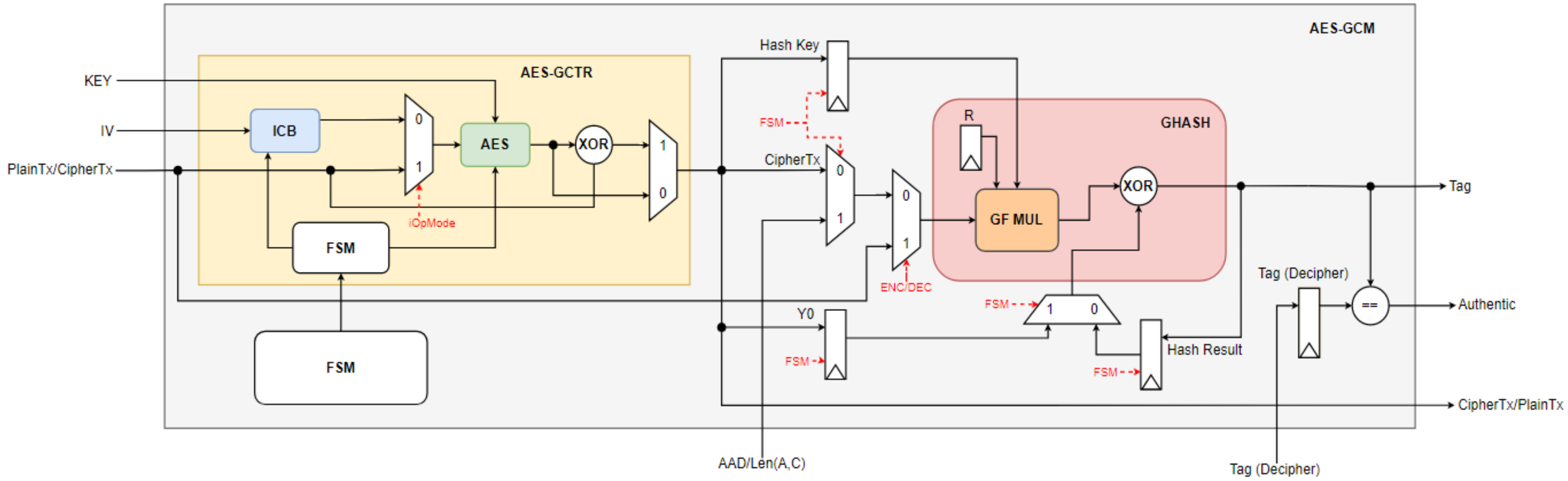
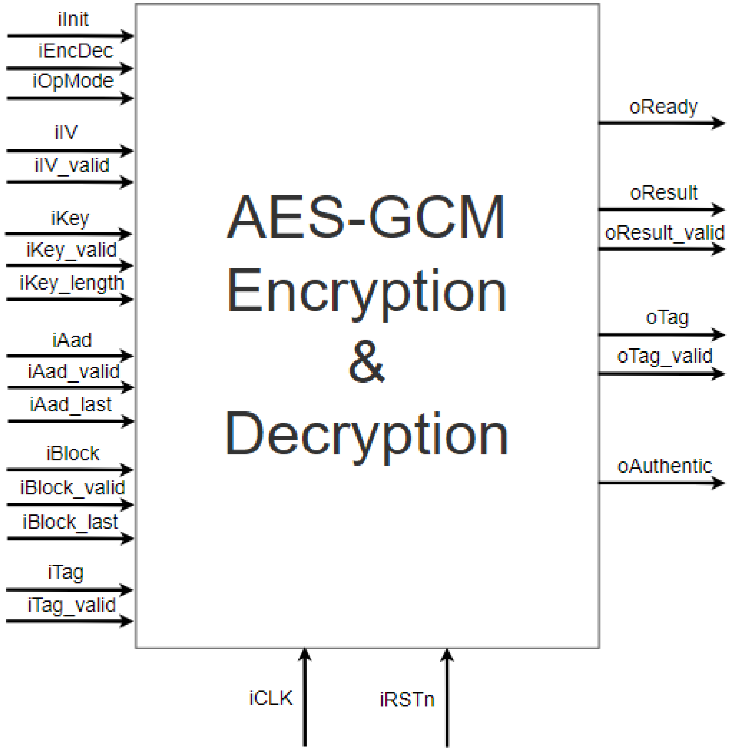
AES GCM

