전방 추돌 방지 기능 개발

경남대학교 김진호

RTES
Real Time Embedded System

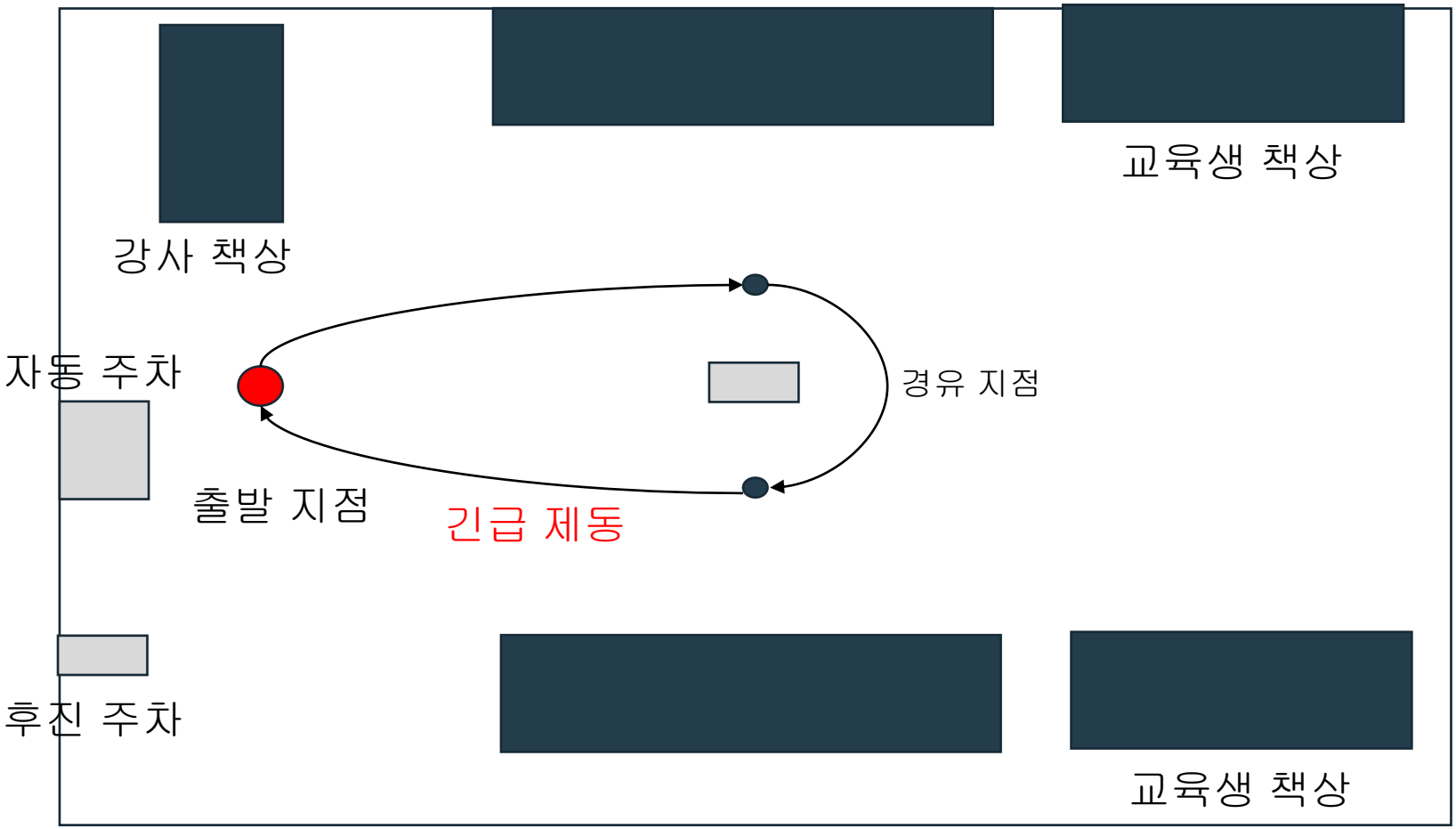
2023.11.15

프로젝트 개요

- 팀 구성 : 개인

- 프로젝트 최종 목표
  - 유선 시리얼 통신으로 차량 조작 : 키는 자유롭게 설정 가능
    - 전진, 후진, 좌회전, 후회전
  - 전방 추돌 방지 기능
    - 전방 ToF 센서를 이용한 긴급 제동
    - 차량 속도에 따른 제동 거리
  - 후방 주차
    - 후방 초음파 센서를 이용한 경고음 출력 및 정지
  - 자동 주차
    - 시리얼 통신으로 'p' 입력 시 자동 주차

