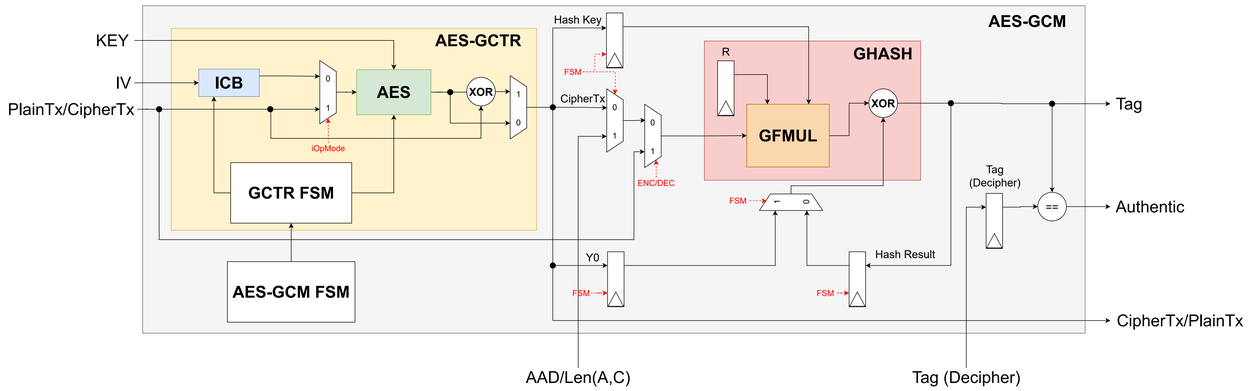
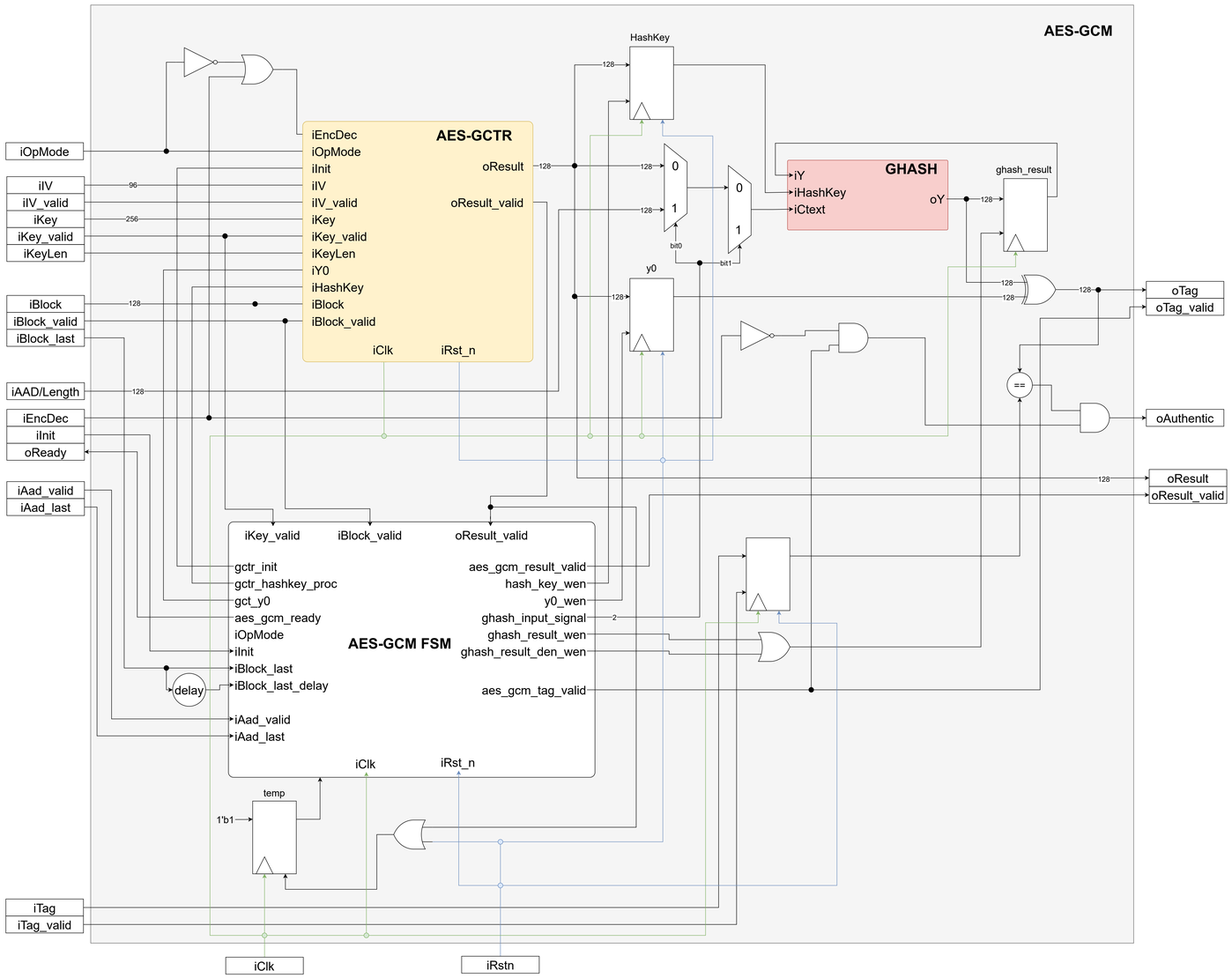




