TI DSP, MCU, Xilinx Zynq FPGA
프로그래밍 전문가 과정

10주차 발표

2018.09.13

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

01 진행상황 및 문제점

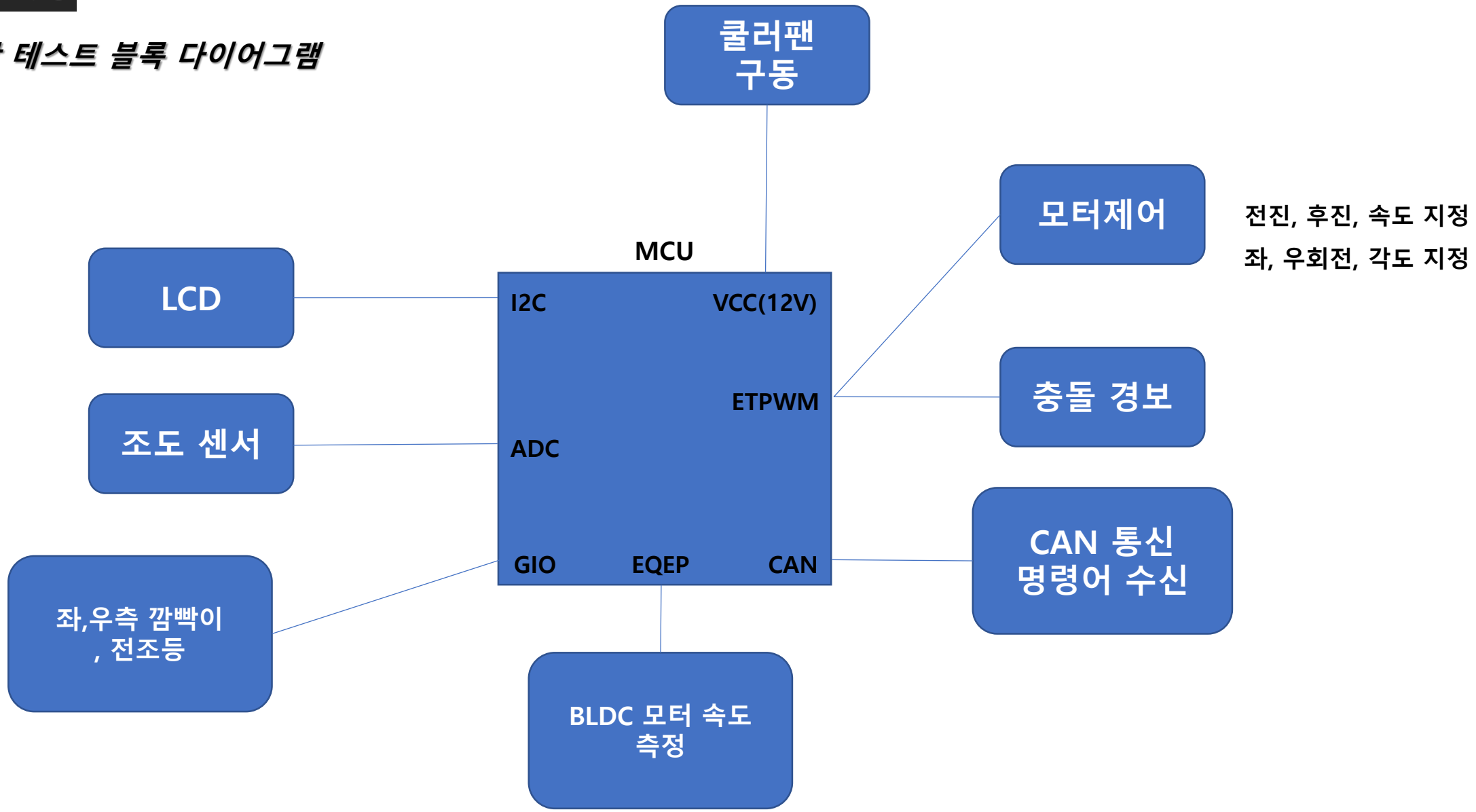
안상재

1. MCU 통합 테스트

2. IR2110 테스트

MCU 통합 테스트

**** 통합 테스트 블록 다이어그램***



**** RTOS 통합 소스 코드 현황***

CAN 인터럽트 서비스 루틴

CAN 통신 – 명령어 데이터 수신

Task1 → **모터 제어 - 전진, 후진, 속도 지정**
좌,우회전, 각도 지정

Task2 → **EQEP**

Task3 → **충돌 경보 (사이렌)**

Task4 → **조도센서값 수신, 좌/우측 LED, 전조등**

Task5

* 현재 사용 중인 MCU Peripheral 및 핀 정리

*** ADC1 Group1 (전조등)**
- Pin0 : 조도센서 (전조등)

*** ADC2 Group1 (가속도 페달)**
- Pin0 : 압력센서 (가속도 페달)

*** ETPWM**
- ETPWM1A : 사이렌
- ETPWM2B : 차 뒷바퀴(BLDC) 제어
- ETPWM3B : 차 앞바퀴(서보모터) 제어

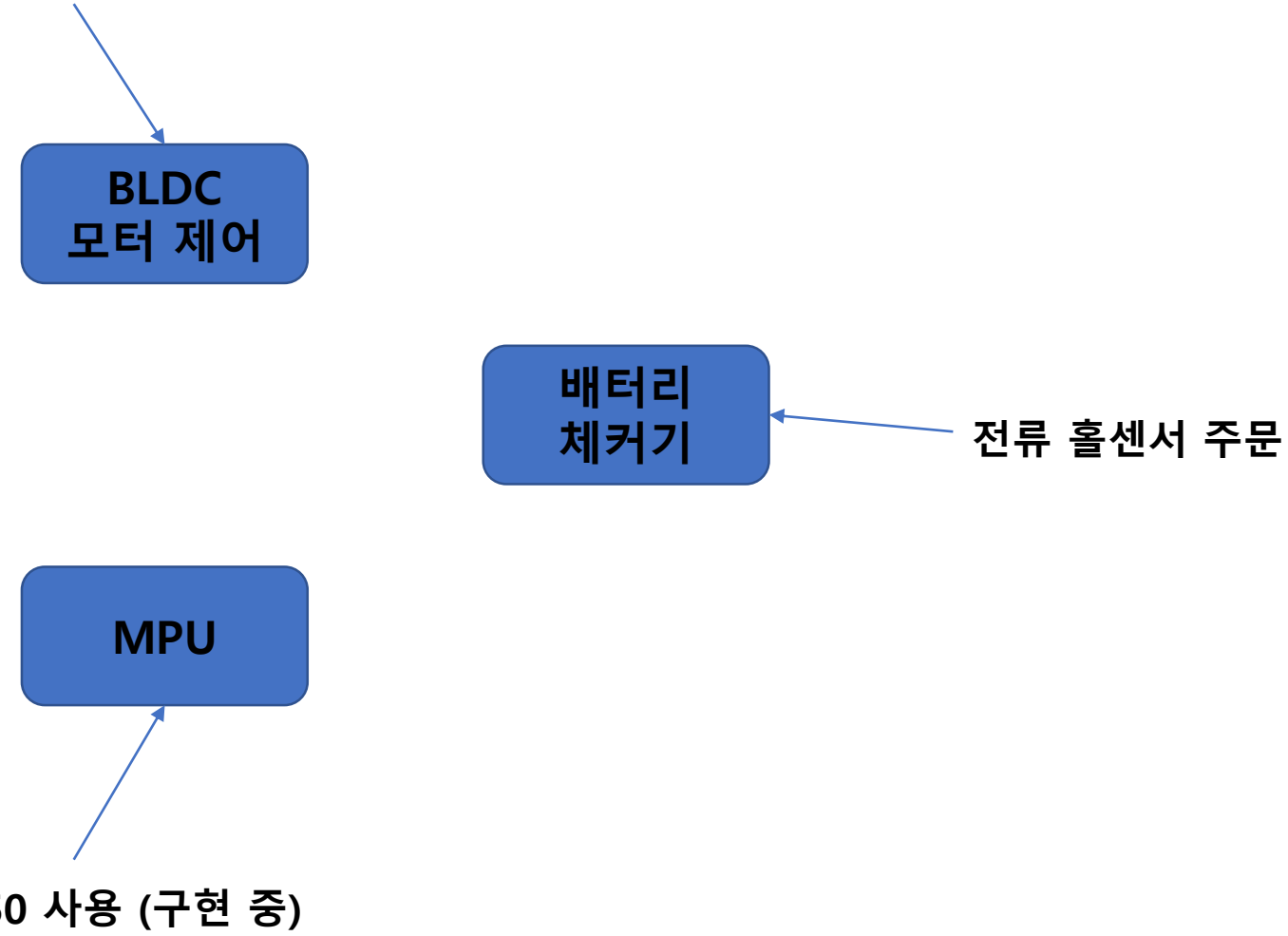
*** GIO**
- gioPORTB0 : ADC 트리거용 (ADC1, ADC2) - (전조등, 가속도 페달)
- gioPORTB1 : 조도센서 밝기에 따른 전조등 제어(TR 베이스) 핀
- gioPORTB2 : 조도센서 VCC 용
- gioPORTB3 : 좌측 깜빡이
- gioPORTB4 : 우측 깜빡이
- gioPORTB5 : 가감속 상태 확인등

