AXI기반
SPI & I2C 통신 설계 프로젝트

채민석

CONTENTS

1. 개 요

● 프로젝트 목표

- SPI / I2C 통신 Protocol을 이해하고 AXI4-Lite Interface와 연동 가능한 Module 구현
- 설계한 통신 Module에 대해 Synopsys Tools(VCS, Verdi)를 이용하여 UVM 환경 구성 및 검증

● 개발 환경

SystemVerilog

Vivado

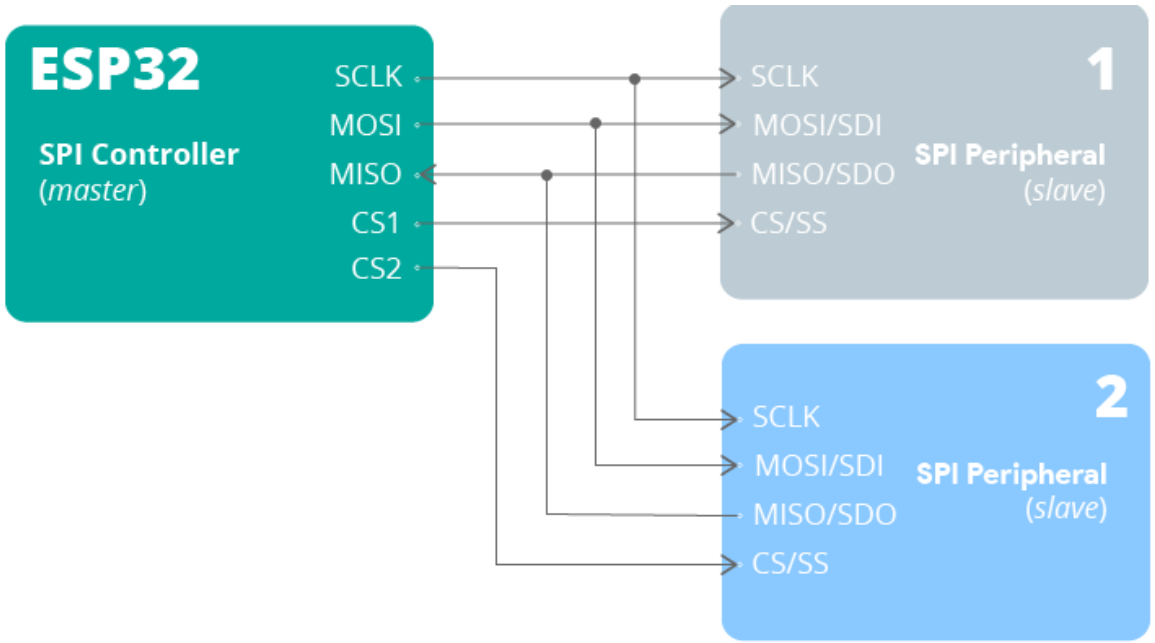
AMBA AXI4-Lite

VCS / Verdi

2. SPI (Serial Peripheral Interface)

● SPI (Serial Peripheral Interface)

- 동기식 직렬 통신 방식 (Full-Duplex 지원)
- 4개의 wire 사용 (SCLK, MOSI, MISO, SS) + Slave마다 SS 추가
- UART, I2C 대비 빠른 속도 Digital 로직 구조 Simple
- Display 패널 및 SD카드에서 주로 사용



SCLK : (Serial Clock)

MOSI : (Master Out Slave In)

MISO : (Master In Slave Out)

SS : (Slave Select)

2. SPI (Serial Peripheral Interface)

● SPI (Serial Peripheral Interface)

CPOL (Clk Polarity) : clk 기본상태 결정 (H/L)

CPHA (Clk Phase) : Data 타이밍 위치 결정

Table 18-2. SPI Modes

SPI Mode	Conditions	Leading Edge	Trailing eDge
0	CPOL=0, CPHA=0	Sample (rising)	Setup (falling)
1	CPOL=0, CPHA=1	Setup (rising)	Sample (falling)
2	CPOL=1, CPHA=0	Sample (falling)	Setup (rising)
3	CPOL=1, CPHA=1	Setup (falling)	Sample (rising)