AES core

The AES core is reused and downloaded from the below URL.

URL: [tee-hardware/hardware/teehw/optvsrc/AES at master · uec-hanken/tee-hardware (github.com)]

Use guide waveforms are presented below.

AES-GCM FSM

1. IDLE:

- ----------ghash----------
- ghash_result_wen = 1'b0
- ghash_result_dec_wen = 1'b0
- temp_wen = 1'b0
- hash_key_wen = 1'b0
- ghash_input_signal[0] = 1'b0
- ghash_input_signal[1] = 1'b0
- ----------gctr----------
- gctr_init = 1'b0
- gctr_hashkey_proc = 1'b0
- gctr_y0 = 1'b0
- y0_wen = 1'b0
- ----------aes_gcm----------
- aes_gcm_ready = 1'b1
- aes_gcm_tag_valid = 1'b0
- aes_gcm_result_valid = 1'b0

2. HASHKEY

- ----------ghash----------
- (oResult_valid) ? hash_key_wen = 1'b1 : hash_key_wen = 1'b0
- ----------gctr----------
- (oResult_valid) ? gctr_init = 1'b0 : gctr_init = 1'b1
- gctr_hashkey_proc = 1'b1

3. AAD

- ----------ghash----------
- ghash_input_signal[0] = 1'b1
- (iAad_valid)? ghash_result_wen = 1'b1 : ghash_result_wen = 1'b0
- hash_key_wen = 1'b0
- ----------gctr----------
- gctr_init = 1'b0
- gctr_hashkey_proc = 1'b0
- ----------aes_gcm----------
- aes_gcm_ready = 1'b1

4. CIPHER

- ----------ghash----------
- ghash_input_signal[0] = 1'b0
- (iEncDec)? ghash_input_signal[1] = 1'b0 : ghash_input_signal[1] = 1'b1
- temp_wen =1'b1
- (iEncDec & oResult_valid) ghash_result_wen = 1'b1 : ghash_result_wen = 1'b0
- (~iEncDec) ghash_result_dec_wen = ~temp & ~ghash_result_wen : ghash_result_dec_wen = 1'b0
- ----------gctr----------
- (gctr_result_valid & iBlock_last_delay | ~iBlock_valid) gctr_init = 1'b0 : gctr_init = 1'b1
- ----------aes_gcm----------
- (gctr_result_valid) aes_gcm_ready = 1'b1 : aes_gcm_ready = 1'b0
- aes_gcm_result_valid = oResult_valid

5. TAG1

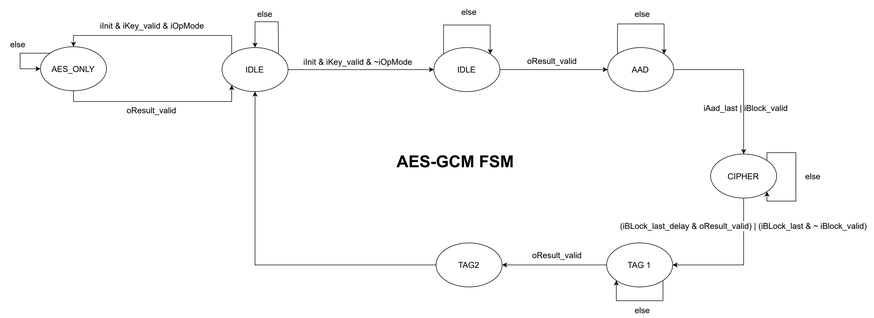
- ----------ghash----------
- ghash_input_signal[0] = 1'b0
- ghash_input_signal[1] = 1'b0
- ghash_result_wen = 1'b0
- ----------gctr----------
- (oResult_valid) ? gctr_init = 1'b0 : gctr_init = 1'b1
- gctr_y0 = 1'b1
- (oResult_valid) ? y0_wen = 1'b1 : y0_wen = 1'b0
- ----------aes_gcm----------
- aes_gcm_ready = 1'b0
- aes_gcm_result_valid = 1'b0