*** CAN**
- CAN1 : DSP->MCU 프로토콜

*** I2C**
-I2C2 : LCD 제어용

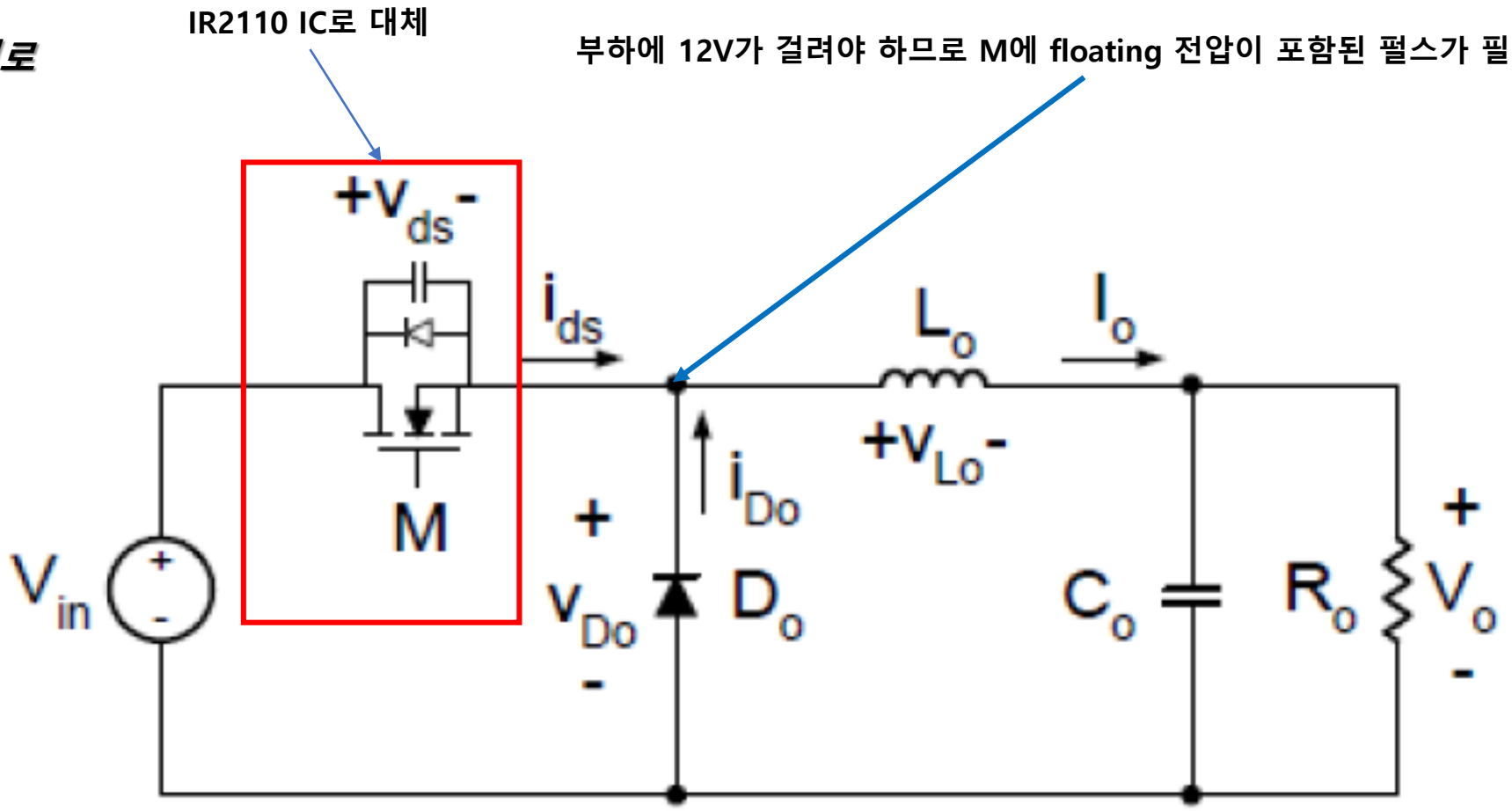
**** 남은 기술***

PID 알고리즘 필요

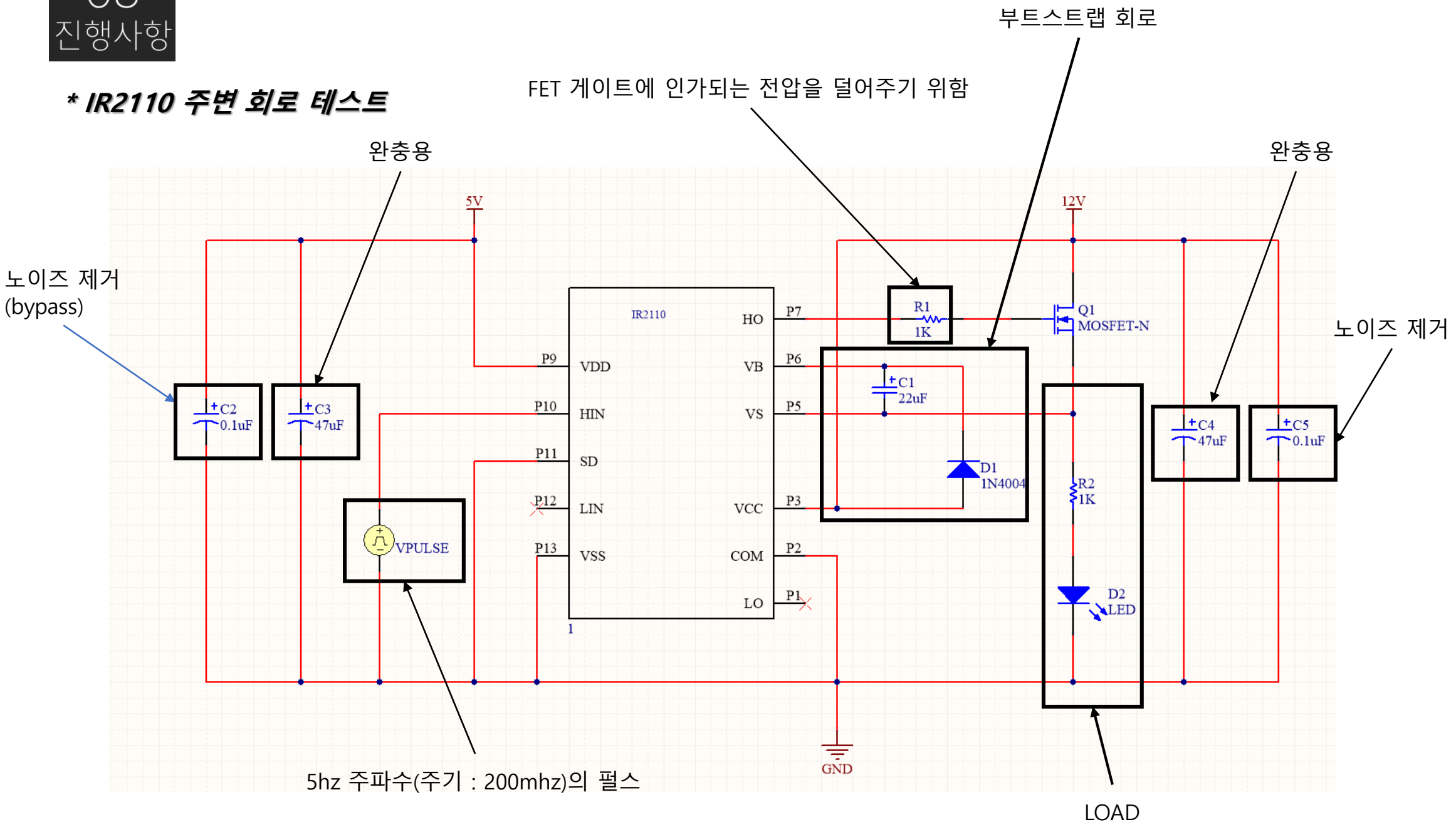


IR2110 테스트

**** DC-DC 컨버터 회로***


IR2110 IC로 대체
부하에 12V가 걸려야 하므로 M에 floating 전압이 포함된 펄스가 필요함
+Vds-
ids
Lo
Io
+VLo-
iDo
M
+
vDo
-
Do
Vin
+
-
Co
Ro
+
Vo
-

