creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

공학석사 학위논문

선박용 Night Vision System의 제어부 설계 및 구현에 관한 연구

A Study on the Control Unit Design and Realization of Shipboard Night Vision System

지도교수 황 승 욱

2008년 8월

한국해양대학교 대학원

제어계측공학과 백 승 훈

본 논문을 백승훈의 공학석사 학위논문으로 인준함

위원장 김 기 문 

위 원 이 서 정 

위 원 황 승 욱 

2008년 6월 25일

한국해양대학교 대학원

제어계측공학과

목 차

ABSTRACT

Due to the increasing demands of monitoring video and collecting evidence for the enhance of security, safety and surveillance of vessels, Various types of Night Vision System(NVS) have been developed as an effective solution.

The design of a NVS needs to adopt high precision in manufacturing and ability to control the Pedestal which a searchlight and day/night camera are mounted on.

It is essential that the NVS is able to track a moving target by the receipt of a set-point change signal from an ARPA(Automatic Radar Printing Aids) radar. The addition, it should keep to maintain the directivity angle I spite of the position change and movement of vessel in the most gruelling environment of sea.

In the thesis ar designed the controllable units of the NVS such as dat/night CCD(Change Coupled Device) Camera, 2-Axis Pedestal and Zoom lens. And also these units are applied to the NVS and is confirmed the performance of controlling NVS by the experiment.

제 1 장 서 론

1.1. 연구배경

오늘날 지속되는 경제성장은 첨단기술의 발달과 함께 해운의 환경을 끊임없이 변화시키고 있다. 세계 물동량의 98% 이상을 처리하는 선박은 점점 더 대형화 · 고속화 되고 있다. 이에 따라 좌초, 침몰, 충돌로부터 운항중인 선박의 안전성을 확보하기 위하여 항해용 Night Vision System(이하 NVS)등의 항해 보조 장비들의 선박 내 탑재가 시작되고 있다.

또한, 해양이 영토로서의 기능이 점차 중요시되면서, 배타적 경제수역(EEZ) 일대의 어업권의 확보와 해상보안 문제가 세계적인 관심사로 대두되고 있다. 우리나라는 지리적인 특성상 반도에 위치하고 있어, 해상에서 중국, 일본, 북한, 러시아 등과 많은 국제분쟁의 요소를 가지고 있는 실정으로 이에 대비한 불법어로의 효율적 단속 및 국제 분쟁 해결을 위한 감시용 NVS과 같은 해상감시 시스템의 개발 및 적용이 시급한 실정이다.

NVS는 ARPA 레이더로 부터 표적 정보, GYRO로부터 자선의 선수 정보, GPS로부터 자선의 위치 정보를 수신하여 표적까지의 상대방위 및 거리를 도출하고, 페데스털이 표적을 지향, 추적하도록 하는 기능을 갖는다. 페데스털에는 주야간 겸용 CCD, IR 카메라 등이 탑재되며, 이러한 센서에 포착된 영상은 전용 모니터에 전송되어 표시되며, 또한 DVR(Digital Video Recorder)에 기록된다.

선박용 NVS는 크게 항해용 및 해상감시용으로 나눌 수 있다. 항해용은 레이더(RADAR)에 포착되지 않는 물체나 근접 해상 상황을 영상으로 관찰하기 위한 장치이며, 해상 감시용은 해상 범죄 행위에 대한 증거물 확보를 위한 장치로서, 고정도 동요안정화(stabilization) 및 표적 추적(target tracking) 기술이[1-2] 요구된다.

선박용 NVS는 항해 및 감시를 위한 장비 표적을 지속적으로 모니터링하기 위해 빠른 시스템 응답성능과 더불어 해상의 파도, 강한바람 등의 외란에서도 짧은 시간 내에 정상 상태로 돌아오는 외란억제성능이 모두 중요시 되는 시스템이다.

선박에서 사용되는 항해 및 감시용 NVS 같은 지향성 탑재장비는 선박의 운동을 실시간으로 보상하여 항상 표적(Target)을 지향하는 기능이 요구된다. 즉, 롤(Roll), 피

치(Pitch), 요(Yaw), 히빙(Heaving), 서징(Surging), 스웨잉(Swaying)과 같은 선박의 6 자유도운동(6-degree movements)을 실시간으로 보상하기 위한 고정도 동요안정화 기능과 표적 추적 기능을 가지는 페데스털 제작 및 제어 기술이[3-4] 요구된다.

선박운동에 대한 동요안정화 방법으로 이전에는 기계적인 구조의 플라이 휠(Fly-wheel)식 관성을 이용하여 수평을 유지하는 수동적 안정화 방식을 사용하였으나, 최근에는 하드웨어 성능과 제어기술[5]의 향상으로 선박의 운동을 계측하여 직접적으로 운동 성분을 보상하는 능동적 안정화 방식[6-7]을 이용하고 있다.

1.2 연구내용

본 논문에서는 선박용 NVS의 하드웨어 설계 및 구현에 관한 내용이다. 또한 NVS의 하드웨어 설계에 앞서 NVS의 개요에 대해서 분석을 한 뒤 하드웨어 설계에 필요한 정보를 얻고 그 정보를 토대로 구현한 하드웨어를 NVS에 탑재하여 실험을 하였다.

본 논문은 먼저 선박의 6 자유도 운동을 축 단위로 분해하여 검출하여 선박의 운동을 보상할 수 있는 고정도 2축 구조의 안정화 페데스털을 제어하는 PCU(Pedestal Control Unit)를 설계 및 제작하였다. PCU는 페데스털의 동요안정화 및 표적추적 기능을 수행하게 하는 제어장치로써, ARM 코어 프로세서[8], PCL6045 모션 콘트롤 IC[9], 광 자이로 센서(FOG Sensor : Fiber Optic GYRO Sensor) 인터페이스로 구성하였다.

또한 NVS의 페데스털에 탑재되어 있는 카메라, 렌즈, 레이저 서치라이트를 제어하는 ECU(Equipment Control Unit)를 설계 및 제작하였다. ECU는 페데스털의 센서부 장비들을 제어하는 장치로써, AVR 프로세서 인터페이스, 장비 제어를 위한 시리얼 통신 인터페이스로 구성하였다.

그리고 NVS의 전체적인 제어를 위한 MCU(Main Control Unit)을 설계 및 제작하였다. MCU는 항해 장비로부터 각종 정보를 얻고, ECU 및 PCU와 통신을 위한 인터페이스, 사용자를 위한 각종 장치들의 인터페이스[10]로 구성하였다.

PCU, MCU, ECU 등의 하드웨어를 설계 구현한 후에, 구현된 하드웨어들을 이용한 시스템 통합 및 NVS의 실선 테스트(Sea Trial)를 수행하였다.

고성능 프로세서를 탑재한 PCU를 사용하게 됨으로써, 페데스털 제어의 주요 기능인

축 변환 알고리즘, 안정화 알고리즘, 표적추종 알고리즘 및 모터제어 알고리즘 등을 효율적으로 구동시킬 수 있었으며, MCU 및 ECU의 구현으로 페데스털, 카메라와 렌즈, 그리고 레이저 서치라이트도 동시에 효율적인 원격제어가 가능하였다.

본 논문은 제 1장 서론에 이어, 제 2장 NVS의 개요와 하드웨어 구조, 제 3장 NVS의 하드웨어 구성 요소들의 설계 및 구현, 제 4장 구현된 구성요소들 이용한 시스템 통합 및 평가 마지막으로 제 5장 결론으로 구성된다.

제 2 장 선박용 Night Vision System의 개요

2.1 Night Vision System 구성요소

2.1.1 선박의 운동 및 좌표계 변환

운항중인 선박은 그림 2.1과 같이 6자유도(6-Degree) 운동을 하게 된다. 선박의 운동은 직교 좌표 상에서 회전운동 및 병진운동 성분으로 나눌 수 있으며, 병진 운동은 서징(Surging), 스웨잉(Swaying) 및 히빙(Heaving)등 3 성분으로, 회전 운동은 롤링(Rolling), 피칭(Pitching) 및 요잉(Yawing) 성분으로 구성된다.

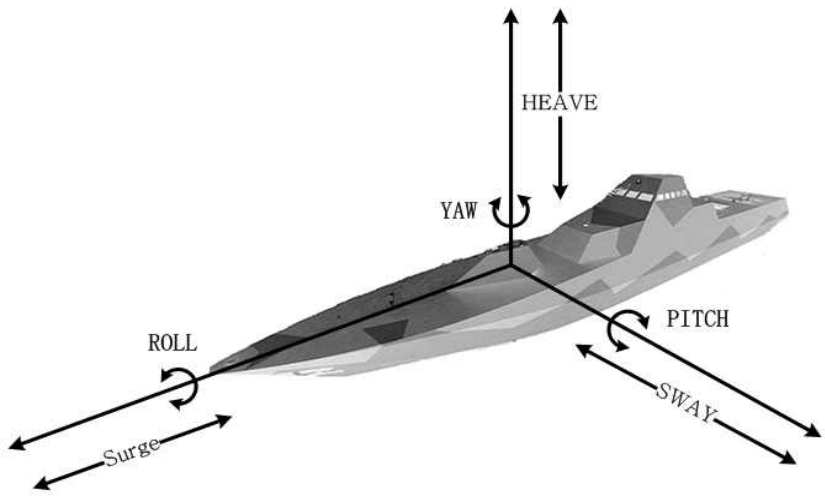


그림 2.1 선박의 6-자유도 운동 성분
Fig. 2.1 6-Degree of freedom motions of ship

선박과 같은 이동 체의 움직임은 표적의 위치를 나타내기 위한 좌표계 기준 프레임의 변화를 유발한다. 이동 체의 운동에 따른 좌표계 기준 프레임의 변화에는 평행이동과 회전등이 있다. 그림 2.2와 같이 프레임에 평행하게 이동하는 좌표계 변위(Coordinate displacement)의 경우, 패럴랙스(Parallax)문제가 유발되며, 그림 2.3 과 같이 방향만이 변하는 좌표계 회전(Coordinate rotation)이 일어났을 때, 안정화(Stabilization)의 문제가 야기된다.

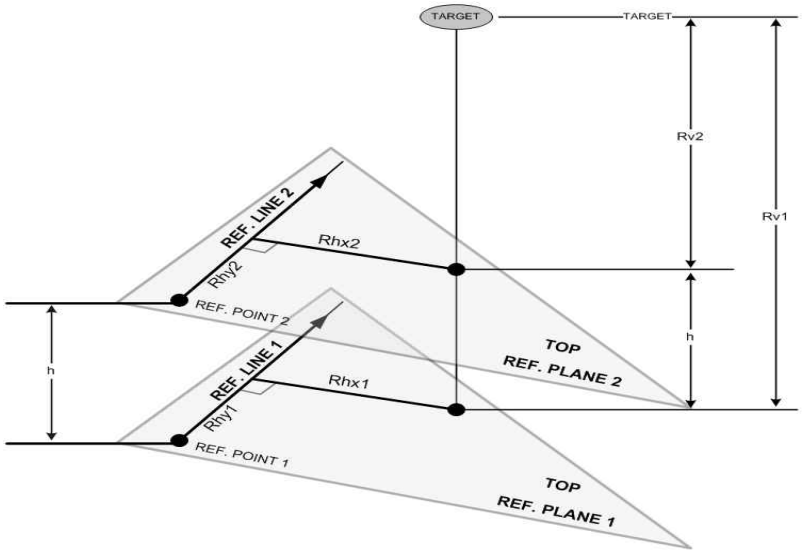


그림 2.2 좌표계의 변위

Fig. 2.2 Displacement of the coordinate system

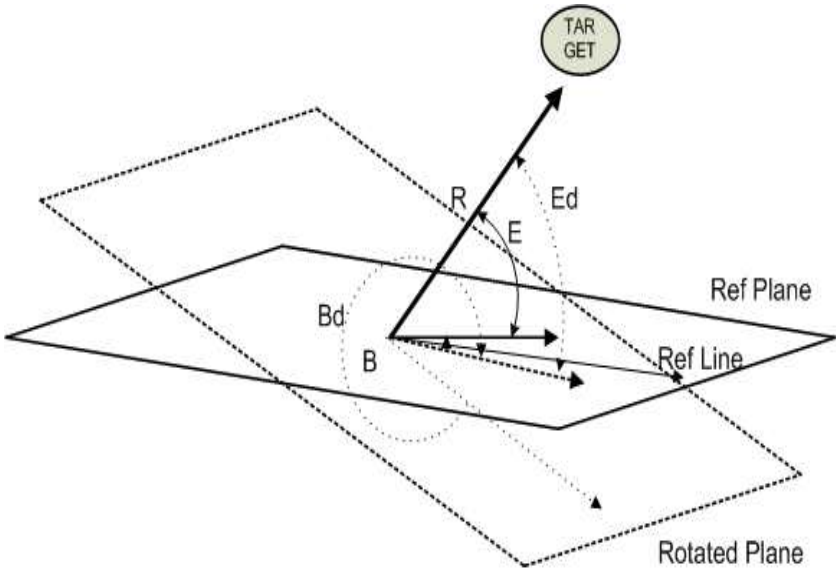


그림 2.3 구면좌표계에서의 회전

Fig. 2.3 Rotation in the spherical coordinate system

이동체용 추적 장치에서 이동 체의 움직임에 따른 기준 프레임의 변화는 패럴랙스의 안정화를 고려한 좌표계의 변환이 필요하게 된다. 본 연구에서 사용한 2축의 페데스털의 구조에 따라 NVS에는 그림 2.4와 같은 3축 회전 운동을 2축 회전 운동으로 변환하는 좌표계 변환 알고리즘(Coordinate Transform Algorithm)이 이전부터 연구되어 져 왔다[1].

