TI DSP, MCU, Xilinx Zynq FPGA
프로그래밍 전문가 과정

MIBSPI Lab

2018.08.08

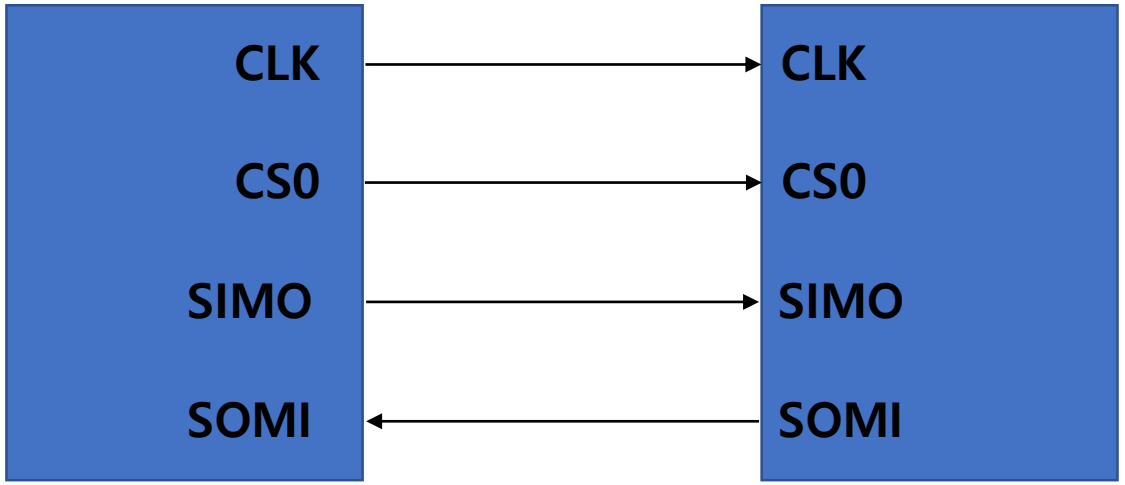
강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com