* IR2110 주변 회로 테스트



**** IR2110 핀 설명***

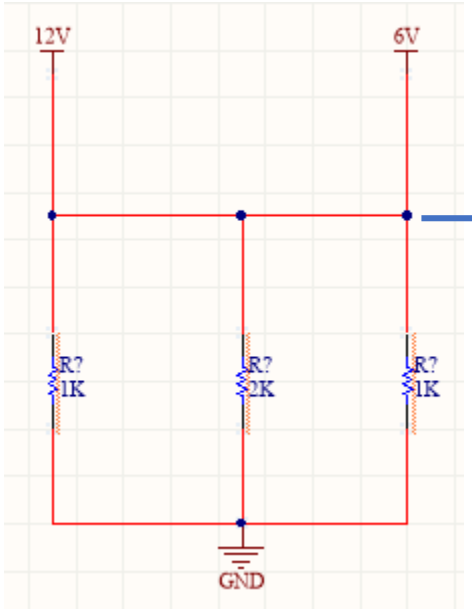
Lead Definitions

Symbol	Description
V_{DD}	Logic supply
HIN	Logic input for high side gate driver output (HO), in phase
SD	Logic input for shutdown
LIN	Logic input for low side gate driver output (LO), in phase
V_{SS}	Logic ground
V_B	High side floating supply
HO	High side gate drive output
V_S	High side floating supply return
V_{CC}	Low side supply
LO	Low side gate drive output
COM	Low side return

INPUT

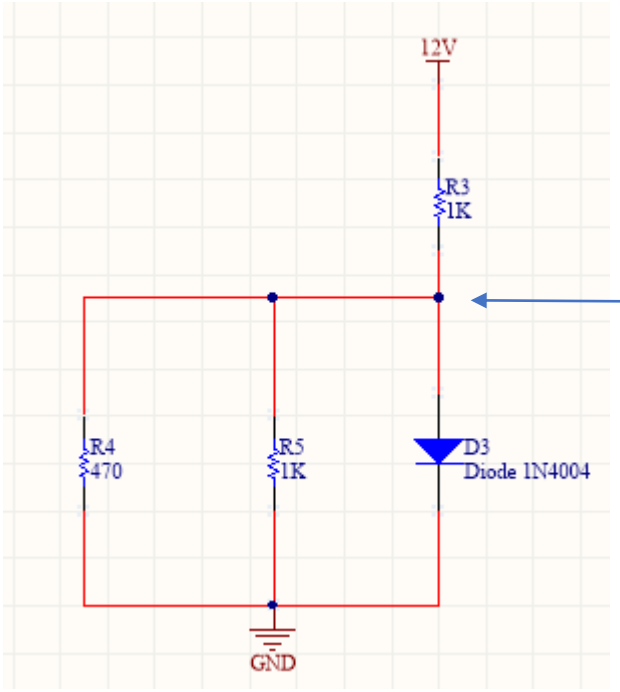
OUTPUT

* 부트스트랩 회로 이해를 위한사전 지식



6V 가 걸림

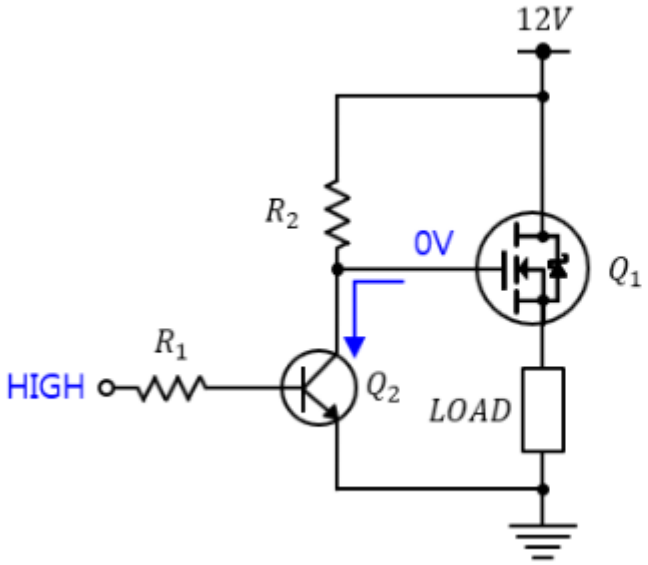
->병렬 연결에서는 낮은 전압으로 통일됨!

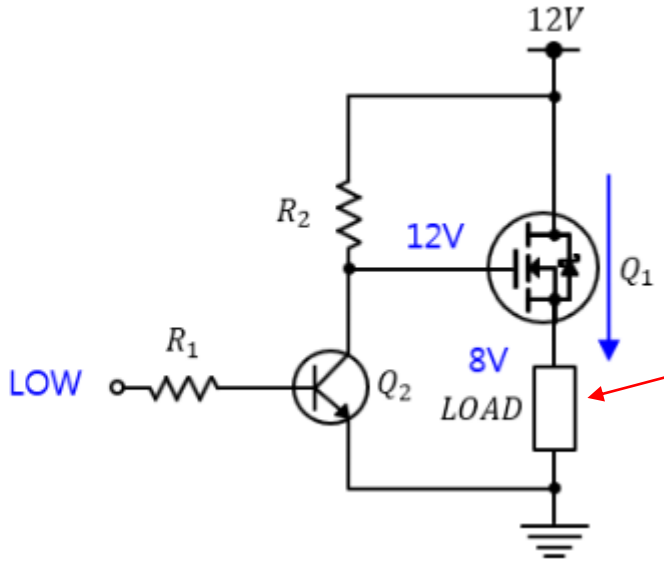


다이오드의 Vd만큼
전압이 걸림

* 부트스트랩 회로

Q1 FET의 소스에 VCC가 걸려 있는 경우,
게이트에 VCC만큼의 전압을 공급해주기 위해 VCC 직류 성분을 floating 시켜줌 (Vb = Vcc+Vs)





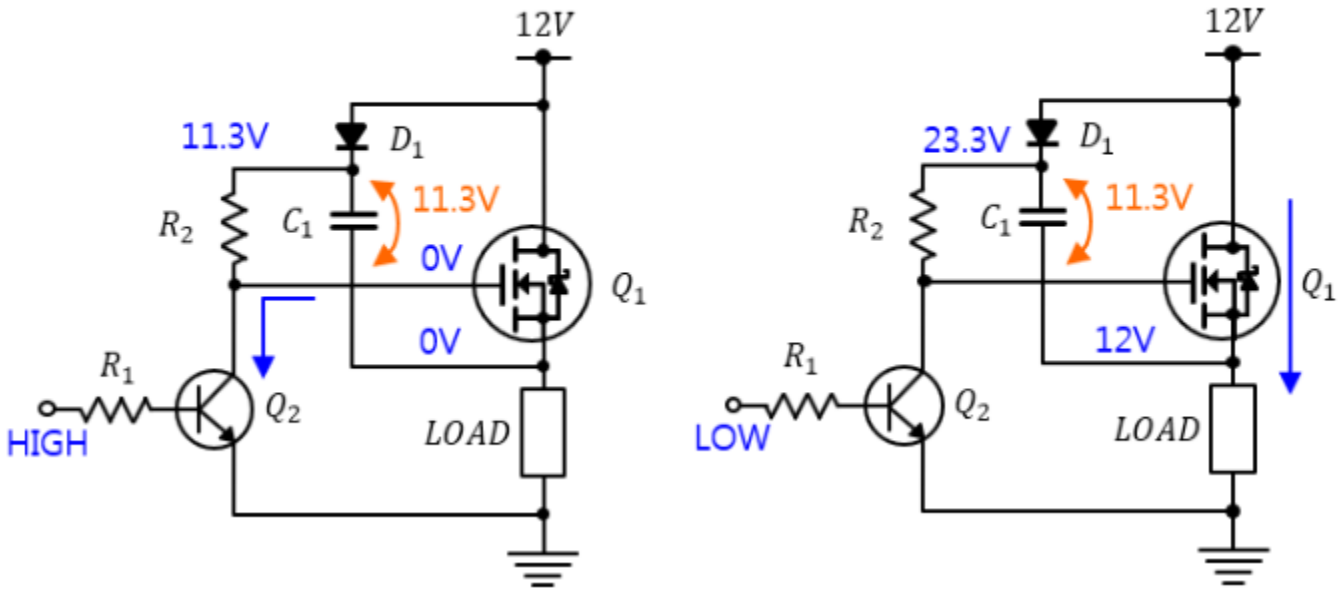
LOAD에 온전히 12V가 걸려서 최대의 효율을 내는게 목적임!

1) TR 베이스에 HIGH 신호가 인가되면 FET 스위칭 OFF됨.