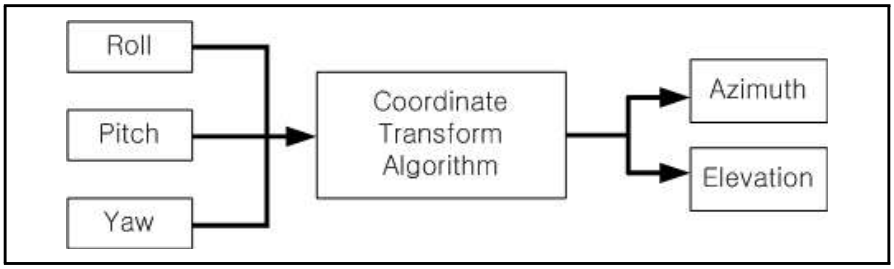


그림 2.4 좌표계 변환
Fig. 2.4 Coordinate Transformation

2.1.2 페데스털 제어 구조

선박용 표적추종 및 동요안정화 시스템은 그림 2.5와 같은 제어과정[6]을 수행한다. 실제 제어를 행하는 전에 언급한 페데스털은 2축 구조로 각 축은 DD(Direct Drive) 모터, 모터 드라이버, 엔코더 그리고 모션 센싱 유닛인 광자이로 센서로 구성된다.

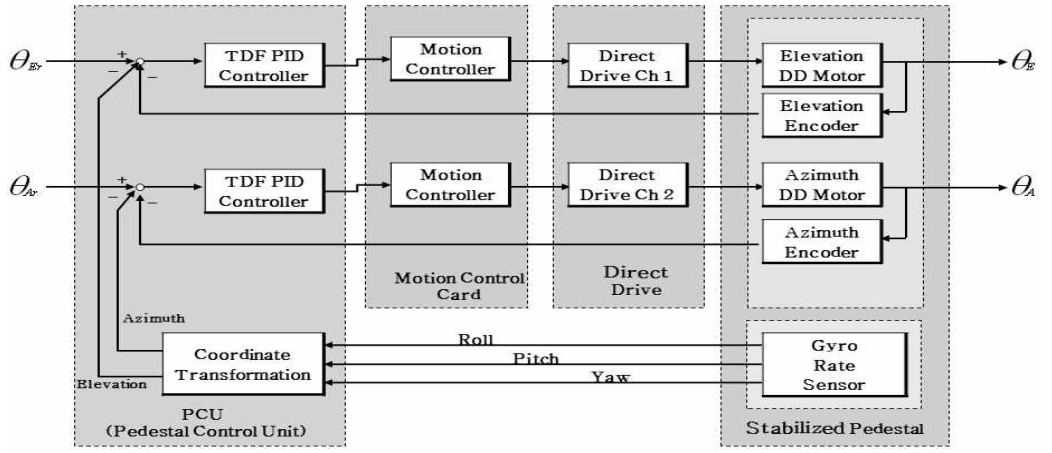


그림 2.5 페데스털의 제어 블록도
Fig. 2.5 Control block diagram of the Pedestal

이 광자이로 센서에서 선박의 3축 회전운동 성분인 롤링(Rolling), 피칭(Pitching), 요잉(Yawing)의 세 가지 회전 운동을 검출하여 PCU로 검출된 데이터를 전달한다. 검출된 3축 회전운동성분은 축 변환(Coordinate Transform)과정을 거치게 된다. 이로써 페데스털의 제어 성분이 고각과 방위각 성분으로 변환되며, 기존의 고각과 방위각의 보상 치에 변환된 보상치가 더해져서 TDF PID controller로 전달하게 된다.

여기서 TDF PID 제어기를 선택한 이유는 페데스털의 제어 알고리즘에는 다양한 방법이 있으나 본 연구에는 종래의 PID 제어기의 특성인 설정치 추종성능이 최적화 되도록 파라메타를 설정하면 외란억제성능이 약해지고, 반대로 외란억제성능이 최적화되도록 파라메타를 설정하면 설정치 추종 시 진동이 되는 단점을 보완한 형태로 설정치 추종성능과 외란억제성능을 동시에 만족시킬 수 있는 TDF PID 제어기를 이용하여 페데스털을 제어하였다[11].

TDF PID Controller는 페데스털부의 엔코더 값과 메인 콘트롤러에서 전달된 보상치를 PCU에 전달하고 PCU는 각도 차원의 입력을 모터 드라이버에 전달한다. 모터 드라이버는 출력을 각 차원의 값을 내보내며, 이는 페데스털의 각축을 구동하는 제어 입력으로 쓰이게 된다. 이와 같은 페데스털을 제어하는 일련의 과정을 반복적으로 수행함으로써, 페데스털의 동요안정화 기능은 유지된다.

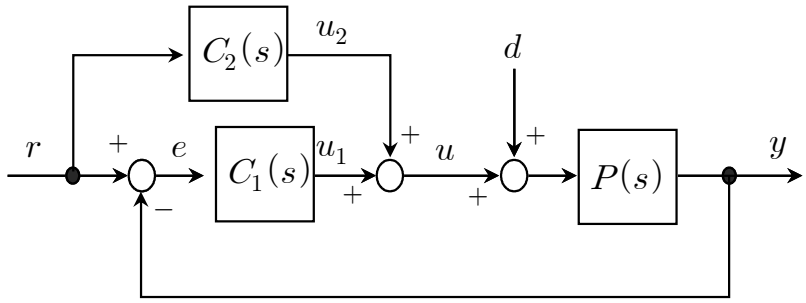


그림 2.6 방위각과 고각 제어용 2 자유도 PID제어 시스템

Fig. 2.6 TDF PID Control system for azimuth and elevation control

이때, r은 설정치, y는 출력, d는 외란, $C_1(s)$와 $C_2(s)$는 TDF PID 제어기이다.

TDF PID 제어기는 기존의 PID 제어기 형태의 주제어기인 $C_1(s)$와 이와는 독립적으로 피드-포워드 게인(Gain)의 역할을 하는 $C_2(s)$로 이루어진 제어기이다. 이러한

제어기의 튜닝은 먼저 주제어기가 이루어진 후 피드-포워드 제어기를 튜닝하는 순서로 이루어진다. $C_1(s)$, $C_2(s)$ 각각의 전달함수는 다음과 같다.

$$C_1(s) = K_P\left\{1 + \frac{1}{\tau_i s} + \tau_d s\right\} \tag{2.1}$$

$$C_2(s) = -K_P\{\alpha + \beta\tau_d s\} \tag{2.2}$$

기존의 PID 제어기의 전달함수와 동일한 식 2.1에는 $K_P\tau_D s$ 형태의 미분기가 들어설정치나 외란이 급격하게 변할 때, 플랜트에 입력되는 u을 지나치게 크게 할 가능성이 있으므로, 필터링 효과를 가미한 식 2.3과 같이 수정하여 이에 대처하도록 한다.

$$C_1(s) = K_P\left\{1 + \frac{1}{\tau_i s} + \tau_d L(s)\right\} \tag{2.3}$$

단, $$L(s) = \frac{s}{1 + \frac{\tau_d}{N}s} \tag{2.4}$$

TDF PID 제어기에서 r에 대한 출력 y의 전달함수 $G_{yr2}(s)$와 외란 d에 대한 출력 y의 전달함수 $G_{yd2}(s)$는 다음과 같이 주어진다. 여기서 아래첨자 '2'는 'Two-Degree-of-Freedom' 의 의미에서 TDF PID 제어기를 지칭한다.

$$G_{yr2}(s) = \frac{P(s)C_1(s) + C_2(s)}{1 + P(s)C_1(s)} \tag{2.5}$$

$$G_{yd2}(s) = \frac{P(s)}{1 + P(s)C_1(s)} \tag{2.6}$$

$C_1(s)$는 기본 파라메타라고 할 수 있는 3가지 파라메타가 있으며 비례이득인 K_P,

적분시간인 τ_i, 미분시간인 τ_d가 이에 해당한다. $C_2(s)$에는 2자유도 파라메타라고 할 수 있는 2가지 파라메타가 있으며, α, β가 이에 해당한다.

TDF PID 제어기는 $C_1(s)$, $C_2(s)$ 두 개의 제어기를 독립적으로 튜닝이 가능하므로 이를 통해 설정치-응답의 전달함수인 $G_{yr2}(s)$와 외란-응답의 전달함수인 $G_{yd2}(s)$를 독립적으로 최적화가 가능하다. 이를 통해 기존의 PID 제어기가 가지는 한계를 극복하여 설정치 추종성능과 외란억제성능을 동시에 만족시킬 수 있다[12-13].

그림 2.7은 TDF PID 제어기의 효과를 나타낸 일러스트이다. 그림에서 보는바와 같이 최적의 기존의 PID 제어기를 통해 발휘할 수 있는 최적의 외란억제성능을 유지하면서 설정치 추종성능을 향상시킬 수 있다.

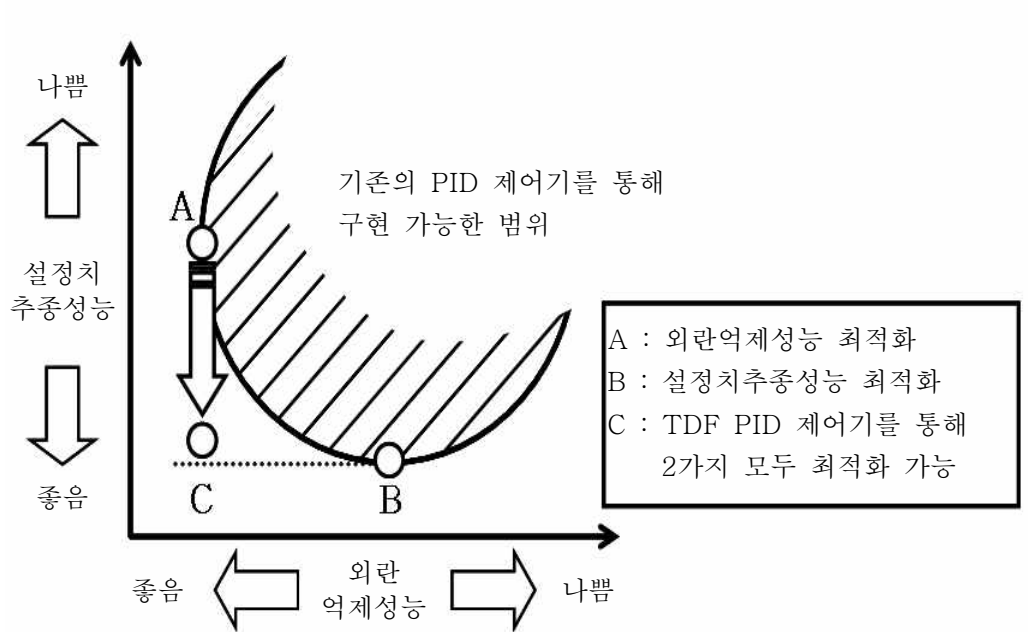


그림 2.7 TDF 구조의 효과의 도시도
Fig. 2.7 Conceptual illustration of the effect of the TDF structure

