SPI
(Serial Peripheral Interface)

2023. 2학기

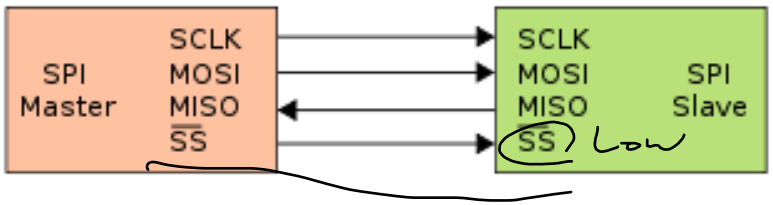
Kookmin Univ. EMCO Lab.

1. SPI 통신

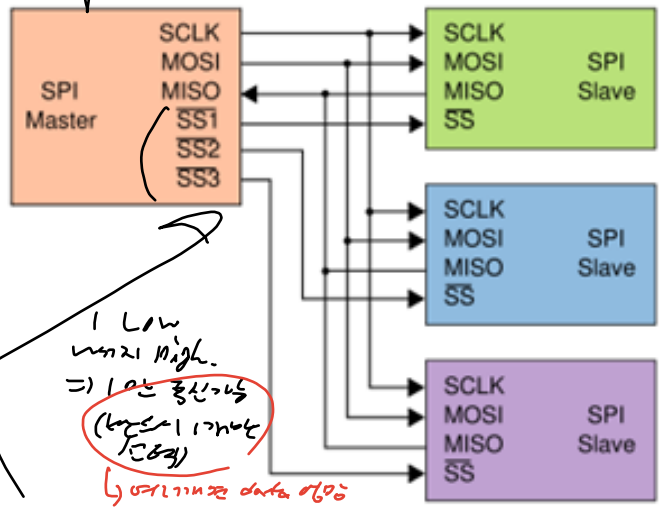
- SPI란?

: 모토롤라가 개발한 단순하고 신뢰성 높은 직렬 통신
: Clock을 사용하여 동기화된 통신을 하고 대역폭은 낮으나 신호 간섭이 적음
: Master-slave 모드로 동작
: Full-duplex* 통신 방식

- 사용 신호



SCLK : Clock
MOSI : Master Output Slave Input
MISO : Master Input Slave Output
SS : Slave Select(Chip Select)