Figure 18-3. SPI Transfer Format with CPHA=0

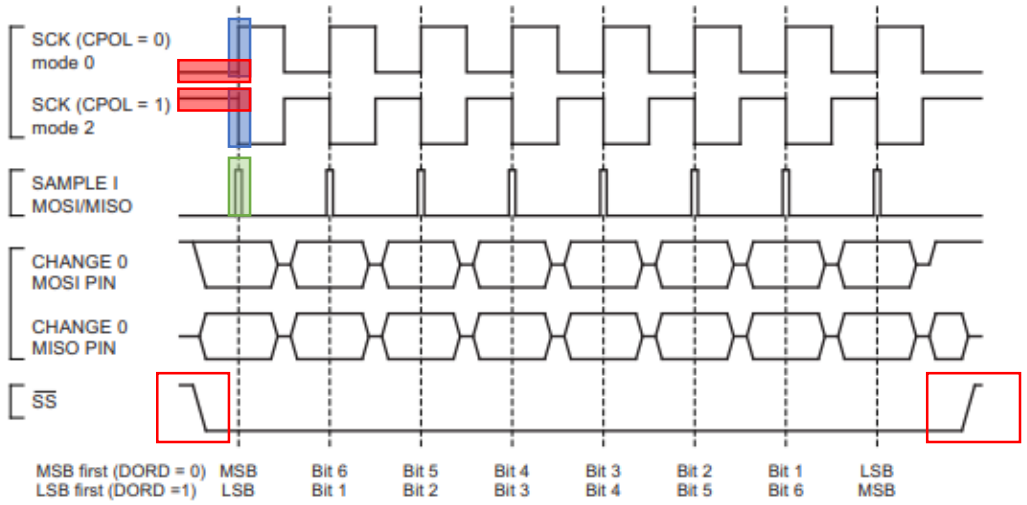
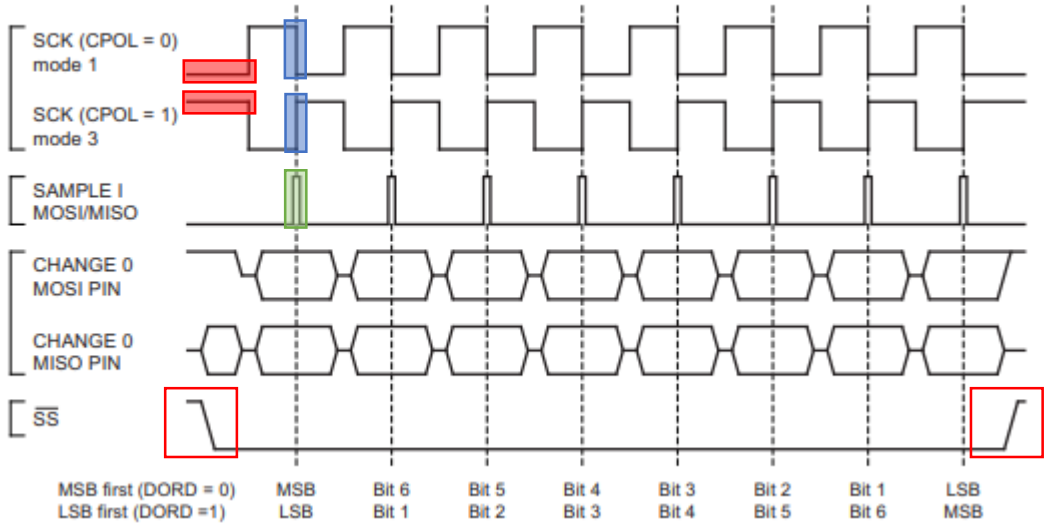
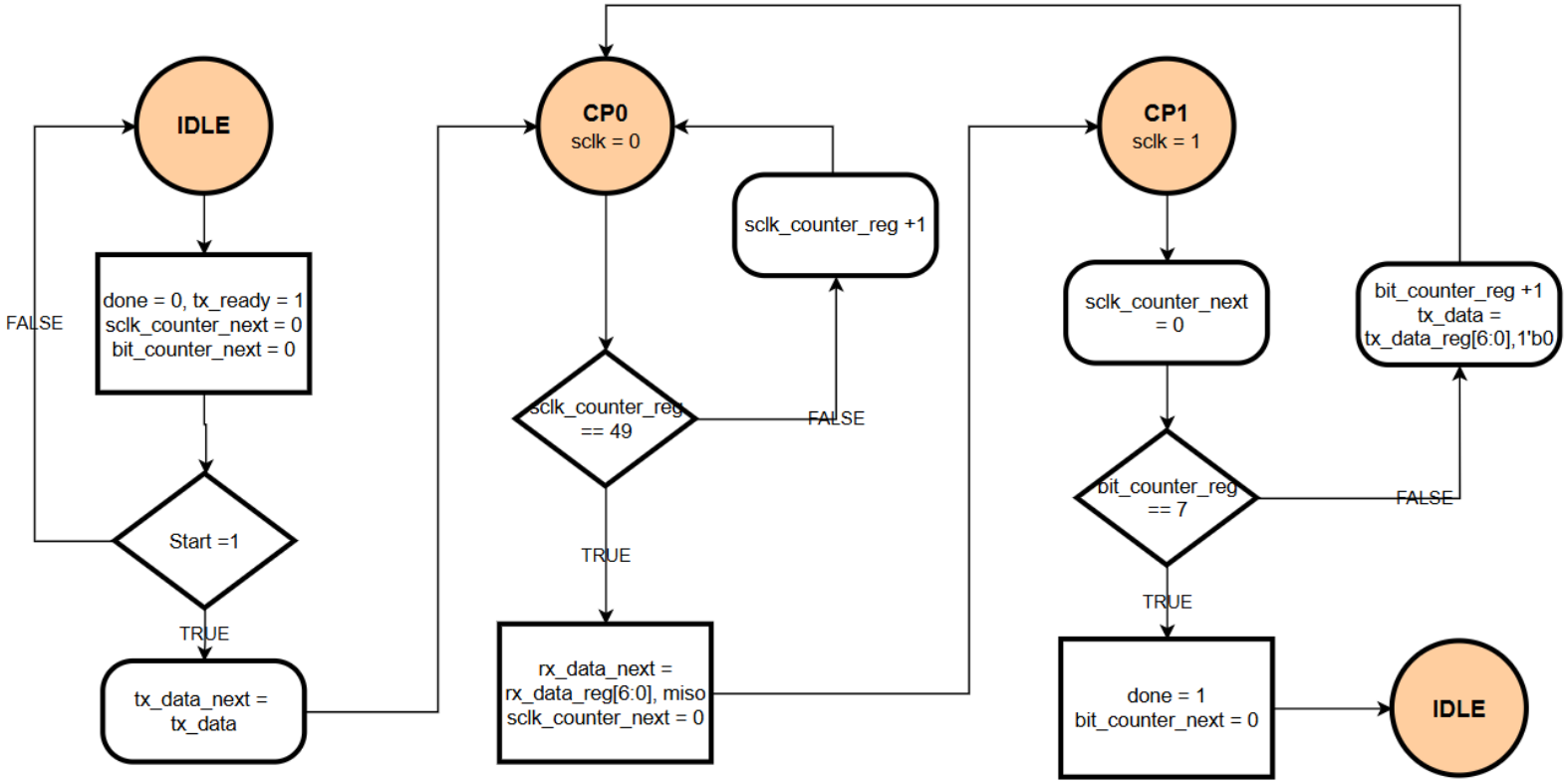


