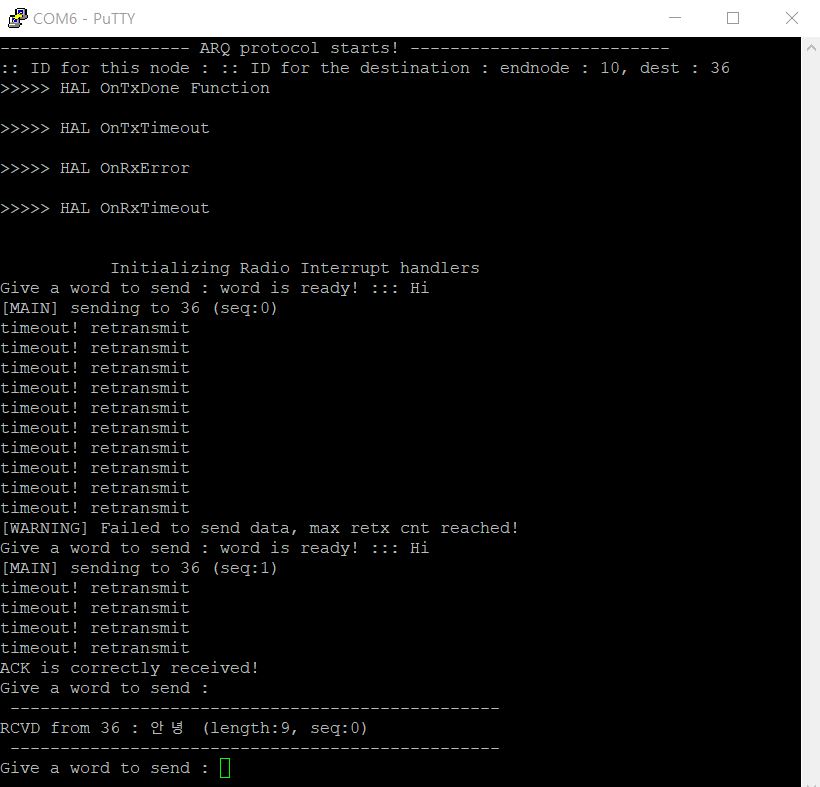
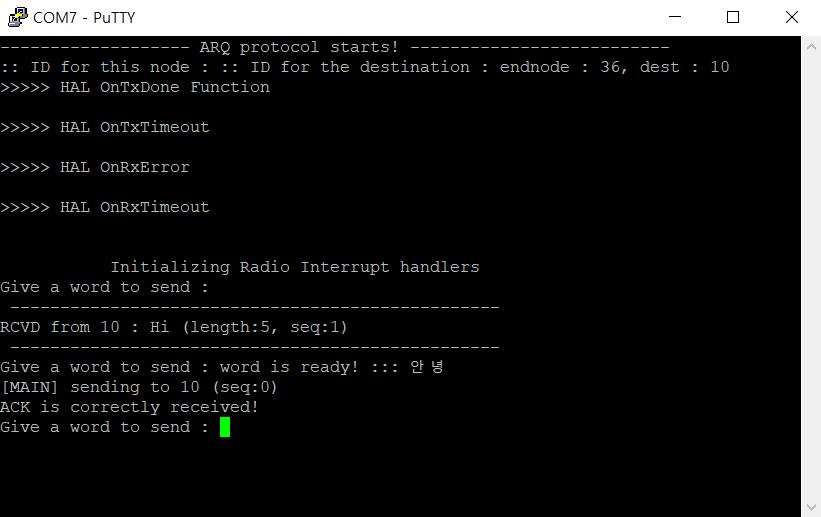
1. 결과 캡쳐, 수정된 코드 부분