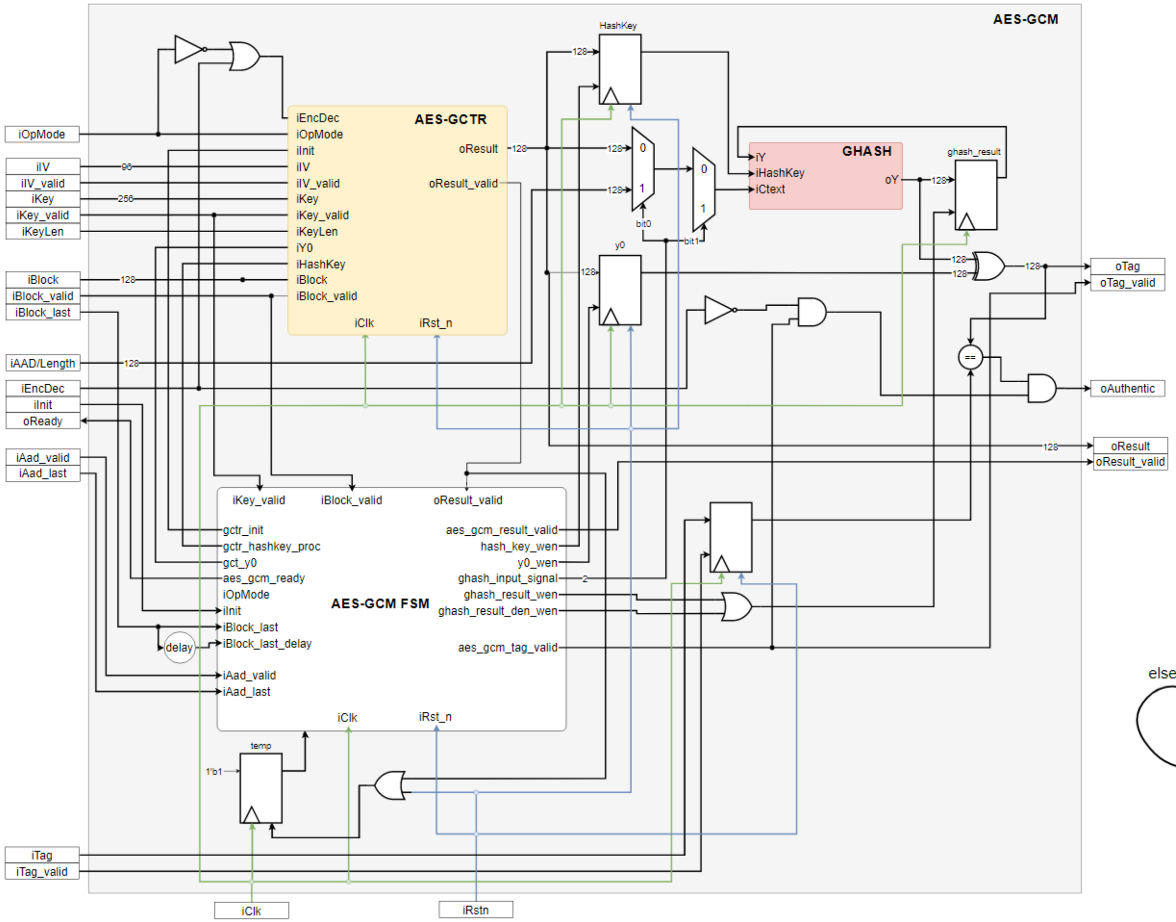
BLOCK DIAGRAM

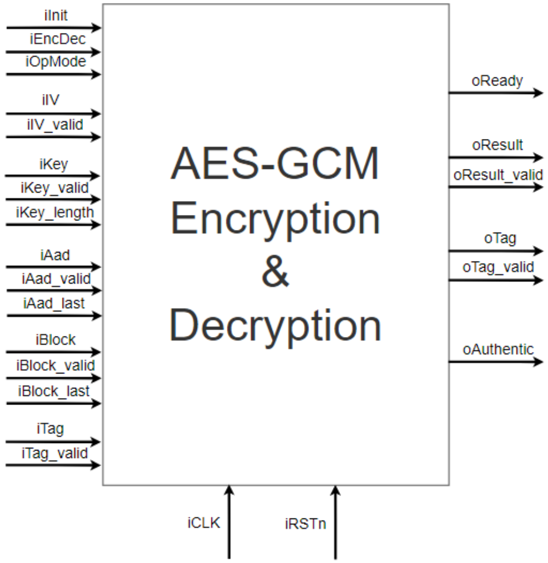
and STATE MACHINE

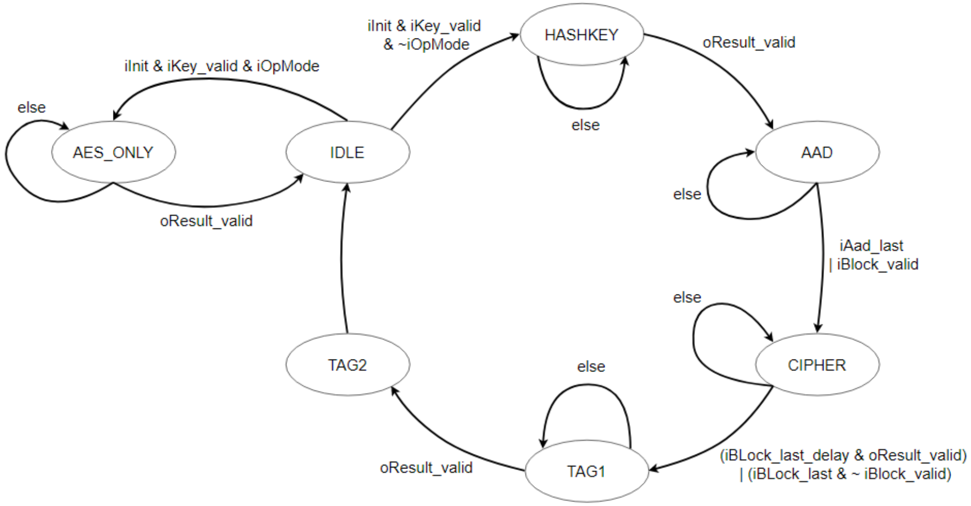

AES-GCM
AES-GCTR
KEY
IV
PlainTx/CipherTx
ICB
0
1
AES
XOR
1
0
iOpMode
FSM
FSM
Hash Key
FSM
CipherTx
0
1
0
1
ENC/DEC
GHASH
R
GF MUL
XOR
FSM
1
0
Y0
FSM
FSM
Hash Result
Tag
Tag (Decipher)
==
Authentic
CipherTx/PlainTx
AAD/Len(A,C)
Tag (Decipher)


iInit
iEncDec
iOpMode
iIV
iIV_valid
iKey
iKey_valid
iKey_length
iAad
iAad_valid
iAad_last
iBlock
iBlock_valid
iBlock_last
iTag
iTag_valid
AES-GCM
Encryption
&
Decryption
oReady
oResult
oResult_valid
oTag
oTag_valid
oAuthentic
iCLK
iRSTn