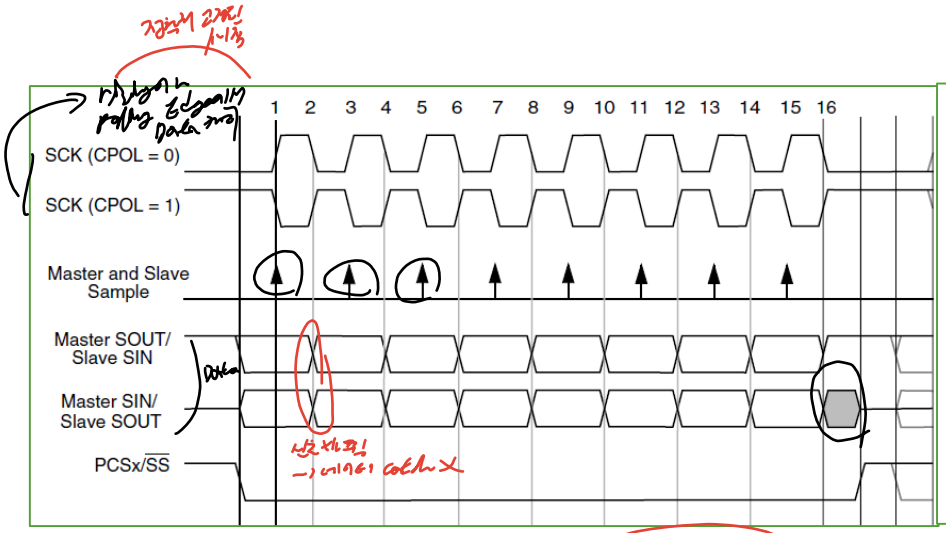
< Multiple Slaves >

*: 데이터를 송수신하기 위한 독립된 회선 보유

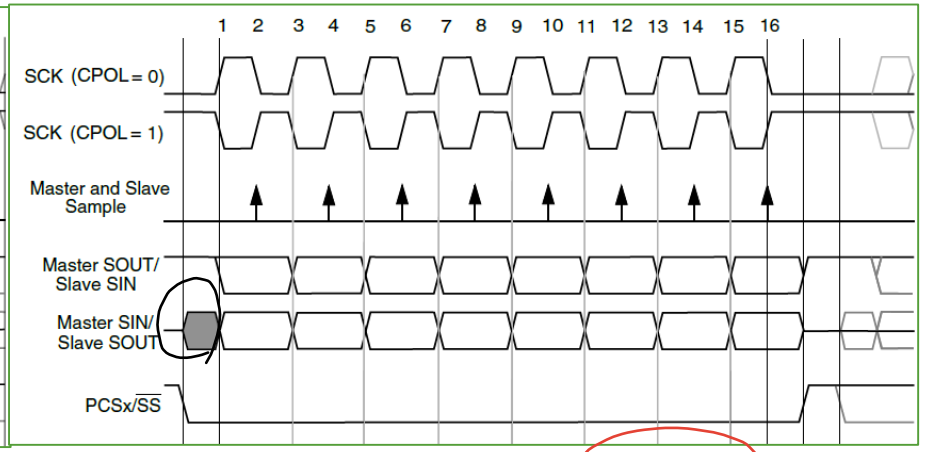
1. SPI 통신

- 통신 신호

- CS는 일반적으로 active low로 동작
- 송수신기가 데이터를 sampling하고 송수신하는 타이밍에 따라서 CPOL과 CPHA 값을 결정
- CPOL : Clock Polarity, SCLK의 극성을 선택
- CPHA : Clock Phase, 데이터를 sampling 하는 시점 선택



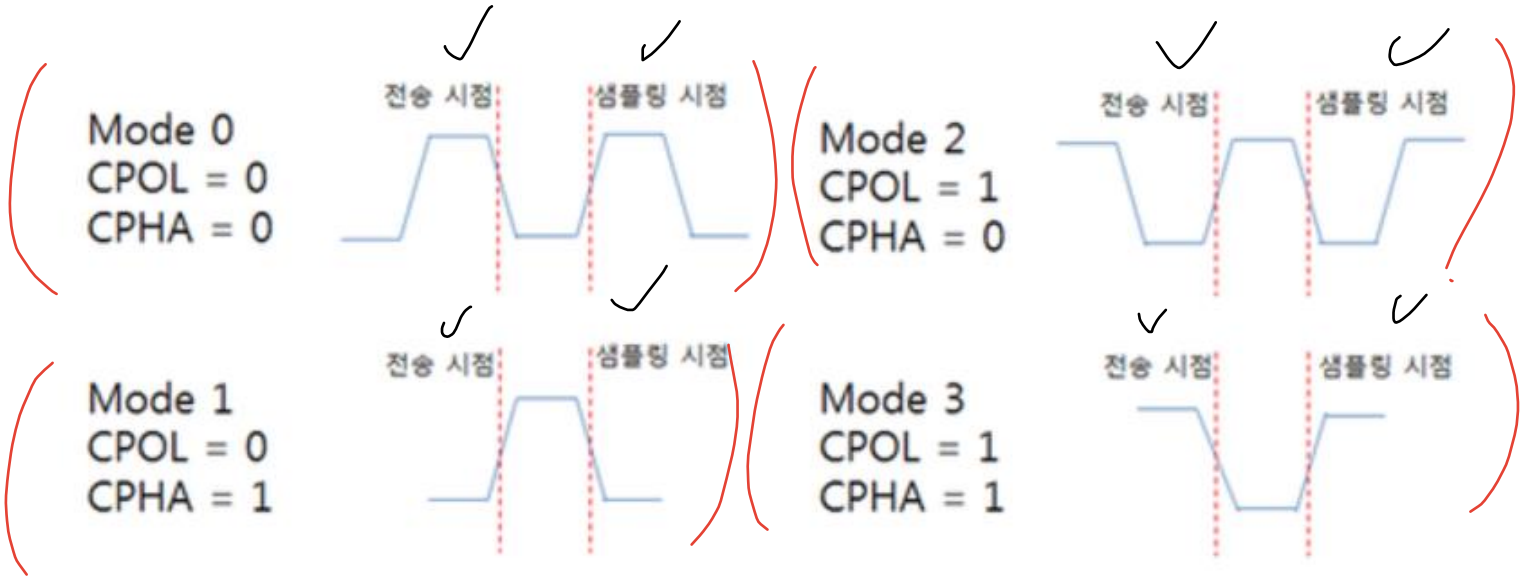
< Transfer timing diagram(CPHA = 0)>



< Transfer timing diagram(CPHA = 1)>

1. SPI 통신

- CPOL과 CPHA



SPI_MODE0	Polarity (CPOL)	0	클록의 상승 에지에서 데이터가 샘플링 되고 하강 에지에서 데이터가 전송된다.
	Phase (CPHA)	0	
SPI_MODE1	Polarity (CPOL)	0	클록의 하강 에지에서 데이터가 샘플링 되고 상승 에지에서 데이터가 전송된다.
	Phase (CPHA)	1	
SPI_MODE2	Polarity (CPOL)	1	클록의 하강 에지에서 데이터가 샘플링 되고 상승 에지에서 데이터가 전송된다.
	Phase (CPHA)	0	
SPI_MODE3	Polarity (CPOL)	1	클록의 상승 에지에서 데이터가 샘플링 되고 하강 에지에서 데이터가 전송된다.
	Phase (CPHA)	1	