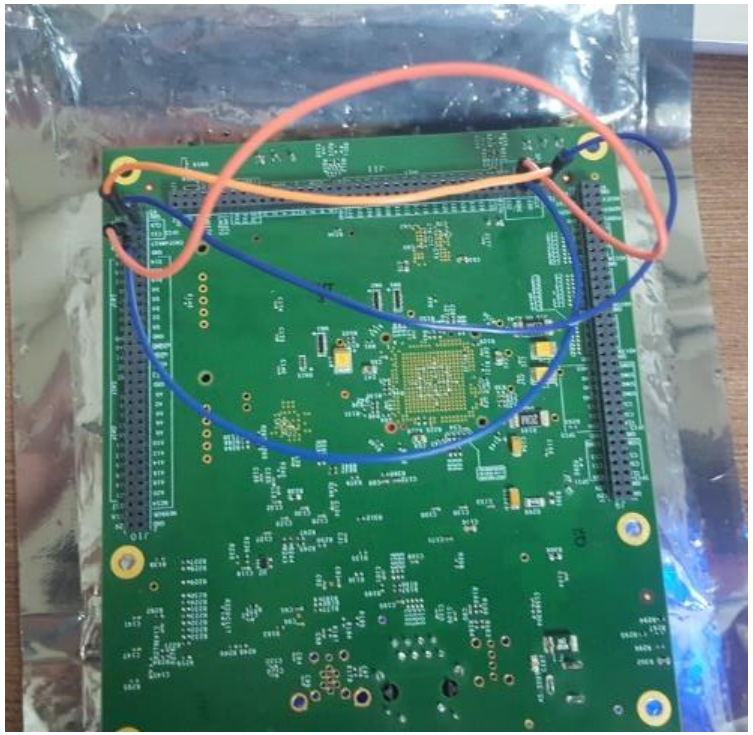
학생 – 안상재
sangjae2015@naver.com

1. TMS570 MCU 내부의 mibspi3 – mibspi2 통신 (페리페럴끼리의 통신)

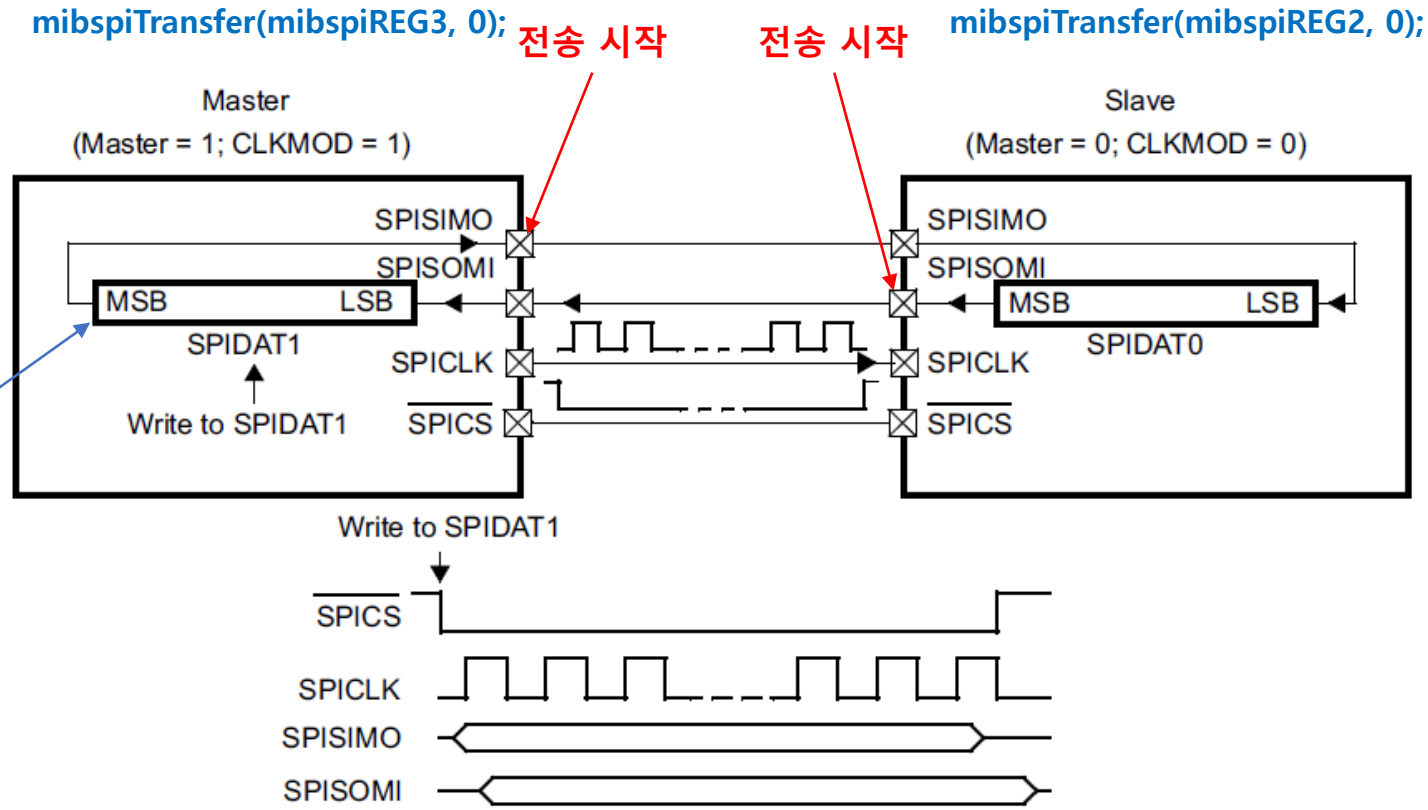
1. TMS570 MCU 내부의 mibspi3 – mibspi2 통신 (페리페럴끼리의 통신)

=> mibspi3 -> mibspi2 데이터 송신





동시에 해줘야 함!



송신과 수신이 동시에 일어남

2. 코드 분석

```
void main(void)
{
    data = &rx_data[0];
    _enable_IRQ_interrupt_();

    mibspiInit();
    sciInit();

    mibspiEnableGroupNotification(mibspiREG2, 0, 1); // mibspi2의 인터럽트 허용
    mibspiEnableGroupNotification(mibspiREG3, 0, 1); // mibspi3의 인터럽트 허용

    while(1)
    {
        mibspiSetData(mibspiREG3, 0, &tx_data3[0]);
// mibspi3에서 전송할 데이터를 송신 버퍼에 셋팅함
        mibspiTransfer(mibspiREG3, 0);
// mibspi3의 송신 버퍼의 데이터를 전송 개시함
        mibspiTransfer(mibspiREG2, 0);
    }
}
```

uint16 tx_data3[D_COUNT] = {'3','3','3','3','3','₩r','₩n','₩n'};

*** mibspi3 에서 mibspi2로 데이터를 전송하는 상황인데**

왜 mibspi2를 Transfer 해야 할까?

=> SPI 통신은 마스터, 슬레이브의 송신과 수신이 항상 동시에 일어난다.

그렇기 때문에 마스터가 전송을 할 때, 슬레이브도 동시에 전송을 해주어야 통신이 된다.

mibspi1~5의 전송이 완료되면 모두 같은 함수를 호출함

```
void mibspiGroupNotification(mibspiBASE_t *mibspi, uint32 group)
{
    uint32 ret1;

    if(mibspi == mibspiREG2)  // mibspi2 의 전송이 완료되면
    {
        ret1 = mibspiGetData(mibspi, group, data);
// mibspi2의 수신 버퍼로 들어온 데이터 저장
        sciSend(sciREG1, 13,"mibspiREG2 = ");
        sciSend(sciREG1, 16, data);  // 수신한 데이터를 터미널창에 출력
    }
}
```