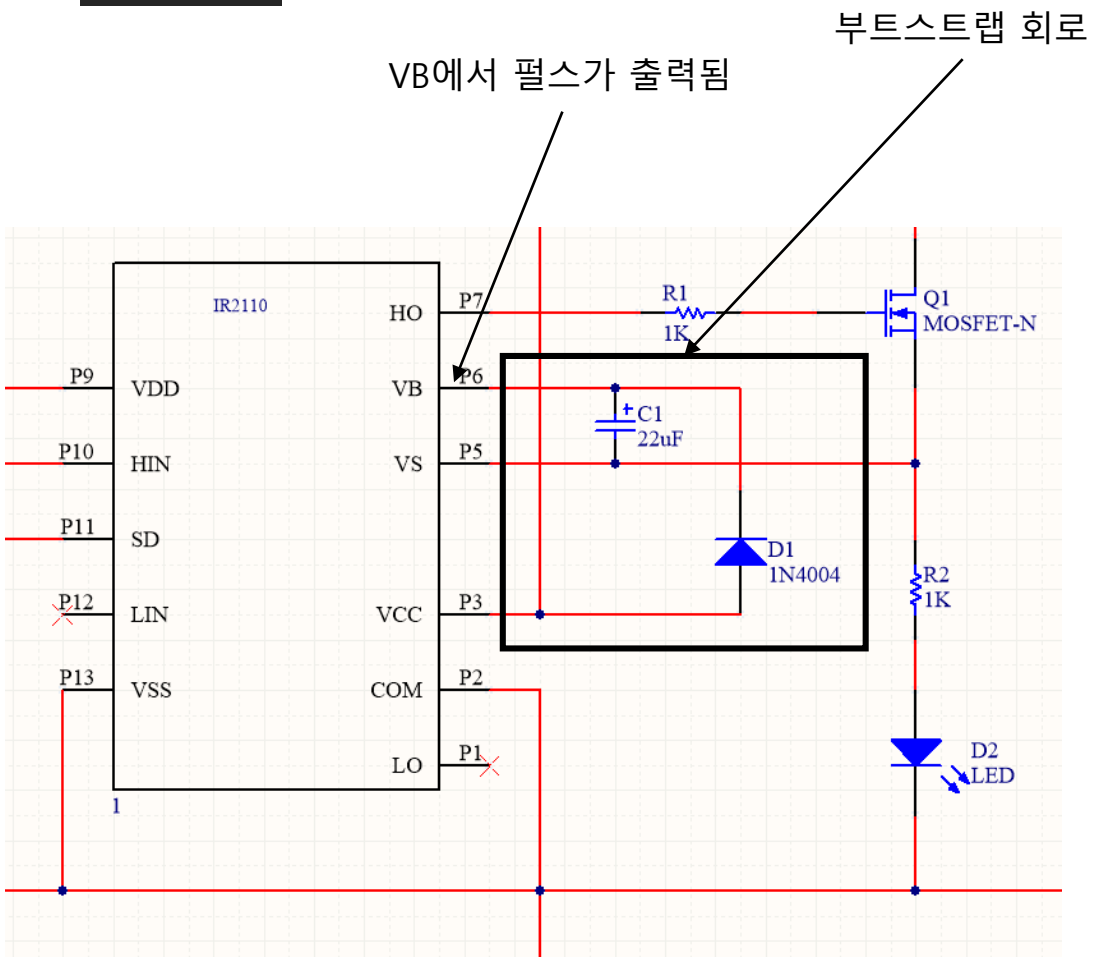
2) TR 베이스에 LOW 신호가 인가되면 FET 스위칭 ON됨.

- TR이 비활성화 되면서 풀업저항 R2에 의해 12V가 FET의 게이트에 인가됨.
- Vgs 값에 의해 LOAD에 온전히 12V가 걸리지 않고 Vgs 값만큼 강하된 값만큼 전압이 걸림. **=> 정상적인 동작을 하지 않을 수도 있음!**

부트스트랩 회로

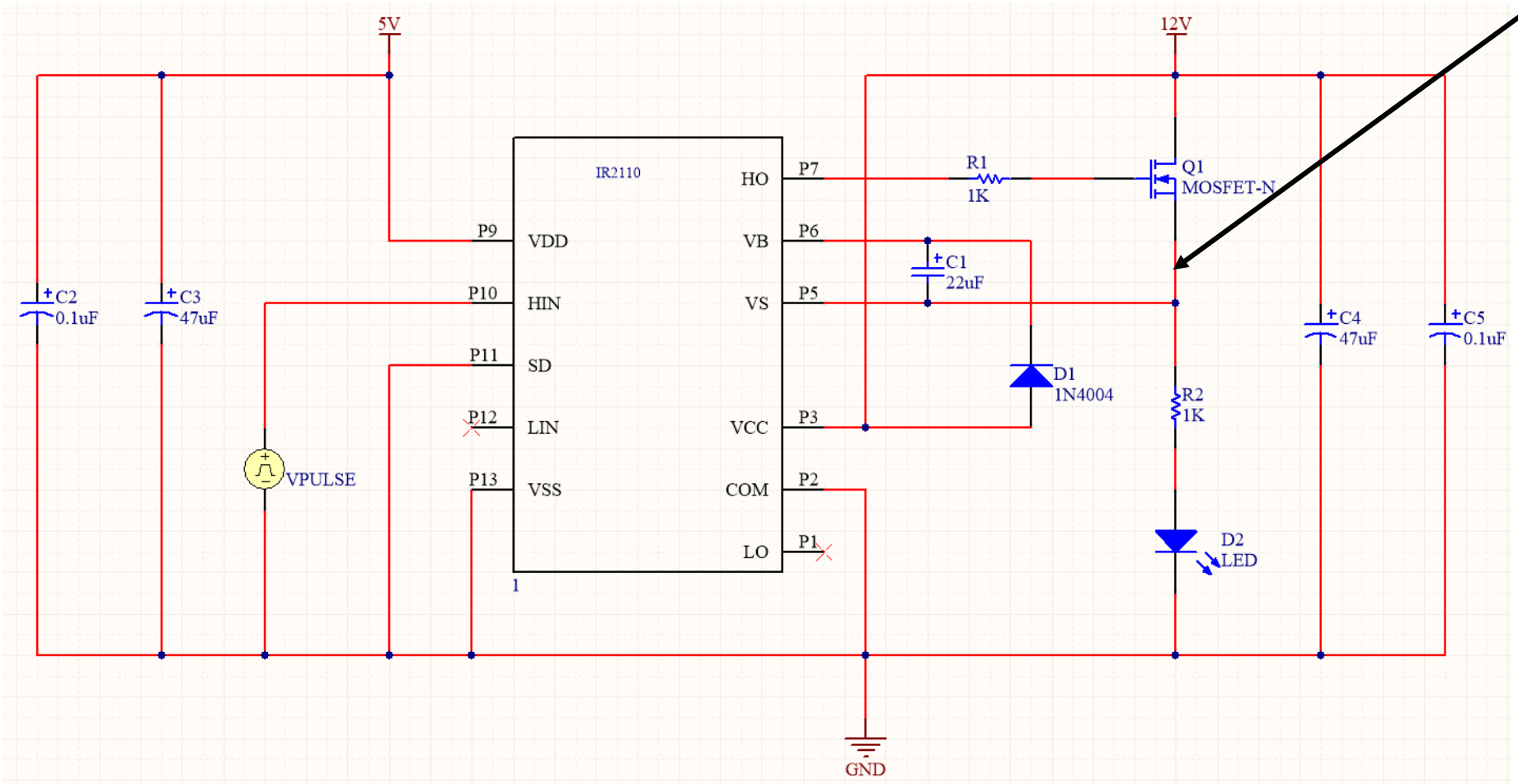


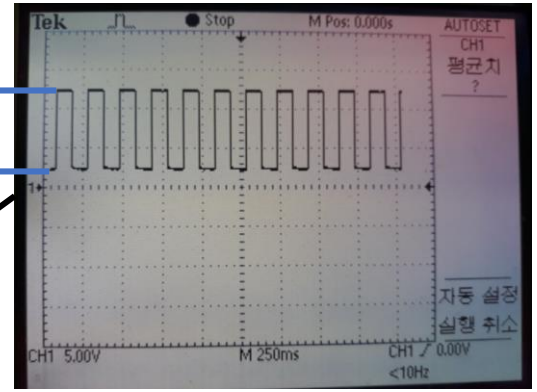
1) TR 베이스에 HIGH 신호가 인가되면 TR이 스위칭 ON 되고, FET는 스위칭 OFF 된다.
   - C1 에 11.3V만큼의 전압이 충전된다.
2) TR 베이스에 LOW 신호가 인가되면 TR이 스위칭 OFF 되고, FET는 스위칭 ON 된다.
   - TR이 OFF되고 FET가 ON되는 상황에서, VCC에서 흐르는 전류는 VCC(12V) 에서 C1을 거쳐 LOAD로 흐르고 R2를 거쳐 FET의 게이트로 흐른다.
   - C1에 이미 11.3V가 충전되어 있기 때문에, 12V 와 합쳐져서 R2의 위에는 23.3V가 걸리게 된다.
   - 23.3V에서 Vgs값을 강하한다고 해도 12V보다 작아지지는 않기 때문에 LOAD에는 VCC만큼의 12V가 안정적으로 공급된다.