RC카 기능 통합 및 데모



목 차

- 레이저 센서 모듈 및 실습
  - 레이저 센서 동작 원리 및 데이터 프레임 구조
  - 레이저 센서 거리 수신 프로그래밍 예제
- 전방 긴급충돌방지(AEB) 기능 구현 프로젝트

레이저 센서 개요

- 레이저 센서
  - 모델명 : TOF Sense
  - 상업적으로 이용 가능한 레이저 파장 중 근적외선을 이용한 센서
    - 940nm의 근적외선 사용
  - 근적외선을 방출하고 수신하는 데에 소요되는 시간으로 거리를 계산
  - 즉, TOF(Time of Flight) 기반의 레이저 거리 측정 센서

레이저 센서 개요

- 레이저 센서 주요 특징
  - UART, CAN 통신 지원
    - 지원 통신 속도

UART Baudrate	Note
115200,230400,460800,921600,1000000,1200000, 1500000,2000000,3000000	Default baud rate: 921600
CAN Baudrate	**Note**
100K、250K、500K、1M	Default baud rate: 1M

  - 측정 범위 1.5cm ~ 5m
  - 거리 오차 ±1.5cm
  - 센서에 내장된 펌웨어가 거리를 계산하여 통신으로 전송함
  - 초당 10번 거리 측정 가능

레이저 센서 개요

- 센서 측정 레벨 설정
  - Short
  - Middle
  - Long

Typical ranging range: m	Short range: 0.015 ~ 2.16
	Mid-range: 0.015 ~ 3.60
	Long range: 0.015 ~ 5.05

- 설정에 따른 특성
  - Blind Zone : 측정 불가한 사각지대
  - Maximum Distance : 최대 측정 가능한 거리
  - Accuracy : 정확도

Mode	Blind zone (cm)	Maximum distance (m)	Accuracy (cm)
Short	1.5	2.16	±1.0
Medium	1.5	3.60	±1.0
Long	1.5	5.05	±1.5

레이저 센서 데이터 프레임 구조

- 레이저 센서 모듈이 출력하는 데이터 프레임 구조
  - UART 인터페이스용 출력 데이터 프레임의 총 길이 : 16Bytes
  - Frame Header : 1 Byte
  - Function Mark : 1 Byte
  - Data : 13 Bytes
    - 거리 측정 값, 신호 세기, 상태 등 데이터가 저장됨
  - Sum Check (CheckSum) : 1Byte
  - 바이트 배열 : Little Endian

레이저 센서 데이터 프레임 구조

- 레이저 센서 모듈이 출력하는 데이터 프레임 구조
  - TOFSense는 측정된 거리에 1,000을 곱하여 3Bytes로 출력

Data	Type	Length (Bytes)
Frame Header	uint8	1
Function Mark	uint8	1
reserved	uint8	1
id	uint8	1
System_time	uint32	4
dis*1000	uint24	3
dis_status	uint8	1
signal_strength	uint16	2
reserved	uint8	1
Sum Check	uint8	1

레이저 센서 데이터 프레임 구조

- **ex)** 아래와 같은 데이터 프레임이 수신되었을 때, 거리는?
  - 57 00 ff 00 9e 8f 00 00 ad 08 00 00 03 00 ff 3°

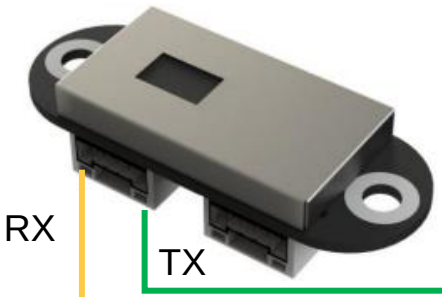
Data	Type	Length (Bytes)	Hex
Frame Header	uint8	1	57
Function Mark	uint8	1	00
reserved	uint8	1	…
id	uint8	1	00
System_time	uint32	4	9e 8f 00 00
dis*1000	uint24	3	ad 08 00

  - 레이저 센서는 데이터를 리틀 엔디안 방식으로 저장함
    - 리틀 엔디안: 변수 값의 우측 바이트부터 메모리에 저장하는 방식
    - 빅 엔디안: 변수 값의 왼쪽 바이트부터 메모리에 저장하는 방식
    - ex) 변수 값이 2인 int형 변수는 16진수 0x00000002로 나타내며,
      만약 리틀 엔디안 방식을 사용하는 경우 메모리에는 02 00 00 00 순서로 저장됨
  - 거리 × 1,000 = $0008ad_H$ = 2,221
  - 거리 = 2.221 m