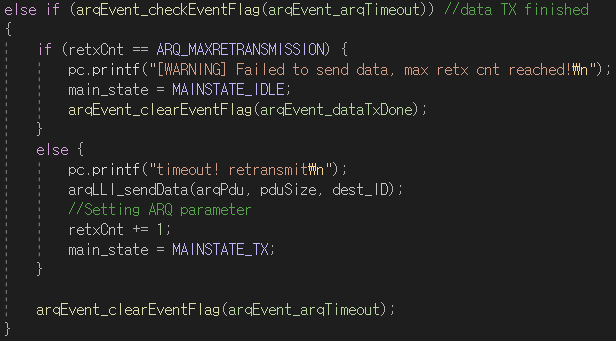
결과화면 1



결과 화면 2



수정된 코드 부분



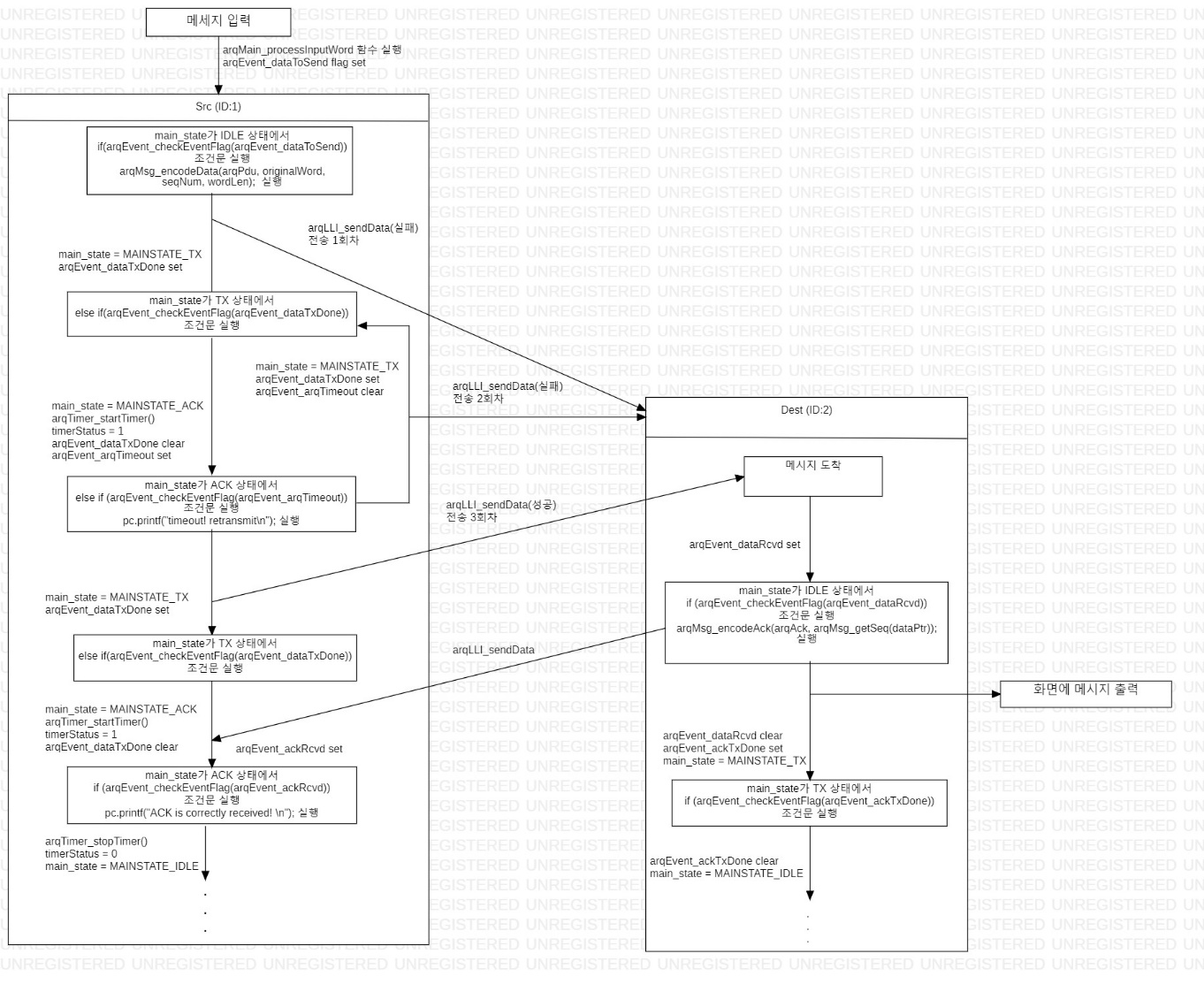
ARQ\_MAXRETRANMISSION의 값이 10이므로, timer가 만료될 때까지 ack를 수신하지 못하는 경우 재전송을 10회까지 합니다. retxCnt 변수는 0으로 초기화 되어있고 재전송을 할 때마다 1씩 값을 증가시키므로, 그 값이 ARQ\_MAXRETRANMISSION의 값과 같아질 때 실패를 알리고 IDLE상태로 전이해야 재전송을 10회까지만 할 수 있습니다. 따라서 if문을 추가해 retxCnt의 값을 확인하고 재전송 횟수를 넘었을 때 pc.printf("[WARNING] Failed to send data, max retx cnt reached!\n"); 명령을 추가해 사용자에게 전송이 실패했음을 알리고, main\_state = MAINSTATE\_IDLE; 명령을 추가해 상태를 전이했습니다. 전송이 실패했을 때 arqEvent\_dataTxDone flag의 경우 setting된 상태로 노드의 상태가 IDLE로 전이되면, 예외처리에 걸려 경고문을 출력합니다. Exercise2에서 원하는 것은 정상 동작이므로 arqEvent\_dataTxDone flag를 clear 해줍니다. 재전송 횟수를 초과하지 않는 경우에는 timer가 만료되어 재전송함을 사용자에게 알리고, 재전송을 수행하며, retxCnt를 1씩 증가시키고, 노드의 상태를 Tx로 전이시킵니다.