Figure 18-4. SPI Transfer Format with CPHA=1

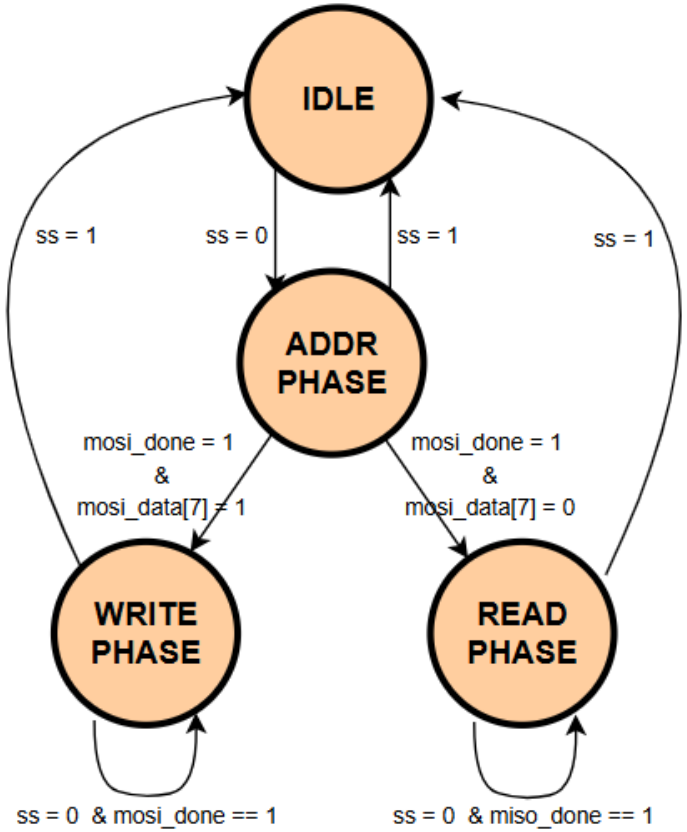


2. SPI ASM/FSM

● SPI Master ASM



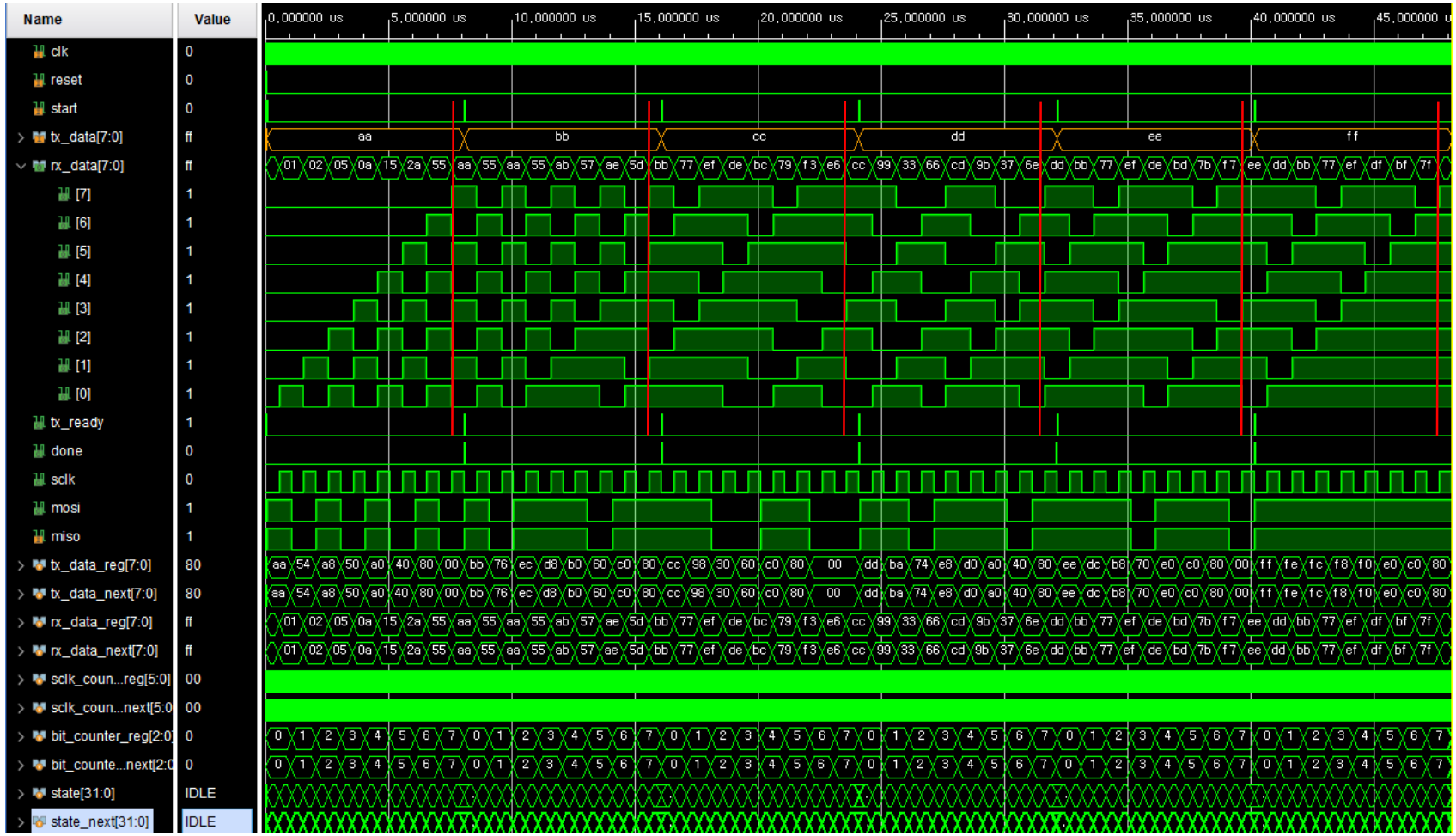
● SPI Slave FSM



2. SPI Simulation

● SPI Simulation

```
initial begin
    repeat (5) @(posedge clk);
    spi_write(8'haa);
    spi_write(8'hbb);
    spi_write(8'hcc);
    spi_write(8'hdd);
    spi_write(8'hee);
    spi_write(8'hff);
    #20;
    $finish;
```



2. SPI UVM

● Verification

```
class spi_seq_item extends uvm_sequence_item;
         bit        cpol;
         bit        cpha;
         bit        start;
    rand bit  [7:0] tx_data;
         bit  [7:0] rx_data;
         bit        done;
         bit        ready;
         bit        SS;

    function new(string name = "spi_ITEM");
        super.new(name);
    endfunction
```

Transaction 생성

- tx_data random 설정

ScoreBoard 비교

tx_data == rx_data

```
class spi_scoreboard extends uvm_scoreboard;
    `uvm_component_utils(spi_scoreboard)

    uvm_analysis_imp #(spi_seq_item, spi_scoreboard) recv;

    spi_seq_item spi_item;

if (spi_item.tx_data == spi_item.rx_data) begin
    `uvm_info("SCB", $sformatf("*** spi TEST PASSED *** tx_data:%0d, rx_data:%0d",
                               spi_item.tx_data, spi_item.rx_data), UVM_LOW);

end
else begin
    `uvm_info("SCB", $sformatf("*** spi TEST FAILED *** Received tx_data:%0d, rx_data:%0d",
                               spi_item.tx_data, spi_item.rx_data), UVM_LOW);

end

        // spi_item.print(uvm_default_line_printer);

        if (spi_item.tx_data == spi_item.rx_data) begin
            `uvm_info("SCB", $sformatf("*** spi TEST PASSED *** tx_data:%0d, rx_data:%0d",
                                       spi_item.tx_data, spi_item.rx_data), UVM_LOW);

        end
        else begin
            `uvm_info("SCB", $sformatf("*** spi TEST FAILED *** Received tx_data:%0d, rx_data:%0d",
                                       spi_item.tx_data, spi_item.rx_data), UVM_LOW);

        end

    endfunction
endclass
```