레이저 센서 개요

- **UART 연결**
  - UART3 (TC275) ↔ Putty (PC)
  - UART1 (TC275) ↔ 레이저 센서 (ToF Sense)

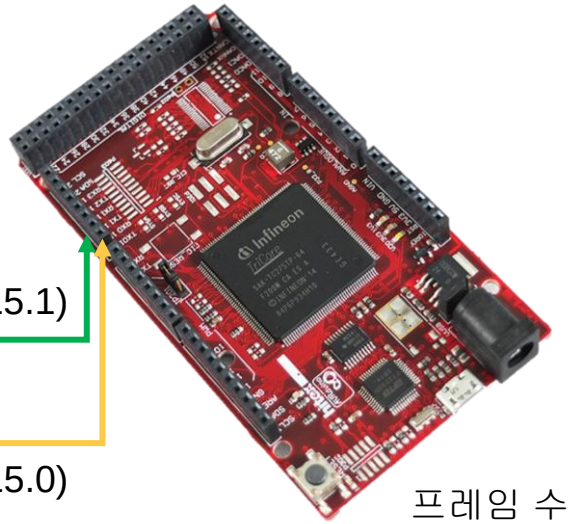
100ms마다 데이터 프레임 전송



RX

TX

UART1 RX(P15.1)



UART1 TX(P15.0)

프레임 수신

Frame Header	+	Function Mark	+	Data	+	Sum Check

16Byte Data Frame

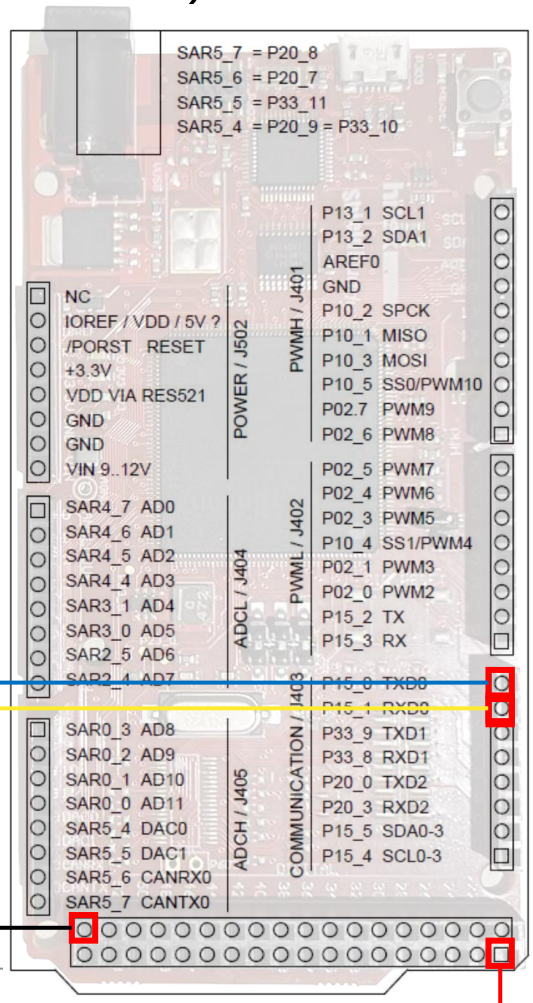
목 차

- 레이저 센서 모듈 및 실습
  - 레이저 센서 동작 원리 및 데이터 프레임 구조
  - 레이저 센서 거리 수신 프로그래밍 예제

- 전방 긴급충돌방지(AEB) 기능 구현 프로젝트

레이저 센서를 이용한 거리 수신 예제

- **TC275 ShieldBuddy와 결합**
  - P15.0 – ASC1_TX(레이저 센서의 RX에 연결)
  - P15.1 – ASC1_RX(레이저 센서의 TX에 연결)
  - UART1 → TOF센서와 연결
  - UART3 → PC와 연결