- 특징

: Full-duplex, 3-wire 동기 통신
: Master, Slave 모드
: 4~16bit로 통신 데이터 설정 가능
: 1개의 모듈당 8개의 CS

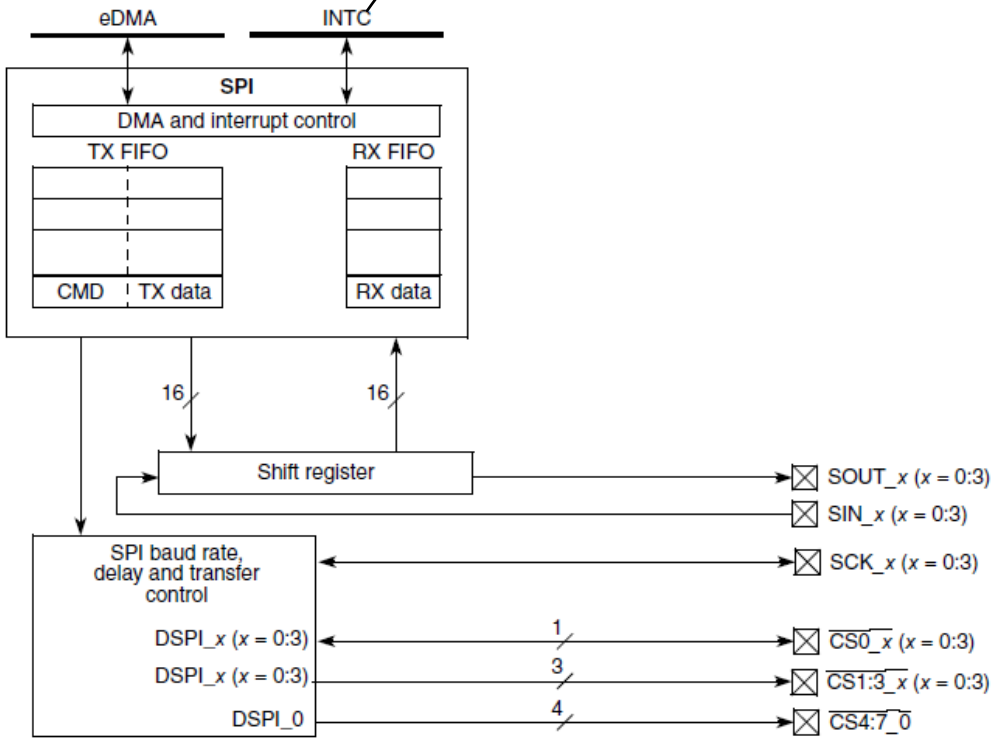
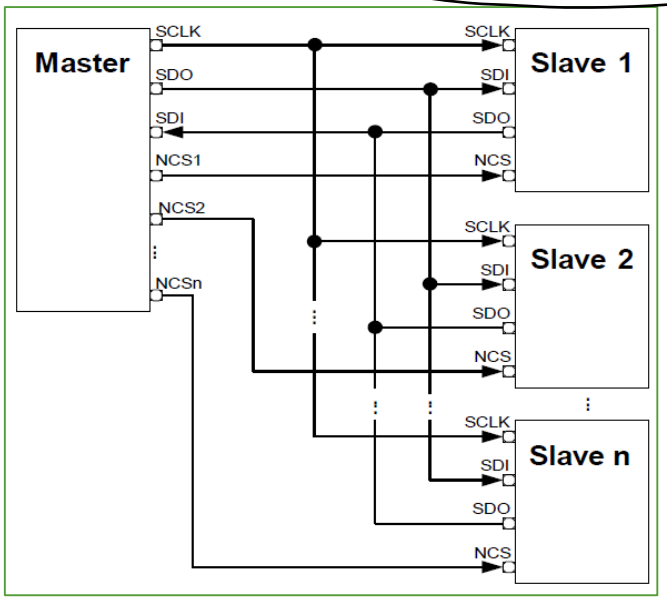
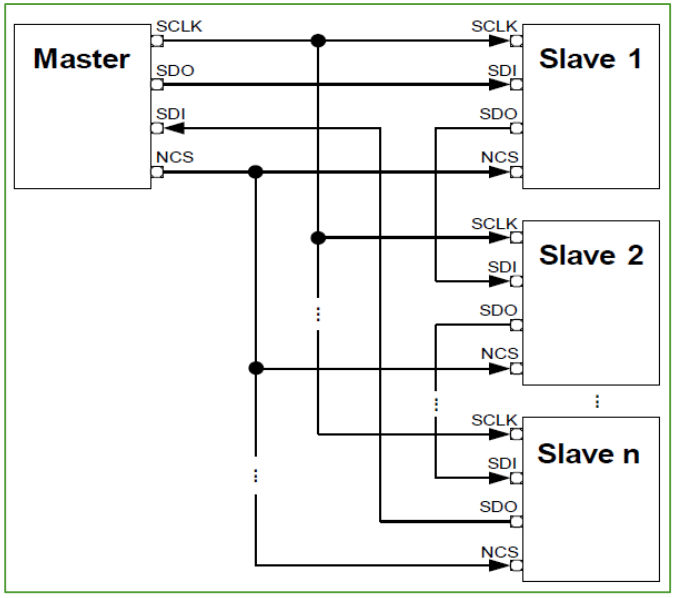