2. SPI UVM

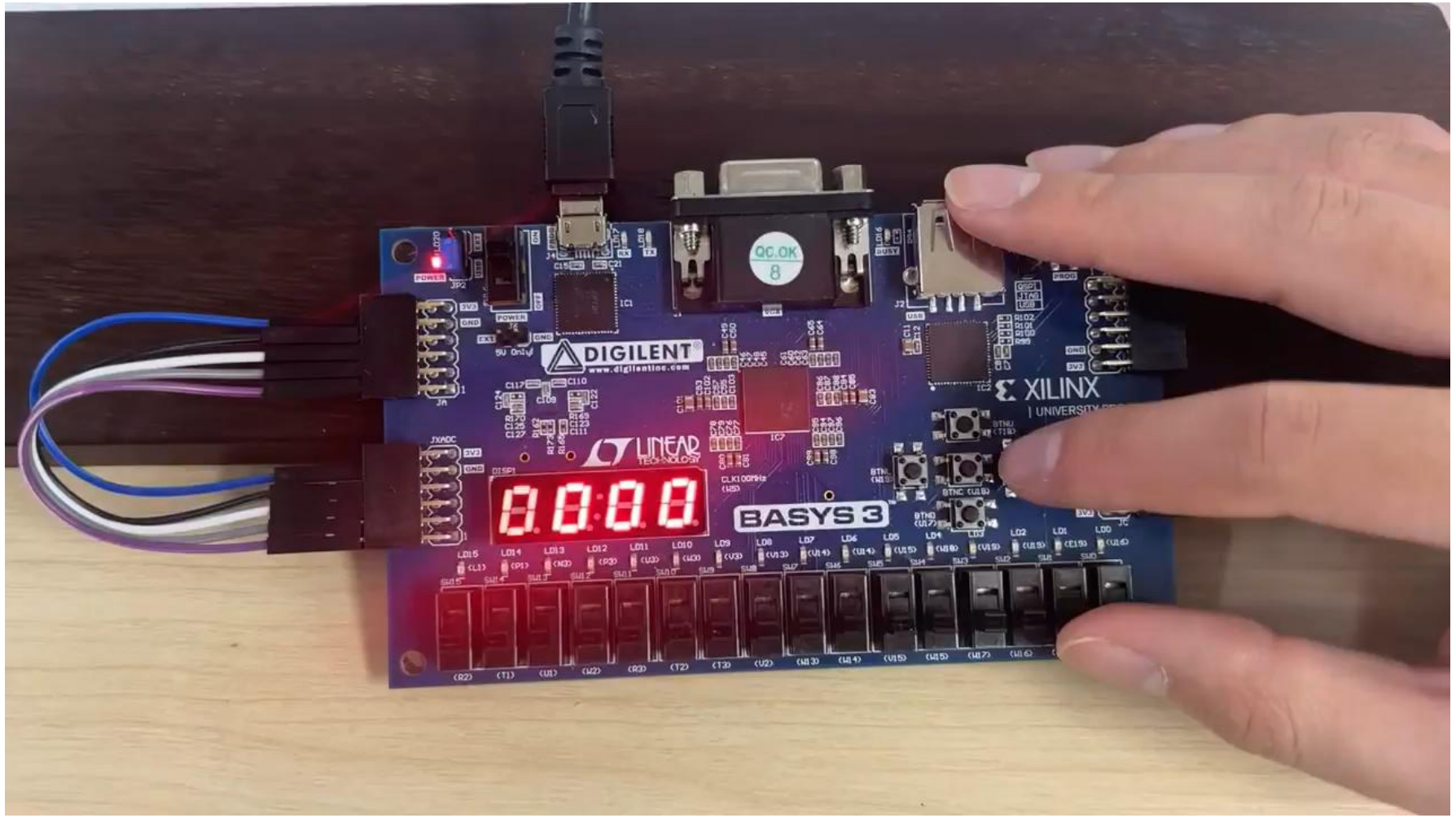
● Verification

```
--- UVM Report Summary ---

** Report counts by severity
UVM_INFO :  104
UVM_WARNING :    0
UVM_ERROR :    0
UVM_FATAL :    0
** Report counts by id
[DRV]     10
[MON]     40
[RNTST]      1
[SCB]     40
[SEQ]     10
[TEST_DONE]      1
[UVM/RELNOTES]      1
[UVM/REPORT/CATCHER]      1
```

```
) @ 3000455000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:191, done:0, ready:1
) @ 3000455000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:0, rx_data:0
) @ 3008475000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:0, done:0, ready:1
) @ 3008475000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** tx_data:0, rx_data:0
) @ 3016485000: uvm_test_top.ENV.AGT.DRV [DRV] Drive DUT cpol:0, cpha:0, start:0, tx_data:180, rx_data:0, done:0, ready:0
) @ 3016495000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:0, tx_data:180, rx_data:0,  , done:0, ready:1
) @ 3016495000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:180, rx_data:180
@ 3016495000: uvm_test_top.ENV.AGT.SQR@@SEQ [SEQ] spi item to driver cpol:0, cpha:0, start:0, tx_data:89, rx_data:0, done:0, ready:0
) @ 3024525000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:89, rx_data:89, done:0, ready:1
) @ 3024525000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:89, rx_data:89
) @ 3032545000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:52, done:0, ready:1
) @ 3032545000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:52,  rx_data:52
) @ 3040565000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:0, done:0, ready:1
) @ 3040565000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** tx_data:0, rx_data:0
) @ 3048575000: uvm_test_top.ENV.AGT.DRV [DRV] Drive DUT cpol:0, cpha:0, start:0, tx_data:89, rx_data:0, done:0, ready:0
) @ 3048585000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:0, tx_data:89, rx_data:89, done:0, ready:1
) @ 3048585000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** tx_data:89, rx_data:89
@ 3048585000: uvm_test_top.ENV.AGT.SQR@@SEQ [SEQ] spi item to driver cpol:0, cpha:0, start:0, tx_data:210, rx_data:0, done:0, ready:0
) @ 3056615000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:210, rx_data:89, done:0, ready:1
) @ 3056615000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:210, rx_data:210
) @ 3064635000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:89, done:0, ready:1
) @ 3064635000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:89,rx_data:89
) @ 3072655000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:255, rx_data:255, done:0, ready:1
) @ 3072655000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:255, rx_data:255
) @ 3080665000: uvm_test_top.ENV.AGT.DRV [DRV] Drive DUT cpol:0, cpha:0, start:0, tx_data:210, rx_data:0, done:0, ready:0
) @ 3080675000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:0, tx_data:210, rx_data:210, done:0, ready:1
) @ 3080675000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** tx_data:210, rx_data:210
@ 3080675000: uvm_test_top.ENV.AGT.SQR@@SEQ [SEQ] spi item to driver cpol:0, cpha:0, start:0, tx_data:96, rx_data:0, done:0, ready:0
) @ 3088705000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:96, rx_data:2196, done:0, ready:1
) @ 3088705000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:96, rx_data:96
) @ 3096725000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:0,  done:0, ready:1
) @ 3096725000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:0, rx_data:0
) @ 3104745000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:255, rx_data:255, done:0, ready:1
) @ 3104745000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:255, rx_data:255
) @ 3112755000: uvm_test_top.ENV.AGT.DRV [DRV] Drive DUT cpol:0, cpha:0, start:0, tx_data:96, rx_data:0, done:0, ready:0
) @ 3112765000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:0, tx_data:96, rx_data:96, done:0, ready:1
) @ 3112765000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:96, rx_data:96
@ 3112765000: uvm_test_top.ENV.AGT.SQR@@SEQ [SEQ] spi item to driver cpol:0, cpha:0, start:0, tx_data:170, rx_data:0, done:0, ready:0
) @ 3120795000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:170, rx_data:170, done:0, ready:1
) @ 3120795000: uvm_test_top.ENV.SCB [SCB] *** spi TEST PASSED *** Received tx_data:170, rx_data:170
) @ 3128815000: uvm_test_top.ENV.AGT.MON [MON] sampled cpol:0, cpha:0, start:1, tx_data:0, rx_data:224, done:0, ready:1
```