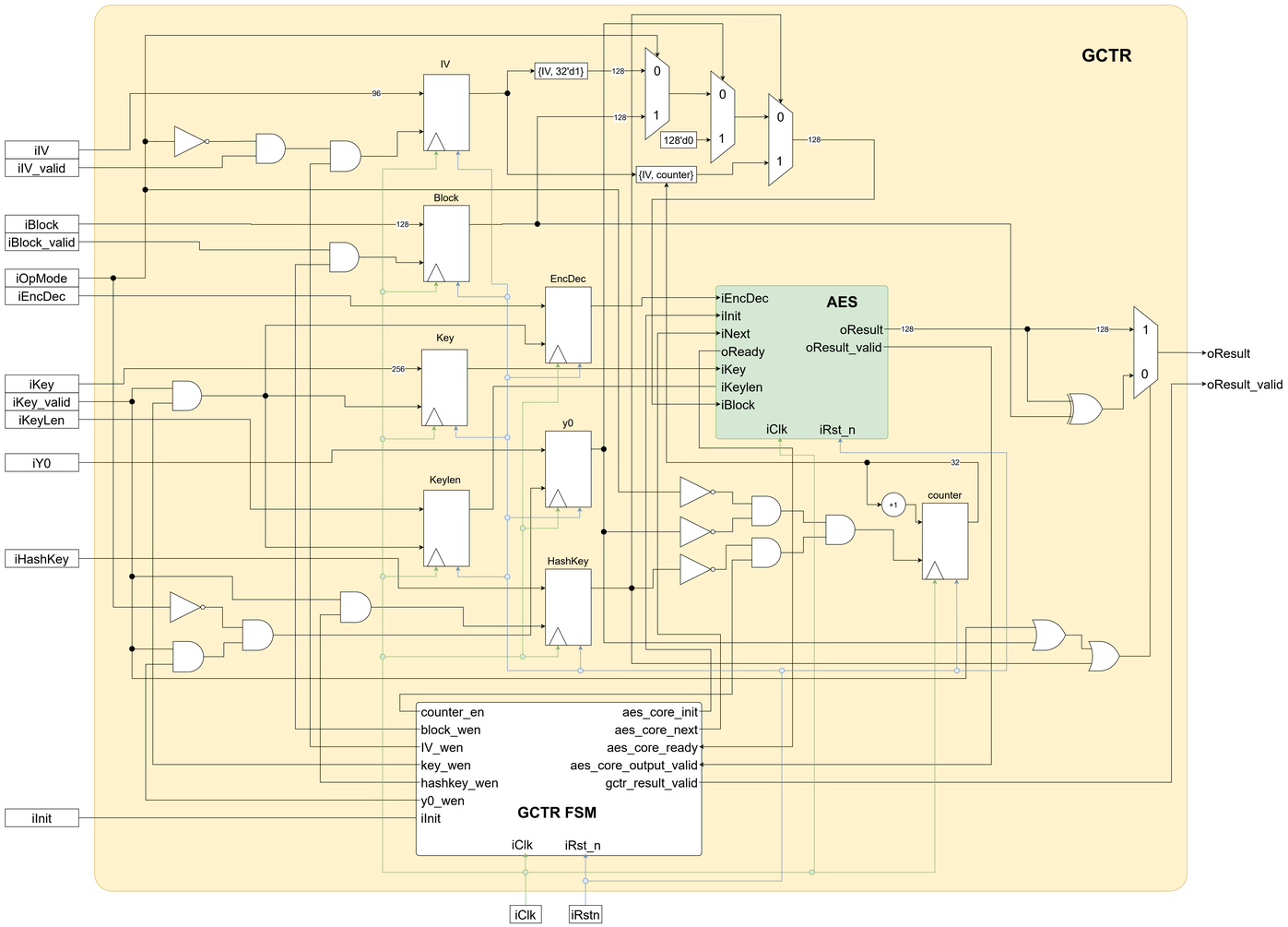
6. TAG2

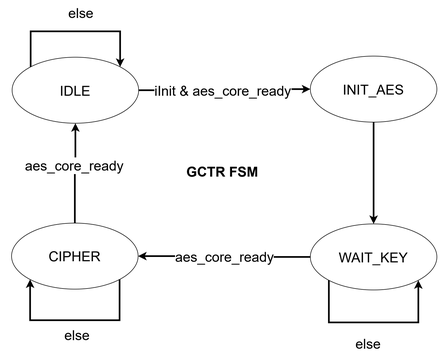
- ----------ghash----------
- ghash_input_signal[0] = 1'b1
- ghash_result_wen = 1'b1
- ----------gctr----------
- gctr_init = 1'b0
- gctr_y0 = 1'b0
- y0_wen = 1'b0
- ----------aes_gcm----------
- aes_gcm_tag_valid = 1'b1

7. AES_ONLY:

- ----------gctr----------
- gctr_init = 1'b1
- ----------aes_gcm----------
- (oResult_valid) aes_gcm_ready = 1'b1 : aes_gcm_ready = 1'b0
- (oResult_valid) aes_gcm_result_valid = 1'b1 : aes_gcm_result_valid = 1'b0







GCTR FSM

1. IDLE

- counter_wen = 1'b0
- block_wen = 1'b0
- IV_wen = 1'b0
- key_wen = 1'b0
- hashkey_wen = 1'b0
- aes_core_init = 1'b0
- aes_core_next = 1'b0
- gctr_result_valid = aes_core_output_valid

2. INIT_AES

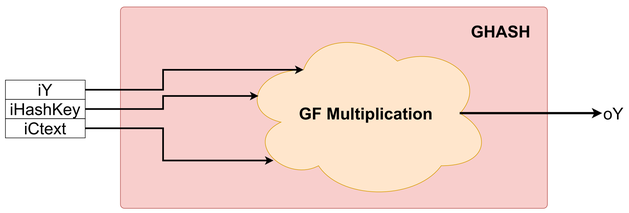
- block_wen = 1'b1
- IV_wen = 1'b1
- key_wen = 1'b1
- hashkey_wen = 1'b1
- aes_core_init = 1'b1