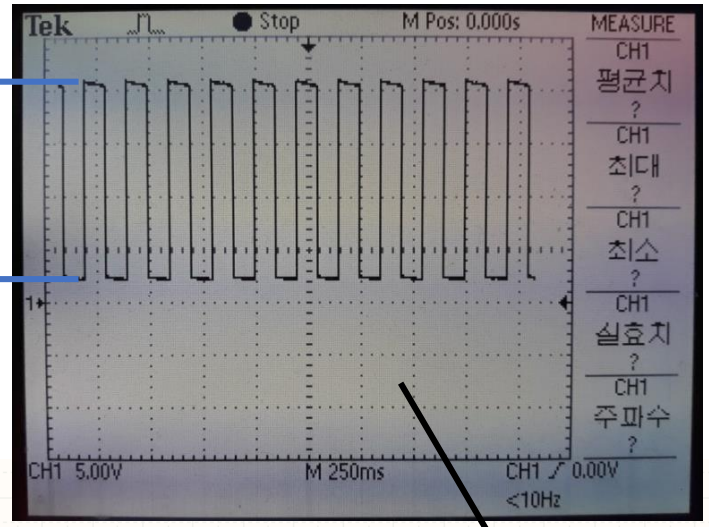
- VCC에서 다이오드를 거쳐서 C1에 충전되어 있는 전압과 합쳐짐.
- VB에서 합쳐진 전압만큼 HO 에서 나가는 펄스를 floating 시켜줌.
  => HO에서 나가는 펄스에 약 VCC 만큼의 전압이 floating 되기 때문에 FET의 소스 부분에 LOAD가 있더라도 안정적으로 VCC만큼의 전압을 공급해줄 수 있음

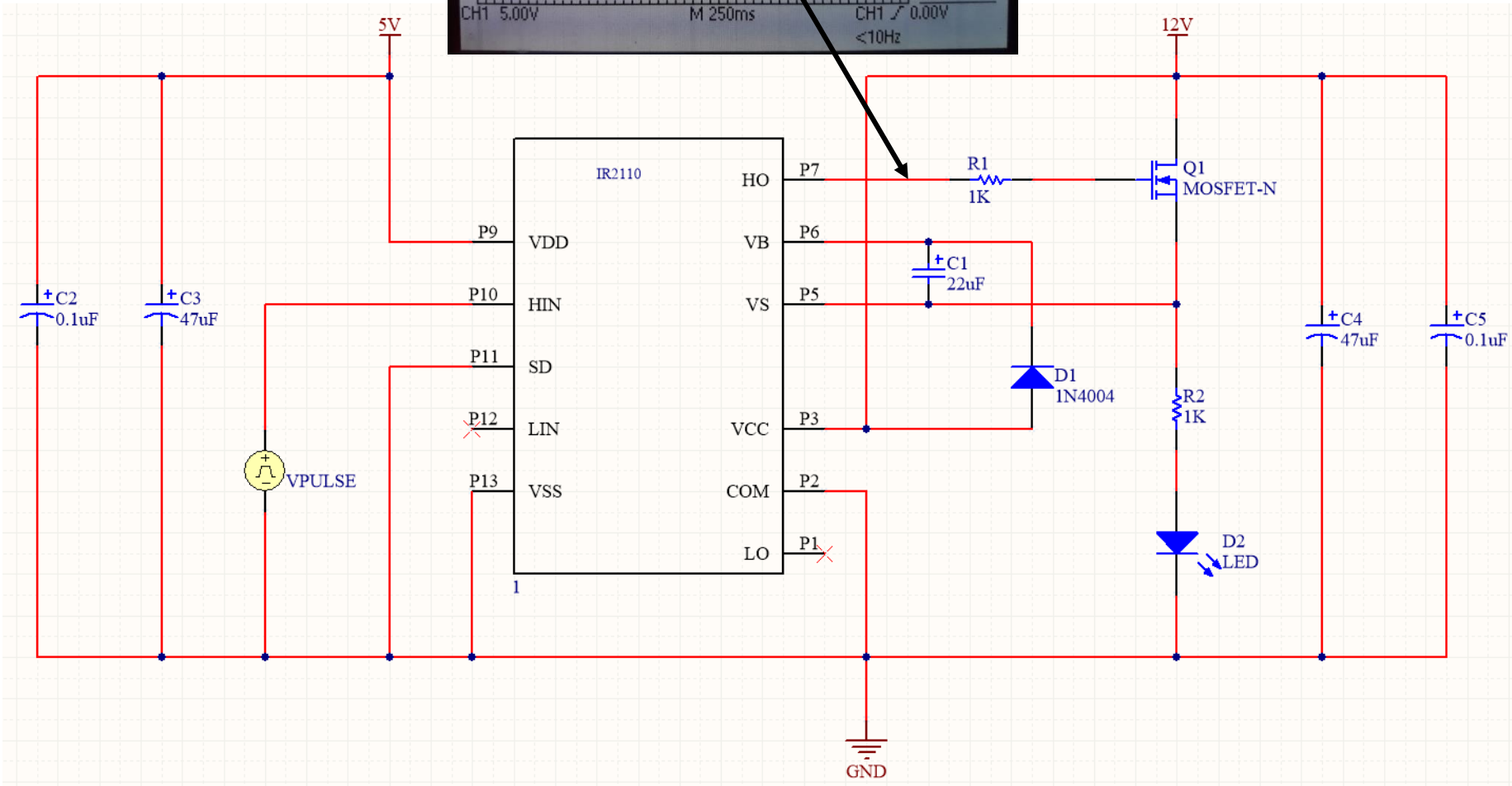
* IR2110 실험 결과


5V
12V
IR2110
P9
VDD
P10
HIN
P11
SD
P12
LIN
P13
VSS
HO
VB
VS
VCC
COM
LO
P7
P6
P5
P3
P2
P1
1
R1
1K
Q1
MOSFET-N
C1
22uF
C2
0.1uF
C3
47uF
C4
47uF
C5
0.1uF
D1
1N4004
R2
1K
D2
LED
VPULSE
GND


12V
2V
Tek
Stop
M Pos: 0.000s
AUTOSET
CH1
평균치
자동 설정
실행 취소
CH1 5.00V
M 250ms
CH1 0.00V
<10Hz

03 진행사항

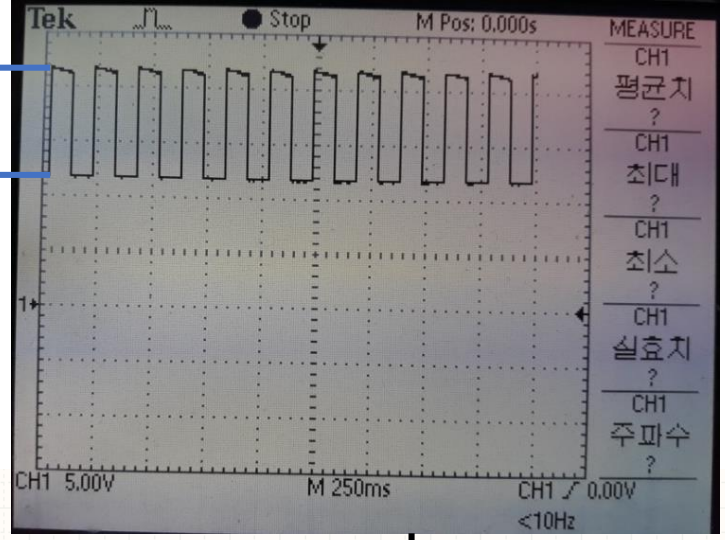

21V
2V
Tek
Stop
M Pos: 0.000s
MEASURE
CH1 평균치
CH1 최대
CH1 최소
CH1 실효치
CH1 주파수
CH1 5.00V
M 250ms
CH1 0.00V
<10Hz

