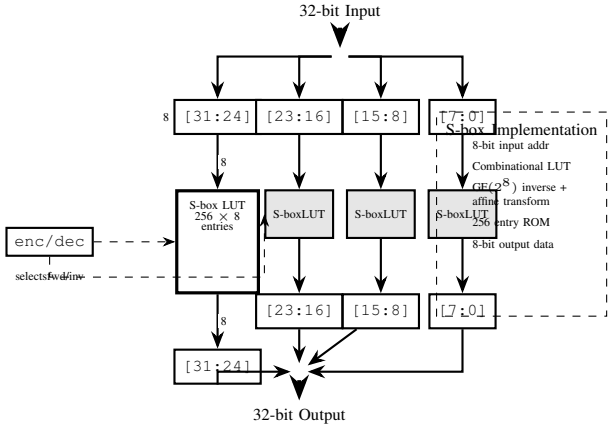

Fig. 4: SubBytes hardware with four parallel S-box lookup tables. Each S-box is a 256×8-bit combinational ROM implementing the AES substitution. The enc/dec control selects forward or inverse S-box.

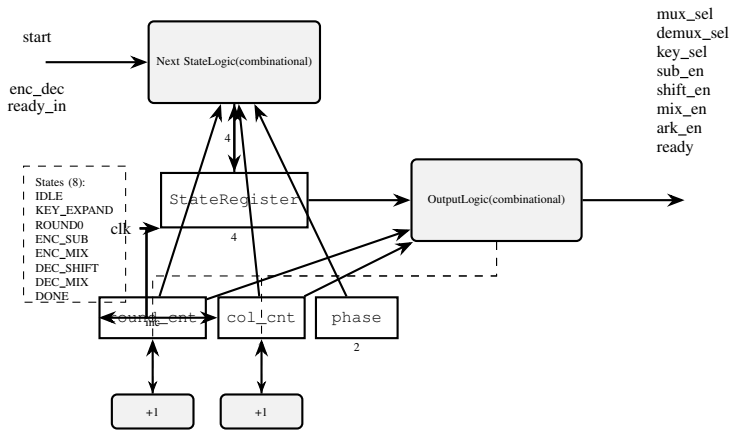

Fig. 6: Control FSM hardware showing state register, next-state logic, output logic, and round/column counters. All control signals are generated combinationally from current state and counter values.