Figure 21-1. DSPI block diagram

- Daisy Chain
  - Slave 장치가 많아질수록 master에 요구되는 SPI CS 라인의 수가 많음
  - Daisy chain 기능이 지원되는 소자는 slave 끼리 연쇄적으로 데이터를 전달하도록 통신
  - 첫 번째 slave가 데이터를 받고 마지막 slave가 데이터를 master로 전달
  - 주로 master에서 slave로 단방향 통신을 할 때 사용하기 좋음
  - 회선 중간 부분에 이상이 생기면 여러 장치가 동작이 안 됨.



< SPI regular bus topology >



< SPI daisy-chain bus topology >

```
void init_DSPI_1(void)
{
#if SPImode
    DSPI_1.MCR.R = 0x80010001;       // Configure DSPI_0 as master
    DSPI_1.CTAR[0].R = 0x7A0A7727;   // Configure CTAR0
                                     //       : SCK inactive state LOW
                                     //       : LSB first
                                     //       : 16bit data frame
                                     //       : Baud rate prescaler 5
                                     //       : Baud Rate Scaler 128
                                     //       : Sclk Duty cycle 50/50
                                     // Fclk : 100kHz
//  DSPI_1.RSER.B.RFDFRE = 1;        // Recieve FIFO drain request enable
    DSPI_1.MCR.B.HALT = 0x0;         // Exit HALT mode: go from STOPPED to RUNNING state

    SIU.PCR[5].R = 0x0604;       // Config pad as DSPI_1 CS0 output
    SIU.PCR[6].R = 0x0604;       // Config pad as DSPI_1 SCK output
    SIU.PCR[7].R = 0x0604;       // Config pad as DSPI_1 SOUT output
    SIU.PCR[8].R = 0x0103;       // Config pad as DSPI_1 SIN input
#else
    DSPI_1.MCR.R = 0x00010001;       // Configure DSPI_0 as slave
    DSPI_1.CTAR[0].R = 0x7A000000;   // Configure CTAR0
                                     //       : SCK inactive state LOW
                                     //       : MSB first
    DSPI_1.RSER.B.RFDFRE = 1;        // Recieve FIFO drain request enable
    DSPI_1.MCR.B.HALT = 0x0;         // Exit HALT mode: go from STOPPED to RUNNING state

    SIU.PCR[5].R = 0x0504;       // Config pad as DSPI_1 CS0 input
    SIU.PCR[6].R = 0x0504;       // Config pad as DSPI_1 SCK input
    SIU.PCR[7].R = 0x0604;       // Config pad as DSPI_1 SOUT output
    SIU.PCR[8].R = 0x0103;       // Config pad as DSPI_1 SIN input

    INTC_InstallINTCInterruptHandler(SlaveDSPI1, 98, 5);
#endif
}
```

- DSPI Module Configuration Registers(DSPIx_MCR)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	MSTR	MONT_SCKE	DCONF[0:1]		FRZ	MTFE	PCSSE	ROOE	PCSIS7	PCSIS6	PCSIS5	PCSIS4	PCSIS3	PCSIS2	PCSIS1	PCSIS0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	0	MDIS	DIS_TXF	DIS_RXF	CLR_TXF	CLR_RXF	SMPL_PT[0:1]		0	0	0	0	0	0	0	HALT
W					w1c	w1c										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Port GPIO Pad Data Output Register 0-3(GPDO0_3)

