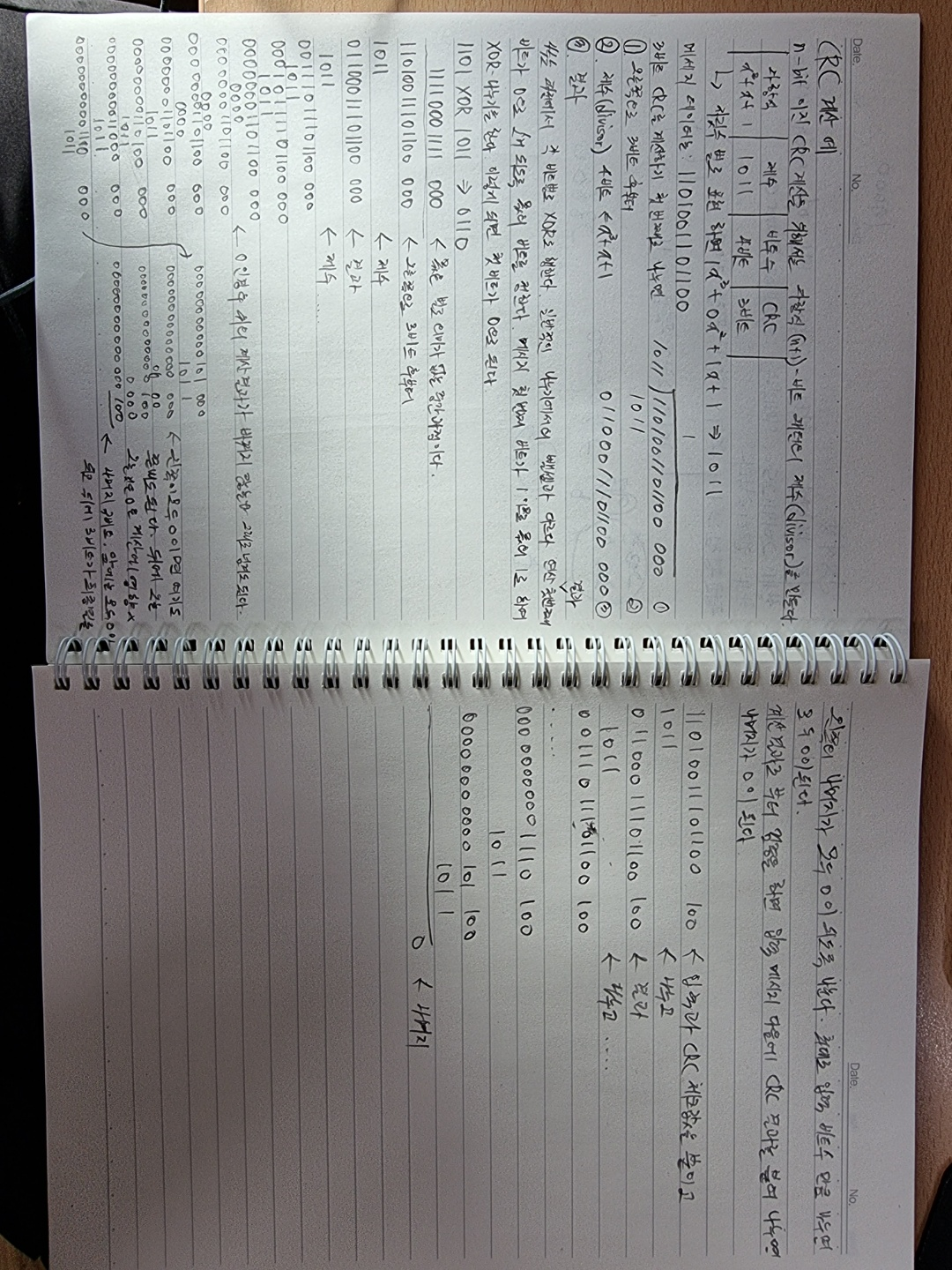
CRC코드  생성 및 검증하는 과정을 풀이하여 제출하기 바랍니다.

컴퓨터공학과 2학년 20191125 박형민



**순환 중복 검사**(巡環重復檢査), **CRC**(cyclic redundancy check)는 [네트워크](https://ko.wikipedia.org/wiki/%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC) 등을 통하여 데이터를 전송할 때 전송된 데이터에 오류가 있는지를 확인하기 위한 체크값을 결정하는 방식을 말한다.

데이터를 전송하기 전에 주어진 데이터의 값에 따라 CRC 값을 계산하여 데이터에 붙여 전송하고, 데이터 전송이 끝난 후 받은 데이터의 값으로 다시 CRC 값을 계산하게 된다. 이어서 두 값을 비교하고, 이 두 값이 다르면 데이터 전송 과정에서 잡음 등에 의해 오류가 덧붙여 전송된 것 임을 알 수 있다.

CRC는 [이진법](https://ko.wikipedia.org/wiki/%EC%9D%B4%EC%A7%84%EB%B2%95) 기반의 하드웨어에서 구현하기 쉽고, 데이터 전송 과정에서 발생하는 흔한 오류들을 검출하는 데 탁월하다. 하지만 CRC의 구조 때문에 의도적으로 주어진 CRC 값을 갖는 다른 데이터를 만들기가 쉽고, 따라서 [데이터 무결성](https://ko.wikipedia.org/wiki/%EB%8D%B0%EC%9D%B4%ED%84%B0_%EB%AC%B4%EA%B2%B0%EC%84%B1)을 검사하는 데는 사용될 수 없다. 이런 용도로는 [MD5](https://ko.wikipedia.org/wiki/MD5) 등의 함수들이 사용된다.

순환 중복 검사를 계산하는 과정은 하드웨어적 방식과 소프트웨어적 방식을 생각할 수 있다. 하드웨어적 방식을 말할 때, 직렬데이터를 계산하는 것이 단순하다. 통신시스템에서 프로토콜 계층에서 물리층에 가까울수록 하드웨어 접근을 그리고 상위계층에 가까울 수록 소프트웨어적인 방식이 적용된다.

통신시스템에서 물리계층에 가까울수록 직렬데이터를 사용하는 경향이 있다. 따라서 하드웨어적 계산방식을 사용한다. 전송라인은 거의 직렬 데이터이기 때문이다. 이런 경우 순환 중복 검사는 비트단위의 입력에 대한 출력을 얻는다. 논리 회로를 만들면 간단해진다. 그러나 높은 계층으로 갈수록 병렬데이터(octet 단위, 8비트)를 사용한다. 이런경우는 소프트웨어적 접근으로 주로 바이트 단위로 계산한다. 순환 중복 검사는 결국 비트단위 입력에 대한 각 비트별 [XOR](https://ko.wikipedia.org/wiki/XOR) 연산이므로 한 바이트 계산도 소프트웨어적 고속계산에 한계가 있다. 이런 경우 주로 미리계산을 한 테이블 형태를 사용한다