PIN	Direction	Width	Description
iClk	Input	1	Clock
iRstn	Input	1	Negative edge reset: reset after using the core
iInit	Input	1	Initialize signal: assert when using the core
iEncDec	Input	1	Assert for Encryption, deassert for Decryption
iOpMode	Input	1	Operation Mode: assert for AES-GCM, deassert for AES only
oReady	Output	1	Output ready signal: input new data when ready is asserted
iIV	Input	96	96-bit length IV
iIV_valid	Input	1	When asserted, IV is valid
iKey	Input	256	Key
iKey_valid	Input	1	When asserted, Key is valid
iKey_len	Input	1	When asserted, Key length is 256-bit. When deasserted, Key length is 128-bit.
iAad	Input	128	Additional Authentic Data or Length(A,C): when Ready = 1, change data every clock cycle
iAad_valid	Input	1	When asserted, AAD is valid.
iAad_last	Input	1	Asserted when input the last AAD data.
iBlock	Input	128	Input Plaintext or Ciphertext.
iBlock_valid	Input	1	When asserted, Plaintext or Ciphertext is valid.
iBlock_last	Input	1	Asserted when input the last Block data.
iTag	Input	128	Input Tag for authentication in Decryption mode
iTag_valid	Input	1	When asserted, Tag is valid
oResult	Output	128	Output Ciphertext or Ciphertext
oResult_valid	Output	1	When asserted, Result is valid
oTag	Output	128	Output Tag in Encryption mode.
oTag_valid	Output	1	When asserted, output Tag is valid
oAuthentic	Output	1	When asserted, indicating authentic Block in Decryption mode.






HASHKEY
AES_ONLY
IDLE
AAD
CIPHER
TAG1
TAG2
iInit & iKey_valid & ~iOpMode
iInit & iKey_valid & iOpMode
else
oResult_valid
else
else
oResult_valid
iAad_last
| iBlock_valid
else
else
(iBLock_last_delay & oResult_valid)
| (iBLock_last & ~ iBlock_valid)
oResult_valid

Submodule	File
AES_GCTR	aes_gcm.v
GHASH	ghash_block.v

Name	AES_GCM
File	aes_gcm.v

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

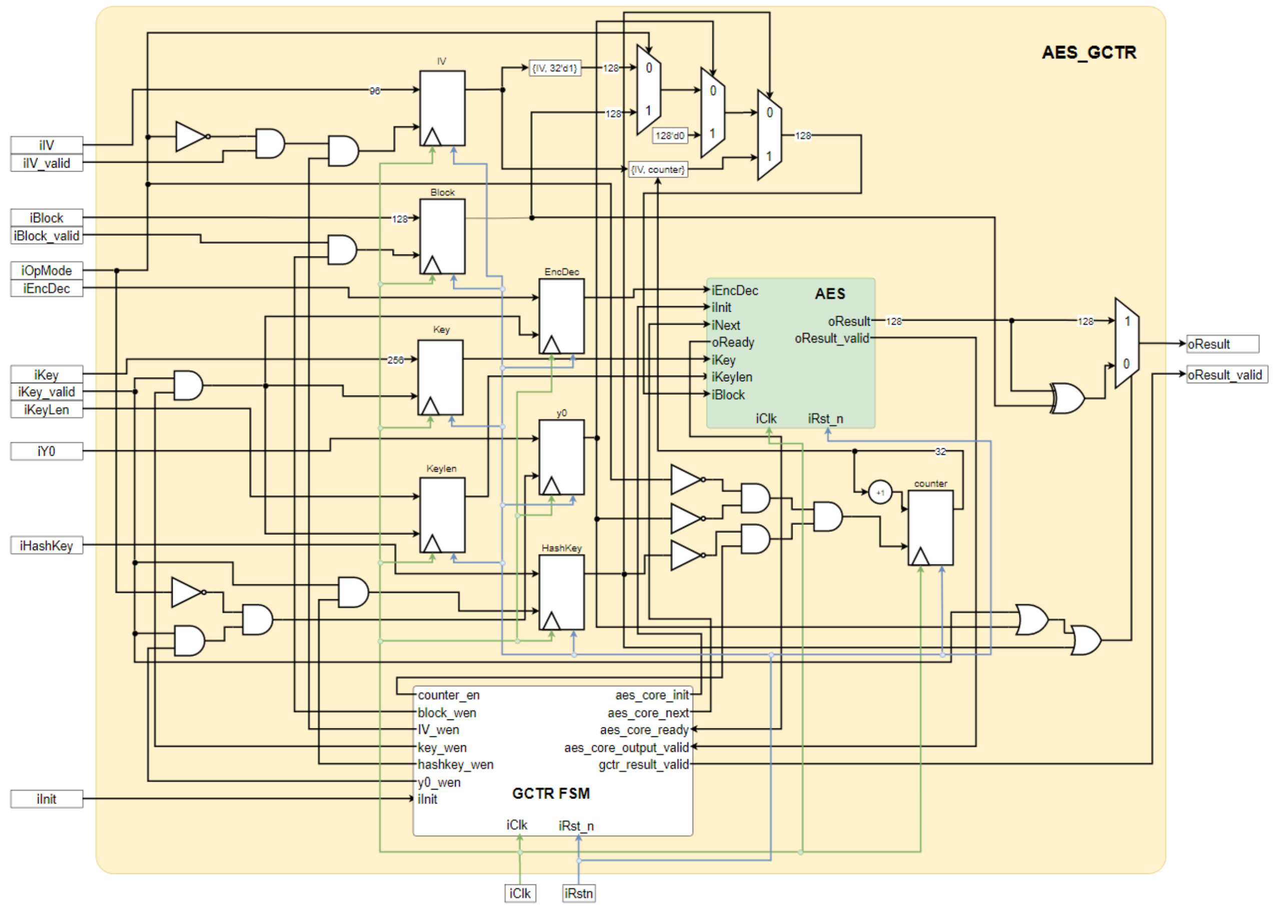
- -------------ghash-------------
- ghash_input_signal[0] = 1'b0
- ghash_input_signal[1] = 1'b0
- ghash_result_wen = 1'b0
- ---------------gctr--------------
- (oResult_valid) ? gctr_init = 1'b0 : gctr_init = 1'b1
- gctr_y0 = 1'b1
- (oResult_valid) ? y0_wen = 1'b1 : y0_wen = 1'b0
- ---------------aes_gcm-----------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