2.1.3 표적 추종 기능(레이더 데이터 처리)

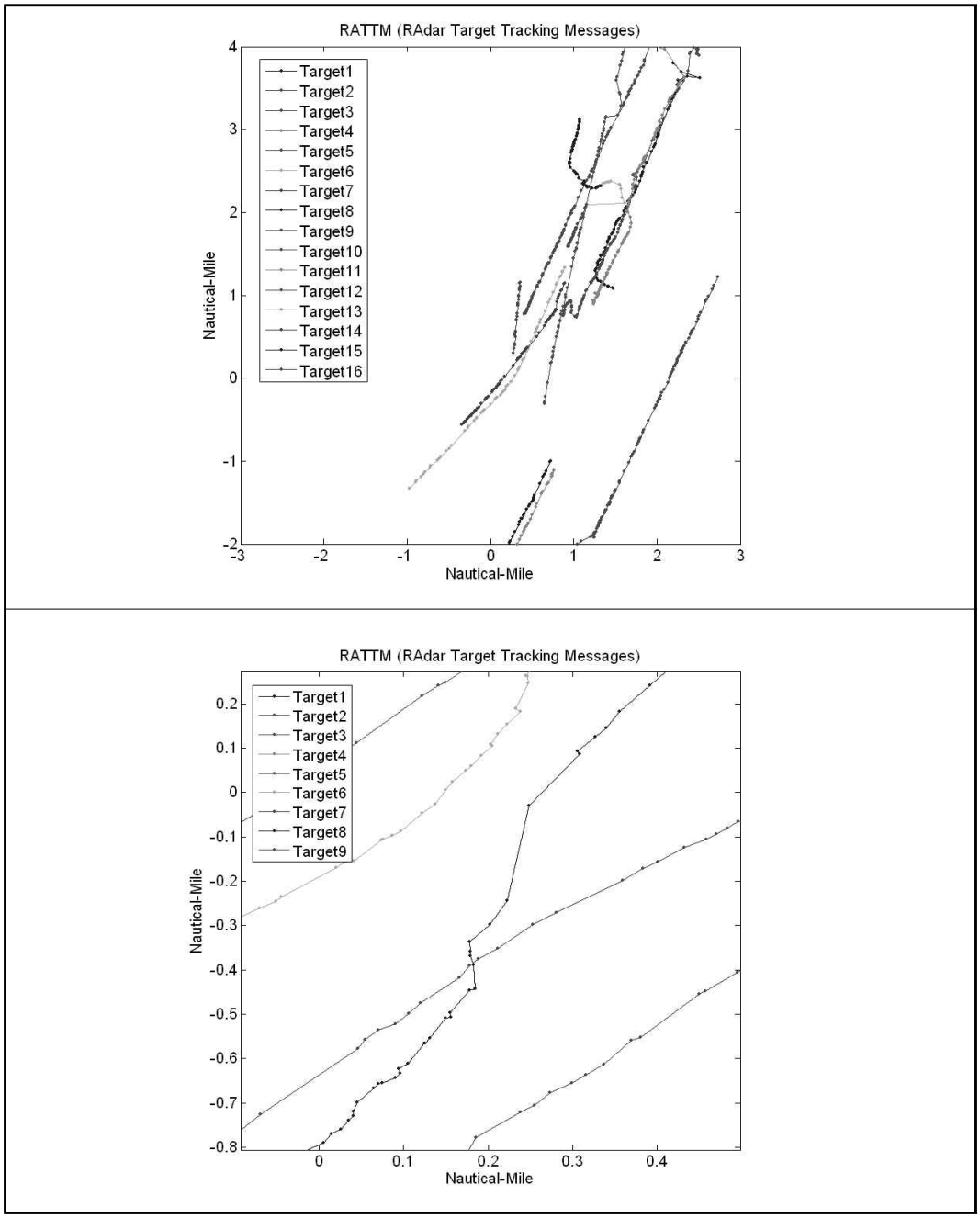


그림 2.8 RATTM 그래프

Fig. 2.8 RATTM graph

표적 추종에는 다양한 요소가 필요하나, 그 중에서 ARPA 레이더를 이용한 표적 추종방법을 이용했다.

그림 2.8은 ARPA 레이더로부터 수신한 RATTM(RADAR Target Tracking Message)을 그래프로 나타낸 것이다. RATTM은 전송 주기가 길고 데이터의 선형성이 떨어진다(그림 2.8). 그래서 다음의 2가지 방법을 통하여 오차 제거 및 선형화를 수행하였다.

첫 번째, 측정위치와 측정 벡터를 바탕으로 Tayler series 방법을 통하여 보간법을 수행하고, LPF(Low Pass Filter), Kalman filter 등을 통한 오차 제거 및 선형화를 하였다.

두 번째, 벡터를 계산적으로 구하기 위해 미분을 수행할 경우, 측정 위치의 오차가 미분에 더욱 큰 오차 성분으로 작용하기 때문에 LPF, Kalman filter등을 통한 오차 제거 및 선형화를 먼저 수행하고, 측정 위치와 측정 위치의 미분을 통해 계산적으로 얻어진 벡터를 바탕으로 Tayler series 방법을 통한 보간법을 수행하고, 다시 LPF, Kalman filter 등을 통한 오차 제거 및 선형화를 수행 하였다.

$$Taylor series \ : \ x(t_k+\Delta t) = x(t_k)+\Delta t x'(t_k)+0.5\Delta t^2 x''(t_k)+\cdots \qquad (2-7)$$

어떤 방법이 더 우수한지 판단하는 손쉬운 방법은 현재위치를 토대로 각각의 방법을 이용해 다음위치를 예측하고 예측된 위치와 다음 측정위치간의 오차가 작은 방법을 선택하는 것이다.

먼저 측정 위치와 계산된 벡터(미분)를 바탕으로 구한 다음 위치의 값을 구하기 위해 측정 위치에서 LPF를 수해하였다. LPF의 Cutoff frequency는 많은 반복을 통해 0.024Hz를 선택하였다. 측정값의 오차를 충분히 줄이기 위해 매우 작은 값으로 선택하였으며, 선택을 위한 목적 함수는

$$f_1(j) = \sum_{t_k=0}^{end\ time} |(Position(t_k+\Delta t)) - (Predicted\,Position(t_k))| \qquad (2-8)$$

를 최소화 하는 값이다.

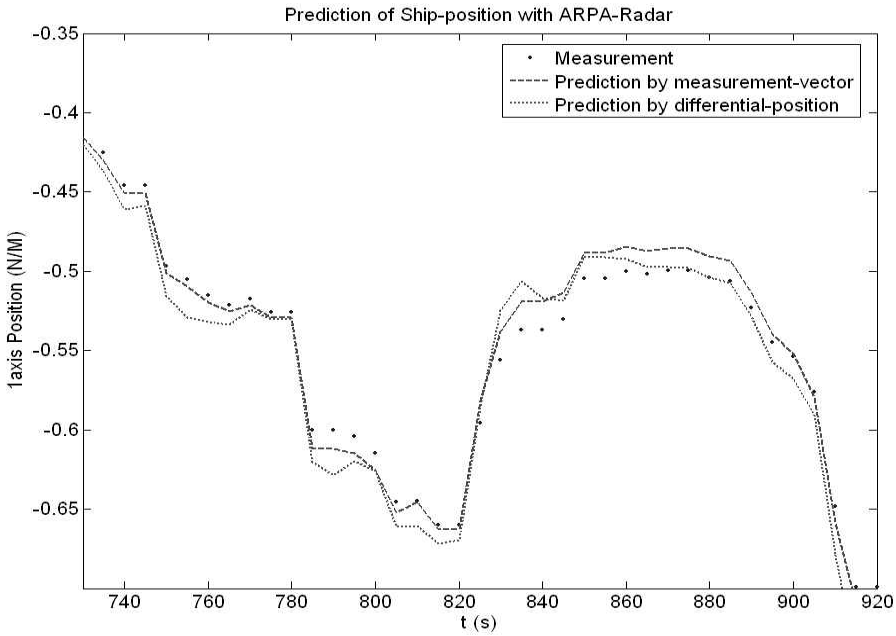


그림 2.9 ARPA 레이더에서 선박 위치의 예측

Fig. 2.9 Prediction of Ship-position with ARPA-Radar

그림 2.9에서 보는 것과 같이 먼저 측정벡터에 의해 예측된 위치는 비교적 오차가 작은 반면, 벡터에 생기는 오차가 일정하여 결과적으로 예측된 위치에 편향성(위 혹은 아래)이 드러난다. 반대로 측정된 위치를 바탕으로 미분을 이용해 예측한 위치는 비교적 오차가 큰 반면, 예측된 위치에 방향성은 존재하지 않는다. 즉 실제위치를 기준으로 위, 아래를 지그재그로 지나간다. 이 때문에 두 가지 방법 가운데 선택을 위해 사용한 목적함수,

$$f_2(j) = \sum_{t_k=0}^{end\ time} |(Position(t_k + \Delta t)) - (Predicted\,Position(t_k))| \qquad (2\text{-}9)$$

에서는 두 가지 방법이 비슷한 수치를 보였지만, 그래프를 통해 확인한 결과 측정벡터를 이용한 방법이 일정한 편향성이 존재하는 단점을 안고서도 다음 측정위치와의 오차가 작았으며, 보다 선형적인 예측 값을 이끌어 낼 수 있다.

2.2 Night Vision System의 구성

NVS는 그림 2.10과 같이 DVR, Display Monitor, Image Stabilizer, MCU(Main Control Unit)이 설치되어 있는 Remote Operator Workstation 과 카메라, 줌 렌즈, 레이저 서치라이트를 제어하는 ECU(Equipment Control Unit), 두 축의 DD(Direct-Drive)모터와 광 자이로 센서가 인터페이스 되어 있는 PCU(페데스털 Control Unit), Stabilized Precision 페데스털로 구성하였다. Stabilized Precision 페데스털에는 ECU와 PCU의 두 제어 유닛이 장착되도록 하였다.

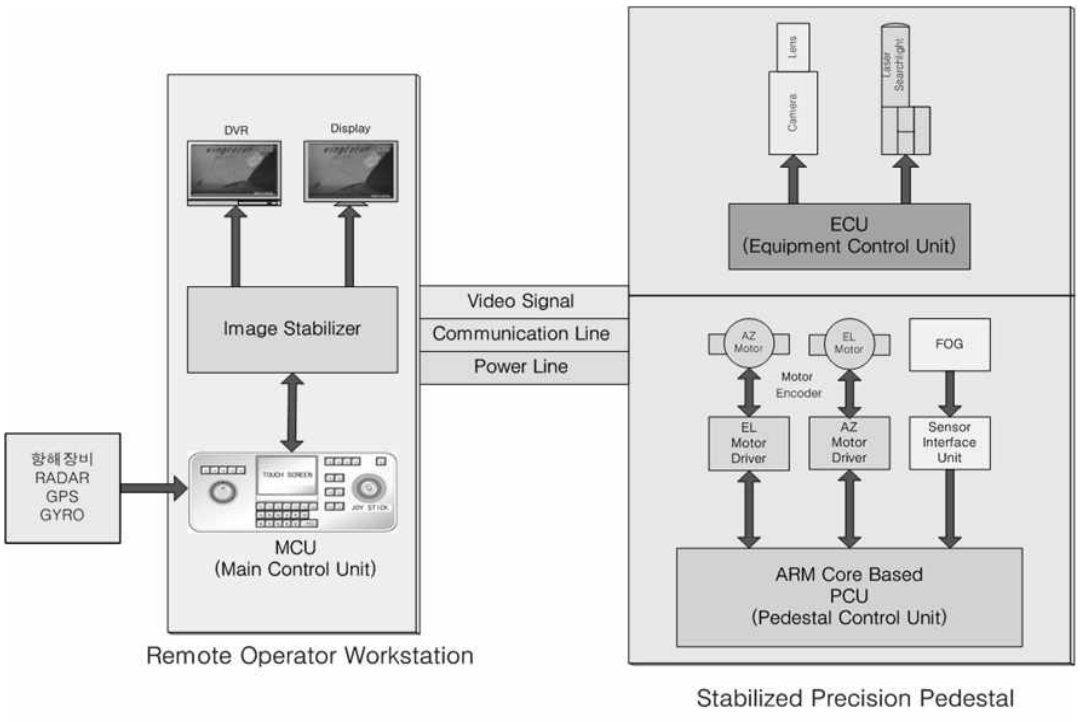


그림 2.10 Night Vision System의 구성도
Fig. 2.10 Night Vision System construction Diagram

NVS는 MCU를 통하여 ECU와 PCU를 제어함으로써 동작이 된다. 그림 2.11에서 보는 것과 같이 제어 신호는 RS422 방식으로 MCU에서 ECU와 PCU로 송신되며,

ECU와 PCU는 그 제어 신호의 명령과 정보를 분석하여 액츄에이터 및 센서 장치들을 구동 시킨다.

