586:AL
265:AS
key_init = start;
587:BL
588:IF
!(!rst_n)
(!rst_n)
591:IF
!((state == IDLE) || (state == GEN_KEY3) && last_round))
((state == IDLE) || (state == GEN_KEY3) && last_round)
589:NS
rd_count <= INITIAL_ROUND;
594:IF
rd_count_en
592:NS
rd_count <= INITIAL_ROUND;
595:NS
rd_count <= rd_count + 1'b1;
Leaf_586:AL
598:AS
round = rd_count;
599:AS
first_round = rd_count == INITIAL_ROUND;
680:AS
last_round = (rd_count == NUMBER_ROUND_I) || (rd_count == NUMBER_ROUND_INC);
262:AS
mode_ctr = aes_mode == CTR;
267:AS
op_mode = (mode_cur)? ENCRYPTION : operation_mode;
260:AS
op_key_derivation = op_mode == KEY_DERIVATION;
270:AL
271:BL
next_state = state;
273:CS
state
274:CA
275:BL
276:IF
(!start)
!(!(!start))
279:CS
op_mode
277:BS
next_state = IDLE;
284:CA
284:BS
next_state = IDLE;
282:CA
282:BS
next_state = GEN_KEY0;
281:CA
281:BS
next_state = ROUND0_COL3;
280:CA
280:BS
next_state = ROUND0_COL0;
283:CA
283:BS
next_state = GEN_KEY0;
337:CA
338:BL
next_state = GEN_KEY1;
346:BL
next_state = GEN_KEY3;
345:CA
322:CA
341:CA
342:BL
next_state = GEN_KEY2;
323:BL
next_state = (enc_dec)? ROUND_COL2 : ROUND_COL0;
291:CA
303:CA
326:CA
304:BL
327:BL
next_state = (enc_dec)? ROUND_COL3 : ROUND_COL1;
292:BL
next_state = (enc_dec)? ROUND0_COL2 : ROUND0_COL0;
299:CA
318:CA
330:CA
331:BL
395:IF
!(!first_round))
(!first_round)
310:BL
next_state = (enc_dec)? ROUND_COL0 : ROUND_COL3;
306:BL
next_state = (last_round)? READY : NOP;
332:IF
!((last_round && enc_dec))
(last_round && enc_dec)
335:BS
next_state = (enc_dec)? ROUND_KEY0 : ROUND_COL2;
333:BS
next_state = READY;
300:BL
next_state = (enc_dec)? ROUND_KEY0 : ROUND0_COL2;
319:BL
next_state = (enc_dec)? ROUND_COL1 : ROUND_KEY0;
287:CA
288:BL
next_state = (enc_dec)? ROUND_COL1 : ROUND_KEY0;
356:CA
357:BL
next_state = IDLE;
295:CA
296:BL
next_state = (enc_dec)? ROUND_COL3 : ROUND0_COL1;
314:CA
311:BL
next_state = (enc_dec)? ROUND_COL0 : ROUND_COL3;
349:CA
350:BL
351:IF
last_round
(!last_round)
352:BS
next_state = (op_key_derivation)? READY : ROUND0_COL3;
354:BS
next_state = GEN_KEY0;
Leaf_270:AL
243:AL
244:BL
245:IF
!(!(!rst_n))
(!rst_n)
248:IF
!(disable_core)
disable_core
251:NS
state <= next_state;
249:NS
state <= IDLE;
296:NS
state <= IDLE;
Leaf_243:AL
258:AS
key_gen = state == ROUND_KEY0;
365:AS
end_comp = (state == READY)? ENABLE : DISABLE;
254:AS
encrypt_decrypt = (op_mode == ENCRYPTION) || (state == GEN_KEY0) || (state == GEN_KEY1) || (state == GEN_KEY2) || (state == GEN_KEY3);
257:AS
enc_dec = encrypt_decrypt | mode_ctr;
263:AS
mode_cbc = aes_mode == CBC;
392:AL
393:BL
sbox_sel = COL_0;
rk_sel = COL;
bypass_rk = DISABLE;
key_out_sel = KEY_0;
col_sel = INPUT;
key_sel = KEY_HOST;
key_en = KEY_DIS;
col_en = DISABLE;
rd_count_en = DISABLE;
iv_cnt_en = DISABLE;
iv_cnt_sel = IV_RUS;
bypass_key_en = DISABLE;
key_derivation_en = DISABLE;
408:CS
state
570:CA
571:BL
key_en = EN_KEY_3;
key_sel = KEY_OUT;
bypass_key_en = ENABLE;
419:CA
420:BL
sbox_sel = COL_1;
rk_sel = COL;
bypass_rk = ENABLE;
bypass_key_en = ENABLE;
key_out_sel = KEY_1;
col_sel = ADD_RK_OUT;
col_en = EN_COL_1;