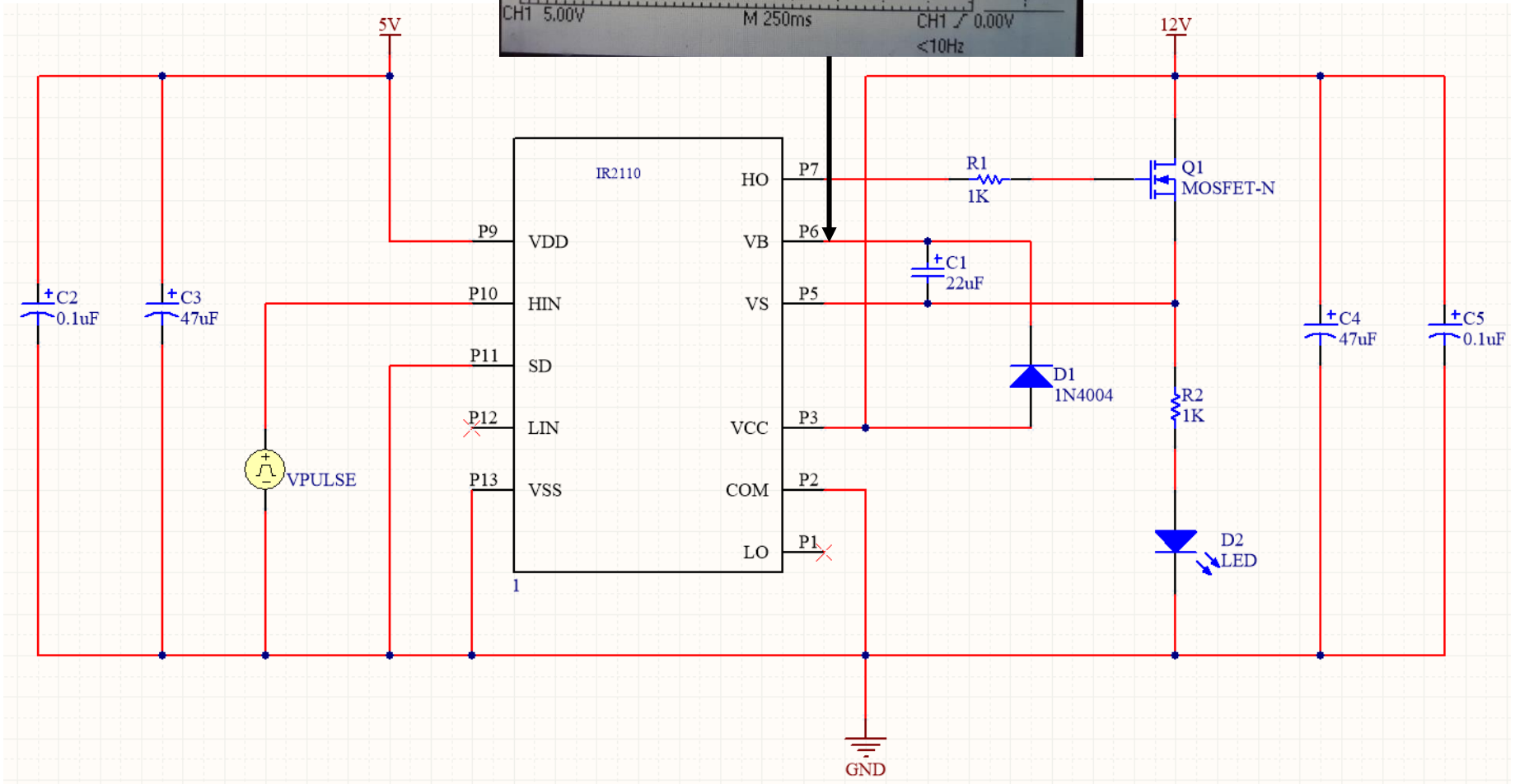

5V
12V
IR2110
HO
VB
VS
VCC
COM
LO
VDD
HIN
SD
LIN
VSS
P7
P6
P5
P3
P2
P1
P9
P10
P11
P12
P13
R1
1K
Q1
MOSFET-N
C1
22uF
C2
0.1uF
C3
47uF
C4
47uF
C5
0.1uF
D1
1N4004
R2
1K
D2
LED
VPULSE
GND