Field	Description	
MSTR	Master/slave mode select DSPI모듈을 master/slave 모드 선택	1 : Master 모드 0 : Slave 모드
PCSISx	Peripheral chip select inactive state CS의 비활성화 상태 결정	1 : CS0_x가 high일 때 비활성화 0 : CS0_x가 low일 때 비활성화
HALT	Halt DSPI의 전송을 start/stop (Register 설정 시 stop 후 설정)	1 : Stop 0 : Start

- DSPI Clock and Transfer Attributes Registers(DSPIx_CTARn)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	DBR	FMSZ				CPOL	CPHA	LSBFE	PCSSCK		PASC		PDT		PBR	
Reset	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R W	CSSCK				ASC				DT				BR			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Port GPIO Pad Data Output Register 0-3(GPDO0_3)

Field	Description	
FMSZ	Frame Size 통신 data size 설정	데이터 크기 = FMSZ+1(FMSZ>3)

2. DSPI – 데이터 송신의 상태천이도

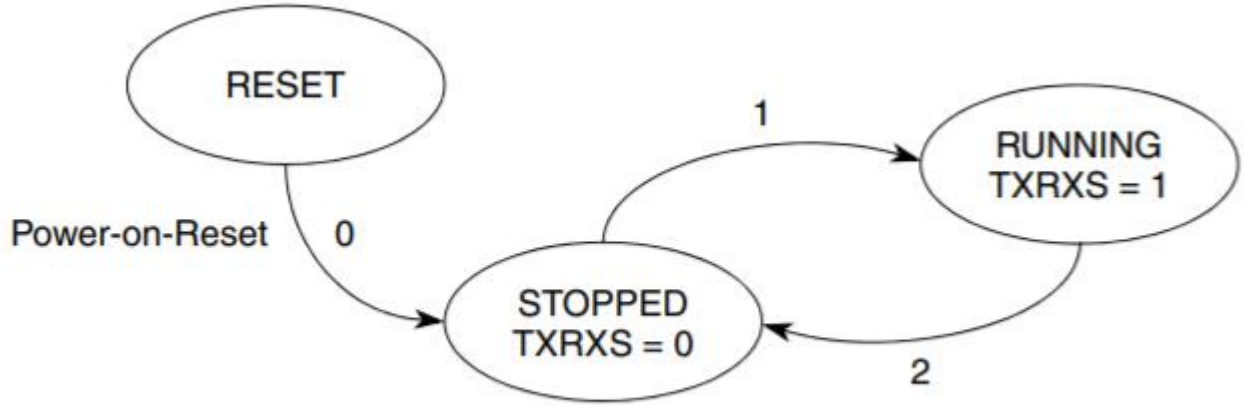


Figure 21-13. DSPI start and stop state diagram

Table 21-18. State transitions for start and stop of DSPI transfers

Transition #	Current State	Next State	Description
0	RESET	STOPPED	Generic power-on-reset transition
1	STOPPED	RUNNING	The DSPI starts (transitions from STOPPED to RUNNING) when all of the following conditions are true: • EOQF bit is clear • Debug mode is unselected or the FRZ bit is clear • HALT bit is clear
2	RUNNING	STOPPED	The DSPI stops (transitions from RUNNING to STOPPED) after the current frame for any one of the following conditions: • EOQF bit is set • Debug mode is selected and the FRZ bit is set • HALT bit is set

2. DSPI – PAD,인터럽트 벡터 세팅

A[5]	PCR[5]	ALT0 ALT1 ALT2 ALT3 —	GPIO[5] CS0 ETC[5] CS7 EIRQ[5]	SIUL DSPI_1 eTimer_1 DSPI_0 SIUL	I/O I/O I/O O I	Slow	Medium	8	14
A[6]	PCR[6]	ALT0 ALT1 ALT2 ALT3 —	GPIO[6] SCK — — EIRQ[6]	SIUL DSPI_1 — — SIUL	I/O I/O — — I	Slow	Medium	2	2
A[7]	PCR[7]	ALT0 ALT1 ALT2 ALT3 —	GPIO[7] SOUT — — EIRQ[7]	SIUL DSPI_1 — — SIUL	I/O O — — I	Slow	Medium	4	10
A[8]	PCR[8]	ALT0 ALT1 ALT2 ALT3 — —	GPIO[8] — — — SIN EIRQ[8]	SIUL — — — DSPI_1 SIUL	I/O — — — I I	Slow	Medium	6	12