428:IF
(!enc_dec)
429:BL
key_sel = KEY_OUT;
key_en = EN_KEY_1;
500:IF
enc_dec
(!enc_dec)
501:BS
key_en = EN_KEY_2;
502:BS
key_en = EN_KEY_1;
507:BS
col_sel = ADD_RK_OUT;
504:IF
!(!(mode_cbc && last_round && !enc_dec || mode_ctr && last_round))
(mode_cbc && last_round && !enc_dec || mode_ctr && last_round)
505:BS
col_sel = INPUT;
495:BL
sbox_sel = COL_1;
rk_sel = (last_round)? MIXCOL_IN : MIXCOL_OUT;
key_out_sel = KEY_1;
key_sel = KEY_OUT;
col_en = EN_COL_1;
494:CA
577:BL
579:IF
(op_mode == KEY_DERIVATION)
580:BS
key_derivation_en = ENABLE;
576:CA
472:BL
sbox_sel = COL_0;
rk_sel = (last_round)? MIXCOL_IN : MIXCOL_OUT;
key_out_sel = KEY_0;
key_sel = KEY_OUT;
498:IF
!(enc_dec)
enc_dec
490:BS
col_en = EN_COL_0;
485:IF
!(!(mode_cbc && last_round && !enc_dec || mode_ctr && last_round))
(mode_cbc && last_round && !enc_dec || mode_ctr && last_round)
483:BL
479:BS
key_en = EN_KEY_1;
484:IF
!(enc_dec)
486:BS
col_sel = (last_round)? ADD_RK_OUT : SHIFT_ROWS;
487:BS
col_sel = ADD_RK_OUT;
481:BS
col_sel = INPUT;
471:CA
553:CA
554:BL
sbox_sel = C_FUNCTION;
rd_count_en = ENABLE;
559:BL
key_en = EN_KEY_1 | EN_KEY_0;
key_sel = KEY_OUT;
bypass_key_en = ENABLE;
558:CA
564:CA
464:CA
465:BL
sbox_sel = C_FUNCTION;
key_sel = KEY_OUT;
key_en = EN_KEY_2;
rd_count_en = ENABLE;
565:BL
key_en = EN_KEY_2;
key_sel = KEY_OUT;
bypass_key_en = ENABLE;
516:IF
enc_dec
!(enc_dec)
517:BS
key_en = EN_KEY_3;
519:BS
key_en = EN_KEY_2;
521:BS
col_sel = INPUT;
523:BS
col_sel = ADD_RK_OUT;
510:CA
511:BL
sbox_sel = COL_2;
rk_sel = (last_round)? MIXCOL_IN : MIXCOL_OUT;
key_out_sel = KEY_2;
key_sel = KEY_OUT;
col_en = EN_COL_2;
520:IF
(mode_cbc && last_round && !enc_dec || mode_ctr && last_round)
!(!(mode_cbc && last_round && !enc_dec || mode_ctr && last_round))
409:CA
434:CA
435:BL
sbox_sel = COL_2;
rk_sel = COL;
bypass_rk = ENABLE;
bypass_key_en = ENABLE;
key_out_sel = KEY_2;
col_sel = ADD_RK_OUT;
col_en = EN_COL_2;
443:IF
(!enc_dec)
444:BL
key_sel = KEY_OUT;
key_en = EN_KEY_2;
526:CA
410:BL
sbox_sel = COL_0;
rk_sel = COL;
bypass_rk = ENABLE;
bypass_key_en = ENABLE;
key_out_sel = KEY_0;
col_sel = (enc_dec)? ADD_RK_OUT : SHIFT_ROWS;
col_en = (enc_dec)? EN_COL_0 : COL_ALL;
544:BS
col_en = (last_round)? EN_COL_3 : COL_ALL;
546:BS
col_en = EN_COL_3;
449:CA
527:BL
sbox_sel = COL_3;
rk_sel = (last_round)? MIXCOL_IN : MIXCOL_OUT;
key_out_sel = KEY_3;
key_sel = KEY_OUT;
542:IF
544:IF
(!enc_dec)
!(!(mode_cbc && last_round && !enc_dec || mode_ctr && last_round))
547:IF
(!enc_dec)
(mode_ctr && last_round)
537:BL
533:BS
key_en = EN_KEY_3;
548:BL
iv_cnt_en = ENABLE;
iv_cnt_sel = IV_CNT;
538:IF
enc_dec
!(enc_dec)
539:BS
col_sel = (last_round)? ADD_RK_OUT : SHIFT_ROWS;
541:BS
col_sel = ADD_RK_OUT;
535:BS
col_sel = INPUT;
450:BL
sbox_sel = COL_3;
rk_sel = COL;
bypass_key_en = ENABLE;
key_out_sel = KEY_3;
col_sel = (enc_dec)? SHIFT_ROWS : ADD_RK_OUT;
col_en = (enc_dec)? COL_ALL : EN_COL_3;
bypass_rk = ENABLE;
458:IF
(!enc_dec)
(mode_cbc && last_round && !enc_dec || mode_ctr && last_round)
459:BS
key_sel = KEY_OUT;
key_en = EN_KEY_3;
Leaf_392:AL