2. SPI 동작영상

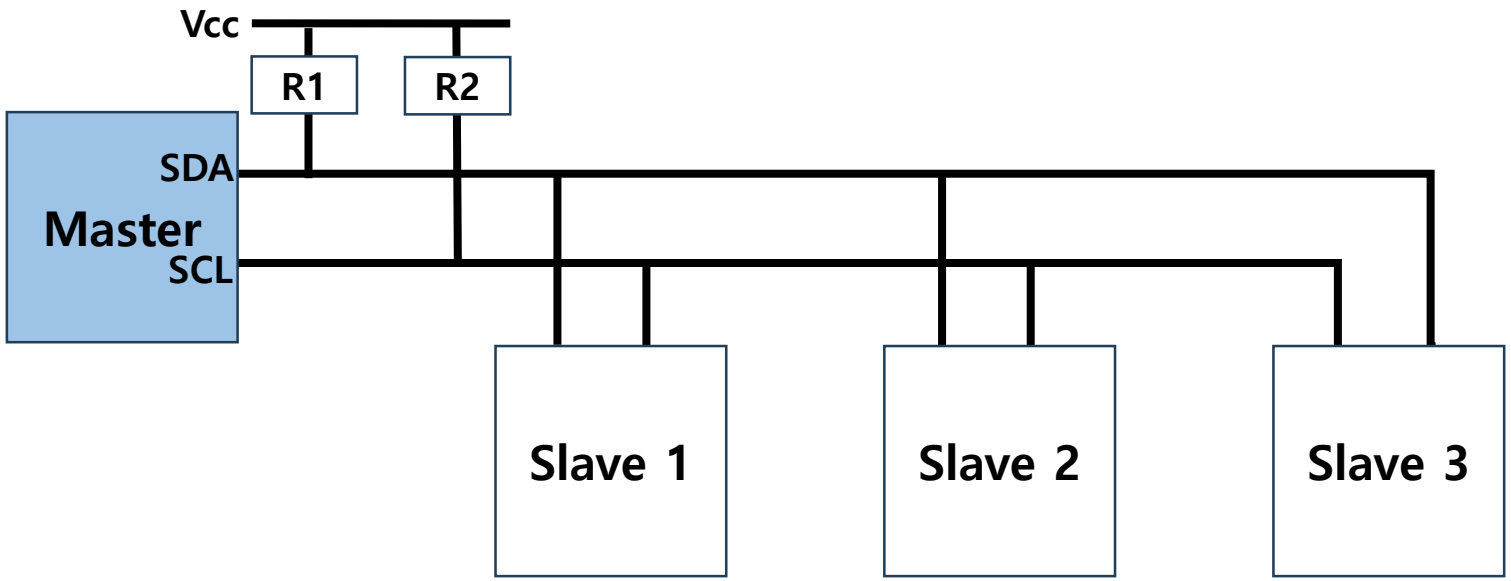
● 동작영상



3. I2C (Inter-Integrated Circuit)

● I2C (Inter-Integrated Circuit)

- 동기식 직렬 통신 방식 (Half-Duplex 지원) Open Dain 구조
- 2개의 wire 사용 (SDA, SCL) + 주소기반으로 다중 Slave 연결
- SPI 대비 속도가 느리고 회로설계가 복잡함
- 여러 저속 센서를 제어에 사용

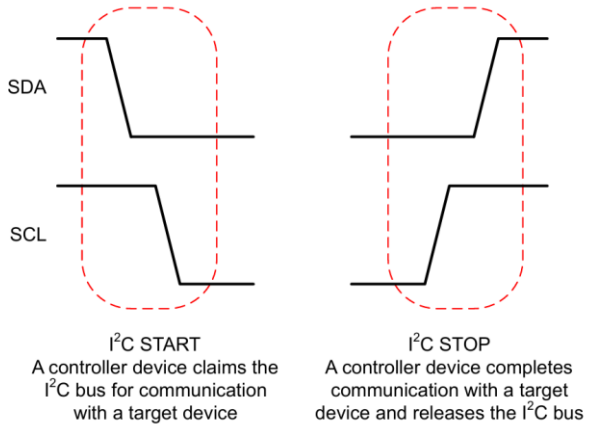


● I2C의 특징

Open-Drain구조의 N-MOS출력으로 구성
Pull-Up저항을 통해 기본 논리 상태 1지정
필요시 N-MOS를 켜서 GND로 당김

3. I2C Protocol

● I2C Protocol



초기상태 : SDA = H SCL = H
START : SDA = H > L SCL = H
STOP : SDA = L > H SCL = H

SCL이 High 일때 SDA는 안정해야한다. (무조건 H/L 유지)
SCL이 L일때만 DATA를 바꿀수 있음.

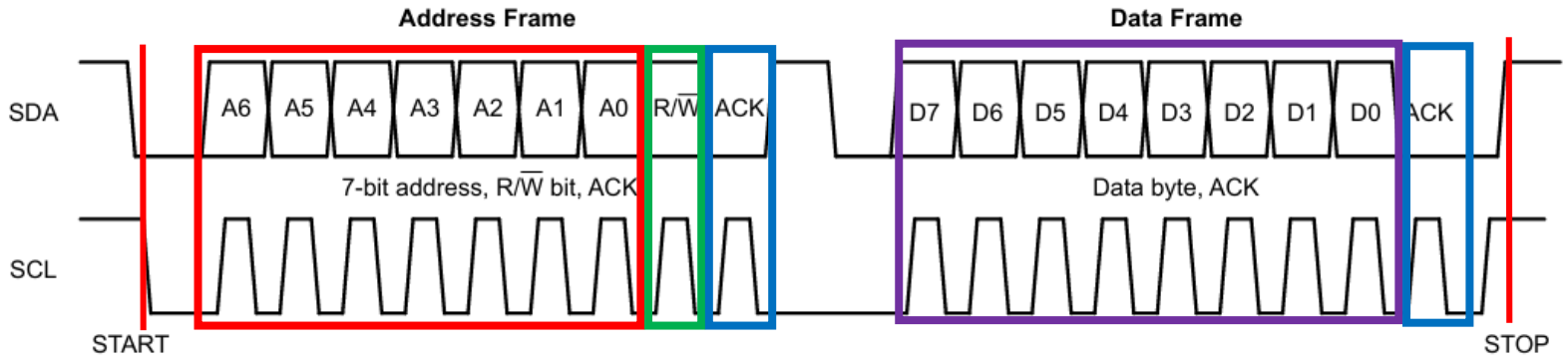
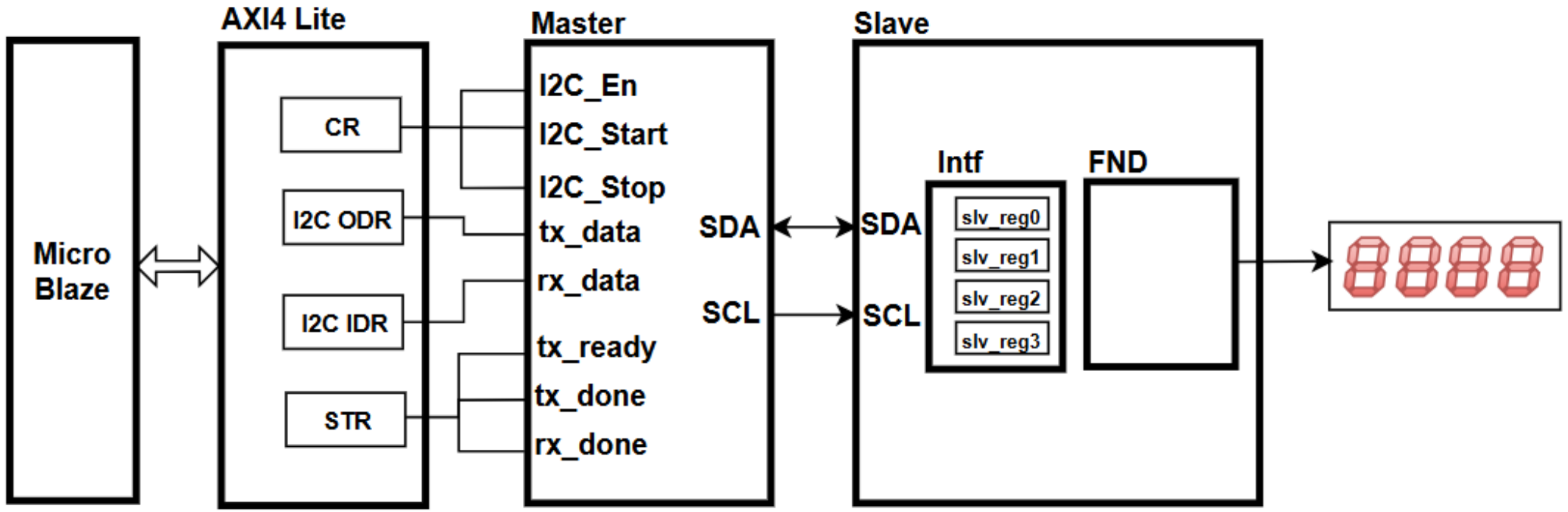