레이저 센서를 이용한 거리 수신 예제

- 레이저 센서 값 수신을 위해 **UART1** 모듈 초기화 필요
  - _init_uart1() 함수 사용 (Drivers/asclin.c)

```
void _init_uart1(void)
{
    IfxAsclin_Asc_Config ascConf;

    IfxAsclin_Asc_initModuleConfig(&ascConf, &MODULE_ASCLIN1);

    ascConf.baudrate.baudrate = TOF_BAUDRATE;
    ascConf.baudrate.oversampling = IfxAsclin_OversamplingFactor_16;
    ascConf.bitTiming.medianFilter = IfxAsclin_SamplesPerBit_three;
    ascConf.bitTiming.samplePointPosition = IfxAsclin_SamplePointPosition_8;

    const IfxAsclin_Asc_Pins pins = {
            .cts         = NULL_PTR,                    .ctsMode    = IfxPort_InputMode_pullUp,
            .rx          = &IfxAsclin1_RXA_P15_1_IN,    .rxMode     = IfxPort_InputMode_pullUp,
            .rts         = NULL_PTR,                    .rtsMode    = IfxPort_OutputMode_pushPull,
            .tx          = &IfxAsclin1_TX_P15_0_OUT,    .txMode     = IfxPort_OutputMode_pushPull,
            .pinDriver   = IfxPort_PadDriver_cmosAutomotiveSpeed1
    };
    ascConf.pins = &pins;

    ascConf.txBuffer = g_uartTxBuffer_1;
    ascConf.txBufferSize = ASC_TX_BUFFER_SIZE;
    ascConf.rxBuffer = g_uartRxBuffer_1;
    ascConf.rxBufferSize = ASC_RX_BUFFER_SIZE;

    IfxAsclin_Asc_initModule(&g_ascHandle1, &ascConf);
}
```

설정값을 저장하기 위한 임시 변수 ascConf 선언 및 초기 설정값 저장

cts, rts 신호선 미사용 설정

ascConf에 설정값 저장

ascConf에 설정된 값으로 모듈 초기화

레이저 센서를 이용한 거리 수신 예제

- 레이저 센서 값 수신을 위해 **UART1** 모듈 초기화 필요
  - _out_uart1, _in_uart1, _poll_uart1 함수 (Drivers/asclin.c)

```
void _out_uart1(const unsigned char chr) {
    IfxAsclin_Asc_blockingWrite(&g_ascHandle1, chr);
}

unsigned char _in_uart1(void) {
    unsigned char ch;
    while (_poll_uart1(&ch) == 0);
    return ch;
}

int _poll_uart1(unsigned char *chr){
    unsigned char ch;
    Ifx_SizeT count = 0;
    int res = 0;
    count = IfxAsclin_getRxFifoFillLevel(g_ascHandle1.asclin);
    if(count >= 1) {
        IfxAsclin_read8(g_ascHandle1.asclin, &ch, 1);
        *chr = ch;
        res = TRUE;
    }
    else
        res = FALSE;
    return res;
}
```

레이저 센서를 이용한 거리 수신 예제 (폴링)

```c
#include "main.h"
#include "asclin.h"
#include "ToF.h"
int core0_main(void) {
   (...)
   _init_uart3();  _init_uart1();
   unsigned char buf_tof[16] = { 0 };
   int tof_distance, tof_strength;
   while (1) {
      for (int i = 0; i < 16; i++) { buf_tof[i] = _in_uart1(); } /* Receive 1 frame */

      /* verify header & CheckSum */
      unsigned char checksum = 0;
      for (int i = 0; i < 15; i++) { checksum += buf_tof[i]; }
      if (buf_tof[0] == 0x57 && buf_tof[1] == 0x0 && buf_tof[2] == 0xFF && checksum == buf_tof[15]) {
         /* save distance, signal strength */
         tof_distance = buf_tof[8] | (buf_tof[9] << 8) | (buf_tof[10] << 16);
         tof_strength = buf_tof[12] | (buf_tof[13] << 8);
         /* when distance over 2m - out of range */
         if (tof_strength != 0 && tof_distance != 0xFFFFF6u) {
            my_printf("Received Frame: ");
            for (int i = 0; i < 16; i++) { my_printf("%02X ", buf_tof[i]); }
            my_printf("\nDistance: %dmm\n", tof_distance);
         } else {
            my_printf("Out of Range!\n");
         }
      } else {
         my_printf("Invalid checksum error!\n");
      }
   }
   return 0;
}
```

16바이트 수신

거리 값 출력

```
Received Frame: 57 00 FF 00 BB D8 43 00 9C 01 00 00 14 00 FF DC
Distance: 412mm
Received Frame: 57 00 FF 00 36 D9 43 00 9C 01 00 00 13 00 FF 57
Distance: 412mm
```

레이저 센서를 이용한 거리 수신 예제

- **Checksum이란?**
  - 수신 데이터가 오류 없이 잘 수신되었는지 검사하는 과정
  - 센서는 데이터 프레임에 checksum을 추가하여 전송함
  - 수신측에서는 받은 데이터의 checksum을 계산하여 송신측에서 보낸 checksum과 비교하여 데이터에 오류가 있는지 판별함