1. 동작과정 설명

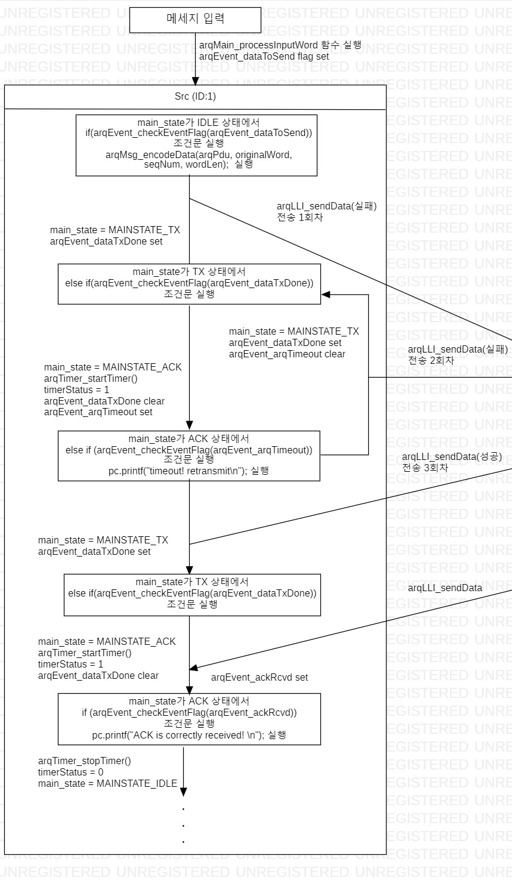
**src노드의 ID 가 1이고, dest노드의 ID가 2일때, src노드에서 dest노드로 "hello" 라는 메시지를 보내는 경우 받는쪽에서 패킷을 2번 받지 못하여 retransmission 2번이 일어나 세번째 전송에 ack수신에 성공하는 경우 모든 동작에 대해서 설명하시오.**

위 동작의 전체적인 흐름을 플로우 차트로 그리면 아래와 같다.