- gctr_result_valid = 1'b0

3. WAIT_KEY

- block_wen = 1'b0
- IV_wen = 1'b0
- key_wen = 1'b0
- hashkey_wen = 1'b0
- aes_core_init = 1'b0

4. CIPHER

- counter_en = 1'b1
- aes_core_next = 1'b1



Multiplication in GF(2^{128})

Each element is a vector of 128 bits. The i^{th} bit of an element X is denoted as X_i. The leftmost bit is X_i, and the rightmost bit is X_{127}. The multiplication operation uses the special element R = 11100001||0, and is defined in Algorithm 1. The argument ***righshift***() moves the bits of its argument one bit to the right. More formally, whenever W = ***rightshift***(V), then $W_i = V_{i-1}$ for 1 <= i <= 127 and $W_0 = 0$.

What we want to compute:

- oY = gf_mul(iHashKey, iCtext) xor iY

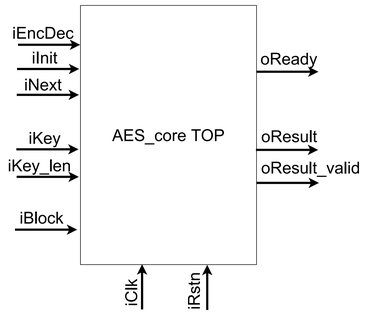
Algorithm 1 Multiplication in $GF(2^{128})$. Computes the value of $Z = X \cdot Y$, where X, Y and $Z \in GF(2^{128})$.