송신측

0x57 + 0x00 + 0xFF … + 0x0C + 0x00 + 0xFF = 0xDB

57	00	FF	00	D8	06	91	00	09	02	00	00	0C	00	FF	DB

수신측

데이터 오류 발생

57	00	FF	00	D8	06	91	00	09	02	00	01	0C	00	FF	DB

0x57 + 0x00 + 0xFF … + 0x01 + 0x0C + 0x00 + 0xFF = 0xDC

레이저 센서를 이용한 거리 수신 예제 (폴링)

```c
#include "main.h"
#include "asclin.h"
#include "ToF.h"
int core0_main (void) {
   (...)
   _init_uart3();  _init_uart1();
   unsigned char buf_tof[16] = { 0 };
   int tof_distance, tof_strength;
   while (1) {
      for (int i = 0; i < 16; i++) { buf_tof[i] = _in_uart1(); } /* Receive 1 frame */

      /* verify header & CheckSum */
      unsigned char checksum = 0;
      for (int i = 0; i < 15; i++) { checksum += buf_tof[i]; }
      if (buf_tof[0] == 0x57 && buf_tof[1] == 0x0 && buf_tof[2] == 0xFF && checksum == buf_tof[15]) {
         /* save distance, signal strength */
         tof_distance = buf_tof[8] | (buf_tof[9] << 8) | (buf_tof[10] << 16);
         tof_strength = buf_tof[12] | (buf_tof[13] << 8);
         /* when distance over 2m - out of range */
         if (tof_strength != 0 && tof_distance != 0xFFFFF6u) {
            my_printf("Received Frame: ");
            for (int i = 0; i < 16; i++) { my_printf("%02X ", buf_tof[i]); }
            my_printf("\nDistance: %dmm\n", tof_distance);
         } else {
            my_printf("Out of Range!\n");
         }
      } else {
         my_printf("Invalid checksum error!\n");
      }
   }
   return 0;
}
```

헤더, 체크섬 확인

Data	Type	Length (Bytes)	Hex
Frame Header	uint8	1	57
Function Mark	uint8	1	00
reserved	uint8	1	…

헤더, 체크섬 확인

```
Received Frame: 57 00 FF                    01 00 00 14 00 FF DC
Distance: 412mm
Received Frame: 57 00 FF 00 36 D9 43 00 9C 01 00 00 13 00 FF 57
Distance: 412mm
```

레이저 센서를 이용한 거리 수신 예제

- 레이저 센서의 유효 측정 거리를 벗어난 경우
  - Short-Range: 거리 값으로 0xFFFFF6 (-0.01) 전송

> In short-range mode, when the range is exceeded, the distance output is a fixed value of -0.01 (0xFFFFF6 in hexadecimal).

```
57 00 FF 00 FB CC B3 00 F6 FF FF 04 00 00 FF C7   Distance: 16777206mm
57 00 FF 00 3F CD B3 00 F6 FF FF 02 00 00 FF 0A   Distance: 16777206mm
57 00 FF 00 C9 CD B3 00 05 03 00 02 01 00 FF A9   Distance: 773mm
```

  - Medium, Long-Range: 1~2미터 랜덤 값 전송
    - 신호 세기를 확인하여 측정 거리를 벗어났는지 확인해야 함
    - 측정 거리를 벗어나면 신호 세기(13번, 14번째 바이트)가 0이 됨

> In medium-range mode, when the range is exceeded, the distance output jumps randomly between 1-2 meters. At this time, you can refer to the signal strength and distance status for judgment.
>
> In long-range mode, when the range is exceeded, the data output jumps randomly between 1-2 meters. At this time, you can refer to the signal strength and distance status for judgment.

레이저 센서를 이용한 거리 수신 예제 (폴링)

```c
#include "main.h"
#include "asclin.h"
#include "ToF.h"
int core0_main(void) {
   (...)
   _init_uart3();  _init_uart1();
   unsigned char buf_tof[16] = { 0 };
   int tof_distance, tof_strength;
   while (1) {
      for (int i = 0; i < 16; i++) { buf_tof[i] = _in_uart1(); } /* Receive 1 frame */

      /* verify header & CheckSum */
      unsigned char checksum = 0;
      for (int i = 0; i < 15; i++) { checksum += buf_tof[i]; }
      if (buf_tof[0] == 0x57 && buf_tof[1] == 0x0 && buf_tof[2] == 0xFF && checksum == buf_tof[15]) {
         /* save distance, signal strength */
         tof_distance = buf_tof[8] | (buf_tof[9] << 8) | (buf_tof[10] << 16);
         tof_strength = buf_tof[12] | (buf_tof[13] << 8);
         /* when distance over 2m - out of range */
         if (tof_strength != 0 && tof_distance != 0xFFFFF6u) {
            my_printf("Received Frame: ");
            for (int i = 0; i < 16; i++) { my_printf("%02X ", buf_tof[i]); }
            my_printf("\nDistance: %dmm\n", tof_distance);
         } else {
            my_printf("Out of Range!\n");
         }
      } else {
         my_printf("Invalid checksum error!\n");
      }
   }
   return 0;
}
```

