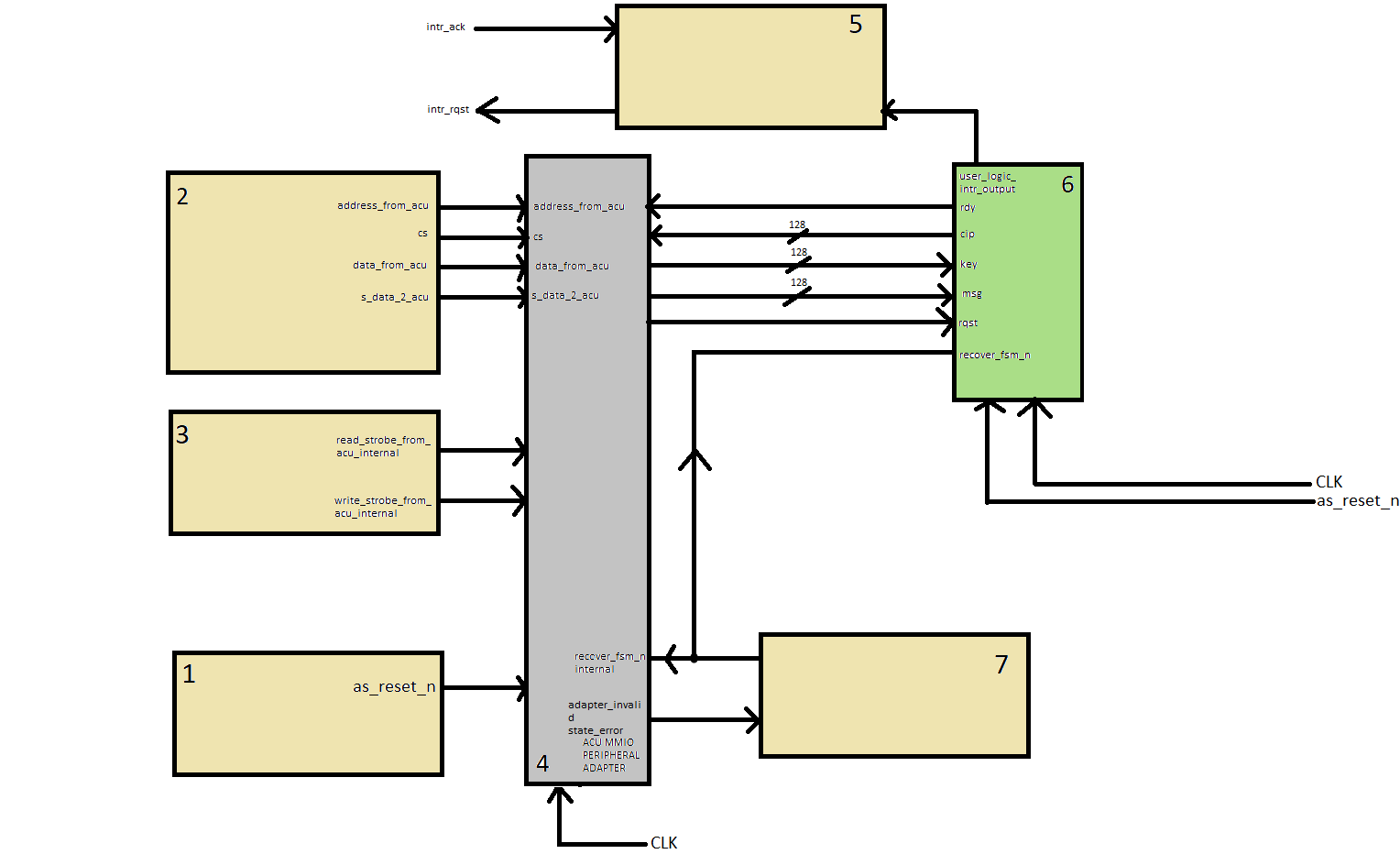
**Rendszerterv – kvázi AES titkosítás**

A féléves projektünk során egy kvázi AES (Advanced Encryption Standard) titkosítást fogunk megvalósítani egy MMIO perifériavezérlő segítségével. A „kvázi” jelző arra utal, hogy nem magára az AES egyedi funkcióira helyezzük a hangsúlyt, hanem mint IP egység az implementálására.