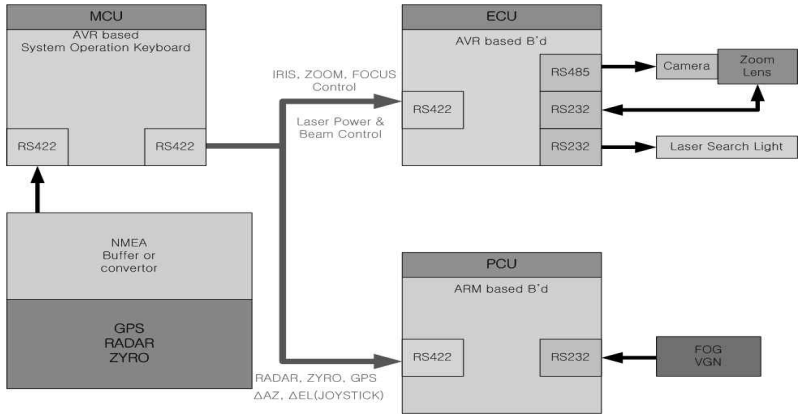


그림 2.11 NVS의 제어와 통신 흐름 블록도
Fig. 2.11 Control & Communication Flow Block Diagram of NVS

2.3 Night Vision System의 기능

NVS에는 3개의 Control Unit이 설치되어 있다. MCU, ECU, PCU 등이 각각 수행하는 기능은 다음과 같다.

표 2.1 Function of Control Units
Table 2.1 Function of Control Units

Function of Control Units	
ECU (Equipment Control Unit)	● 줌 렌즈 제어 ● 레이저 서치라이트 제어 ● 카메라 제어
MCU (Main Control Unit)	● ARPA Radar, GYRO, GPS 등 항해 장비 인터페이스 ● ECU / PCU 와의 통신 인터페이스 ● 사용자 인터페이스 : Joystick, LCD, Keyboard
PCU (페데스털 Control Unit)	● Stabilization : 0.1 degree with OGC ● Coordinate Transformation : – Ship Rotation –> Azimuth & Elevation ● Precision Motor Control : DD-Motor ● ARPA 레이더와 조이스틱 연동

2.3.1 Equipment Control Unit의 기능

ECU는 페데스털의 외부에 있는 카메라, 줌 렌즈, 레이저 서치라이트를 제어하기 위해 설치된 제어 유닛이다. 각 장비들은 RS485 또는 RS232 방식으로 ECU와 연결하였다. ECU는 MCU와의 통신 외에도 각 장비들과도 통신을 수행하여야 하기 때문에 장비의 특성에 맞는 통신 인터페이스를 설계하였다.

CCD 카메라는 빛이 없는 야간에도 촬영이 가능한 극초저조도 CCD 카메라를 사용하였으며, 원격 제어를 위하여 RS485 통신 방식을 적용하였다. 표 2.2는 카메라의 프로토콜에 대한 구조를 나타낸 것이다. 총 11Byte를 가지며 메뉴 ON/OFF, 메뉴 Up/Down/Left/Right 기능을 ECU에서 카메라로 전송하게 된다.

표 2.2 카메라 프로토콜 구조
Table 2.2 Camera Protocol Construction

	Byte1	2	3	4 | 5	6	7	8	9	10	11
Command	STX	ID	ADDR	Command	DATA3	DATA4	DATA5	DATA6	ETX	CHECK SUM
Return	STX	ID	ADDR	Command	DATA3	DATA4	DATA5	DATA6	ETX	CHECK SUM

다음으로 줌 렌즈는 원거리에 있는 물체를 식별하기 위해서 설치하였으며, 아날로그 제어 방식과 RS232를 이용한 제어 방식을 제공한다. 본 연구에서는 RS232를 이용한 방식으로 렌즈를 제어하였다. 표 2.3은 줌 렌즈의 프로토콜 구조를 나타낸 것이다. ECU는 위와 같은 구조로 하여 렌즈에 IRIS, ZOOM, FOCUS 명령을 전송하게 된다.

표 2.3 줌 렌즈 프로토콜 구조
Table 2.3 Zoom Lens Protocol Construction

Command Block Data Construction

Data Length	Function Code	Function Data(Variable Length)	Checksum
1 Byte	1 Byte	0-15 Byte	1 Byte

Data Length

D7	D6	D5	D4	D3	D2	D1	D0

D7-D4 : 0000

D3-D0 : Function Data Length (0-15)

ECU 기능의 마지막으로 레이저 서치라이트 제어가 있다. 서치라이트의 ON/OFF 와 빔 조정(모터 제어), LD 파워 조정에 대한 명령을 RS232 통신 방식으로 전송하게 된다.

표 2.4 레이저 서치라이트 프로토콜 구조
Table 2.4 Laser Searchlight Protocol Construction

Start Code	Command Code	Checksum	Stop code
1 Byte	2 Byte	1 Byte	1 Byte

이렇게 ECU는 NVS에서 표적의 식별을 위한 센서부의 제어가 주 기능이다.

2.3.2 페데스털 Control Unit의 기능

PCU는 페데스털의 초기화, 안정화, 표적추종 기능을 수행한다. 초기화 기능은 각종 하드웨어와 소프트웨어 변수들을 초기화하는 기능이다. 또한 안정화 기능은 광 자이로 센서, 액츄에이터 및 제어 알고리즘으로 구성되며, 광 자이로 센서는 선박의 롤링, 피칭, 요잉의 회전 운동성분을 실시간으로 계측하여 PCU로 전송한다. PCU는 수신된 선박의 3 축 회전 운동성분을 페데스털의 구조에 따라서 방위각과 고각의 2축의 제어성분으로 변환하고 페데스털의 방위각 과 고각 모터를 제어한다. 표적추종 기능은 ARPA 레이더로부터 받은 정보를 이용하여, 선박이 이동하더라도 페데스털이 항상 표적을 지향하도록 한다.

이러한 기능을 하는 PCU는 페데스털 제어를 위한 핵심장치로써, 센서 인터페이스부, 모션 콘트롤 부, 엑츄에이터 구동 부, 통신부 등으로 구성하였다. 센서 인터페이스부는 광 자이로 센서의 출력을 받아서 RS232레벨의 신호를 TTL레벨로 변환하도록 인터페이스를 하였다. 모션 콘트롤 부는 NPM사의 모션 콘트롤러 IC 인 PCL6045를 사용하여 모션 제어를 효율적으로 하도록 설계하였으며, 엑츄에이터 구동 부는 서보모터 드라이버 인터페이스 및 20비트 엔코더 신호의 수신 회로로 구성하였다. 통신부는 MCU로부터 방위각과 고각, 그리고 트랙킹 기능에 필요한 신호를 전송 받으며, 한편, 테스트 프로그램의 다운로드, 메모리 덤프, 테스트 프로그램의 실행 기능을 수행한다.

2.3.3 Main Control Unit의 기능

MCU는 먼저 ARPA 레이더, GYRO, GPS와 같은 항해장비가 인터페이스 되어, ARPA 레이더로 부터 표적 정보, GYRO로부터 자선의 선수 정보, GPS로부터 자선의 위치 정보를 수신하여 표적까지의 상대방위 및 거리를 도출하고 그 정보를 PCU로 전송을 하는 기능을 한다.

ARPA 레이더의 표적정보는 아래 그림 2.12와 같이 IMO Resolution 820(고속정용 항해레이더 설비의 성능기준):1995에 정의된 TTM(Tracked Target Message) 포맷으로 전송되는데, 이 메시지에는 표적번호, 자선의 위치 및 방위로부터 표적까지의 거리 및 상대 방위각 등을 포함한다.

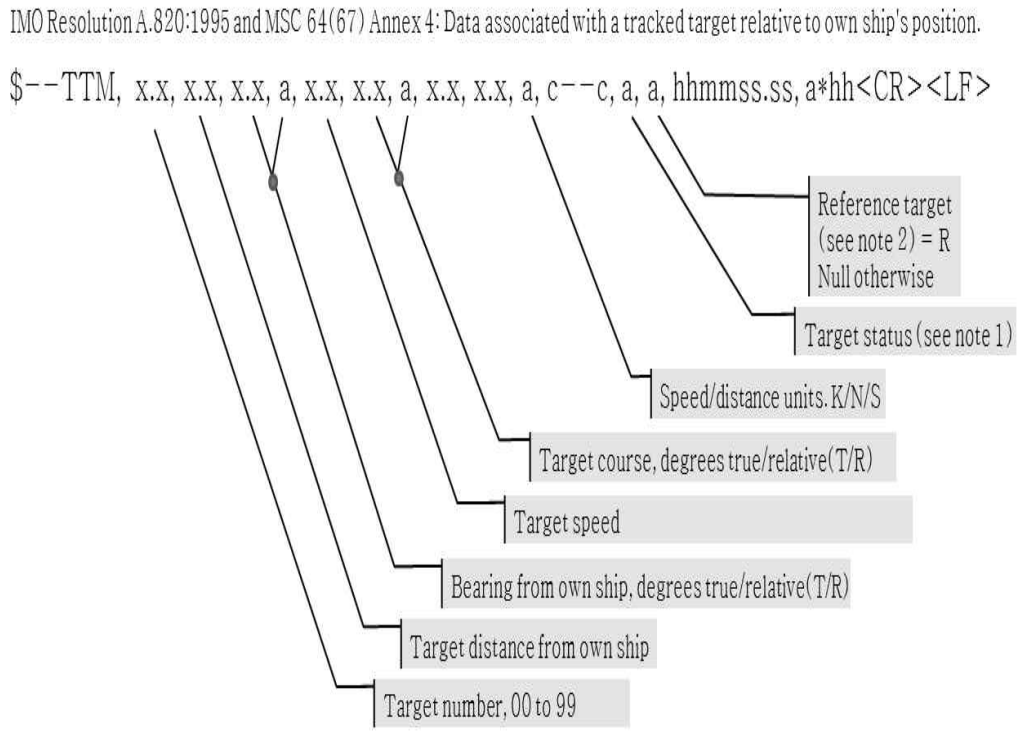


그림 2.12 ARPA 레이다의 TTM 메시지 포맷
Fig. 2.12 TTM message format of ARPA Radar

IMO Resolution 820:1995 및 IEC 61162-1(항법 장치 인터페이스 국제 표준 만족)에 정의된 GYRO의 선수정보 및 GPS의 위치정보의 표준 메시지 포맷은 아래 그림 2.13, 그림 2.14와 같다.