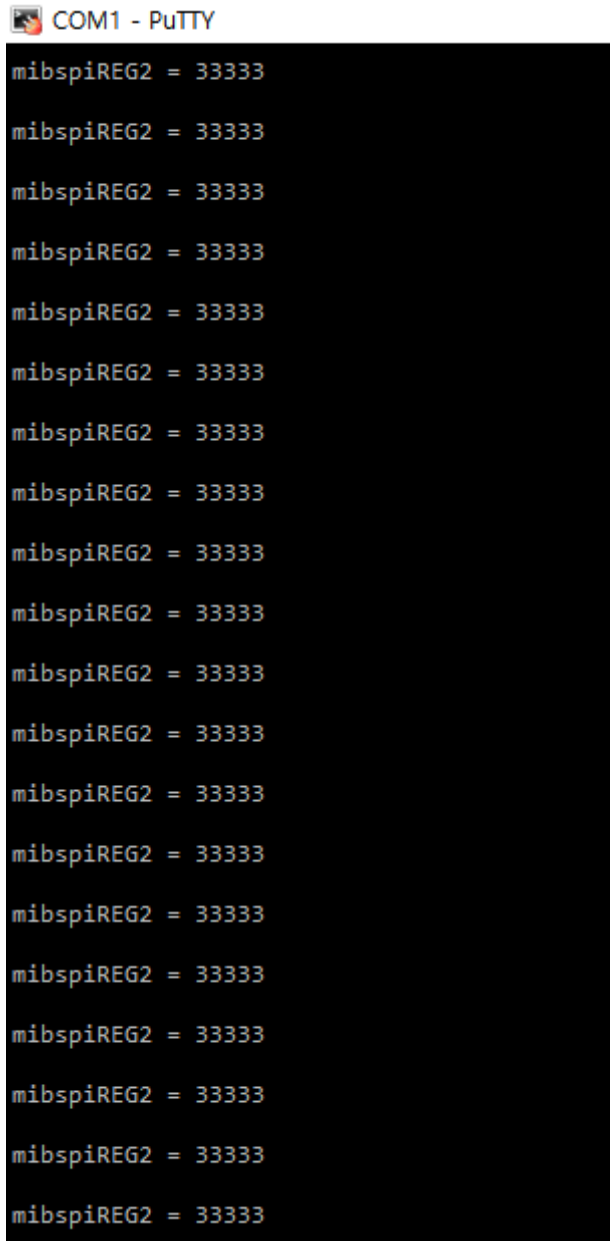
*** 코드 작성 시 주의 사항**

=> 절대로 LoopBack 을 enable 시켜서는 안됨!

=> LoopBack을 enable 시키게 되면 송신 시, 데이터가 SIMO핀으로 나가는 것이 아니라 LoopBack 경로로 이동함

=> 수신 시, 슬레이브에서 데이터가 SIMO핀에서 RX SHIFT REG로 이동하는 것이 아니라 LoopBack 경로로 이동함

3. 결과 화면

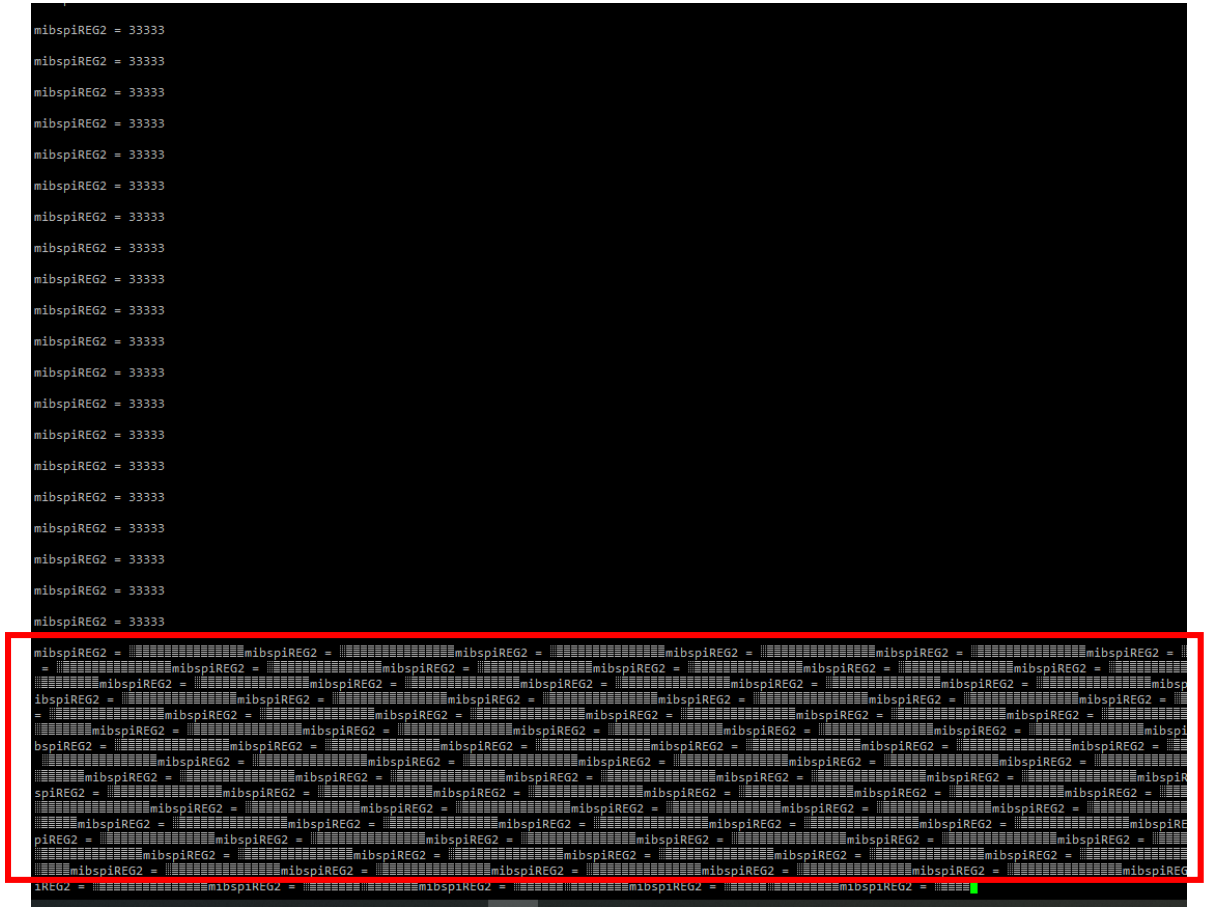


슬레이브(mibspi2)의 수신 상태

"333333₩r₩n₩n" 의 데이터를 mibspi3->mibspi2 로 전송했기 때문에
mibspi2가 수신한 데이터는
"333333₩r₩n₩n" 이라는 것을 확인할 수 있음!!!