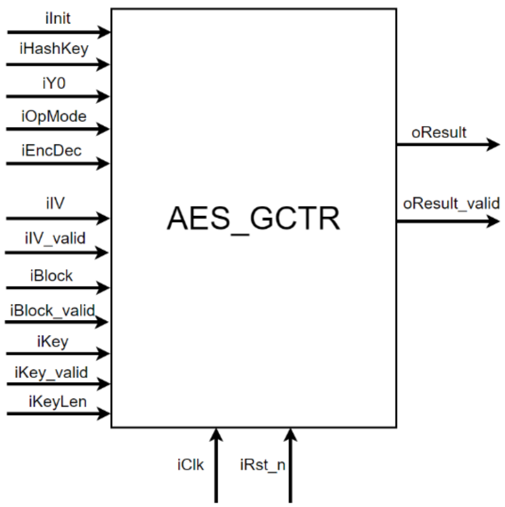
6. TAG2

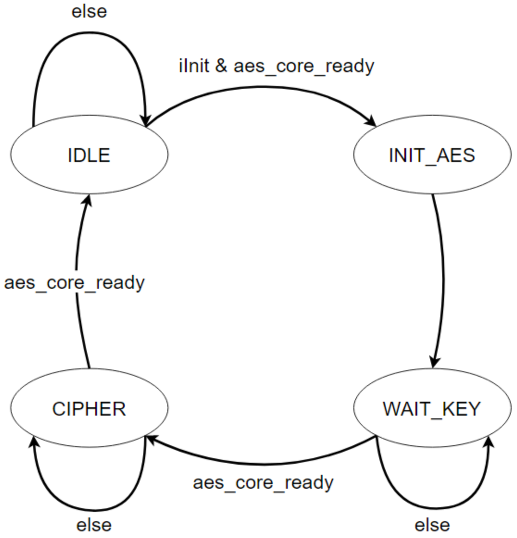
- -------------ghash-------------
- ghash_input_signal[0] = 1'b1
- ghash_result_wen = 1'b1
- ---------------gctr--------------
- gctr_init = 1'b0
- gctr_y0 = 1'b0
- y0_wen = 1'b0
- ---------------aes_gcm-----------
- aes_gcm_tag_valid = 1'b1