DSPI1			
94	0x0978	DSPI_SR[TFUF] DSPI_SR[RFOF]	DSPI 1
95	0x097C	DSPI_SR[EOQF]	DSPI 1
96	0x0980	DSPI_SR[TFFF]	DSPI 1
97	0x0984	DSPI_SR[TCF]	DSPI 1
98	0x0988	DSPI_SR[RFDF]	DSPI 1

2. DSPI – 송수신함수

```
void MasterDSPI1(void)
{
    MasterCnt++;
    DSPI_1.PUSHR.R = (0x00010000|Send_Data);
    Read_Data = (uint16_t)(DSPI_1.POPR.R&0xffff);     /* Read data received by SPI */
    DSPI_1.SR.R = 0x80020000;                         /* Clear TCF, RDRF flags by writing 1 to them */
}

void SlaveDSPI1(void)
{
    SlaveCnt++;
    DSPI_1.PUSHR.R = Send_Data;
    Read_Data = (uint16_t)(DSPI_1.POPR.R&0xffff);     /* Read data received by SPI */
    DSPI_1.SR.R = 0x80020000;                         /* Clear TCF, RDRF flags by writing 1 to them */
}
```

Address Base + 0x0034 Access: User read/write

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CONT	CTAS			EOQ	CTCNT	0	0	PCS7	PCS6	PCS5	PCS4	PCS3	PCS2	PCS1	PCS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	TXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 21-8. DSPI PUSH TX FIFO Register (DSPI*x*_PUSHR)

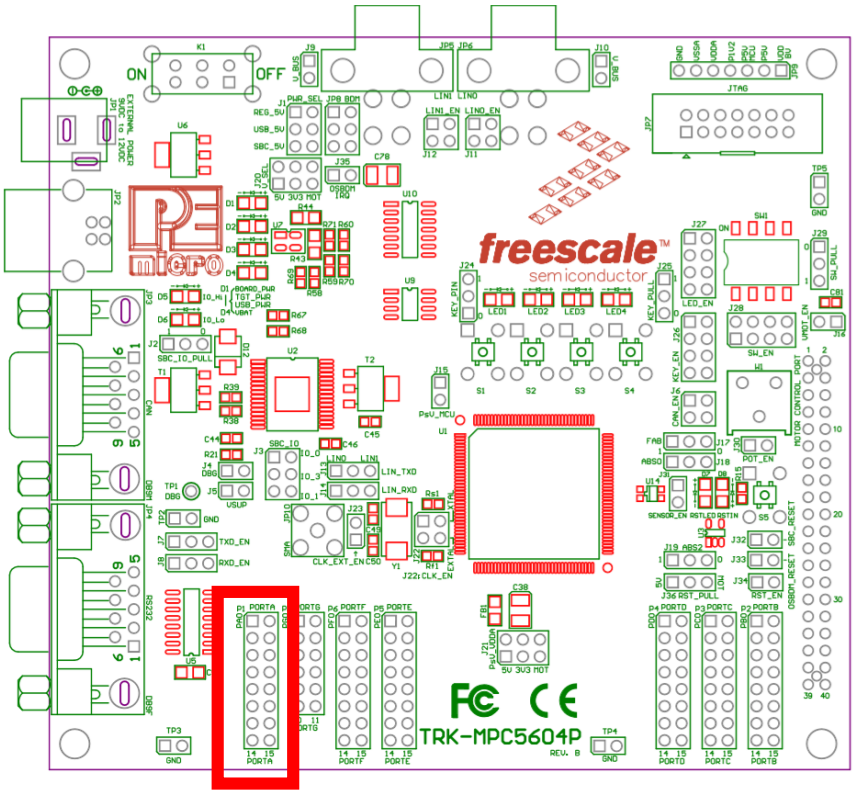
Address: Base + 0x0038 Access: User read-only