4. 통신 중에 연결 되어 있는 선을 뽑으면?

1) 통신 중에 clk 선을 뺀 경우

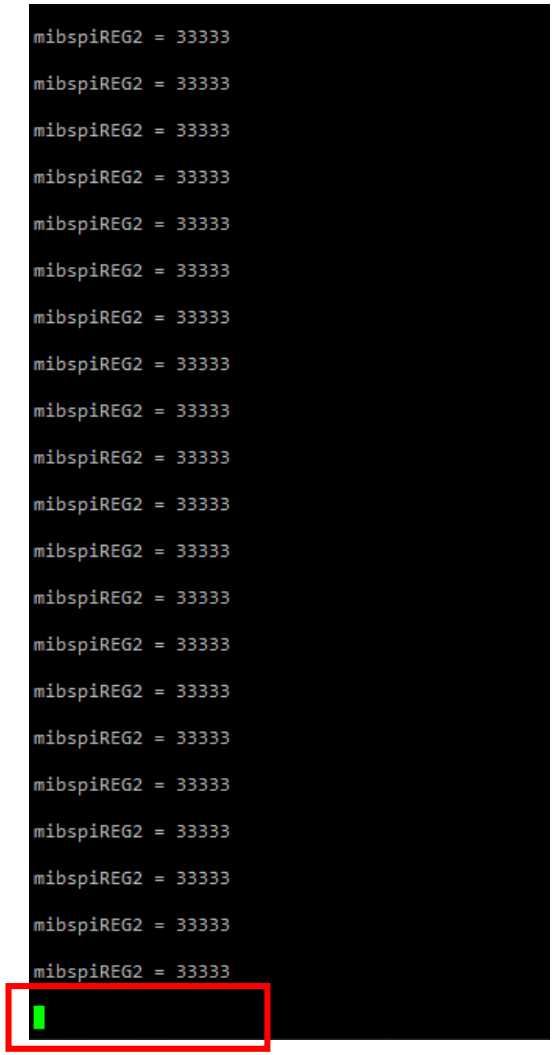


2) 통신 중에 SIMO 선을 뺀 경우

쓰레기 값이 들어감

3) CS 선을 뺀 경우

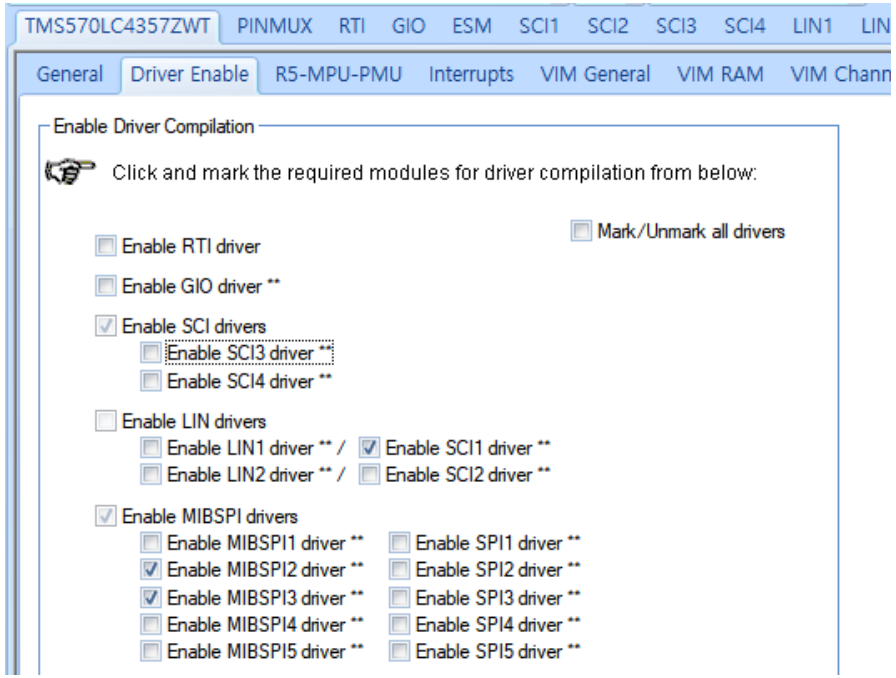
- 통신이 중단됨