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

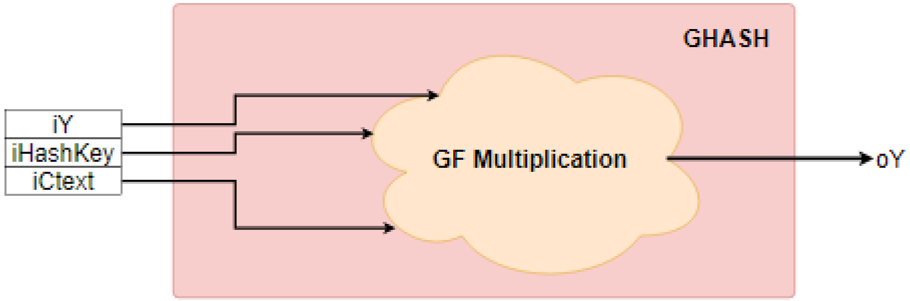
- block_wen = 1'b0
- IV_wen = 1'b0
- key_wen = 1'b0
- hashkey_wen = 1'b0
- aes_core_init = 1'b0

4. CIPHER

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

Submodule	File
AES_CORE	aes_core.v

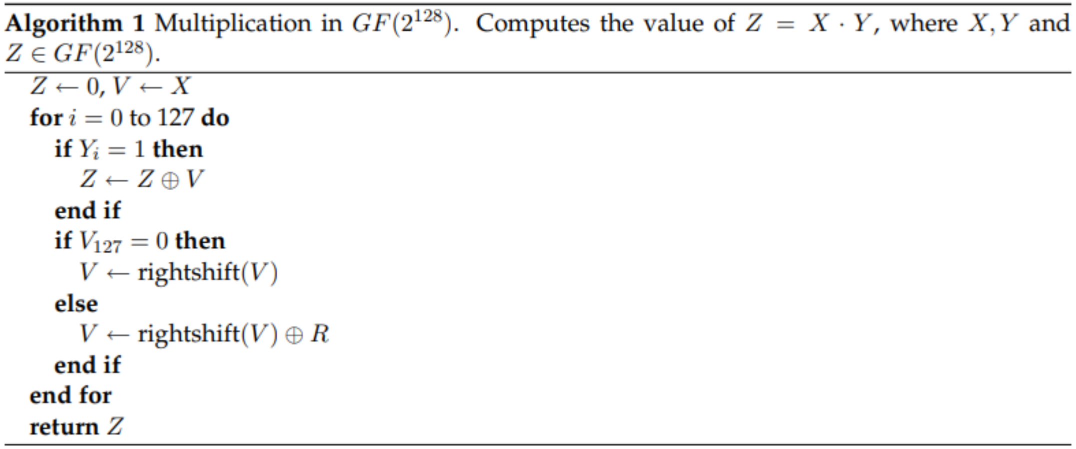
Name	AES_GCM
File	aes_gcm.v



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 = 1110001||0**, and is defined in Algorithm 1. The argument **rightshift()** 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 $W_0 = 0$.

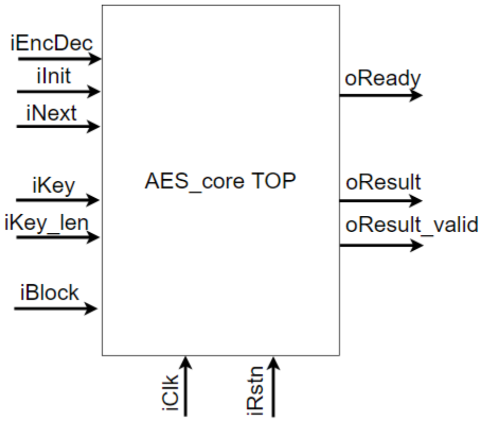
What we want to compute: oY = gf_mul(iHashKey, iCtext ^ iY)