Figure 3-3. I²C Address and Data Frames

ADDR : 주소 7BIT로 Slave 선택
R/W : Read / Write 신호 (1/0)
ACK : Data 수신 여부 확인
DATA : 8 BIT

3. I2C BlockDiagram

● I2C BlockDiagram



3. I2C (Inter-Integrated Circuit)

● I2C Master ASM

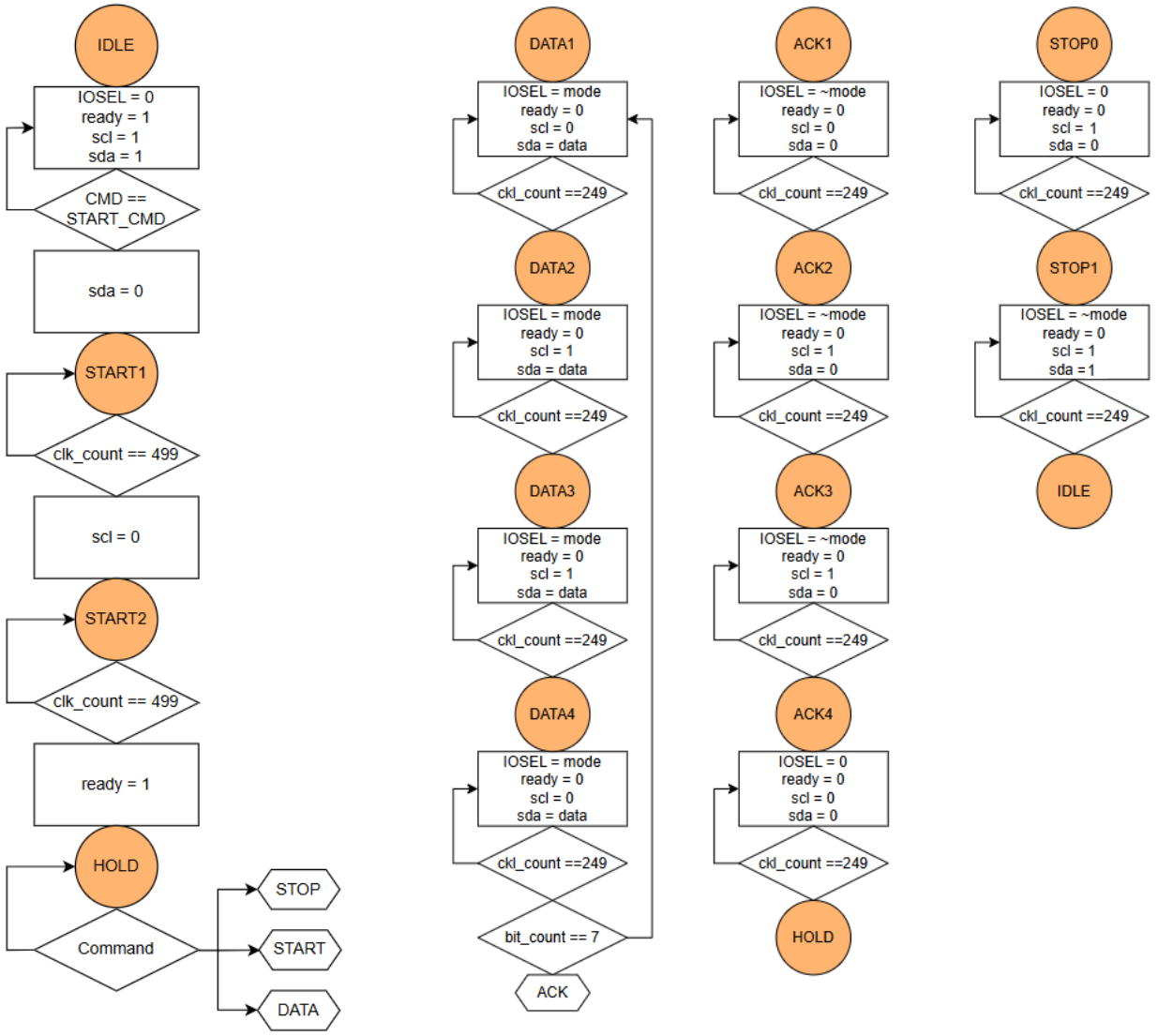
IDLE / START
- 값 상태 초기화
- SDA = 0으로 START

HOLD
- command에 따라 상태변화

DATA (write = master sda, read = slave sda)
- DATA1 값 준비
- DATA2,3 SCL = 1로 DATA 유지
- DATA4 SCL = 0으로 떨어지고 bit 전송마무리

ACK
- write 일때는 slave로부터 ack받음
- read는 master가 보냄
- SDA : 0 이면 상태 지속, 1이면 write/read 종료