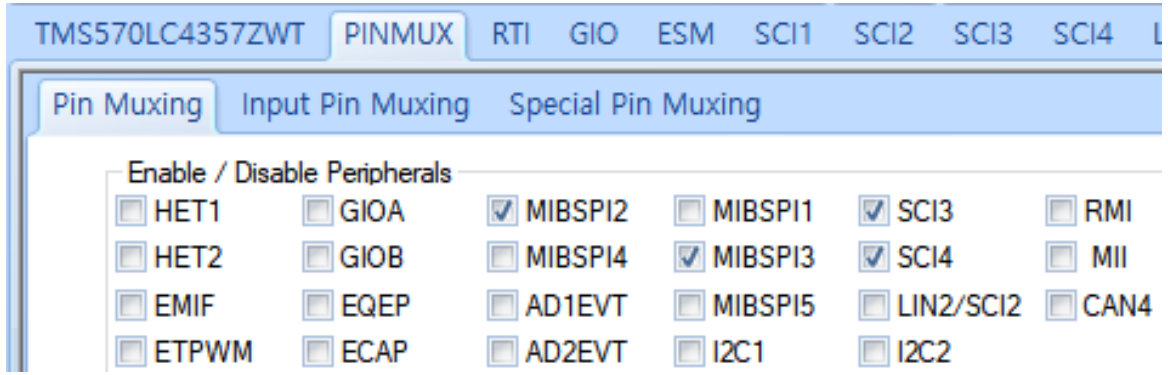


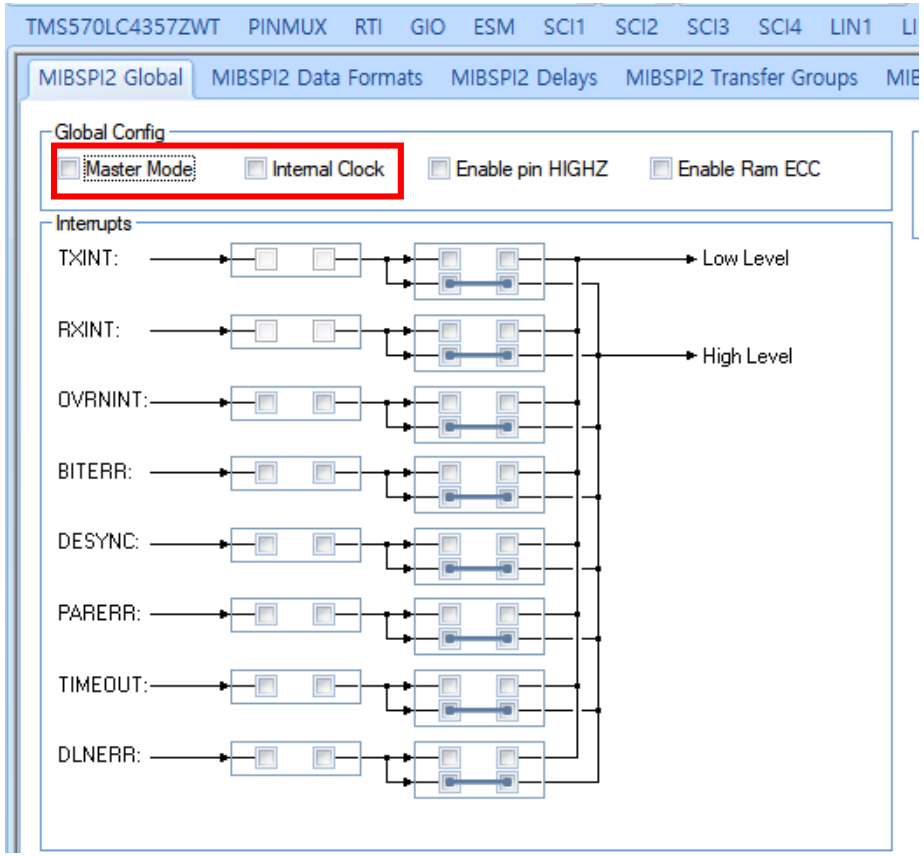
4) SOMI 선을 뺀 경우

- **정상 동작함.** (SOMI 선이 없더라도 마스터->슬레이브의 통신에는 지장이 없음)
- **슬레이브에서 마스터로 데이터를 송신하지 않아도 마스터가 송신하는 데이터를 잘 수신함.**