거리, 신호 세기 저장

거리 신호 세기

```
Received Frame: 57 00 FF 00 BB D8 43 00 9C 01 00 00 14 00 FF DC
Distance: 412mm
Received Frame: 57 00 FF 00 36 D9 43 00 9C 01 00 00 13 00 FF 57
Distance: 412mm
```

레이저 센서를 이용한 거리 수신 예제 (폴링)

```c
#include "main.h"
#include "asclin.h"
#include "ToF.h"
int core0_main(void) {
   (...)
   _init_uart3();  _init_uart1();
   unsigned char buf_tof[16] = { 0 };
   int tof_distance, tof_strength;
   while (1) {
      for (int i = 0; i < 16; i++) { buf_tof[i] = _in_uart1(); } /* Receive 1 frame */

      /* verify header & CheckSum */
      unsigned char checksum = 0;
      for (int i = 0; i < 15; i++) { checksum += buf_tof[i]; }
      if (buf_tof[0] == 0x57 && buf_tof[1] == 0x0 && buf_tof[2] == 0xFF && checksum == buf_tof[15]) {
         /* save distance, signal strength */
         tof_distance = buf_tof[8] | (buf_tof[9] << 8) | (buf_tof[10] << 16);
         tof_strength = buf_tof[12] | (buf_tof[13] << 8);
         /* when distance over 2m - out of range */
         if (tof_strength != 0 && tof_distance != 0xFFFFF6u) {
            my_printf("Received Frame: ");
            for (int i = 0; i < 16; i++) { my_printf("%02X ", buf_tof[i]); }
            my_printf("\nDistance: %dmm\n", tof_distance);
         } else {
            my_printf("Out of Range!\n");
         }
      } else {
         my_printf("Invalid checksum error!\n");
      }
   }
   return 0;
}
```

유효 측정 거리 검사 및 출력

유효 거리를 벗어난 경우 데이터 프레임

```
Received Frame: 57 00 FF 00 BB D8 43 00 F6 FF FF 00 00 00 FF 1F
```

레이저 센서를 이용한 거리 수신 예제

- 폴링 기반 센서 값 수신의 문제점
  - main 함수에 다른 센서 또는 로직을 수행하는 코드가 추가되는 경우, 레이저 센서는 주기적으로 데이터를 전송하지만 UART1 모듈의 수신 버퍼에서 데이터를 제때 수신하지 않아 오버플로우가 발생하여 제대로 수신되지 않을 수 있음

- 인터럽트 방식의 수신으로 변경 필요
  - UART1 모듈에서 데이터 수신 인터럽트가 발생하면, 인터럽트 핸들러에서 수신 데이터를 바로 저장함

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- **UART1** 모듈 초기화 코드에 인터럽트 핸들러 등록
  - _init_uart1() 함수 변경 (Drivers/asclin.c)

```
IFX_INTERRUPT(asclin1TxISR, 0, ISR_PRIORITY_ASCLIN_TOF_TX);
void asclin1TxISR(void)
{
    IfxAsclin_Asc_isrTransmit(&g_ascHandle1);
}

void _init_uart1(void)
{
     (...)

    ascConf.interrupt.txPriority = ISR_PRIORITY_ASCLIN_TOF_TX;
    ascConf.interrupt.rxPriority = ISR_PRIORITY_ASCLIN_TOF_RX;
    ascConf.interrupt.typeOfService = IfxSrc_Tos_cpu0;

     (...)

    IfxAsclin_Asc_initModule(&g_ascHandle1, &ascConf);
}
```

인터럽트 핸들러 등록

송신을 위한
인터럽트 핸들러 함수 정의

인터럽트 우선순위 및
처리할 cpu 설정

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- **UART1** 수신 인터럽트 핸들러 함수 추가
  - asclin1RxISR() 함수 사용 (IO/ToF.c)

```
IFX_INTERRUPT(asclin1RxISR, 0, ISR_PRIORITY_ASCLIN_TOF_RX);
void asclin1RxISR(void)
{
    static unsigned char rxBuf[16] = { 0 };

    unsigned char c = (unsigned char) _in_uart1();

    rxBuf[rxBufIdx] = c;
    ++rxBufIdx;

    if (rxBufIdx == TOF_length) {
        memcpy(gBuf_tof, rxBuf, TOF_length);
        rxBufIdx = 0;
    }
}
```