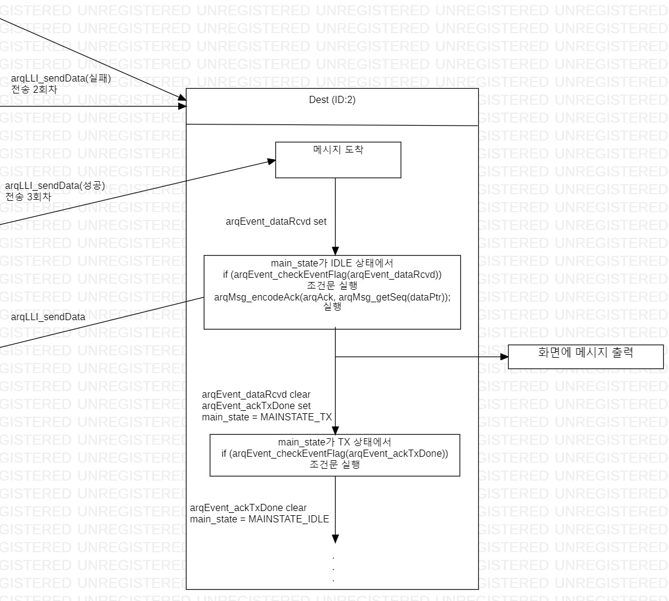
1. 전체 흐름도



* 1. Tx측 흐름도



* 1. Rx측 흐름도



•**초기화**

Src 노드의 경우 endNode\_ID : 1, dest\_ID : 2, dest 노드의 경우 endNode\_ID : 2, dest\_ID : 1, 두 노드 모두 flag\_needPrint : 1, prev\_state : 0 으로 초기화 되었습니다. 그리고 두 노드 전부 arqEvent\_clearAllEventFlag();를 수행하여 모든 event flag를 clear 합니다.

arqLLI\_initLowLayer(endNode\_ID) 함수를 통해 PHY MAC 주소를 초기화, 입력을 받을 수 있도록 pc.attach(&arqMain\_processInputWord, Serial::RxIrq);를 수행합니다. 정상 실행을 가정하였으므로 prev\_state와 main\_state의 값이 달라도 debug\_if문을 실행하지 않고, prev\_state = main\_state;를 실행합니다. Switch문에서 현재의 main\_state가 MAINSTATE\_IDLE이고 모든 flag는 clear 되어있기 때문에 else if (flag\_needPrint == 1) 조건문을 실행하여 flag\_needPrint의 값을 0으로 바꾸고 입력을 하라는 문장을 화면에 출력하고 대기하고 있는 상황입니다.

•**Src노드(In MAINSTATE\_IDLE)**

Src 노드의 화면에 “hello”가 입력되면, arqMain\_processInputWord() 함수를 통해 입력을 받으며, 함수를 통해 originalWord에는 ‘hello\0’이 wordLen에는 6이 저장되고, arqEvent\_dataToSend flag가 set 됩니다.

이로 인해 case MAINSTATE\_IDLE:문에서 else if (arqEvent\_checkEventFlag(arqEvent\_dataToSend)) 조건문을 실행하고, pduSize = arqMsg\_encodeData(arqPdu, originalWord, seqNum, wordLen); 실행을 통해 arqPdu : { 0(Msg type: data),0(seqNum),‘h’,‘e’, ‘l’, ‘l’, ‘o’, ‘\0’,}이 , pduSize : 8이 저장됩니다.

arqLLI\_sendData(arqPdu, pduSize, dest\_ID); 실행을 통해 dest\_ID로 arq\_Pdu가 전송되고, txType : 0(data tx)이 저장됩니다. txType변수에 따라 arqEvent\_dataTxDone이 set됩니다. seqNum : 1이, retxCnt : 0이 저장되고 data 전송 정보가 화면에 출력됩니다. 마지막으로 main\_state : MAINSTATE\_TX를 통해 상태를 TX로 전이하며 flag\_needPrint: 1을 저장하여 입력요청 메시지를 화면에 출력합니다. 마지막으로 arqEvent\_dataToSend flag를 clear합니다.

•**Src노드(In MAINSTATE\_TX)**

case MAINSTATE\_TX:문에서 else if (arqEvent\_checkEventFlag(arqEvent\_dataTxDone)) 조건문을 실행합니다. main\_state : MAINSTATE\_ACK를 저장하여 상태를 ACK로 전이, arqTimer\_startTimer();로 timer를 작동시키고, timerStatus : 1이 저장됩니다. 마지막으로 arqEvent\_dataTxDone flag를 clear 합니다.

•**Src노드(두번째, 세번째 재전송: In MAINSTATE\_ACK)**

3번 재전송을 하므로, timer가 만료될 경우에 해당합니다. 시간 내에 ack가 도착하지 못하였으므로, arqEvent\_arqTimeout flag가 set됩니다.

따라서 else if (arqEvent\_checkEventFlag(arqEvent\_arqTimeout)) 조건문을 실행합니다. retxCnt 값이 0이므로 "timeout! retransmit\n"을 화면에 출력하고 arqLLI\_sendData(arqPdu, pduSize, dest\_ID);함수를 실행합니다. 이때 첫번째 전송과 마찬가지로 txType에는 0이 저장되어 있으므로, arqEvent\_dataTxDone flag가 set 됩니다. retxCnt : 1을 저장하고, arqEvent\_arqTimeout flag를 clear 하고, main\_state : MAINSTATE\_TX를 저장하여 상태를 TX로 전이 시킵니다.