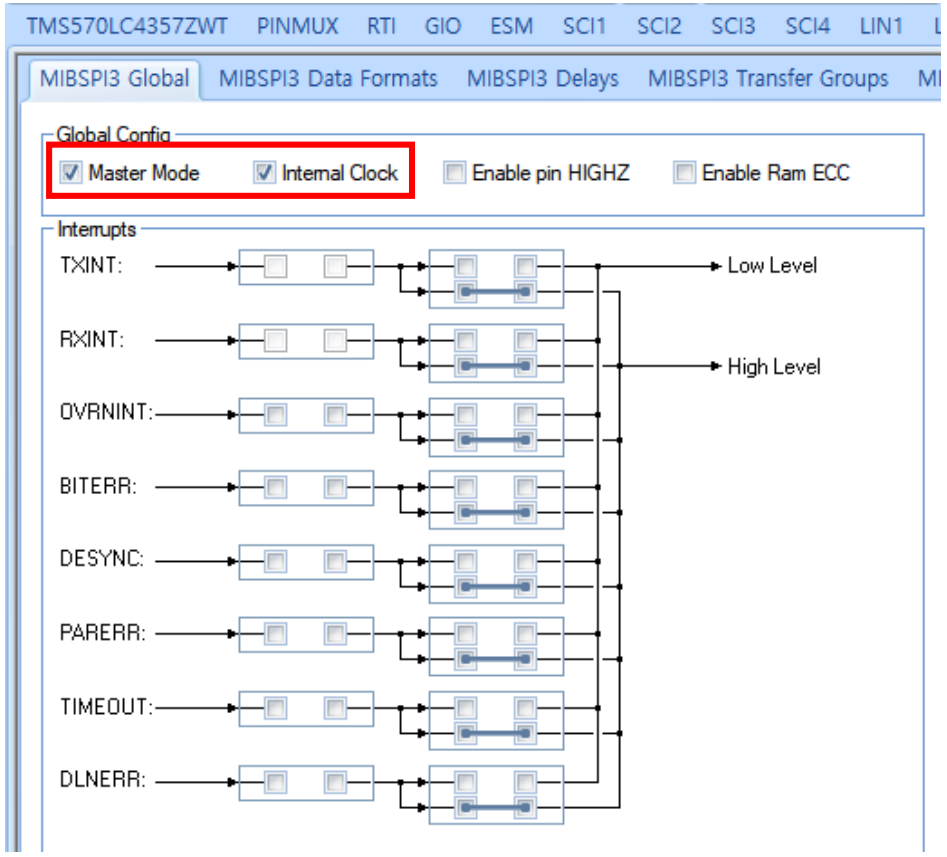
5. HALCoGen 설정



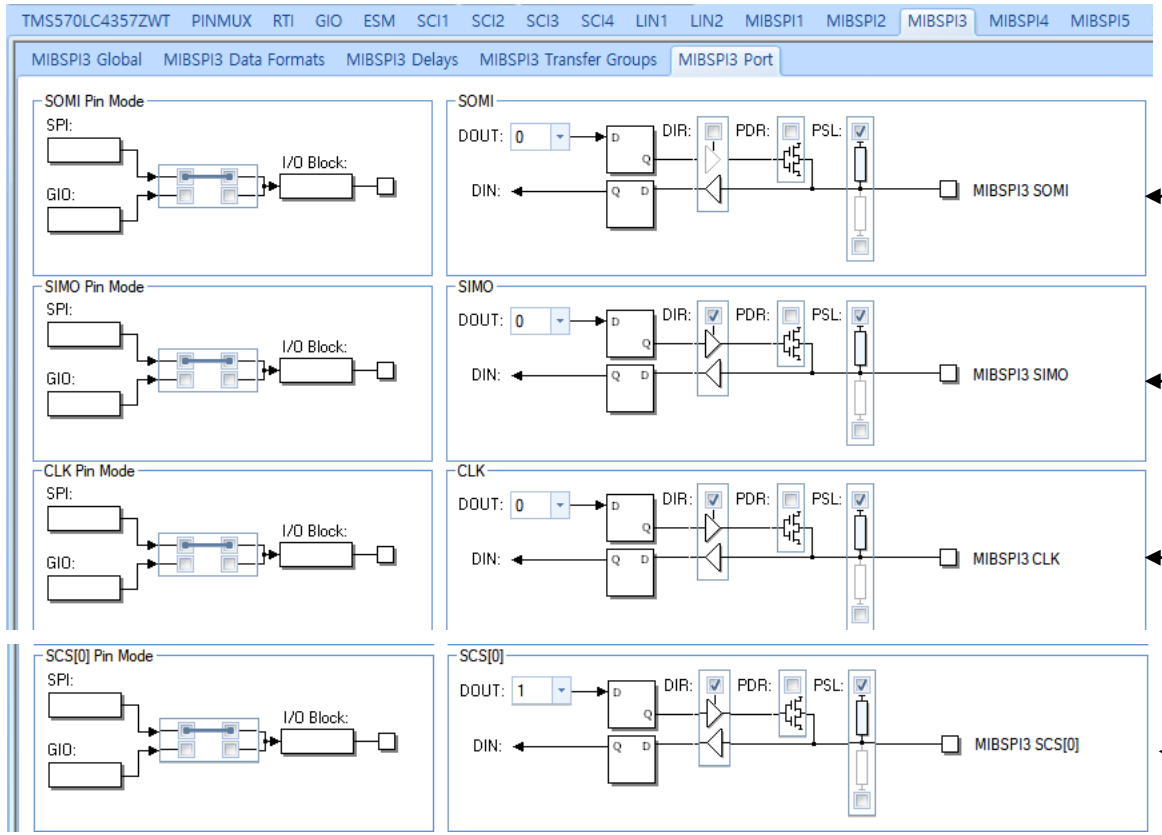




슬레이브 설정(Master = 0, CLKMOD = 0)



마스터 설정(Master = 1, CLKMOD = 1)