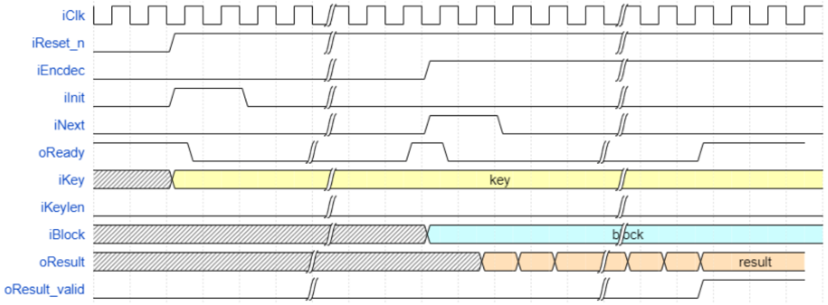


Algorithm 1 Multiplication in $GF(2^{128})$. Computes the value of $Z = X \cdot Y$, where X, Y and $Z \in GF(2^{128})$.

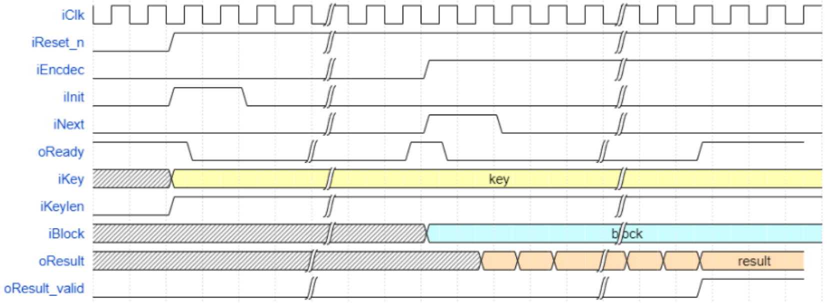
```
$Z \leftarrow 0, V \leftarrow X$
for $i = 0$ to 127 do
    if $Y_i = 1$ then
        $Z \leftarrow Z \oplus V$
    end if
    if $V_{127} = 0$ then
        $V \leftarrow \text{rightshift}(V)$
    else
        $V \leftarrow \text{rightshift}(V) \oplus R$
    end if
end for
return $Z$
```

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

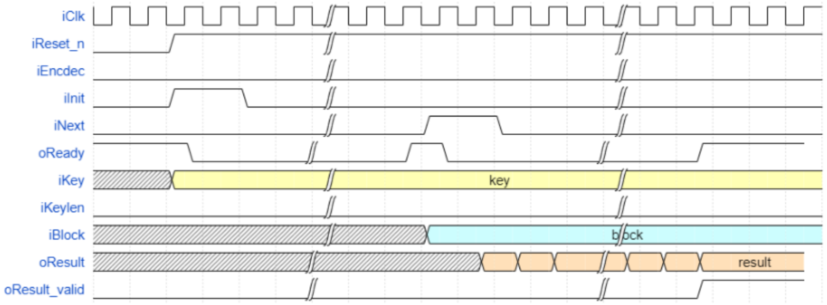




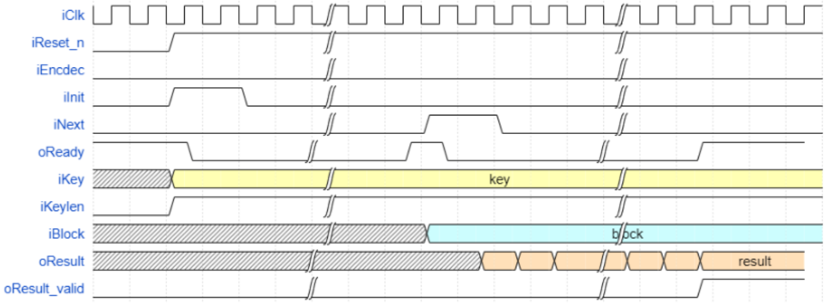
AES Encryption 128b Key



AES Encryption 256b Key



AES Decryption 128b Key



AES Decryption 256b Key