Az AES funkciói az alábbiak: üzenet és kulcs bitenkénti XOR-ozása, ezen kódolt üzenet karaktercseréje egy meghatározott look-up table segítségével (ún. S-box). Egy 4x4 mátrixként értelmezhető a 128 bites adatunk, melyen előbb soronkénti shiftelést, majd az oszlopok variálását hajtjuk végre. Ez utóbbit egy speciális konstansokból álló mátrix-szal történő szorzás testesíti meg. Ezután csak iteráljuk ezen lépéseket 10-szer, így kapjuk meg a kódolt végeredményt, az úgynevezett Cypher Text-et.

Esetünkben a kulcsunk 1db 1-es, tehát a kulccsal történő művelet tulajdonképpen egy összeadássá egyszerűsödik. Az S-box nem-szimmetrikus mivolta biztosítja számunkra a műveleteink valódi titkosító jellegét (azaz nincsen olyan bemeneti metódus, amelyekre „hasonló”, akár ugyanolyan eredményt kapunk).

******

1. **Double-flop reset-szinkronizáló áramkör:**

Az ACU mikrokontrollerben minden almodul (a processzormag és a különböző perifériavezérlők) saját reset-szinkronizáló fokozattal van ellátva, ami megkönnyíti a toplevel-integrációt, ahol már nem kell a reset-hierarchia kialakításával foglalkozni, elegendő az almodulokat példányosítani. Az almodulonkénti reset-szinkronizáció a reset jel terheléskiegyenlítését is szolgálja.

1. **Lokális címdekóder:**

Az ACU mikrokontroller elosztott címdekódolási sémát valósít meg (lásd 6.1. pont). Minden perifériavezérlő saját címdekódert tartalmaz, amely a processzor címbuszának értéke alapján előállít egy lokális chip-select (cs) jelet. E chip-select jel logikai magas szintű, ha a processzor címbuszán lévő cím az adott perifériavezérlő címterébe tartozik. A lokális chip-select jellel vezérelt M1 és M2 jelű multiplexerek biztosítják, hogy a perifériavezérlő a ready\_2\_acu és a data\_2\_acu kimeneteket mindaddig logikai alacsony értéken tartsa, ameddig a processzor „meg nem szólítja” a perifériát. A lokális címdekóder egy kombinációs hálózat.

1. **MMIO strobe jelek metastabil szűrői:**

Az MMIO perifériavezérlők és a processzormag aszinkron órajeltartományokat képviselhetnek, ezért a strobe jeleket metastabil szűrőkkel kell ellátni. Ezek a szűrő fokozatok nem valósítják meg a jel megérkezését nyugtázó visszacsatolást (vö. 2.1. pont: trigger\_ack), mert ugyanezt az információt az MMIO interfész ready\_2\_acu jele is hordozza.

1. **MMIO adapter FSM:**

Az MMIO interfész írás/olvasás kéréseit egy állapotgép fordítja le az adott perifériaáramkör (pl. UART, SPI, GPIO stb.) „nyelvére”; az MMIO írás/olvasás kéréseket értelmezve vezérli a perifériaáramkört.

1. **Megszakításgenerálás**:

Bizonyos perifériavezérlők nemcsak az MMIO buszon keresztül, hanem a megszakítási rendszeren keresztül is kapcsolódnak az ACU processzorhoz. A 15. ábrán bemutatott megszakításgeneráló áramkör teljesíti a processzor megszakításkérő és nyugtázó interfészének ütemezési kritériumait (3. ábra), így hozzákapcsolható közvetlenül a processzorhoz és a 6.5.1. pontban bemutatott MMIO megszakításvezérlőhöz is. A megszakítás forrása az ábrán vázolt elrendezésben a „USER LOGIC” modul user\_logic\_intr\_output kimenete, amelyről feltételezzük, hogy 0 → 1 átmenete az az esemény, amelynek meg kell szakítania a processzor műveletvégzését. A különböző perifériavezérlőknél ez az esemény más és más lehet, ennek megfelelően a fent vázolt megszakításgeneráló áramkör bemeneti (USER LOGIC-felőli) oldala is változhat. A megszakításgenerálás a generate\_intr generikus paraméter és az általa vezérelt M3 multiplexer segítségével letiltható. Ebben az esetben az MMIO strobe jelek metastabil szűrőinek kiiktatásánál tárgyalt mechanizmussal (lásd 2.1. pont) azonos okok miatt a teljes megszakításgeneráló áramkör kioptimalizálódik a szintézis során.

1. **USER LOGIC – kvázi AES:**

Az egyedi interfésszel rendelkező, újrafelhasználható modul, amelyet a 15. ábrán bemutatott egyéb áramköri részletekkel teszünk ACU MMIO kompatibilissé. Esetünkben ebben a modulban valósítjuk meg a felvázolt kvázi AES IP-nkat.