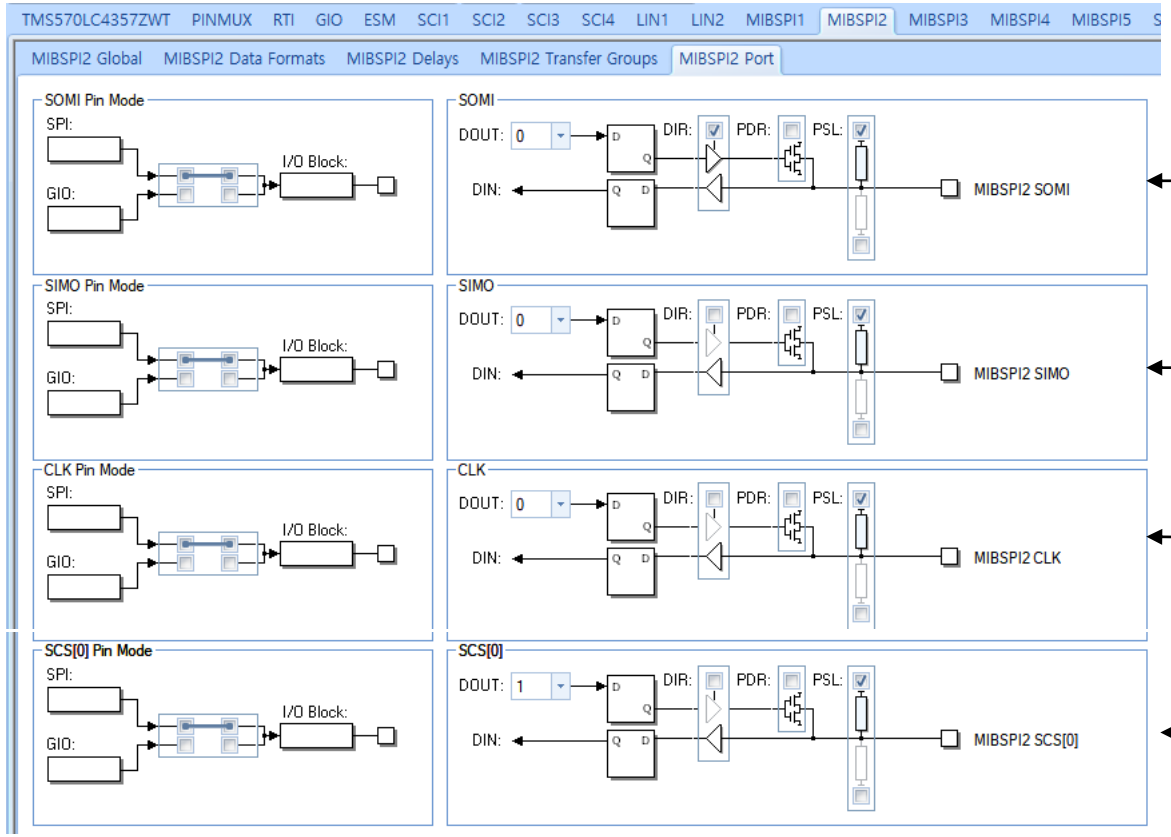
마스터 입장에서는 SOMI 핀은 입력

마스터 입장에서는 SIMO핀은 출력

마스터 입장에서는 CLK 핀은 출력

마스터 입장에서는 CS 핀은 출력

* 사실 마스터랑 똑같이 방향 설정해도 통신은 정상동작함



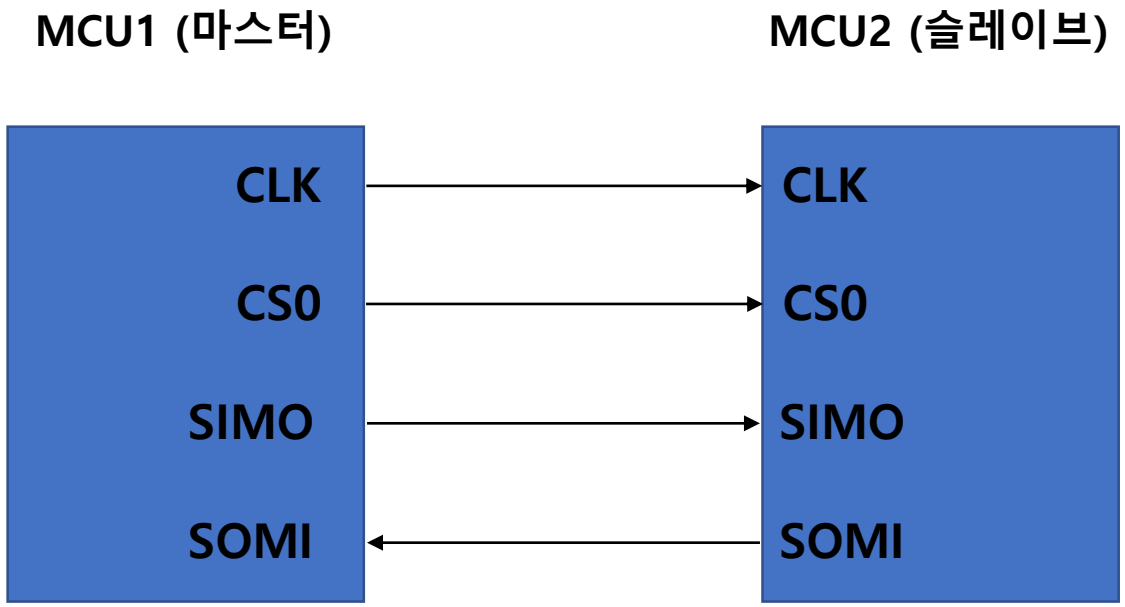
슬레이브 입장에서는 SOMI 핀은 출력

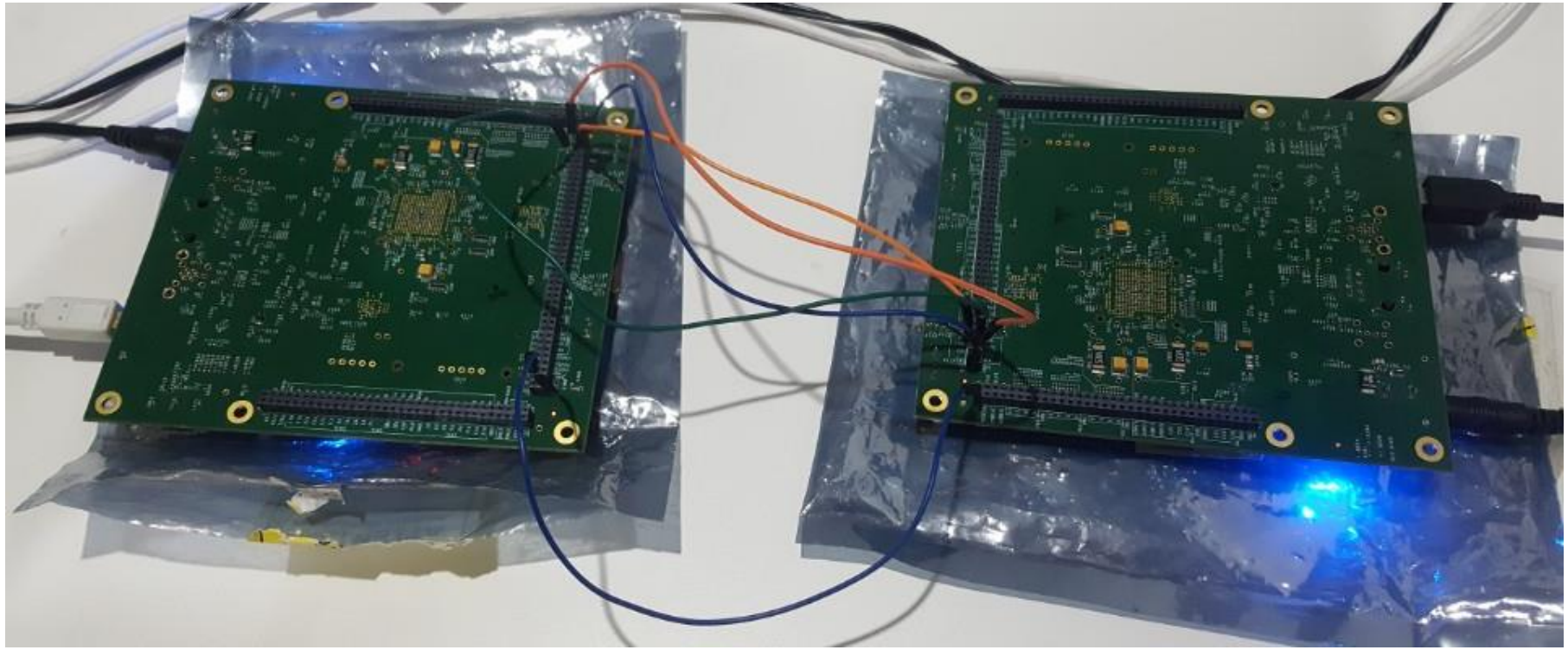
슬레이브 입장에서는 SIMO핀은 입력

슬레이브 입장에서는 CLK 핀은 입력

슬레이브 입장에서는 CS 핀은 입력

1. TMS570 MCU – MCU MIBSPI 통신

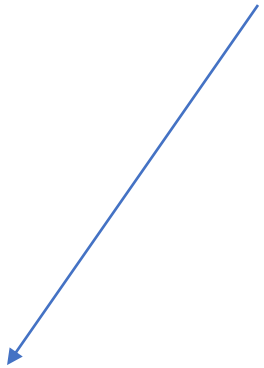

MCU1 (마스터)
MCU2 (슬레이브)
CLK
CS0
SIMO
SOMI
CLK
CS0
SIMO
SOMI



2. 코드 분석

- 마스터(송신)

```
                                    uint16 tx_data[8] = {'a','a','a','a','a','₩r','₩n','₩n'};
int main(void)
{
    data = &rx_data[0];
    _enable_IRQ_interrupt_();
    sciInit();
    mibspiInit();

    while(1)
    {
        mibspiSetData(mibspiREG3, 0, &tx_data[0]);
        // 송신 버퍼에 송신할 데이터 셋팅