수신을 위한
인터럽트 핸들러 함수 정의

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- 체크섬 확인 함수 (IO/ToF.c)
  - 데이터 프레임 헤더 (0x57, 0x00, 0xFF) 검사 및 체크섬 확인
  - 확인 결과 정상이면 1 반환, 그렇지 않으면 0 반환

```
/* 수신 데이터가 정상이면 1, 그렇지 않으면 0 반환 */
static int verifyCheckSum (unsigned char data[])
{
    unsigned char checksum = 0;
    for (int i = 0; i < TOF_length-1; i++)
    {
        checksum += data[i];
    }
    if (data[0] == 0x57 && data[1] == 0x0 && data[2] == 0xFF)
    {
        return checksum == data[TOF_length-1];
    }
    else
    {
        return 0;
    }
}
```

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- 유효 측정 범위 확인 함수 **(IO/ToF.c)**
  - Short-range인 경우, 유효 측정 거리를 벗어나면 0xFFFFF6을 전송함
  - 또한, 유효 측정 거리를 벗어나면 신호 세기가 0이 됨
  - 수신한 거리가 유효한 경우 1 반환, 그렇지 않으면 0 반환함

```
/* 유효 거리인 경우 1 반환, 그렇지 않으면 0 반환 */
int checkTofStrength(unsigned char data[])
{
     TOF_distance = data[8] | (data[9] << 8) | (data[10] << 16);
     TOF_signal_strength = data[12] | (data[13] << 8);
     /* when distance over 2m - out of range */
     if (TOF_signal_strength != 0 && TOF_distance != 0xFFFFF6u) {
          return 1;
     } else {
          return 0;
     }
}
```

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- **레이저 센서 거리 값 반환 함수 (IO/ToF.c)**
  - 레이저 센서 값(mm)을 반환하는 함수
  - 체크섬, 유효 측정 거리를 확인하고 문제가 없으면 거리 값을 반환함
  - 체크섬 실패한 경우 -1을 반환, 측정 거리를 벗어나면 -2를 반환.

```
/* Return Distance(mm) */
int getTofDistance ()
{
    int TOF_distance = 0;
    unsigned char buf_ToF[TOF_length];
    /* copy buf_tof into tmp */
    memcpy(buf_ToF, gBuf_tof, TOF_length);

    if (!verifyCheckSum(buf_ToF))  { return -1; }
    if (!checkTofStrength(buf_ToF)) { return -2; }

    TOF_distance = buf_ToF[8] | (buf_ToF[9] << 8) | (buf_ToF[10] << 16);

    return TOF_distance;
}
```

레이저 센서를 이용한 거리 수신 예제 (인터럽트)

- **main함수에서 ToF 센서 거리 값 측정**

```
#include "main.h"
#include "Drivers/asclin.h"
#include "IO/ToF.h"

/* tof - interrupt */
int core0_main(void)
{
	(...)

	_init_uart3();
	_init_uart1();

	int tof_distance;
	while (1) {
		tof_distance = getTofDistance();
		if (tof_distance == -1) {
			my_printf("Invalid checksum error!\n");
		} else if (tof_distance == 0) {
			my_printf("Out of Range!\n");
		} else {
			my_printf("Distance: %dmm\n", tof_distance);
		}
	}
	return 0;
}
```

레이저 센서 거리 수신 예제

- 장애물과의 거리가 **30cm** 이상일 때, **LED2**을 점등하고 **10cm** 이하일 때 **LED1**를 점등하는 예시

```
#include "main.h"
#include "Drivers/asclin.h"
#include "IO/GPIO.h"
#include "IO/ToF.h"

int core0_main(void) {
   (...)
   _init_uart3();
   _init_uart1();
   Init_GPIO();
   int distance;
   while (1) {
      distance = getTofDistance();
      if (distance >= 300 && distance > 0) {
         setLED2(1);
      } else {
         setLED2(0);
      }
      if (distance <= 100 && distance > 0) {
         setLED1(1);
      } else {
         setLED1(0);
      }
   }
   return 0;
}
```