STOP
- scl = 1 일때 sda 0 > 1로 바꾸어
  IDLE 상태로 회귀



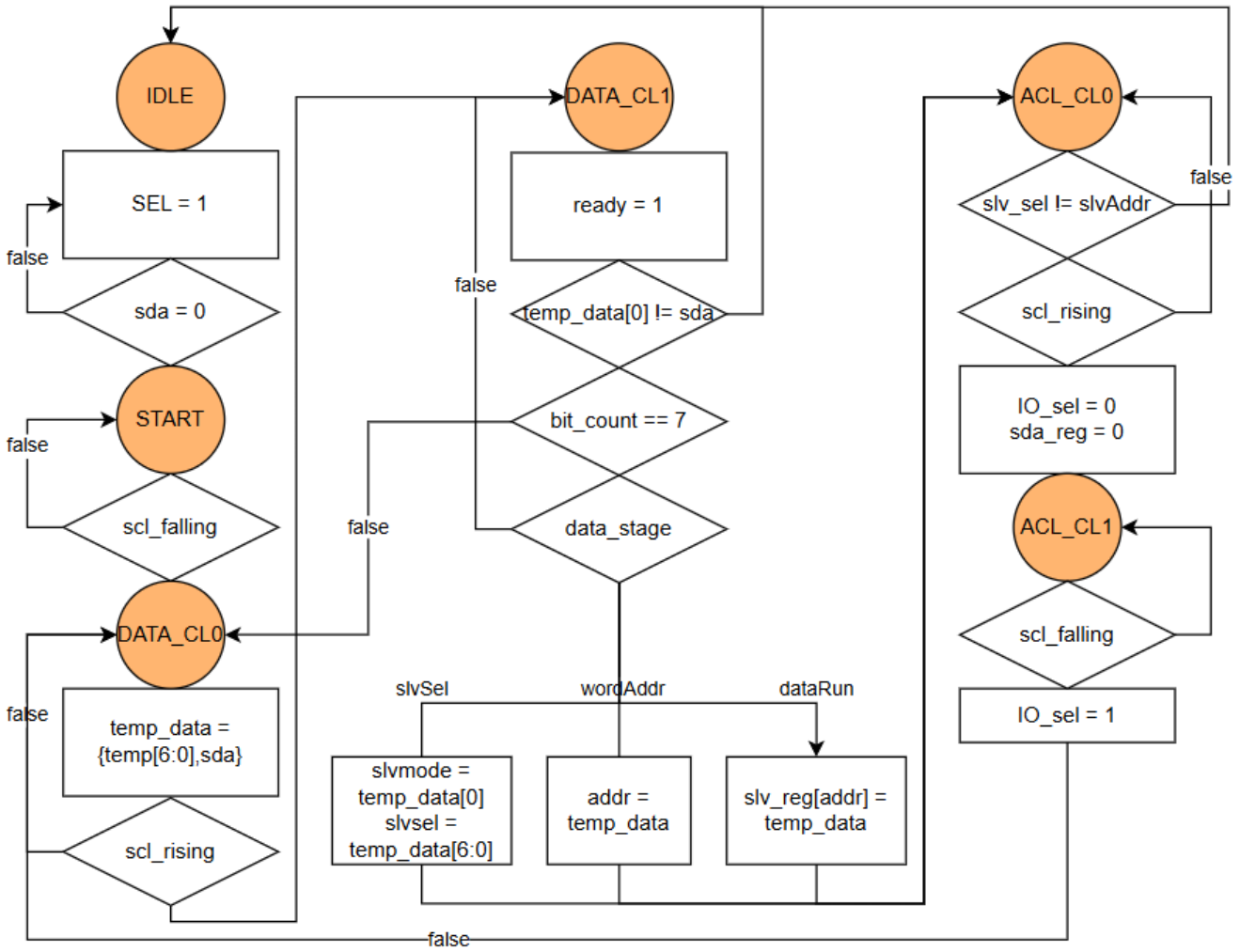
3. I2C (Inter-Integrated Circuit)

● I2C Slave ASM

Slave_Sel
- slave 주소 + r/w bit

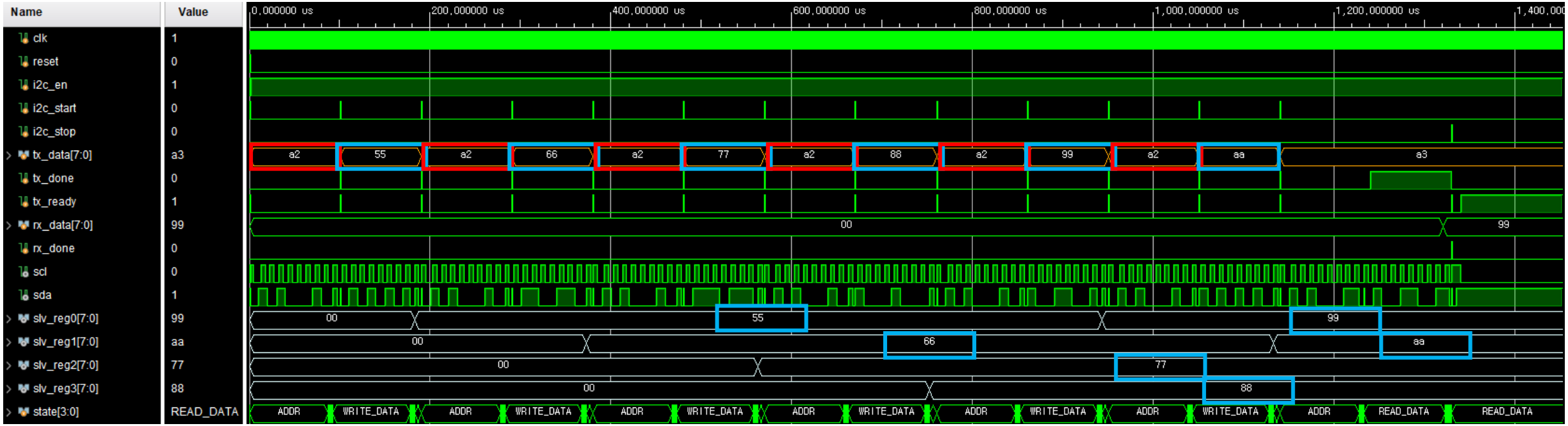
Word_Addr
- 내부 레지스터 주소 지정

Data Run
- 레지스터에 데이터 저장
- 세부구조상 addr를 증가시키면서 순차적으로 Data 저장



3. I2C (Inter-Integrated Circuit)

● Simulation




```
write(8'hA2, 8'h55);
write(8'hA2, 8'h66);
write(8'hA2, 8'h77);
write(8'hA2, 8'h88);
write(8'hA2, 8'h99);
write(8'hA2, 8'hAA);

read(8'hA3);
```