        mibspiTransfer(mibspiREG3, 0);  // 전송 시작
    }

    return 0;
}
```



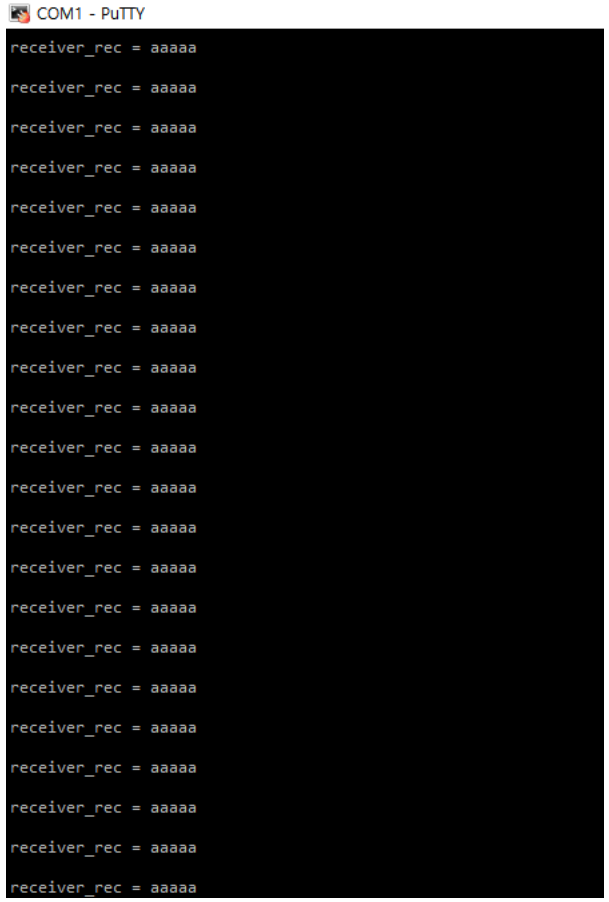
- 슬레이브(수신)

```
int main(void)
{
    data = &rx_data[0];
    _enable_IRQ_interrupt_();
    sciInit();
    mibspiInit();

    mibspiEnableGroupNotification(mibspiREG3, 0, 1);

    while(1)
        mibspiTransfer(mibspiREG3, 0);
// 마스터로부터 데이터가 수신되는 순간을 위해 계속 전송 준비를 함

    return 0;
}
```

```
void mibspiGroupNotification(mibspiBASE_t *mibspi, uint32 group)
{
    if(mibspi == mibspiREG3)
    {
        ret = mibspiGetData(mibspi, group, data);
        // 마스터로부터 수신한 데이터를 저장함
        sciSend(sciREG1, 15, "receiver_rec =  ");
        sciSend(sciREG1, 16, data);    // 디버깅
    }
}
```

**** 결과 화면***

- 슬레이브의 터미널 창



마스터에서 **{'a','a','a','a','a','₩r','₩n','₩n'}** 의 데이터를 송신했으므로
통신이 정상적으로 되고 있다는 것을 알 수 있음.