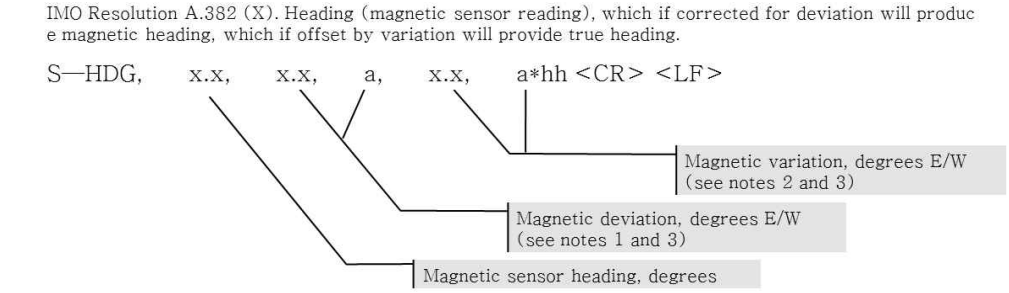


그림 2.13 GYRO의 HDG 메시지 형태
Fig. 2.13 HDG message format of GYRO

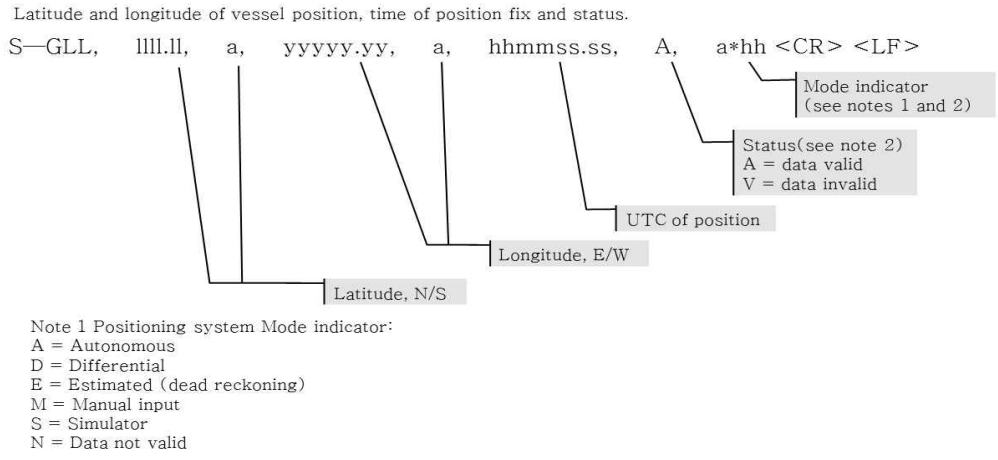


그림 2.14 GPS의 GGL 메시지 형태
Fig. 2.14 GGL message format of GPS

그리고 MCU는 조이스틱, 키보드, 터치 스크린 LCD 모니터와 같이 사용자를 위한 기능을 포함하고 있다. 조이스틱을 사용하여 페데스털을 수동으로 사용자가 조정을 할 수 있고, 키보드를 이용하여 카메라의 셔터 스피드와 주·야간 설정, 그리고 줌 렌즈의 IRIS / FOCUS / ZOOM 조정, 레이저 서치라이트의 조정 등에 대한 명령을 RS422방식의 통신으로 ECU와 PCU에 전송하는 기능을 한다.

제 3 장 Night Vision System Controller H/W 설계 및 구현

NVS의 제어 구조는 그림 3.1과 같다. 이 중에서 마이크로프로세서를 탑재한 장치는 MCU, ECU, PCU의 세 부분의 장치가 있다. 본 논문은 세 부분의 제어 유닛 설계 및 구현에 대한 내용으로써, 본 장에서는 각각의 제어 유닛의 설계 및 구현 과정에 대하여 상술한다.

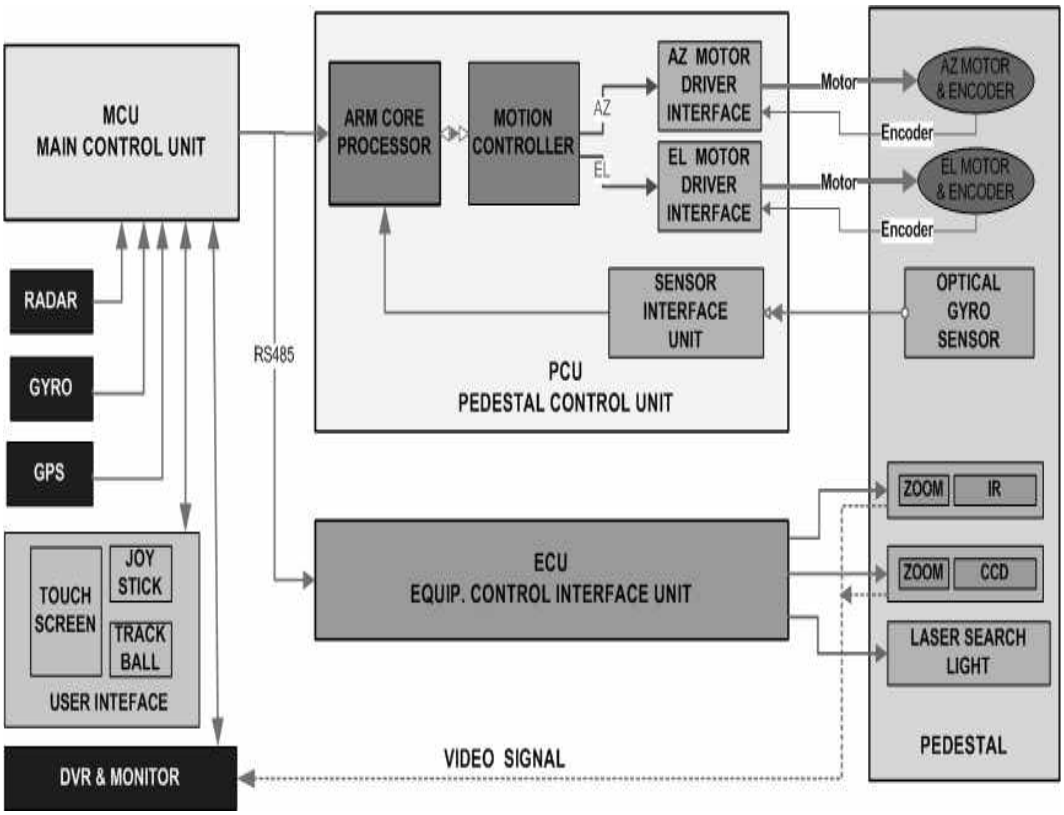


그림 3.1 NVS의 제어 구조
Fig. 3.1 Control Construction of NVS

3.1 Pedestal Control Unit 설계 및 구현

그림 3.2는 PCU의 블록도이다. PCU는 ARM 프로세서가 장착된 콘트롤 보드가 모션 콘트롤러를 제어하도록 설계하고, 모션 콘트롤러의 보호를 위하여 서보 모터 드라이버와의 사이에 포토커플러를 설계하였다. 페데스털에는 2축의 DD-Motor가 설치되어 있어서 모션 콘트롤러는 4축을 지원하지만 2축만 사용하기 때문에 2개의 서보 드라이버 전용 커넥터를 설계하였다. 또한 선박의 운동을 측정하는 3축 광자이로 센서의 정보를 받기 위하여 RS232 방식의 통신 채널 한 개와 MCU와의 통신을 위한 RS422 방식의 통신 채널 한 개를 인터페이스 하였다.

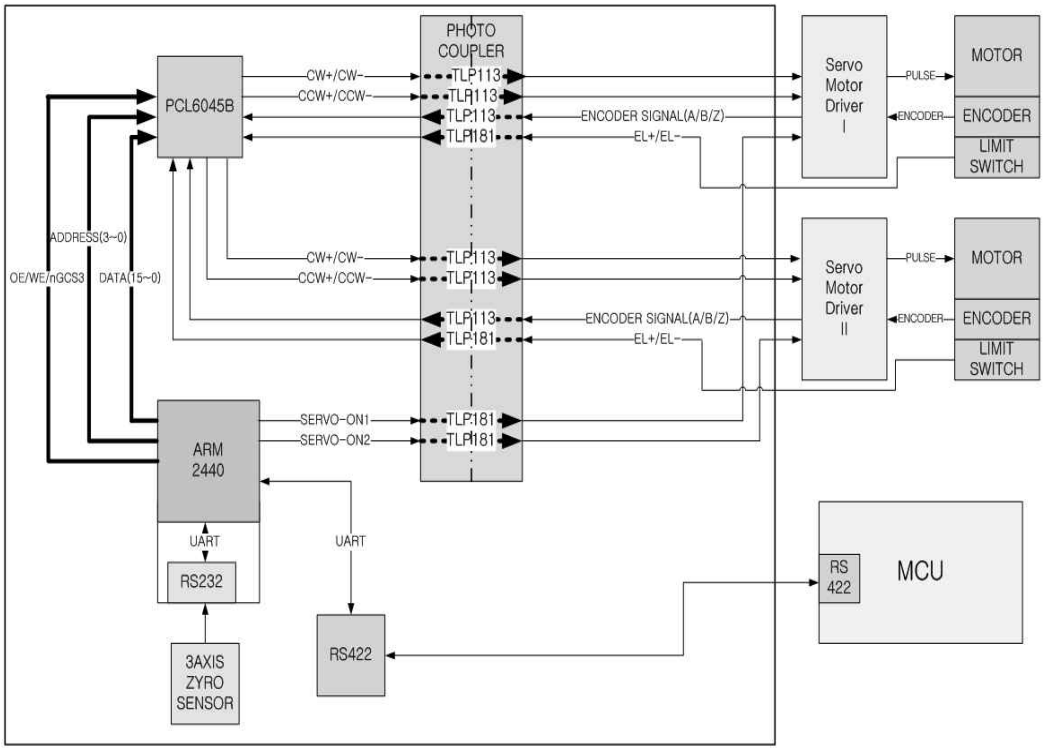


그림 3.2 PCU 블록도
Fig. 3.2 PCU Block Diagram

3.1.1 광자이로 센서 인터페이스

표 3.1 광자이로 센서 커넥터

Table 3.1 Fiber Optic GYRO Sensor Connector

Contact	Name	Description
3	+5V	Power input +5V ±0.25V, 900mA max
4	GND	Power return line, ground, electrically connected to sensor's cover
11	RS232 TXD	Digital output RS232
12	D_GND	Digital ground, connected to "GND"

표 3.1에서 정의된 것과 같이 광자이로 센서의 요구 사항에 맞게 회로를 그림 3.3과 같이 설계하였다.