1. **Állapothelyreállító interfész:**

Az ábrán vázolt elrendezés két állapotgépet feltételez, egyik maga a USER LOGIC jelű áramkör, a másik pedig az MMIO interfész-adapter modul. Mindegyik állapotgép jelzi, ha valamilyen zavar miatt érvénytelen állapotba került. A USER LOGIC jelű modul természetesen nem feltétlenül kell hogy állapotgép legyen, de ha az, akkor a fenti sémát érdemes követni az implementáció során. Mivel a hibadetektáló és naplózó alrendszer ugyancsak különálló órajeltartományt képviselhet, ezért az állapotgépeket alapállapotukba visszaállító bemenet (recover\_fsm\_n) esetén is szükség van a szintézis során kiiktatható double-flop szinkronizáló fokozatra.

**ACU-MMIO FSM Interface portok:**

**entity** **acu\_mmio\_quazi\_aes** **is**

**generic** (

--metastable\_filter\_bypass\_recover\_fsm\_n: boolean := false;

--metastable\_filter\_bypass\_acu: boolean := false;

-- AES address map (128 bit datas separated to 16 bits)

address\_plaintext\_0: **integer** **range** **0** **to** **65535**;

address\_plaintext\_1: **integer** **range** **0** **to** **65535**;

address\_plaintext\_2: **integer** **range** **0** **to** **65535**;

address\_plaintext\_3: **integer** **range** **0** **to** **65535**;

address\_plaintext\_4: **integer** **range** **0** **to** **65535**

address\_plaintext\_5: **integer** **range** **0** **to** **65535**;

address\_plaintext\_6: **integer** **range** **0** **to** **65535**;

address\_plaintext\_7: **integer** **range** **0** **to** **65535**;

address\_key\_0: **integer** **range** **0** **to** **65535**;

address\_key\_1: **integer** **range** **0** **to** **65535**;

address\_key\_2: **integer** **range** **0** **to** **65535**;

address\_key\_3: **integer** **range** **0** **to** **65535**;

address\_key\_4: **integer** **range** **0** **to** **65535**;

address\_key\_5: **integer** **range** **0** **to** **65535**;

address\_key\_6: **integer** **range** **0** **to** **65535**;

address\_key\_7: **integer** **range** **0** **to** **65535**;

address\_cipher\_0: **integer** **range** **0** **to** **65535**;

address\_cipher\_1: **integer** **range** **0** **to** **65535**;

address\_cipher\_2: **integer** **range** **0** **to** **65535**;

address\_cipher\_3: **integer** **range** **0** **to** **65535**;

address\_cipher\_4: **integer** **range** **0** **to** **65535**;

address\_cipher\_5: **integer** **range** **0** **to** **65535**;

address\_cipher\_6: **integer** **range** **0** **to** **65535**;

address\_cipher\_7: **integer** **range** **0** **to** **65535**;

address\_rdy: **out std\_logic**;

address\_rqst: **in std\_logic**;

);

**port** (

clk: **in** **std\_logic**;

raw\_reset\_n: **in** **std\_logic**;

-- ACU memory-mapped I/O interface

read\_strobe\_from\_acu: **in** **std\_logic**;

write\_strobe\_from\_acu: **in** **std\_logic**;

ready\_2\_acu: **out** **std\_logic**;

address\_from\_acu: **in** **std\_logic\_vector** (**15** **downto** **0**);

data\_from\_acu: **in** **std\_logic\_vector** (**15** **downto** **0**);

data\_2\_acu: **out** **std\_logic\_vector** (**15** **downto** **0**);

-- ACU interrupt interface

intr\_rqst: **out** **std\_logic**;

intr\_ack: **in** **std\_logic**;

-- FSM error interface

invalid\_state\_error: **out** **std\_logic**;

recover\_fsm\_n: **in** **std\_logic**;

recover\_fsm\_n\_ack: **out** **std\_logic**

);

**end** **entity** **acu\_mmio\_quazi\_aes**;

**Quazi AES modul interface portok:**

**entity** **quazi\_aes** **is**

**generic** (

--metastable\_filter\_bypass\_host: boolean;

--metastable\_filter\_bypass\_recover: boolean;

);

**port** (

clk: **in** **std\_logic**;

raw\_reset\_n: **in** **std\_logic**;

--AES interface

key: **in** **std\_logic\_vector** (127 downto 0);

plaintext: **in std\_logic\_vector** (127 downto 0);

ciphertext: **out** **std\_logic\_vector** (127 downto 0);

rdy: **out** **std\_logic**;

rqst: **in** **std\_logic**;

intr: **out** **std\_logic**;

**end** **entity** **quazi\_aes**;