Xilinx Zynq FPGA, TI DSP, MCU 기반의 회로 설계 및 임베디드 전문가 과정

강사 - Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 - Hyungjoo Kim(김형주)
mihaelkel@naver.com

1.아래 링크에서 CCS Project를 다운로드 받는다.
https://github.com/KOITT2/RC_Car/tree/master/experiment/mcu/Integration_Test_CAN_2nd

2.지난번 테스트에서는 아래와 같은 프로토콜을 사용하였다

신호 종류	신호 번호	후속 데이터
좌회전	1	없음
우회전	2	없음
전진	3	없음
후진	4	없음
정지	5	없음
충돌 경보	6	없음
좌측 깜빡이	7	없음
우측 깜빡이	8	없음
전조등 신호	9	없음
영상 데이터	10	장애물 존재 여부(0, 1), 근접 여부
레이더 데이터	11	장애물의 방향(각도), 거리
속도 지정	12	특정 속도값 or PWM Duty
각도 지정	13	특정 각도값 or PWM Duty(서보)

13 3을 송신했을 때, 1300,
13 2를 송신했을 때, 1200 등 100단위로 듀티비를 제어하였는데, 2차 테스트에서는 1단위로 듀티비를 변경할 수 있도록 했다.
즉, 13 1 3 0 0를 송신하면 1300이 되는 방식이다.

3.만들면서 실수했던 것들.

```
#include <HL_etpwm.h>
#include <HL_hal_stdtypes.h>
#include <HL_reg_sci.h>
#include <HL_sci.h>
#include "HL_can.h"
#include "HL_gio.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
void send_data(uint8* msg, int length);
int main(void)
{
    uint8 rx_data[32] = { 0, };

    uint8 tx_data[8] = { 1, 1, 1, 1, 1, 1, 1, 1 };
    int data = 0;
    int i, j;
    etpwmInit();
    sciInit();
    canInit();
    gioInit();

    etpwmStartTBCLK();
    canEnableErrorNotification(canREG1); //canIoSetDirection(canREG1, canMESSAGE_BOX1, canMESSAGE_BOX2);
    gioSetBit(gioPORTA, 0, 1);
```

```
    for (i = 0; i < 10; i++)
    {
        for (j = 0; j < 50000000; j++)
            ;

        canTransmit(canREG1, canMESSAGE_BOX1, (const uint8*) &tx_data[0]);
    }
    while (1)
    {
        while (!canIsRxMessageArrived(canREG1, canMESSAGE_BOX2))
            ;
//        {
        canGetData(canREG1, canMESSAGE_BOX2, (const uint8*) &rx_data[0]);

        switch (rx_data[0])
        {
        case 13:
            data = (rx_data[1] - 48) * 1000 + (rx_data[2] - 48) * 100
                    + (rx_data[3] - 48) * 10 + (rx_data[4] - 49);
            data *= 1.25;
            break
        case 7:
            gioSetBit(gioPORTA, 0, rx_data[1]);
            break
        }
//              data = (rx_data[0] - 48)*1000 + (rx_data[1] - 48)*100 + (rx_data[2] - 48)*10 + (rx_data[3] -48);

//              data = atoi(rx_data);

        send_data(rx_data, 8);
        etpwmREG1->CMPA = data;

        memset(rx_data, 0, sizeof(rx_data));
        data = 0;
//        }
    }

    return 0;
}

void send_data(uint8* msg, int length)
{
    int i;
    for (i = 0; i < length; i++)
    {
        sciSendByte(sciREG1, msg[i] + 48);
        sciSendByte(sciREG1, ' ');
    }
    sciSendByte(sciREG1, '\r');
    sciSendByte(sciREG1, '\n');
}
```

위의 코드에서 아래 부분을 보면,

```
data = (rx_data[1] - 48) * 1000 + (rx_data[2] - 48) * 100 + (rx_data[3] - 48) * 10 + (rx_data[4] - 48);
```

각 받은 데이터에서 -48을 해주고 있다.

CAN 통신 수신시 받는 데이터는 Char타입이기 때문에, 실제로 1을 받는다면 아스키코드 49를 받는 것과 같다.

따라서 -48을 해줌으로서 데이터 타입을 정수형으로 변환해주어야 한다.

이를 하지 않아, 테스트 초반에 듀티비가 이상하게 설정되곤 했다.

4.3차 테스트 계획

현재 조향 완성되지 않아, 2차 테스트 시 조향은 사람이 따라다니면서 수동으로 조작하였다.

3차 테스트시에는 조향의 기구설계를 완성하고, duty cycle에 따른 조향각을 계산하여

프로토콜에 추가할 것이다. 즉, 3차 테스트시에는 완벽한 수동 RC카가 완성될 것이다.

또한, 2차 테스트 당시 발열이 매우 심했다. 가능하면 보드들, PDB 등을 담을 만한 기구를 만들고,

Fan을 달아 발열량을 줄일 것이다.

마지막으로, DSP - Smartphone의 통신이 불안정할 때가 있었다. 안테나의 위치에 따라 불안정할 수도 있을 것 같아,

보드의 위치를 선정하는 데에 이를 고려할 것이다.