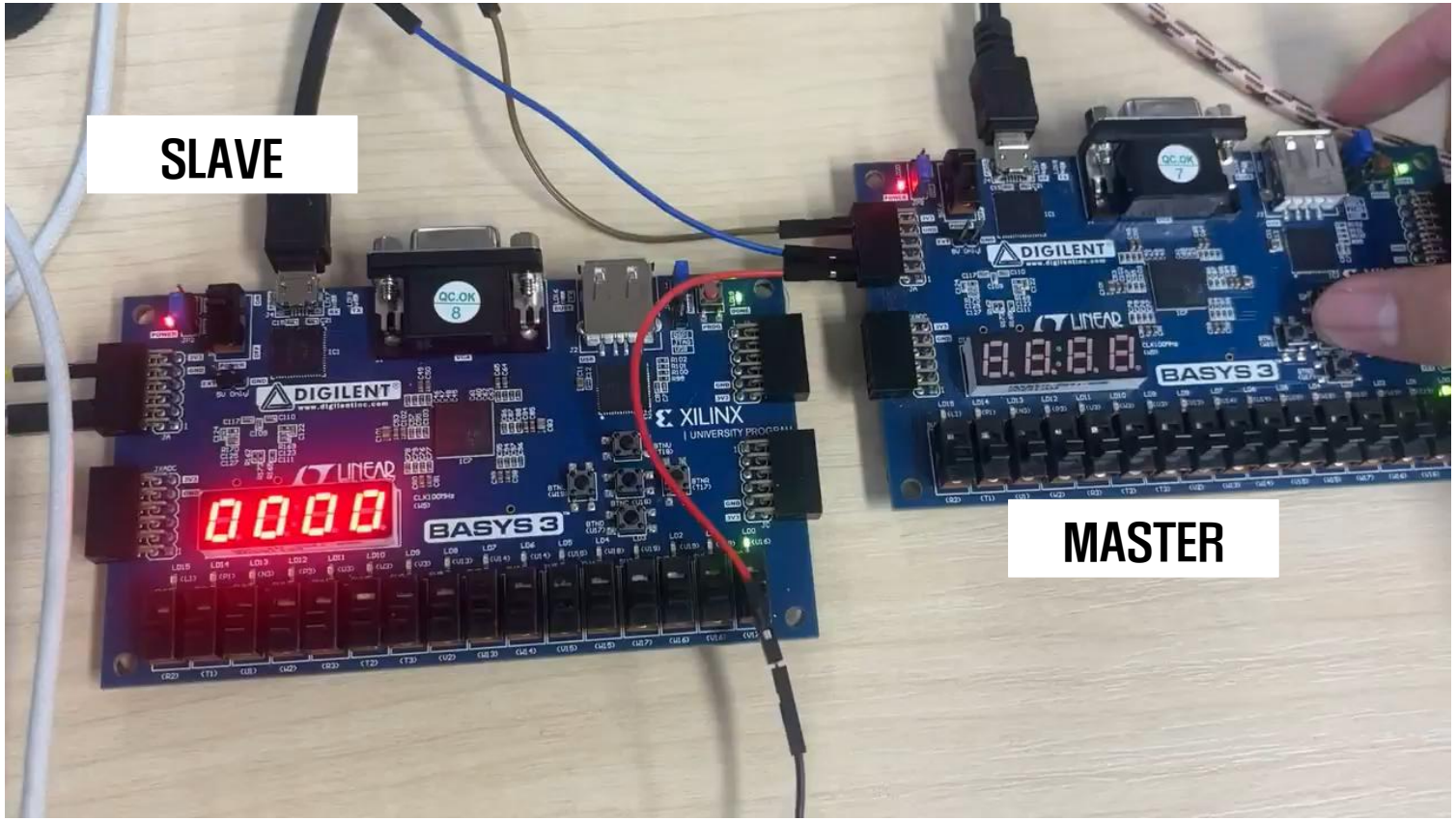
3. I2C 동작영상

● C Application 및 동작영상

C Application

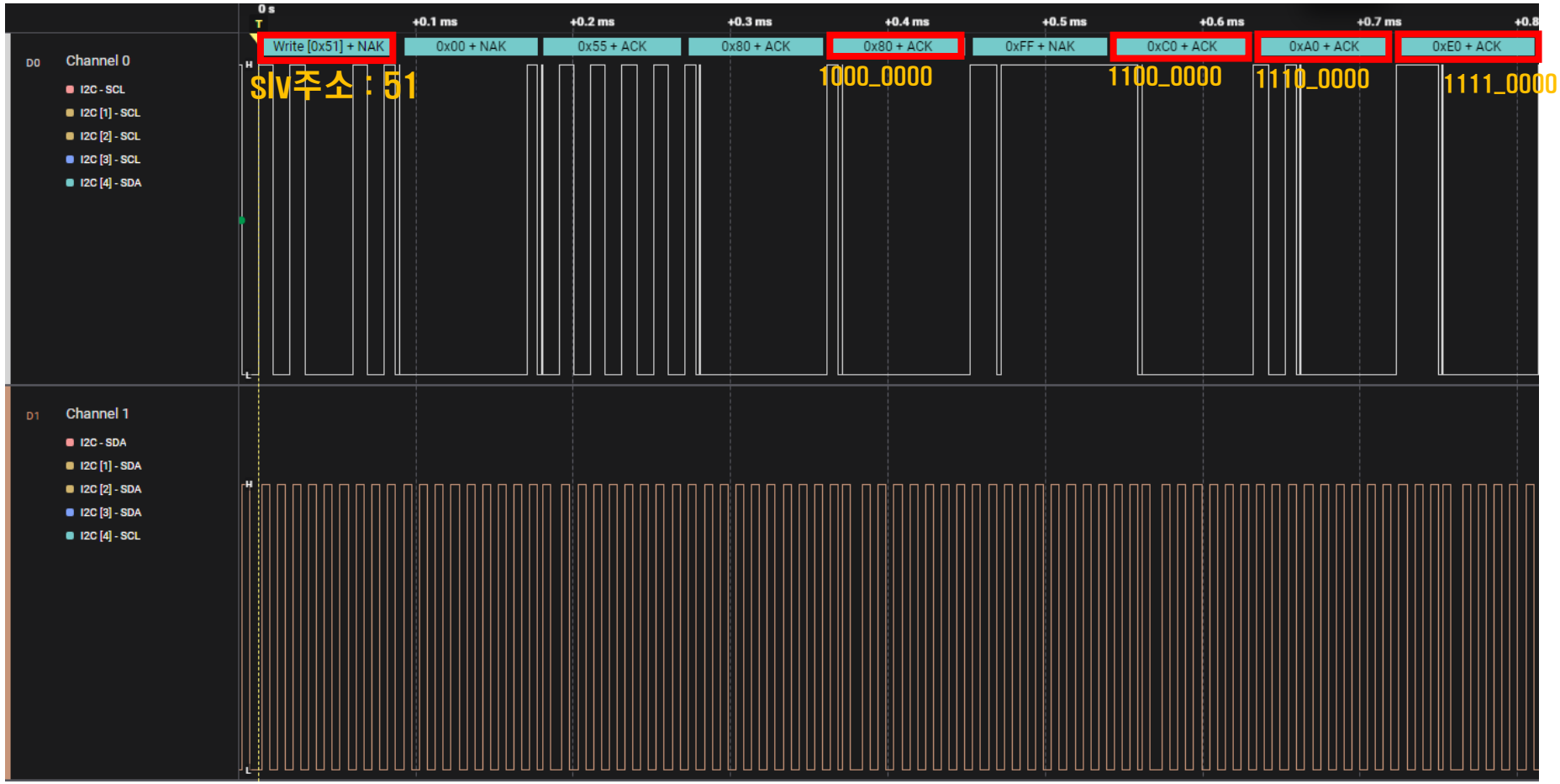
- tx_data random 설정

```c
int main() {
    init_platform();

    uint32_t target_slaveNum = 0b00000000;
    uint32_t start_regAddr   = 0b00000000;

    start_transmission(IIC, target_slaveNum, start_regAddr);
    print("Start Transmission!\n");

    for (int i = 0; i < 100; i++) {
        write_IIC(IIC, i);
        usleep(1000000);
    }
    stop_transmission(IIC);
    print("Stop Transmission!\n");

    cleanup_platform();
    return 0;
}
```



3. I2C Logic Analyzer

● Logic Analyzer



4. Trouble shooting

● Master에서 Addr 이후 NACK 나오는 증상

Master가 주소를 보내고 난 뒤 NACK STATE가 나오는 현상
WaveForm에서 SDA가 ACK에서 High로만 유지

● 해결 방법

Master에서 ACK를 받을 때는 SEL = 0 으로 두고 SDA = High로 둠
SLAVE에서 SCL이 High 일때만 SDA drive 신호를 주어 0으로 당김.

5. 고 찰

- 이번 프로젝트를 통해 I2C 프로토콜과 UVM 구조를 충분히 이해하지 못하면 설계시에 어려움을 겪음
- 현재 I2C Read가 홀수 번에서만 동작하는 현상이 발생, 해당 부분을 다시 정리하고 수정 예정
- 추후 I2C Read/Write 시퀀스를 규격 기준으로 다시 정리하고, UVM 테스트를 재구성하여 현재 Read 버그를 해결하고 검증 신뢰도를 높일 예정