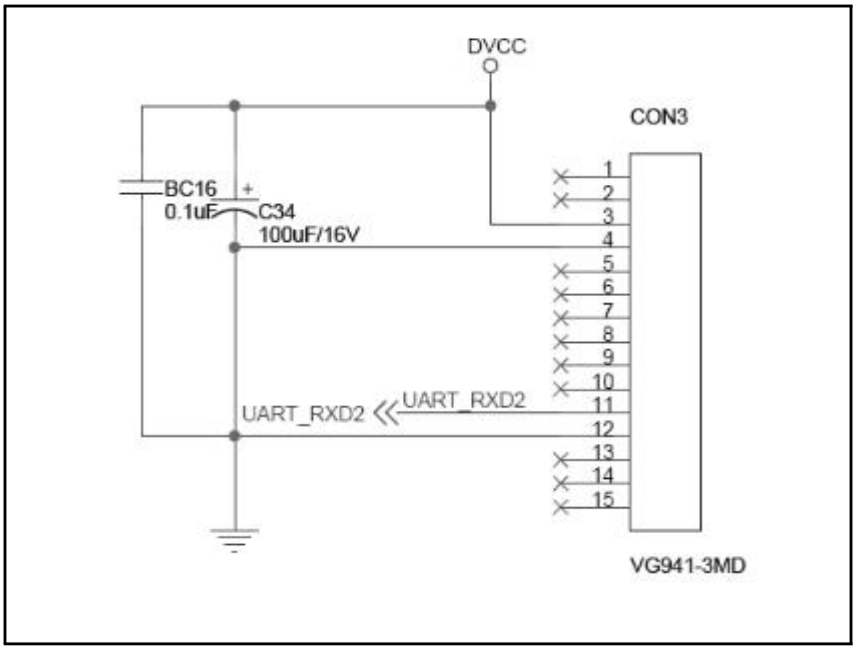


그림 3.3 광자이로 센서 인터페이스 회로

Fig. 3.3 Fiber Optic GYRO Sensor Interface Circuit

3.1.2 포토커플러 인터페이스

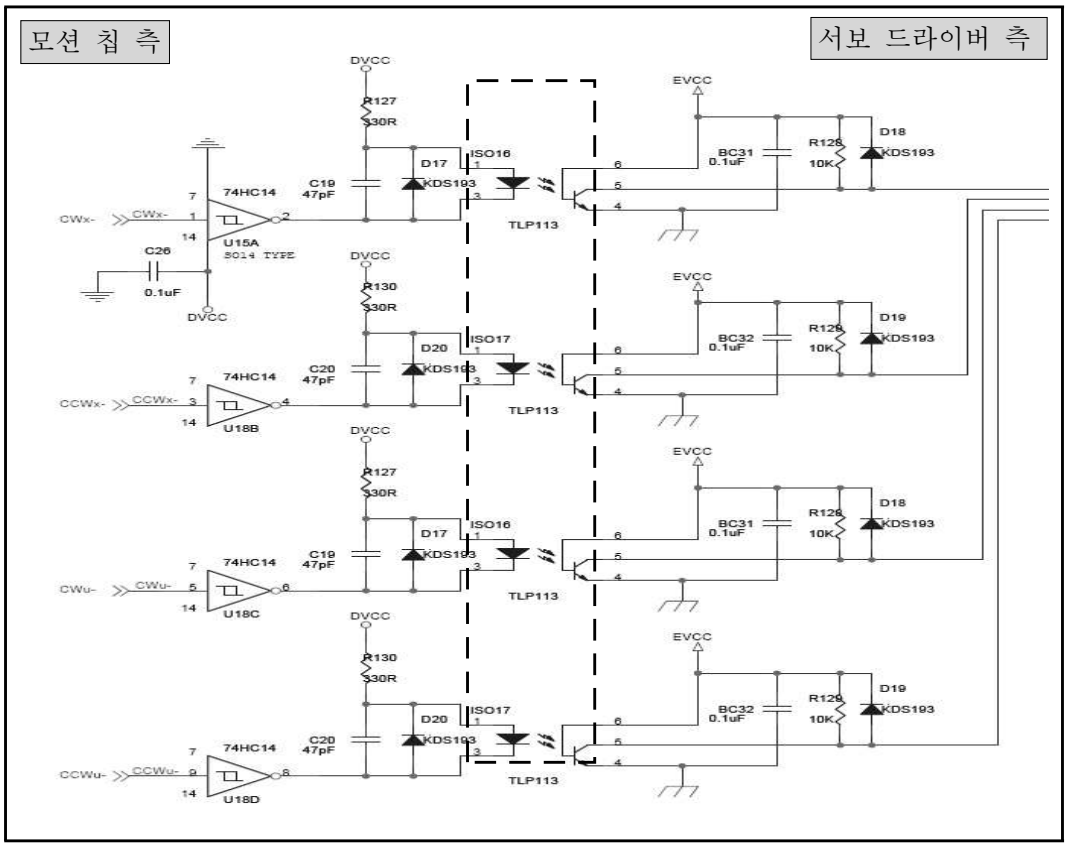


그림 3.4 포토커플러 인터페이스 회로

Fig. 3.4 Photocoupler Interface Circuit

그림 3.4는 PCU에 설계한 포토커플러의 한 예시 회로이다. 비교적 작은 전력에서 회로가 동작하는 마이크로프로세서나 OP앰프의 출력신호로 대전력의 모터를 구동한다고 할 경우 전기적으로만 접속 되었을 때는 모터에서 발생하는 큰 유도 전류가 그라운드나 전원 혹은 신호선을 역으로 타고 들어와 작은 전력회로를 파괴해 버리거나 비교적 빠른 주파수를 취급하는 로직제어회로의 신호의 일부처럼 들어가 회로를 오동작 시키기도 한다.

3.1.3 어드레스 / 데이터 버스 버퍼 인터페이스

PCU에서 ARM 보드로부터 오는 어드레스와 데이터 버스에 버퍼를 모션 콘트롤러 동작에 오류가 발생하지 않게 하였다. 그림 3.5는 ARM 보드로부터 오는 데이터와 어드레스 버스에 3-STATE Octal Bus Transceiver 칩을 설치하여 버퍼의 역할을 수행하게 하였다. 각 칩마다 8bit 버스 라인을 가지고 있기 때문에 16bit의 data bus를 위해 2개의 버퍼 칩을 설치하고, address bus를 위해 1개의 버퍼 칩을 설치했다. 그리고 data 버스는 양방향 전송을 하고, address 버스는 단방향 전송을 하기 때문에 칩의 사양에 맞게 설계를 하였다.

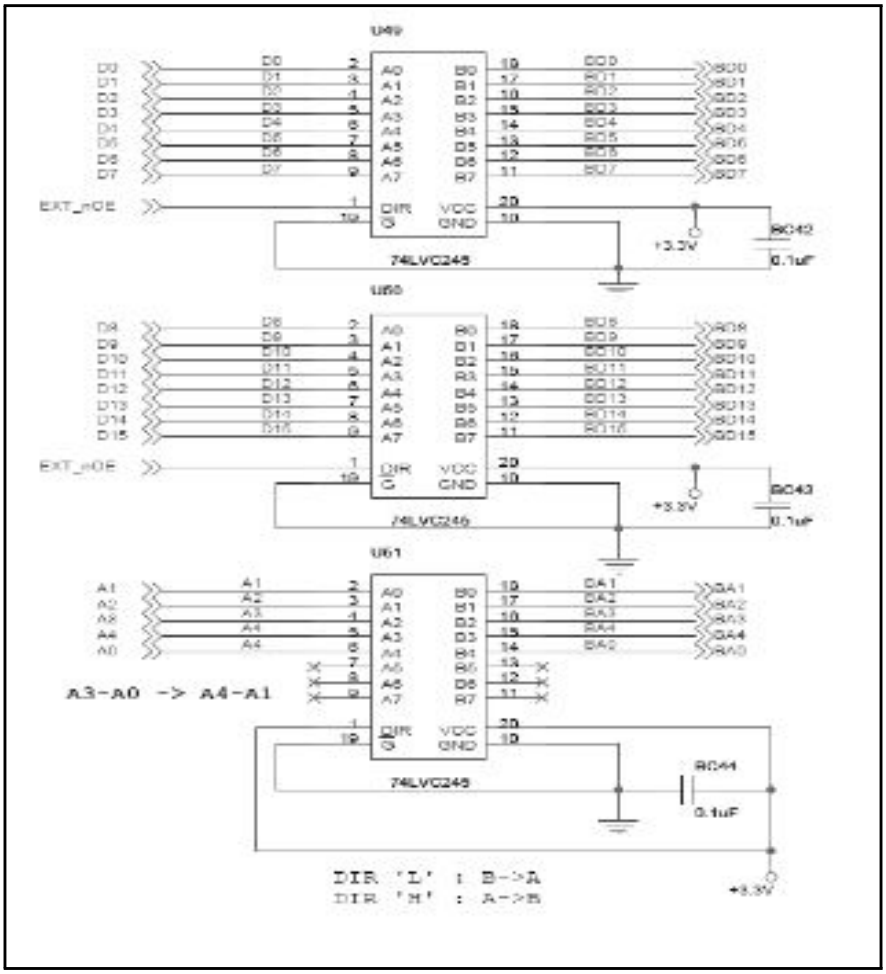


그림 3.5 PCU 어드레스와 데이터 버스 버퍼
Fig. 3.5 PCU Address & Data Bus Buffer

3.1.4 모션 콘트롤러 인터페이스

모션 콘트롤러는 CPU 버스(Bus) 인터페이스에 따라 각종 명령을 이용하여 Stepping Motor, Servo Motor (펄스열 입력) 구동용의 고속 펄스(Pulse) 발진을 목적으로 한 CMOS (complementary metal-oxide semiconductor)구성의 LSI (Large-Scale Integration)이다. 정속, 직선 가/감속, S자 가 감속에 의한 다종·다양한 연속 동작, 위치 결정 동작, 원점 복귀동작 등의 제어를 행할 수 있다. 제어 축수는 4축으로 2~4축의 직선 보간, 임의의 2축의 원호 보간, PCL동작 상황 확인, 각종 조건에 의한 인터럽트(Interrupt) 출력을 행 할 수 있다. 그림 3.7의 회로도는 모션 콘트롤러인 PCL6045 인터페이스 회로를 설계한 것이다.

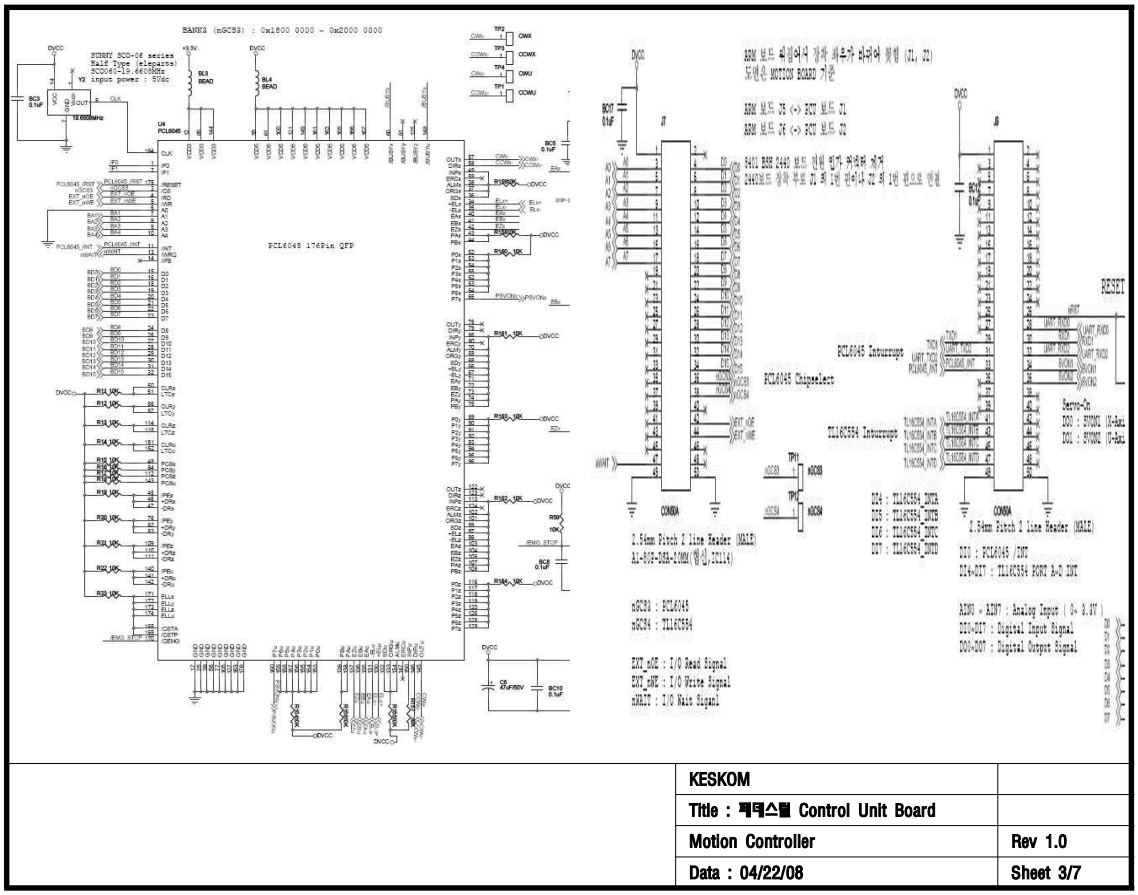


그림 3.6 PCU 모션 콘트롤러 인터페이스 회로

Fig. 3.6 PCU Motion Controller Interface Circuit

3.1.5 RS-422 인터페이스

PCU와 MCU 간의 통신을 위하여 표 3.2에서 보는 것과 같이 1:N 통신과 원거리 통신이 가능한 RS-422 을 사용하였다. ARM Board에서는 기본적으로 제공되는 시리얼 통신 채널이 3개가 있는데, 그 중에서 1개의 채널은 ARM Board의 디버깅용으로 두고, 나머지 2개의 채널을 이용하여 PCU의 통신이 이루어진다. 앞에서 말했듯이 한 채널은 FOG Sensor용의 RS-232 채널이고, 나머지 1개의 채널을 MCU와의 통신을 위한 것으로 할당하여 설계하였다.