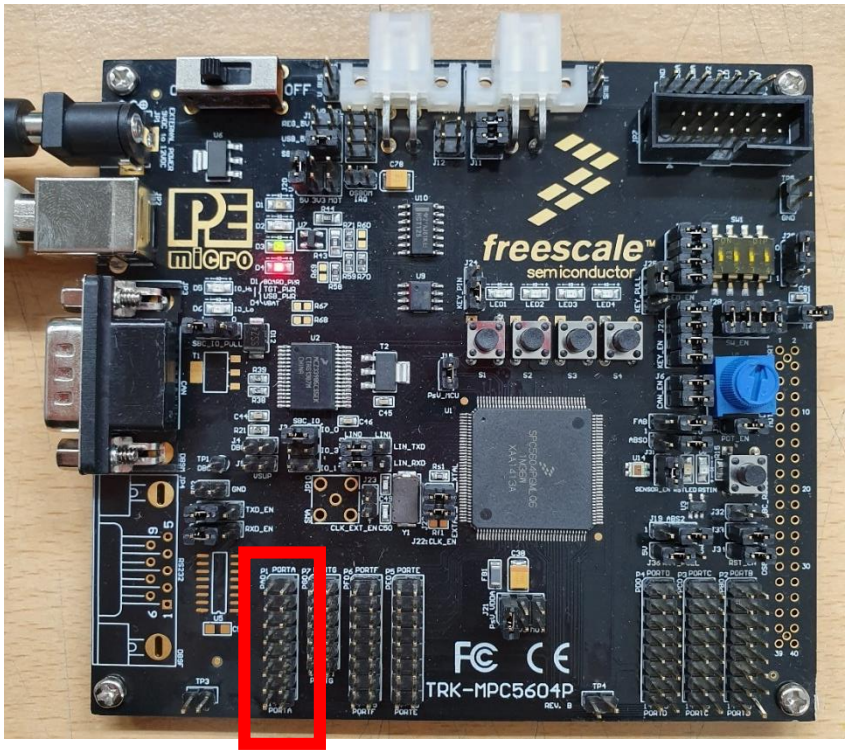
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

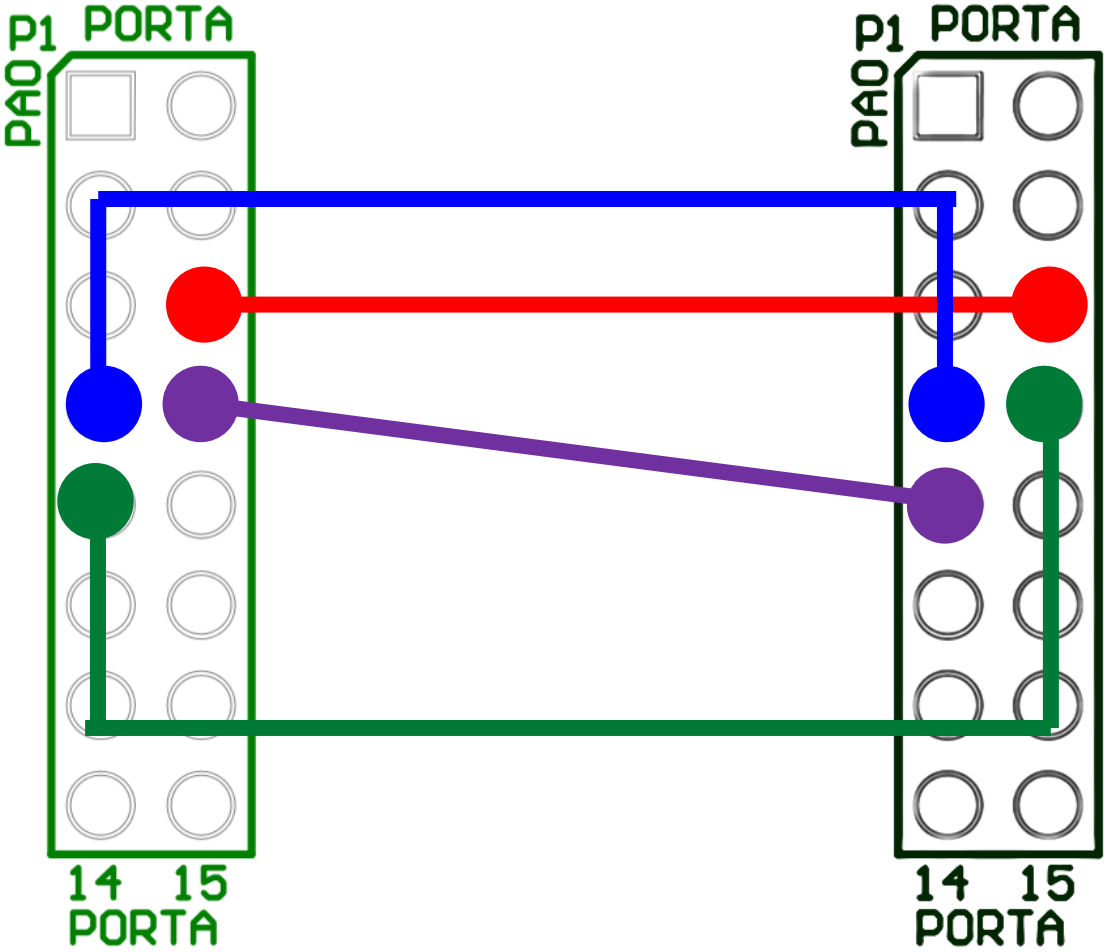
Figure 21-9. DSPI POP RX FIFO Register (DSPI*x*_POPR)