03 진행사항

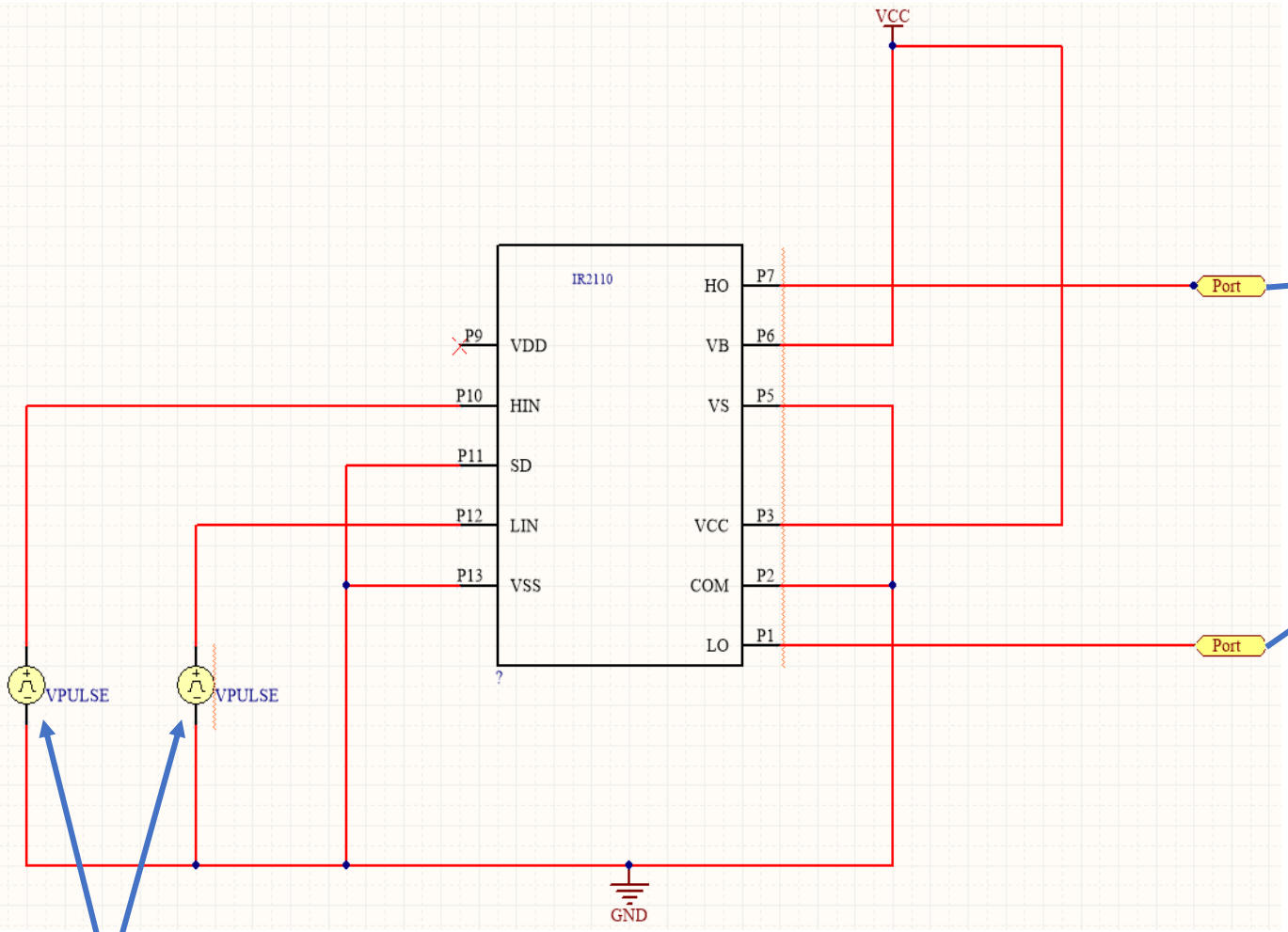
22V

12V



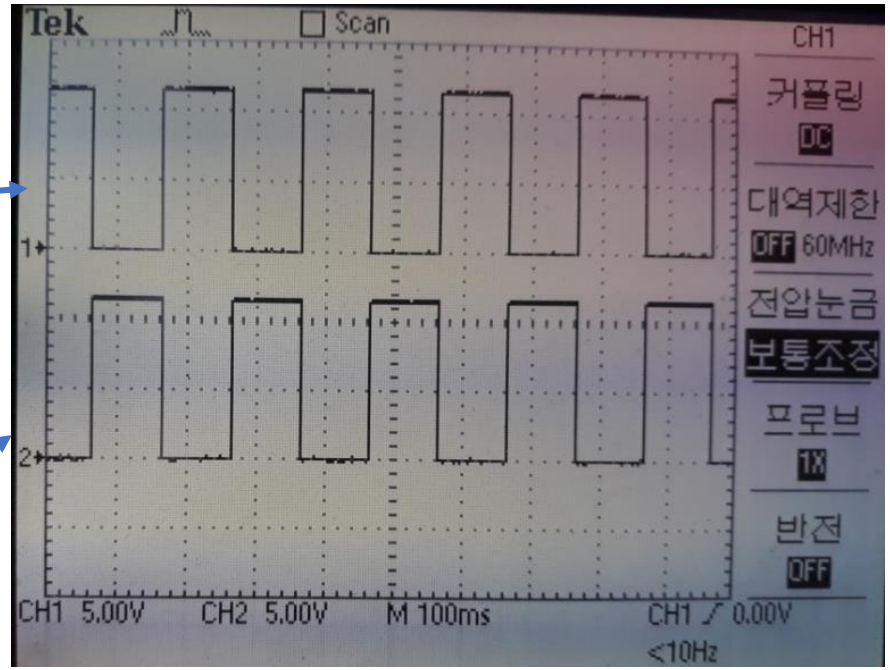


* IR2110 테스트



반전된 펄스

HO, LO 핀에서 반전된 펄스가 나옴



* 문제점

- BLDC 모터를 차의 바퀴부분에 고정시키고 모터를 구동하면 같은 듀티비의 정방향, 역방향 시 속도가 많이 차이남

=> 듀티비 보정 필요

- RTOS 통합 테스트시 문제가 발생하는지 지속적인 TEST

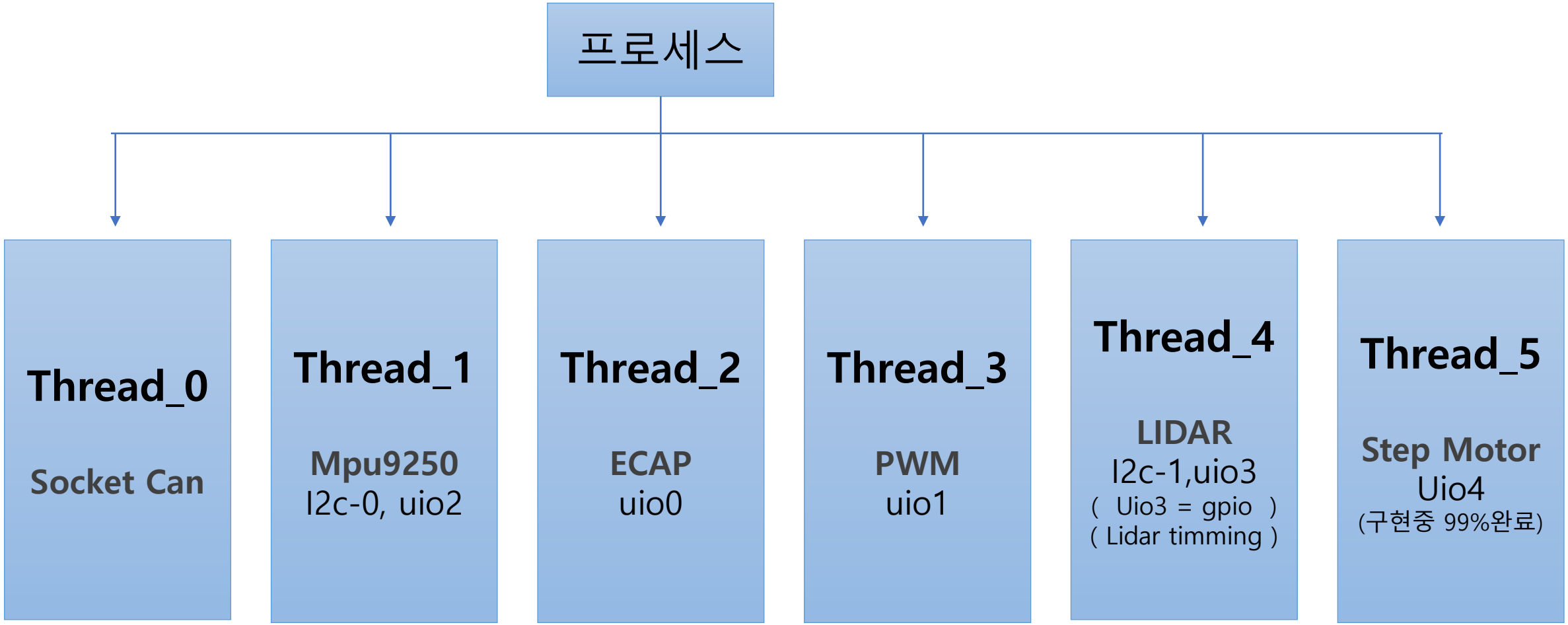
감사합니다

02 진행상황 및 문제점

김시윤

FPGA ALL Test

기존에 fork 로 통합했던 통합 코드를 thread 로 통합.



FPGA ALL Test

참과 거짓의 지옥 이였던 복잡한 fork 에서 thread 로 변환한 코드

```c
int main(void)
{
        int pid;
        int pid2;
        int pid3;
        int pid4;

        if((pid = fork())==0){
        //자식프로세스1

/*****************************************/
                if((pid2= fork()) == 0){
                        //child2
                        if((pid3 = fork()) == 0){
                                //child3
                                        if((pid4 = fork()) == 0){
                                                //child4 logic
                                                while(1);
                                        }
                                                else if(pid4 >=1){

                                        }

                                        //child3 logic
                                        while(1);

                                }
                                else if(pid3 >= 1){
                                }
                        //child2 logic
                        while(1){
                        printf(" i am child2 Hi \n");
                        sleep(3);
                        }
                }
                else if(pid2 >= 1){

                }
/*****************************************/

                //child1 logic
                //mpu9250 logic
                while(1){
                printf("i am child1 Hi \n");
                sleep(3);
                }
        }
        else if(pid >=1){
        //부모프로세스

                }


return 0;
}
```

```c
int main(void)
{
        pthread_t p_thread[5];
        int thr_id;
        int status;

        int fd = 1;
        int fd2 = 2;
        int fd3 = 3;
        int fd4 = 4;

        thr_id = pthread_create(&p_thread[0], NULL, thread_1,(void *)&fd);
        if(thr_id <0)
        {
                perror("thread create error :");
                exit(0);
        }

        thr_id = pthread_create(&p_thread[1], NULL, thread_2,(void *)&fd2);
            if(thr_id <0)
    {
        perror("thread create error :");
        exit(0);
    }

    thr_id = pthread_create(&p_thread[2], NULL, thread_3,(void *)&fd3);
        if(thr_id <0)
    {
        perror("thread create error :");
        exit(0);
    }

    thr_id = pthread_create(&p_thread[3], NULL, thread_4,(void *)&fd4);
        if(thr_id <0)
    {
        perror("thread create error :");
        exit(0);
    }
    pthread_join(p_thread[0], (void **)&status);
    pthread_join(p_thread[1], (void **)&status);
    pthread_join(p_thread[2], (void **)&status);
    pthread_join(p_thread[3], (void **)&status);
return 0;

}
```

FPGA ALL Test

```
/*
    i2c-0 = fd_buf[0] -> mpu9250
    uio2 = fd_buf[1] -> timer
    uio0 = fd_buf[2] -> ECAP
    uio1 = fd_buf[3] -> PWM
    i2c-1 = fd_buf[4] -> Lidar
    uio3 = fd_buf[5] -> gpio
    uio4 = fd_buf[6] -> step*/

system("ifconfig eth0 192.168.7.33");
sleep(10);

if((fd_buf[0] = open("/dev/i2c-0", O_RDWR)) <0)
{
    perror("Open Device Error! \n");
    return -1;
}
printf("fd_buf(i2c-0) Open complete!\n");

if((fd_buf[1] = open("/dev/uio2",O_RDWR)) <0)
{
    perror("Open Time Device Error!! \n");
    return -1;
}
printf("fd_buf2(uio2) Open complete!\n");

if((fd_buf[2] = open("/dev/uio0",O_RDWR))<0)
{
    printf("Invalid UIO Device File uio0\n");
return -1;
}
printf("fd_buf3(uio0) Open complete!\n");



if((fd_buf[3] = open("/dev/uio1",O_RDWR)) <0)
{
    printf("Invalid UIO Device File uio1\n");
    return -1;
}

printf("fd_buf4(uio1) Open complete!\n");

if((fd_buf[4] = open("/dev/i2c-1", O_RDWR)) < 0
{
    perror("---OPEN DEVICE ERROR ");
    return -1;
}
printf("fd_buf5(i2c-1) Open complete!\n");

if((fd_buf[5] = open("/dev/uio3",O_RDWR))<0)
{
    perror("Invalid UIO Device file uio3\n");
    return -1;
}
if((fd_buf[6] = open("/dev/uio4",O_RDWR))<0)
{
    perror("Invalid UIO Device file uio4\n");
    return -1;
}
```

```
/*****************************************************************
    Thread Create Under Code
*****************************************************************/

thr_id = pthread_create(&p_thread[0], NULL, thread_0,(void *)fd_buf);
if(thr_id <0)
{
    perror("thread create error :");
    exit(0);
}

thr_id = pthread_create(&p_thread[1], NULL, thread_1,(void *)fd_buf);
if(thr_id <0)
{
    perror("thread create error :");
    exit(0);
}

thr_id = pthread_create(&p_thread[2], NULL, thread_2,(void *)fd_buf);
    if(thr_id <0)
{
    perror("thread create error :");
    exit(0);
}

thr_id = pthread_create(&p_thread[3], NULL, thread_3,(void *)fd_buf);
    if(thr_id <0)
{
    perror("thread create error :");
    exit(0);
}

thr_id = pthread_create(&p_thread[4], NULL, thread_4,(void *)fd_buf);
    if(thr_id <0)
{
    perror("thread create error :");
    exit(0);
}

thr_id = pthread_create(&p_thread[5], NULL, thread_5,(void *)fd_buf);

if(thr_id<0)
{
    perror("thread create error :");
    exit(0);
}
```

```
/*****************************************************
    Wait thread until turnoff
*****************************************************/

    pthread_join(p_thread[0], (void **)&status);
    pthread_join(p_thread[1], (void **)&status);
    pthread_join(p_thread[2], (void **)&status);
    pthread_join(p_thread[3], (void **)&status);
    pthread_join(p_thread[4], (void **)&status);
    pthread_join(p_thread[5], (void **)&status);
```

Main문 첫 시작에 fd_buf 라는 배열에 각 장치파일에 대한 open을 실시해 저장한다.

Thread를 생성하여 각 thread 마다 배열자체를 전달한다. 2개의 장치가 필요할 경우를 대비.

Join을 하여 프로세스는 스레드들의 종료를 기다린다.

FPGA Step motor Test

Diagram | Address Editor | ad.xdc *

Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
My_ECAP_Core_0	S00_AXI	S00_AXI_reg	0x43C1_0000	64K	0x43C1_FFFF
My_PWM_Core_0	S00_AXI	S00_AXI_reg	0x43C0_0000	64K	0x43C0_FFFF
My_Time_Core_0	S00_AXI	S00_AXI_reg	0x43C2_0000	64K	0x43C2_FFFF
axi_gpio_0	S_AXI	Reg	0x4120_0000	64K	0x4120_FFFF
My_STEP_Core_0	S00_AXI	S00_AXI_reg	0x43C3_0000	64K	0x43C3_FFFF

```c
#include "xparameters.h"
#include "xil_io.h"

//#define MY_PWM XPAR_MY_PWM_CORE_0_S00
#define MY_PWM 0x43C00000 //This value

int main(){
    int num=0;
    int i;

    while(1){
        num = 34000;

        Xil_Out32(MY_PWM, num/2);
        Xil_Out32(MY_PWM+4, num);

        for(i=0;i<300000; i++);
    }
}
```

Step Motor 제어를 위한 기본적인 PWM Step motor control IP 설계, PWM의 주기에 따라 속도가 변하기 때문에 최대속도일때의 주기를 SDK로 구하였다.

