AVR을 이용한 CHAM64구현

유튜브: https://www.youtube.com/watch?v=0tw9PEU1seE

AVR 기초

CHAM 암호

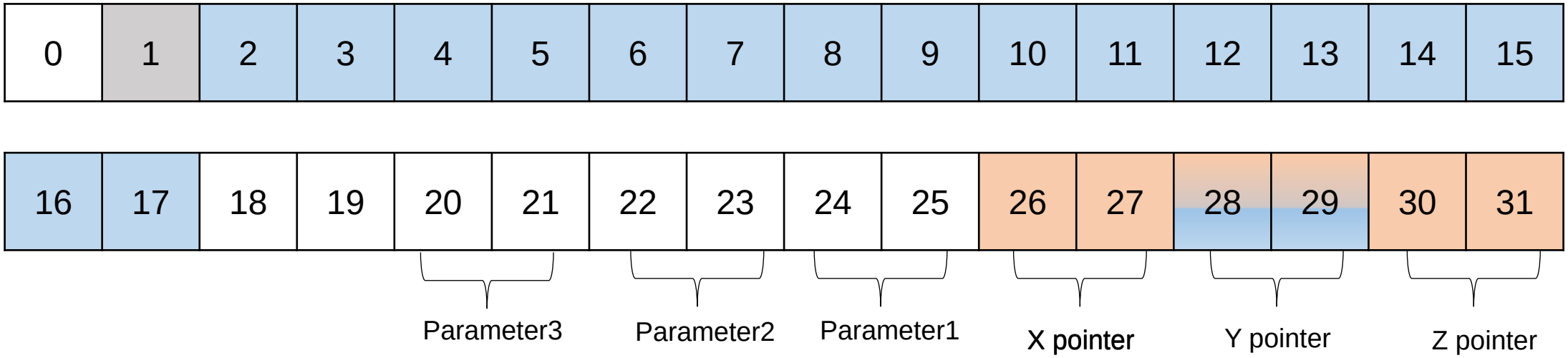
CHAM 구현

AVR 기초

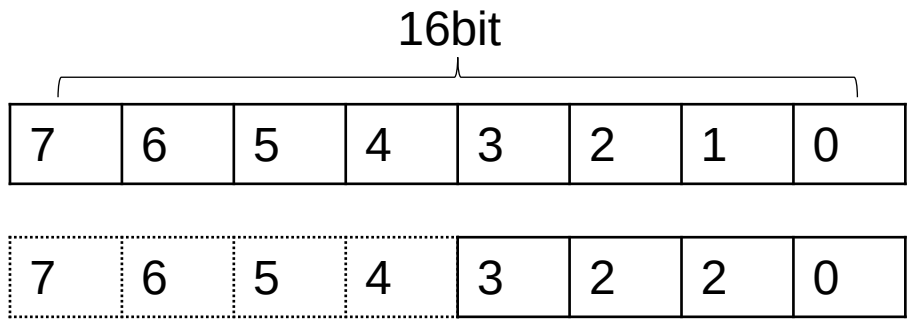
- ATMEL 사의 AVR 프로세서 (tiny,classic,mega 계열)

- ATmega128
  - 하버드 버스구조,(명령어 버스, 데이터 버스)
  - 8비트 단위 연산
  - 32개의 범용 레지스터
  - 81개의 명령어
    - - 분기 명령어
    - - MCU 제어 명령어
    - - 데이터 전송 명령어
    - - 산술 논리연산 명령어
    - - 비트조작명령어

AVR 기초



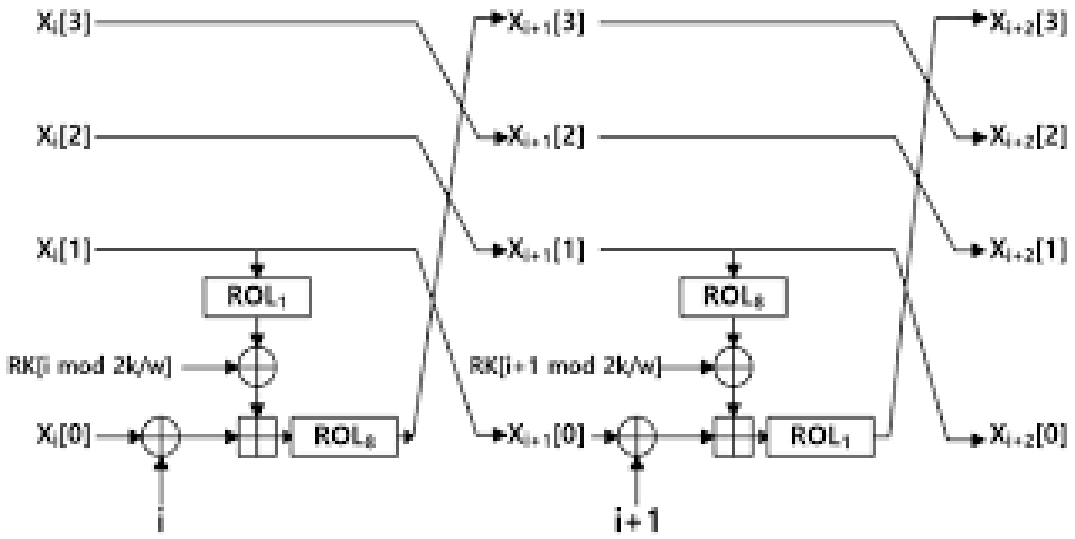
R1: Zero
R2~ R17, R28, R29: Callee Saved Register
R26,R27 :X pointer
R28, R29: Y pointer
R30, R31: Z pointer



ADD Rd, Rr
ADC Rd, Rr

CHAM 암호

국산 경량 블록암호
CHAM 64/128, CHAM128/128,
CHAM 128/256



ODD ROUND: ROL1 을 먼저 해 줌.
EVEN ROUND : ROL8 을 먼저 해 줌.

```
71  void cham64_encrypt(uint8_t* dst, const uint8_t* src, const uint8_t* rks)
72  {
73      uint16_t blk[4] = {0};
74      memcpy(blk, src, BLOCKSIZE_64);
75
76      const uint16_t* rk = (const uint16_t*) rks;
77      uint16_t rc = 0;
78
79      for (size_t round = 0; round < CHAM_64_128_ROUNDS; round += 8) {
80          blk[0] = rol16((blk[0] ^ (rc++)) + (rol16(blk[1], 1) ^ rk[0]), 8);
81          blk[1] = rol16((blk[1] ^ (rc++)) + (rol16(blk[2], 8) ^ rk[1]), 1);
82          blk[2] = rol16((blk[2] ^ (rc++)) + (rol16(blk[3], 1) ^ rk[2]), 8);
83          blk[3] = rol16((blk[3] ^ (rc++)) + (rol16(blk[0], 8) ^ rk[3]), 1);
84
85          blk[0] = rol16((blk[0] ^ (rc++)) + (rol16(blk[1], 1) ^ rk[4]), 8);
86          blk[1] = rol16((blk[1] ^ (rc++)) + (rol16(blk[2], 8) ^ rk[5]), 1);
87          blk[2] = rol16((blk[2] ^ (rc++)) + (rol16(blk[3], 1) ^ rk[6]), 8);
88          blk[3] = rol16((blk[3] ^ (rc++)) + (rol16(blk[0], 8) ^ rk[7]), 1);
89
90          rk = (rk == (const uint16_t*) rks) ? rk + 8 : rk - 8;
91      }
92
93      memcpy(dst, blk, BLOCKSIZE_64);
94  }
```

Q & A