3. HW연결


freescale
semiconductor
TRK-MPC5604P

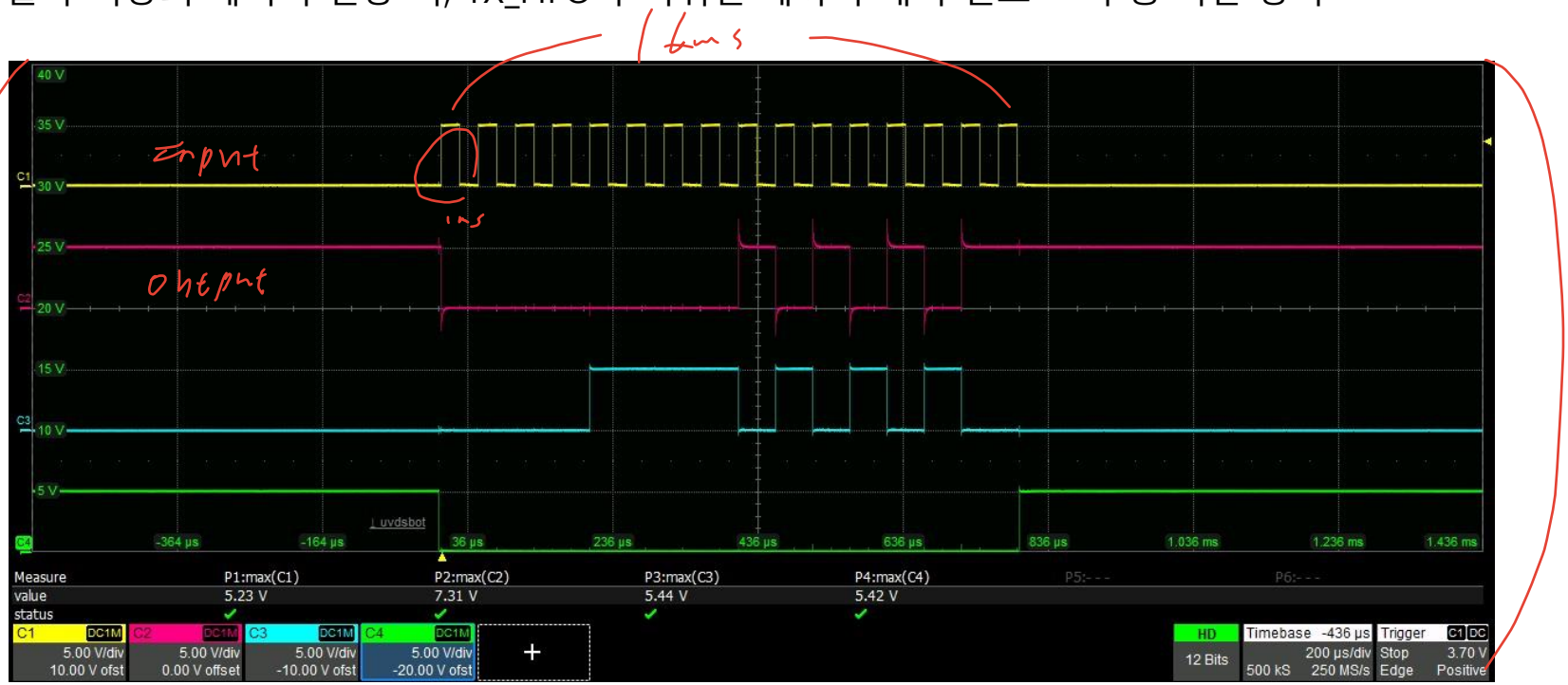

freescale
semiconductor
TRK-MPC5604P

3. HW연결



3. HW연결

- SPI 설정의 SCLK 주파수는 19.53kHz
- SCLK 16 clock + CS ON/OFF delay time = 0.82ms + 0.1ms = 0.92ms
- 데이터 한 블록 당 약 1ms의 시간 소요
- 5 블록 이상의 데이터 전송 시, TX_FIFO가 비워질 때까지 대기 필요 → 수행 시간 증가



3. HW연결

- SPI Clock frequency를 1.042MHz로 선정, Clock 당 주기 1us
- t_{CSC} : 1.92us로 선정
- t_{ASC} : 1.92us로 선정
- t_{DT} : 20.48us로 선정
- 하나의 데이터 블록 당 16 SCLK + t_{CSC} + t_{ASC} + t_{DT} 시간을 소모하므로, 약 40us의 수행시간 소모