표 3.2 RS422 설명
Table 3.2 RS422 Specification

Specification	RS422
동작모드	Differential
최대 Driver/Receiver 수	1Driver 32 Receivers
최대 통달거리	약 1.2km
최고 통신속도	10 Mb/s
지원 전송방식	Full Duplex
최대 출력전압	-0.25 to +6V
최대 입력전압	-7V to +7V

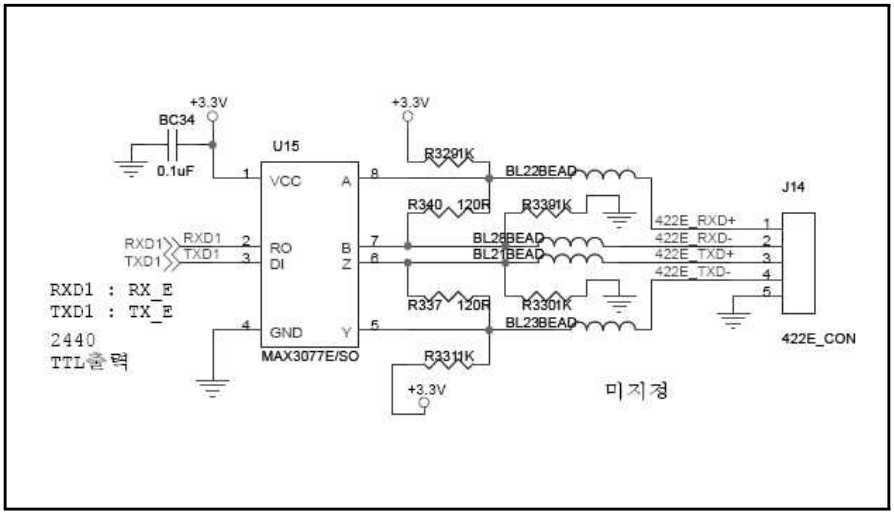


그림 3.7 RS-422 인터페이스 회로도
Fig. 3.7 RS-422 Interface Circuit

3.1.6 PCU 레이아웃과 제작

그림 3.8와 그림 3.9는 PCU PCB 레이아웃 및 구현된 PCU 사진이다. 레이아웃을 할 때 PCU의 디지털 로직 전원(DVCC)과 드라이버단 구동 전원(EVCC)의 절연을 고려하여 포토커플러의 배치를 하였다.

PCU는 페데스털 내에 설치가 되기 때문에 크기와 높이, 그리고 커넥터들의 위치 등을 고려하여 각 칩들을 배치하였다.

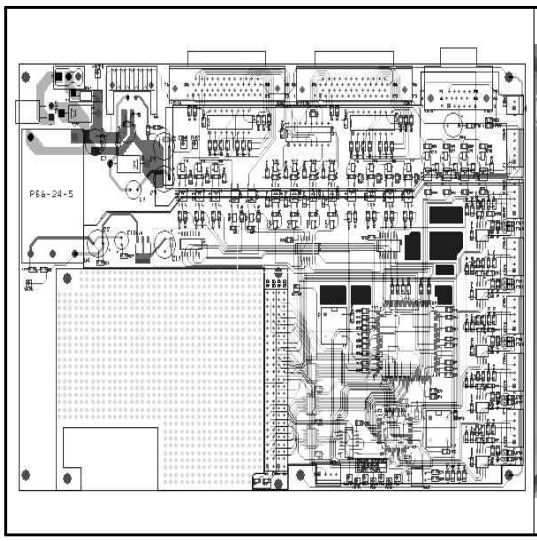


그림 3.8 PCU 레이아웃
Fig. 3.8 PCU Layout

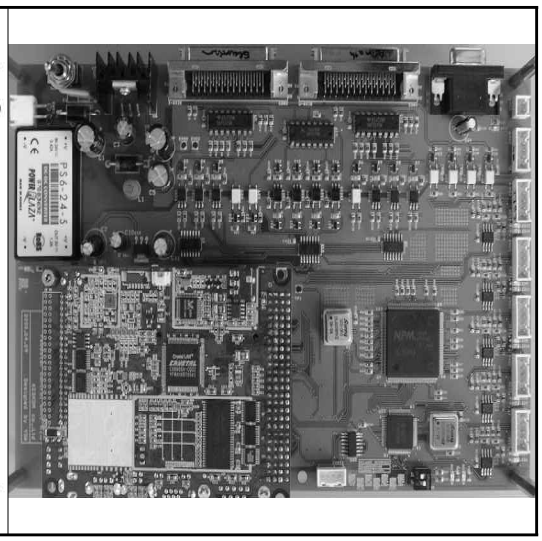


그림 3.9 제작된 PCU 보드
Fig. 3.9 Produced PCU Board

3.2 Main Control Unit 설계 및 구현

MCU용 콘트롤러는 8Bit Microprocessor인 Atmega128(이하 AVR)을 사용하였다, NVS의 정보와 상태들을 모니터링하기 위해서 240×320 픽셀을 가지는 그래픽 LCD을 제어 할 수 있게 Address와 Data Bus, 그리고 read/write 신호를 그래픽 LCD 모듈과 인터페이스 하였다. 조이스틱 정보를 받을 수 있도록 AVR에 내장되어 있는 A/D Converter 중 4채널을 사용할 수 있게 설계하였고, 항해장비의 정보를 받을 수 있는 RS422방식의 통신 인터페이스와 ECU/PCU와의 통신을 위한 RS422를 각각 인터페이스 하였다.

그림 3.10은 MCU의 블록도이다.

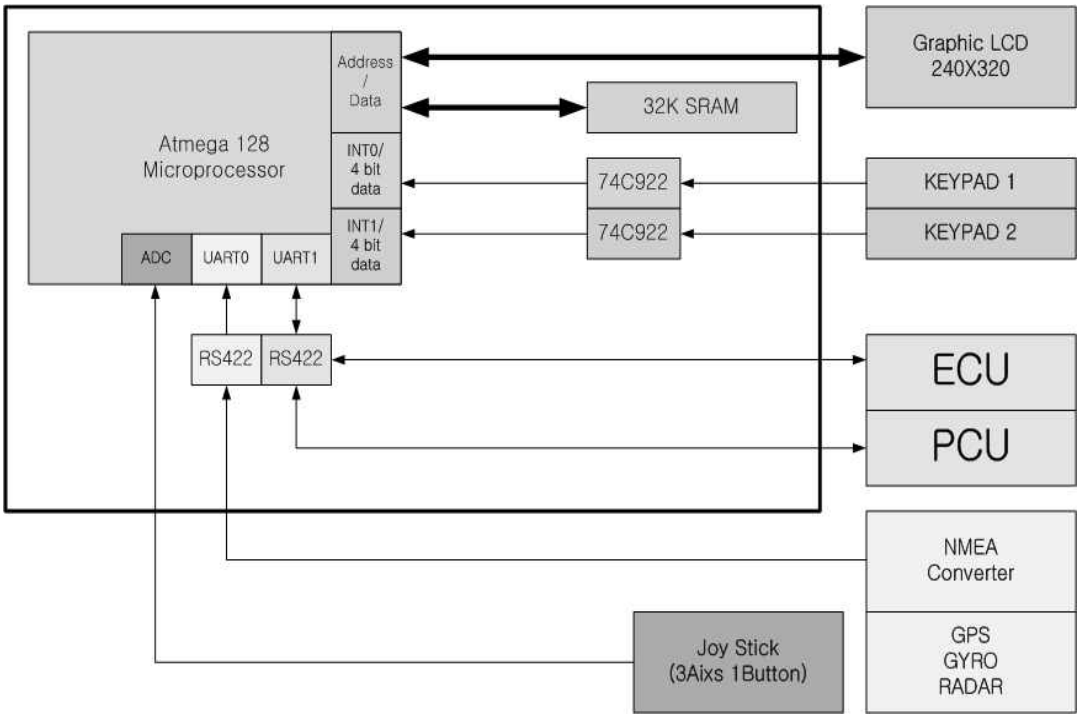


그림 3.10 MCU 블록도
Fig. 3.10 MCU Block Diagram

3.2.1 키패드 인터페이스

2개의 Keypad를 AVR의 포트를 이용하여 Data를 받게 되면 16개의 포트가 필요하게 된다. 그래서 AVR의 사용되는 포트를 줄이고자 16-Key Encoder를 사용하여 12개의 포트만 사용하여 Keypad로부터 Data를 받을 수 있게 하였다. 4개의 Data 출력 핀을 AVR의 포트 중 4핀을 연결하고, Data available 핀은 AVR의 외부 인터럽트 핀에 연결, 그리고 AVR의 1개의 핀을 Encoder의 Output Enable핀에 연결하였다.

그림 3.11는 16-Key Encoder의 블록도이다. 4×4 키 매트릭스 구조를 갖는 키패드를 16-Key Encoder와 어떻게 인터페이스 하는지를 그림 3.12를 통해 알 수 있다.

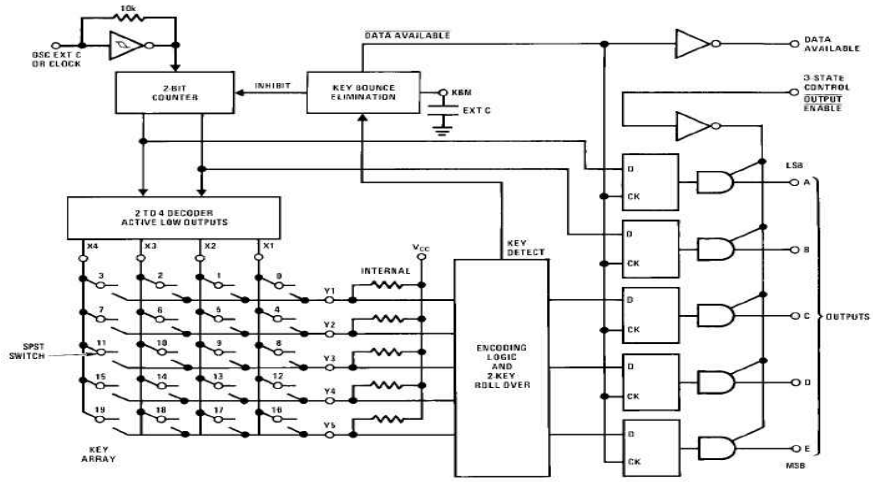


그림 3.11 16-Key Encoder 블록도

Fig. 3.11 16-Key Encoder Block Diagram

3.2.2 LCD 인터페이스

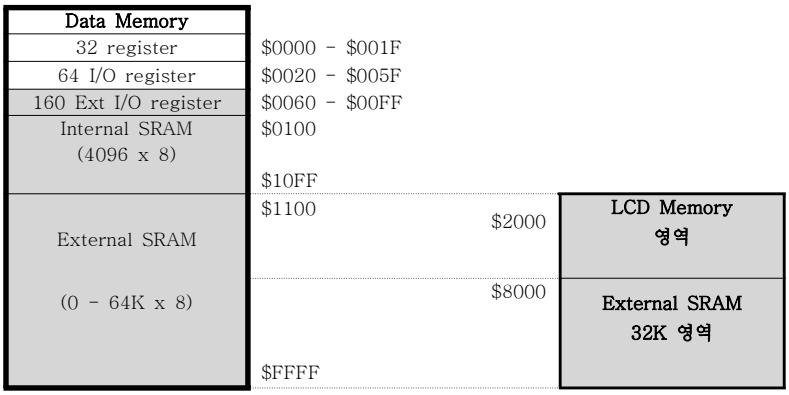


그림 3.12 MCU 메모리 맵

Fig. 3.12 MCU Memory Map

표 3.3은 그래픽 LCD를 MCU에 인터페이스 할 때 필요한 핀들의 배치이다. 리셋은 AVR의 리셋과 동일하게 하였으며, RD/WR은 AVR의 RD/WR에 연결하고, CS는 MCU의 메모리 맵(그림 3.12)에 따라서 AVR의 A15의 값이 '0' 일 때 선택된다. 그리고 Latch A0의 값에 따라 Data / Instruction 을 선택하도록 LCD 모듈의 A0와 연결하였다. DB0-DB7은 AVR의 AD0-AD7에 연결하였다.