다음 과정은 src노드(In MAINSTATE\_TX)의 과정과 동일 하므로 생략하겠습니다. else if (arqEvent\_checkEventFlag(arqEvent\_arqTimeout)) 조건문의 실행이 한번 더 일어나므로 data의 세번째 재전송이 수행되고 화면에 "timeout! retransmit\n" 출력합니다. 이때 retxCnt의 값은 2가 되고, main\_state : MAINSTATE\_TX를 저장하여 상태를 TX로 전이 시킵니다.

•**Dest 노드(In MAINSTATE\_IDLE)**

수신한 msg[ARQMSG\_OFFSET\_TYPE]의 값이 0이므로, arqLLI\_dataIndFunc() 함수에의해서arqEvent\_dataRcvd flag가 set 됩니다. 이로 인해 case MAINSTATE\_IDLE: 문에서 if (arqEvent\_checkEventFlag(arqEvent\_dataRcvd)) 조건문이 실행됩니다. srcId에는 src 노드의 ID가 arqLLI\_getSrcId(); 함수를 통해 저장되고, dataPtr에는 arqLLI\_getRcvdDataPtr(); 함수를 통해 arqPdu의 포인터 변수가 저장되고, size에는 arqLLI\_getSize(); 함수를 통해 pduSize의 값인 8이 저장됩니다. 그리고 pc.printf 함수를 통해 화면에 srcId, 수신받은 메시지, pduSize, seqNum이 출력됩니다. arqMsg\_encodeAck(arqAck, arqMsg\_getSeq(dataPtr)); 함수의 실행으로 arqAck : { 1(Msg type: ack), 0(seqNum), 1} 이 저장되고, arqLLI\_sendData(arqAck, ARQMSG\_ACKSIZE, srcId); 함수를 통해 ack를 src 노드로 전송하여 txType에는 1이 저장됩니다. txType변수에 의해서 arqEvent\_ackTxDone flag가 set 됩니다. main\_state : MAINSTATE\_TX를 저장하여 상태를 TX로 전이 시키고, flag\_needPrint : 1을 저장하여 입력 문장을 화면에 출력합니다. 마지막으로 arqEvent\_dataRcvd flag를 clear 합니다.

•**Dest 노드(In MAINSTATE\_TX)**

case MAINSTATE\_TX:에서 if (arqEvent\_checkEventFlag(arqEvent\_ackTxDone)) 조건문을 실행합니다. if (arqTimer\_getTimerStatus() == 1 || arqEvent\_checkEventFlag(arqEvent\_arqTimeout)) 조건문에 해당하지 않으므로 main\_state : MAINSTATE\_IDLE을 저장하여 상태를 IDLE로 전이 시킵니다.

•**Src노드(세번째 재전송 이후: In MAINSTATE\_ACK)**

가정에 의해 세번째 재전송 이후 ack를 수신했고, msg[ARQMSG\_OFFSET\_TYPE]의 값이 1이므로 arqEvent\_ackRcvd flag가 set되고, 이에 따라서 if (arqEvent\_checkEventFlag(arqEvent\_ackRcvd)) 조건문을 실행합니다. dataPtr 변수에 arqAck의 포인터 값을 저장하고 arqAck와 arqPdu의 두번째 요소에 저장된 seqNum의 값이 0으로 동일하므로 if (arqMsg\_getSeq(arqPdu) == arqMsg\_getSeq(dataPtr)) 조건문을 실행합니다. pc.printf 함수를 통해 화면에 ack를 수신했음을 출력하고, arqTimer\_stopTimer(); 함수를 실행하여 timer를 멈추고 timerStatus 변수에 0을 저장하고, main\_state: MAINSTATE\_IDLE을 저장하여 상태를 IDLE로 전이 시킵니다. 마지막으로 arqEvent\_ackRcvd flag를 clear합니다.

위와 같이 src 노드에서 dest 노드로 ‘hello’를 전송했을 때, 세번째 재전송에서 ack 수신을 성공하는 동작입니다.