목 차

- 레이저 센서 모듈 및 실습
  - 레이저 센서 동작 원리 및 데이터 프레임 구조
  - 레이저 센서 거리 수신 프로그래밍 예제

- **전방 긴급충돌방지(AEB) 기능 구현 프로젝트**

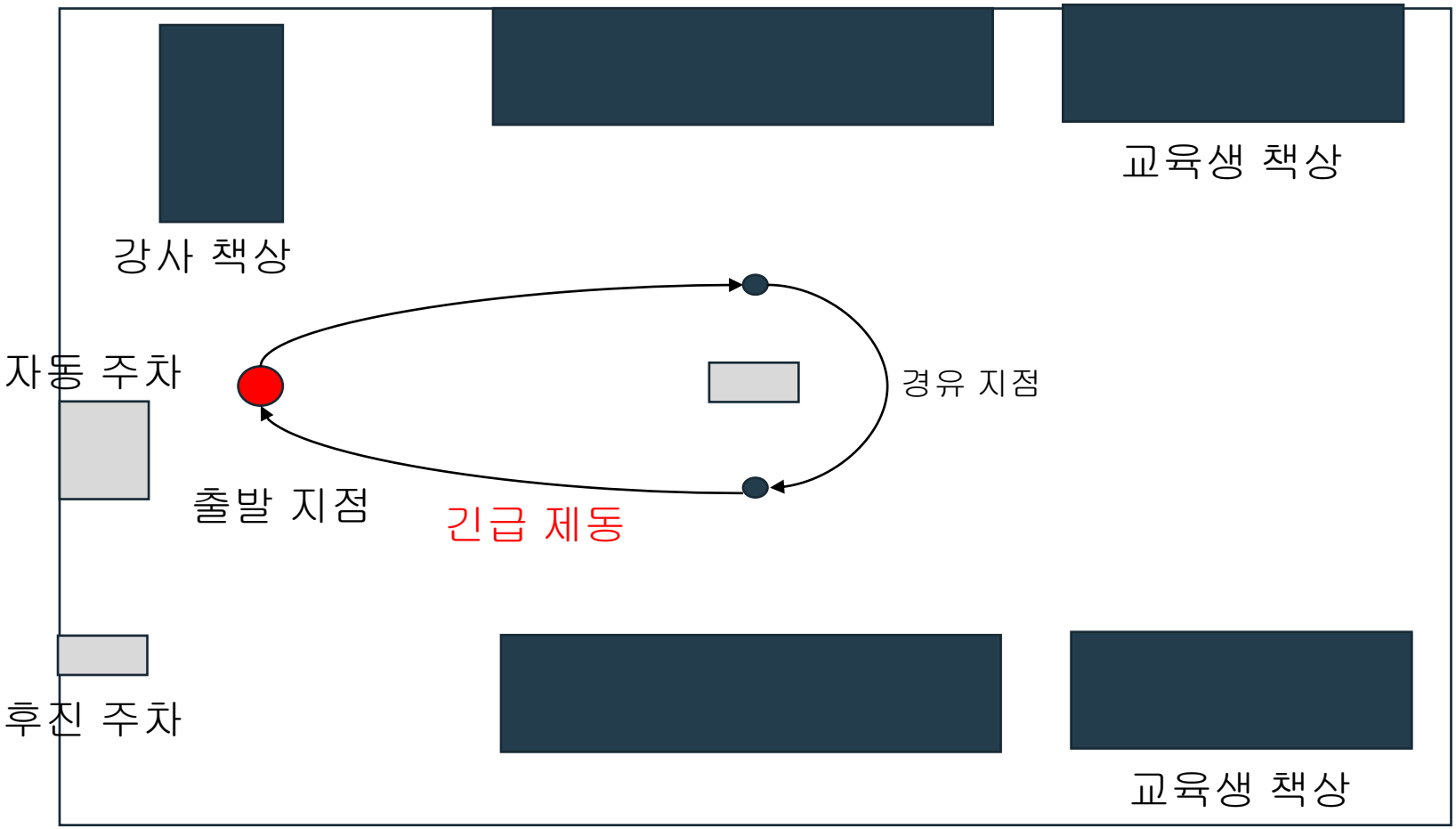
전방 긴급충돌방지 기능 구현 실습

- **RC카를 직진하다가 전방에 물체가 나타나면 긴급 제동하여 차량을 멈추는 기능을 구현하시오.**
  - 현재 모터 속도가 빠르면 제동거리를 더 길게 설정함

전방 긴급충돌방지 (AEB) 기능 구현

- 기존 블루투스 기반 **RC**카 주행 중 전방에 물체가 나타나면 긴급 제동함
  - 긴급 제동이 걸리면 “Emergency Stop!” 등 메시지를 출력하며, 이때는 전진 버튼을 눌러도 전진하지 않으며 후진 등은 가능
  - 후진하여 물체와의 거리가 멀어지면 다시 전진 주행 가능
  - 듀티비에 따라서 긴급제동 거리를 조절하여 장애물에 부딪히지 않도록 할 것
  - 전방 거리 측정은 레이저 센서를 사용함

RC카 기능 통합 및 데모



감사합니다