2019 국가암호공모전 Ⅱ 분야

문제 03

공개키 암호화에서 가장 많이 활용되는 기본 연산은 모듈러 연산이다. 모듈러 연산을 간단히 정의하면 특정한 값 (A)을 또 다른 특정한 값 (B)으로 나눈 나머지 (R)를 구하는 것이다. 이를 식으로 나타내면 다음과 같다.

$$R = A \mod B$$

본 문제에서는 5개의 54-비트 양의 정수 (A, B, C, D, E)를 더한 이후에 이를 $2^{56}-1$로 나눈 나머지 (R)를 8-비트 프로세서 상에서 구현해 보도록 한다. 이를 식으로 나타내면 다음과 같다.

$$R = (A+B+C+D+E) \mod 2^{56}-1$$

○ 힌트#1: 위의 나눗셈은 다음과 같은 특성을 이용하면 보다 편하게 계산할 수 있다.

$$2^{56} \equiv 1 \pmod{2^{56}-1}$$

○ 힌트#2: 54-비트 연산은 8-비트 프로세서에서 한 번에 수행할 수 없으므로 8-비트 단위로 쪼개어서 계산하도록 한다.

○ 힌트#3: 54-비트 덧셈 연산은 특정한 경우에 56-비트 결과값을 도출한다.

해당 모듈러 덧셈에 대한 테스트 벡터는 아래와 같다. 테스트 벡터는 16진수로 표기되어 있다.

Test Vector #1

A	B	C	D	E	R
0x3C70B7259D4F8D	0x296B14ABD055A5	0x2C9C096DF02954	0x3A5453E1663EA7	0x3921FE5717E51B	0x5EE2777DBF249

Test Vector #2

A	B	C	D	E	R
0x2D6D0148F5B644	0x25159EBDB793D0	0x3F884820B92BAF	0x2C5593AFA660D0	0x36E9AD6060023D	0xF54A29376CD8D0

Test Vector #3

A	B	C	D	E	R
0x2D36C57FC52589	0x2DCDFB28D1A5B6	0x234283E941AAA6	0x29D4A208E50F7F	0x35AD21E3E7D30C	0xDDC9087EA55870

○ 구현 및 테스트 환경 상세

- 해당 문제의 구현물은 Arduino-UNO (8-비트 AVR 프로세서) 상에서의 최적화 구현이며 해당 디바이스에 대한 정보는 아래와 같다.

https://store.arduino.cc/usa/arduino-uno-rev3

- Arduino-UNO 상에서의 구현은 Arduino-IDE를 활용한다.
- 사전에 Arduino-UNO 보드를 구하지 못한 경우에는 8-비트 AVR 프로세서의 시뮬레이션 툴인 Atmel Studio 7을 활용하여 테스트 가능하다. 단 최종 결과물은 Arduino-IDE로 프로그래밍하여 프로젝트 형태로 제출한다.

https://www.microchip.com/mplab/avr-support/atmel-studio-7

모듈러 덧셈 연산은 다음 두 함수를 활용하여 구현한다.

```
unsigned char add (unsigned char* R, unsigned char* IN1, unsigned char* IN2){
        //add 함수는 덧셈 코드를 구현함
        //함수 출력 값인 문자열 (R)은 7바이트임
        //함수 입력 값인 문자열 (IN1 그리고 IN2)은 모두 7바이트임
        //함수 리턴 값 (1바이트)은 덧셈에 대한 오버플로우 값 저장용으로 활용
}


void mod (unsigned char* OUT, unsigned char* IN, unsigned char CARRY){
        //mod 함수는 모듈러 리덕션 코드를 구현함
        //함수 출력 값인 문자열 (OUT)은 7바이트임
        //함수 입력 값 중 문자열 (IN)은 7바이트임
        //함수 입력 값 중 문자 (CARRY)는 1바이트임
}
```

연산속도 체크(벤치마크)는 다음 코드를 활용한다. 아래 코드에 명시된 add 함수와 mod 함수의 조합은 기본적인 방법이며 참가자가 원하는 조합으로 얼마든지 두 함수를 활용할 수 있다.

```
  u32 time1;
  u32 time2;

  time1 = millis();
  for(int i=0;i<10000;i++){
    CARRY=add (R1,A,B);
    mod (R1,R1,CARRY);
    CARRY=add (R2,R1,C);
    mod (R2,R2,CARRY);
    CARRY=add (R3,R2,D);
    mod (R3,R3,CARRY);
    CARRY=add (R4,R3,E);
    mod (R,R4,CARRY);
  }
  time2 = millis();
  Serial.println((time2-time1));
```

결과물은 다음 2종을 포함한다.

- Arduino-IDE 전체 프로젝트
  (테스트 벡터 확인 과정, 벤치마크 과정, add, mod 및 main 함수 포함)
- 문서 (구현 기법 상세, 벤치마크 결과, 테스트 벡터 결과)

평가방법은 다음과 같다.

- 테스트 벡터 통과 (절대 평가), 문서화 (상대 평가), 벤치마크 결과 (상대 평가)

○ 주의사항

- 벤치마크 결과는 상대평가이며 연산 속도가 빠른 순서대로 가장 높은 점수를 받는다.
- 테스트 벡터를 통과하지 못한 경우에는 0점 처리된다.
- 단 최종 결과물은 Arduino-IDE로 프로그래밍하여 프로젝트 형태로 제출한다.