

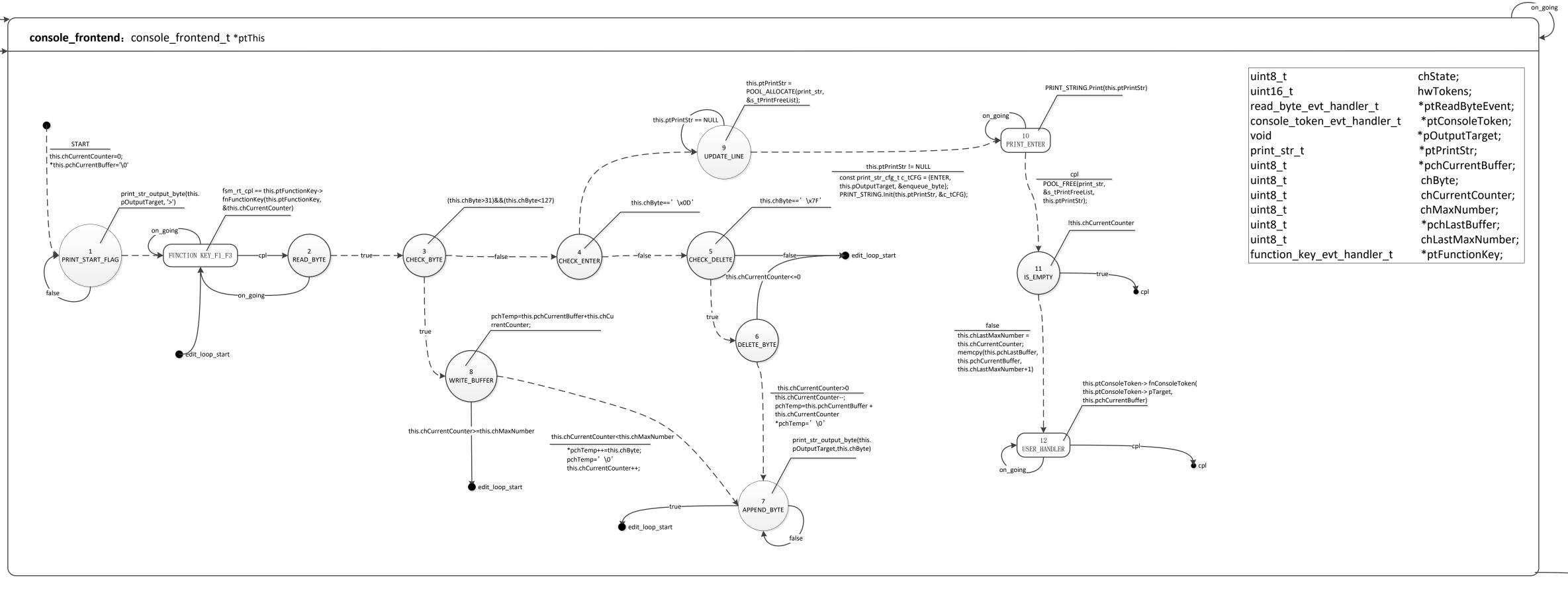
console_frontend

console_token

console_token: void *pThis, uint8_t *pchBuffer

FIND_TOKEN

find_token(pchBuffer, &this.hwTokens)==NULL



chState;

print_token_evt_handler_t *ptPrintToken;

*pTarget;

hwTokens;

uint8_t

uint16_t

this.ptPrintToken-> fnPrintToken(

this.ptPrintToken-> pTarget,

pchBuffer, this.<u>hwTokens)</u> print_token: void *pThis, uint8_t *pchTokens, uint16_t hwTokens

this.hwTokens>0;

START

this.pchTokensArry=pchTokens;

➤ CHECK_TOKEN

_NUMBER

this.hwTokens=hwTokens;

uint8_t

uint16_t

PRINT_TOKEN_ENTER

print_str_t

PRINT_STRING.Print(this.ptPrintStr)

print_str_output_byte(

*this.pchTokensArry++)

this.pTarget,

OUTPUT_BYTE

this.ptPrintStr != NULL const print_str_cfg_t c_tCFG = {ENTER, this.pOutputTarget, &enqueue_byte};
PRINT_STRING.Init(this.ptPrintStr, &c_tCFG);

*this.pchTokensArry=='\0'

this.ptPrintStr =

&s_tPrintFreeList);

this.hwTokens--;

this.ptPrintStr == NULL

END_STRING_ENTER

POOL_FREE(print_str, &s_tPrintFreeList, this.ptPrintStr);

POOL_ALLOCATE(print_str,

chState;

*pTarget;

hwTokens;

*ptPrintStr;

*pchTokensArray;