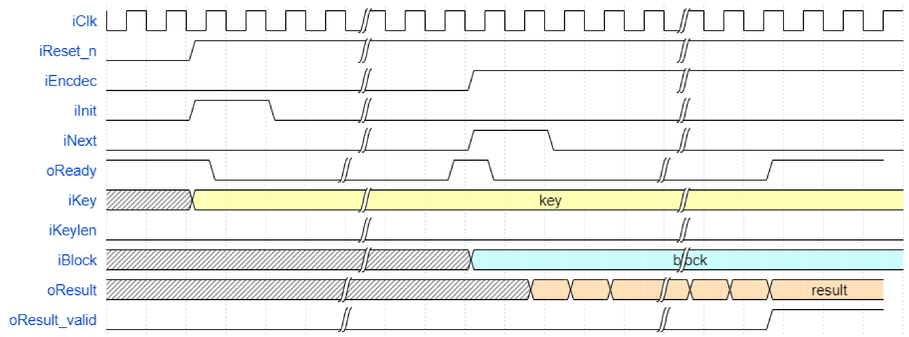
$Z \leftarrow 0, V \leftarrow X$
for $i = 0$ to 127 **do**
  if $Y_i = 1$ **then**
    $Z \leftarrow Z \oplus V$
  end if
  if $V_{127} = 0$ **then**
    $V \leftarrow \text{rightshift}(V)$
  else
    $V \leftarrow \text{rightshift}(V) \oplus R$
  end if
end for
return Z

AES core

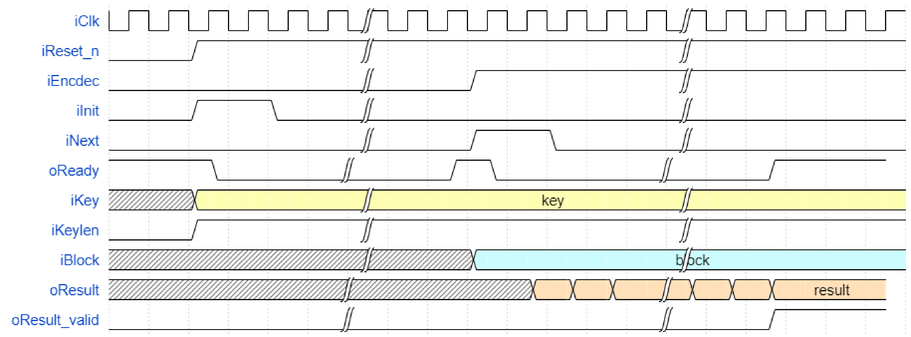
The AES core is reused and downloaded from the below URL.

URL: [tee-hardware/hardware/teehw/optvsrc/AES at master · uec-hanken/tee-hardware (github.com)]

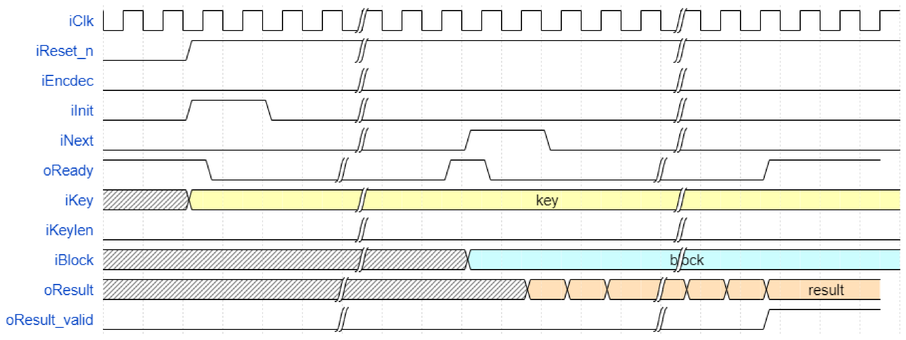




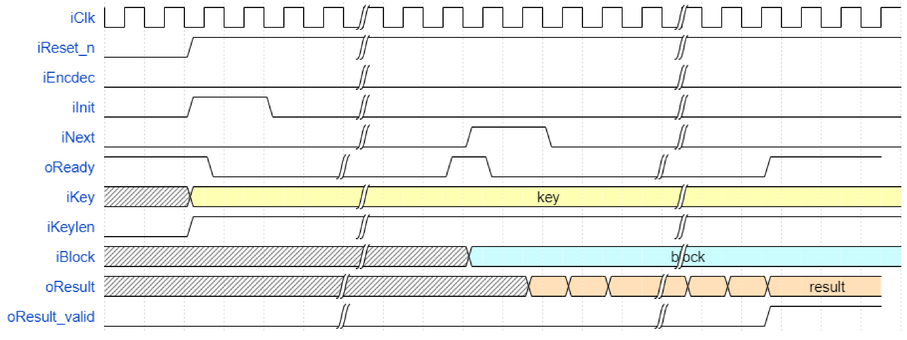
AES Encryption 128b Key



AES Encryption 256b Key



AES Decryption 128b Key



AES Decryption 256b Key