최대 속도일때의 주기는 2.941KHz (0.34msec)이다.

페타리눅스 상에서 mmap을 이용하여 소프트웨어 구현 후 엔코더로 각도,위치를 제어할 예정.

문제점 및 개선

1. 복잡한 fork의 스케줄링 문제.
- Thread로 컨버팅.

2. 스케줄링 시간 육안으로 확인의 어려움
- GPIO IP 설계 후 오실로스코프로 확인.

3. Thread create handler에 여러 개 장치파일 전달
- 배열을 사용하여 배열자체를 전달.

4. Lidar 측정시 250 Cm 이상 오버플로우 발생.
- 데이터시트 확인중.

5. 차량 구성중 서보모터 전원 부족
- DC-DC 컨버터 구매.

6. STEP모터 속도 및 각도 제어
- 진행중.

감사합니다

03 진행상황 및 문제점

문지희

PWM IP Create

my_wm_v1_0

```
timescale 1 ns / 1 ps

    module my_pwm_v1_0 #
    (
        // Users to add parameters here

        // User parameters ends
        // Do not modify the parameters beyond this line
        parameter integer PWM_COUNTER_MAX = 128,

        // Parameters of Axi Slave Bus Interface S00_AXI
        parameter integer C_S00_AXI_DATA_WIDTH  = 32,
        parameter integer C_S00_AXI_ADDR_WIDTH  = 4
    )
    (
        // Users to add ports here

        // User ports ends
        // Do not modify the ports beyond this line

        output wire PWM,
        // Ports of Axi Slave Bus Interface S00_AXI
        input wire  s00_axi_aclk,
```

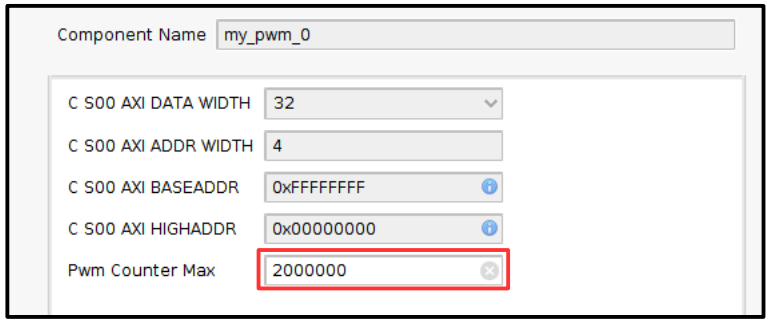
```
// Instantiation of Axi Bus Interface S00_AXI
    my_pwm_v1_0_S00_AXI # (
        .C_S_AXI_DATA_WIDTH(C_S00_AXI_DATA_WIDTH),
        .C_S_AXI_ADDR_WIDTH(C_S00_AXI_ADDR_WIDTH),
    ) my_pwm_v1_0_S00_AXI_inst (
        .PWM(PWM),
        .S_AXI_ACLK(s00_axi_aclk),
        .S_AXI_ARESETN(s00_axi_aresetn),
        .S_AXI_AWADDR(s00_axi_awaddr),
```

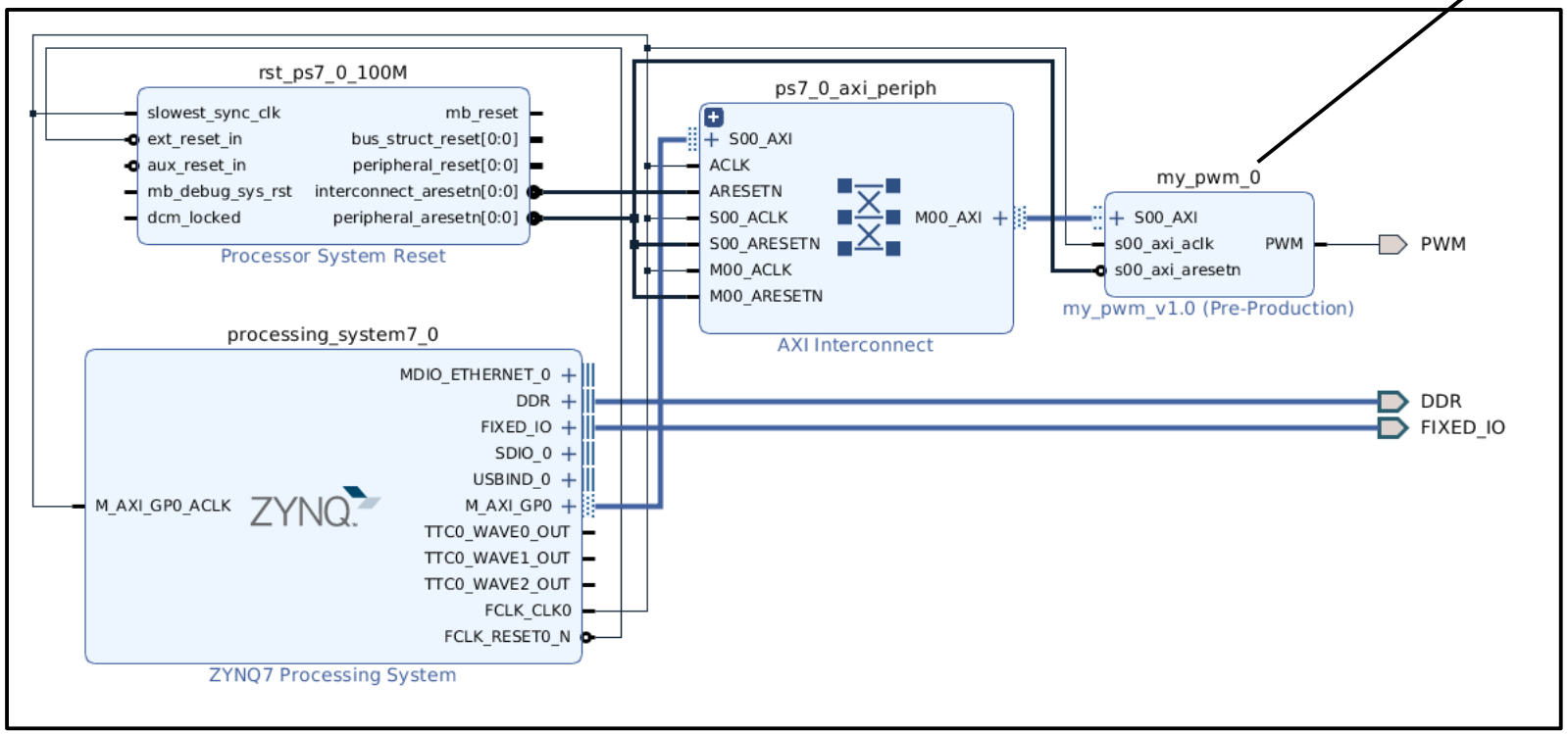
my_pwm_v1_0_S00_AXI_inst

```
timescale 1 ns / 1 ps

    module my_pwm_v1_0_S00_AXI #
    (
        // Users to add parameters here
        parameter integer PWM_COUNTER_MAX = 2000000,
        // User parameters ends
        // Do not modify the parameters beyond this line

        // Width of S_AXI data bus
        parameter integer C_S_AXI_DATA_WIDTH    = 32,
        // Width of S_AXI address bus
        parameter integer C_S_AXI_ADDR_WIDTH    = 4
    )
    (
        // Users to add ports here

        output wire PWM,
        // User ports ends
        // Do not modify the ports beyond this line
```

```
    reg [31:0] counter = 0;
    // Add user logic here
    always @(posedge S_AXI_ACLK) begin
        if(counter < PWM_COUNTER_MAX - 32'd1 )
            counter <= counter + 32'd1;
        else
            counter <= 32'd0;

    end

    assign PWM = slv_reg0 < counter ? 1'b0 : 1'b1;
    // User logic ends
```

Block Design





SDK - 각도제어

```
float duty (int angle)
{
    return ((1.0 + angle/180.0) / 20.0 );
}
//1ms == 0도
//2ms == 180도
```

→ 정확한 제어 불가능
180도 보다 적게 돌게 된다

```
#include "xparameters.h"
#include "xil_io.h"
#define MY_PWM 0x43C00000
#define period  2000000

int main(){
    int i, angle=0;
    float num=0;
    while(1){
        //0
        num = 62800;
        Xil_Out32(MY_PWM, num);
        for(i=0;i<140000000; i++);
        //90
        num = 151400;
        Xil_Out32(MY_PWM, num);
        for(i=0;i<140000000; i++);
        //180
        num = 240000;
        Xil_Out32(MY_PWM, num);
        for(i=0;i<140000000; i++);
    }

}
```

감사합니다