표 3.3 그래픽 LCD 모듈 핀 배치

Table 3.3 Graphic LCD module pin-out assignments

Pin No	Symbol	Function
1	RES	Reset (active low)
2	RD	Read select (active low)
3	WR	Write select (active low)
4	CS	Chip select (active low)
5	A0	Select display data/instruction
6	DB0	Bi-directional Data Bus. Data Transfer is performed once, thru DB0 to DB7, in the case of interface data length is 8-bits.
7	DB1	
8	DB2	
9	DB3	
10	DB4	
11	DB5	
12	DB6	
13	DB7	
14	VCC	Supply terminal of module
15	VSS	Ground terminal of module
16	VEE	Positive supply for Liquid Crystal Drive
17	Vadj	Liquid Crystal Display contrast adjust
18	DISPOFF	Display off signal
19	A	Backlight power supply
20	K	Backlight ground

LCD의 전원은 그림 3.13과 같이 설계하여 LCD의 Contrast(Vadj)를 조정 할 수 있게 하였다. LCD 모듈 내에 전원 회로가 있어서 기본적으로 요구하는 전원만 공급을 해주었다.

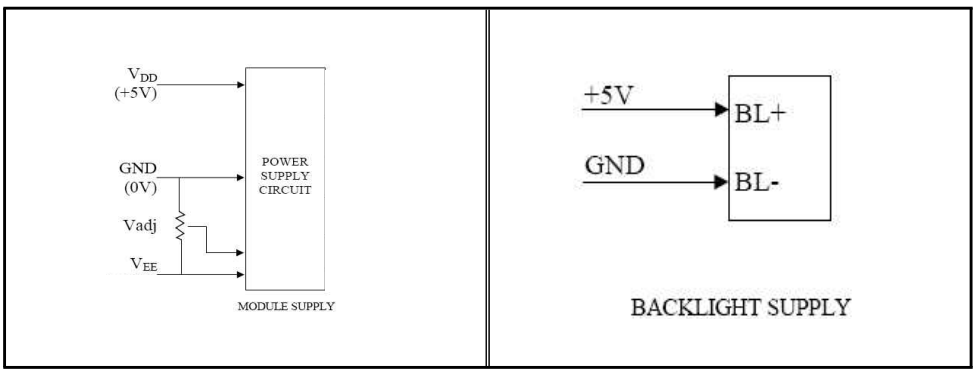


그림 3.13 LCD 전원 공급

Fig. 3.13 LCD Power Supply

3.2.3 Joystick 인터페이스

조이스틱은 3개의 포텐셔미터가 3축의 방향으로 움직이고, 하나의 버튼이 있다. 3개의 포텐셔미터에 각각 전원을 인가하여 포텐셔미터의 아날로그 출력 값을 A/D Converter를 거쳐 디지털 값으로 출력을 할 수 있게 설계를 하였다. 조이스틱의 아날로그 출력 값의 노이즈를 제거하기 위해서 조이스틱의 입력 전압에 간단한 LC 필터(그림 3.14)를 추가 하여 설계하였다.

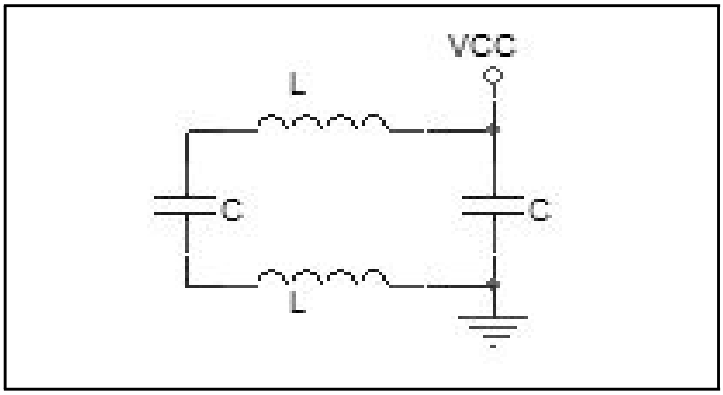


그림 3.14 조이스틱 전원 LC 필터

Fig. 3.14 Joystick Power LC Filter

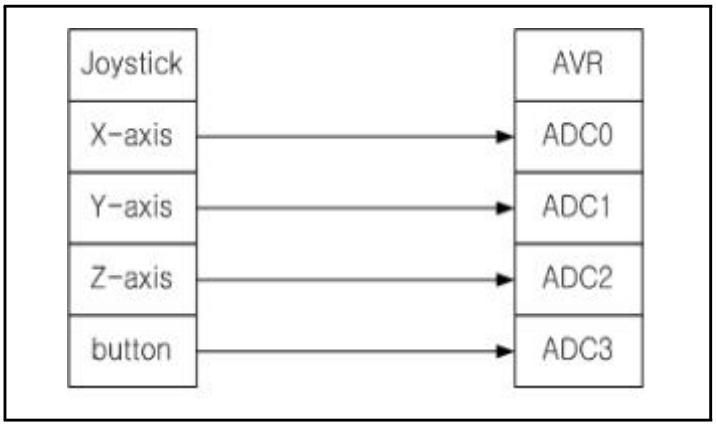
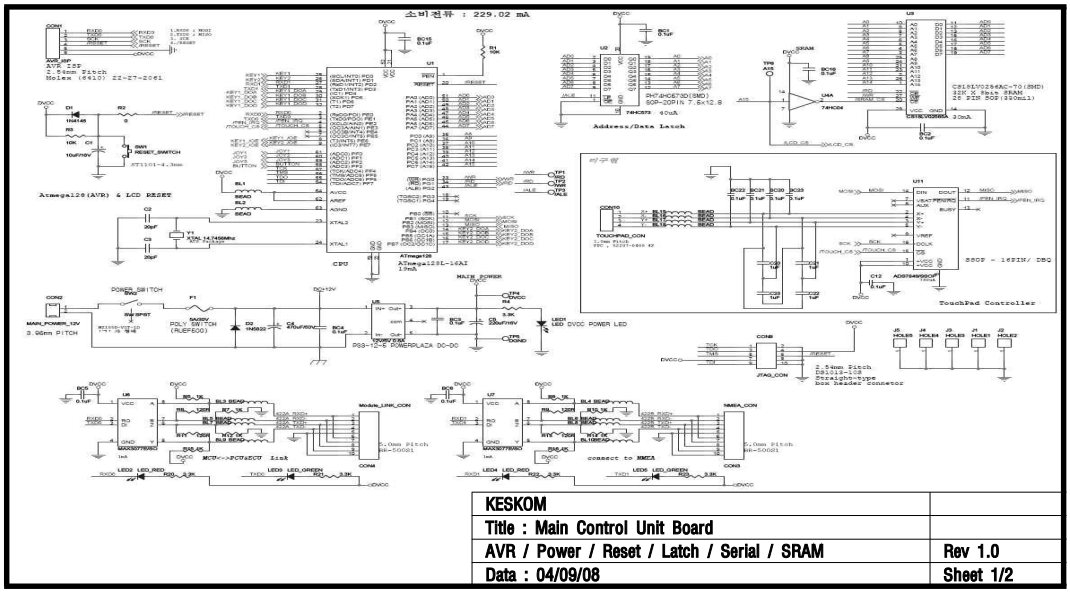


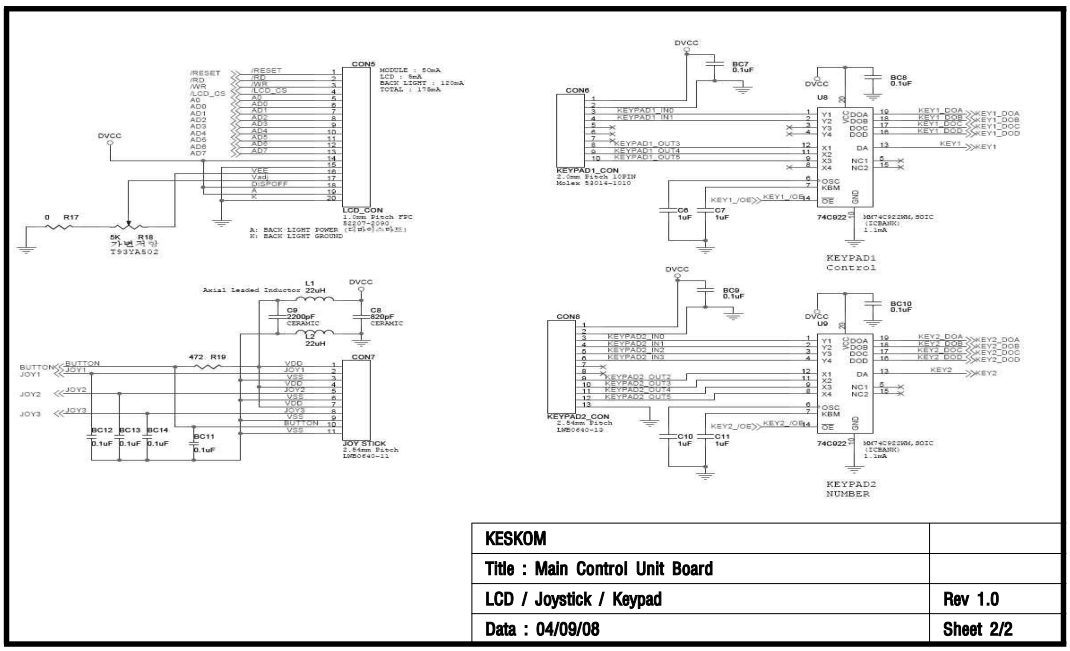
그림 3.15 조이스틱과 AVR 연결

Fig. 3.15 Joystick & AVR Connection

3.2.4 MCU 회로 및 레이아웃



(a) MCU AVR / Powr / Latch / Serial / SRAM Interface Circuit



(b) MCU LCD / Joystick / Keypad Interface Circuit

그림 3.16 MCU 회로도

Fig. 3.16 MCU Schematic

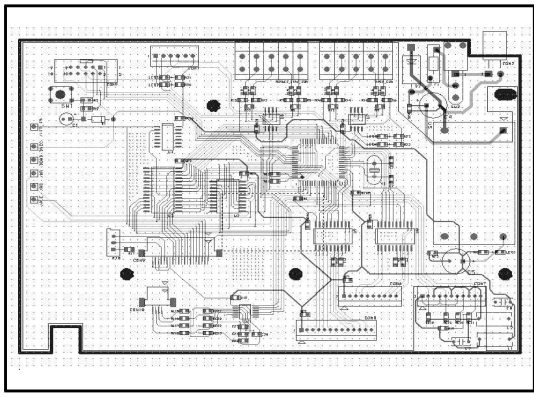


그림 3.17 MCU PCB LAYOUT
Fig. 3.17 MCU PCB LAYOUT

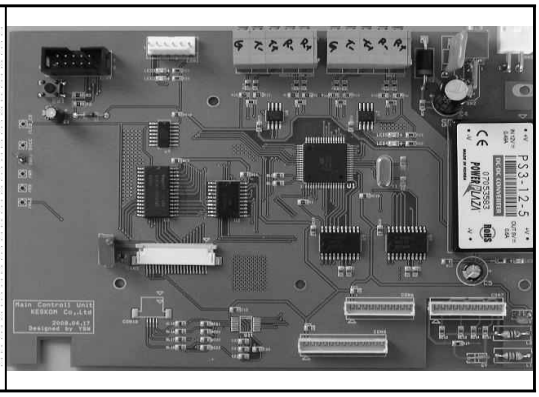


그림 3.18 제작된 MCU 보드
Fig. 3.18 Produced MCU Board

그림 3.16은 앞에서 설명한 각각의 인터페이스를 조합하여 회로를 설계한 것이다. 그림 3.16을 바탕으로 하여 그림 3.17과 같은 MCU 레이아웃을 설계하였고, 그림 3.18의 보드를 제작하였다.

3.3 Equip Control Unit 설계 및 구현

ECU의 설계는 기본적으로 시리얼 통신 인터페이스에 대한 설계이다. MCU와의 통신인 RS422 인터페이스는 PCU에서 설계한 것과 동일하고, 그 외 ECU에 의해서 제어되는 카메라, 줌 렌즈, 레이저 서치라이트를 위한 각각의 사양에 맞는 통신 인터페이스를 설계 하였다. 그림 3.19는 ECU은 블록도이다.

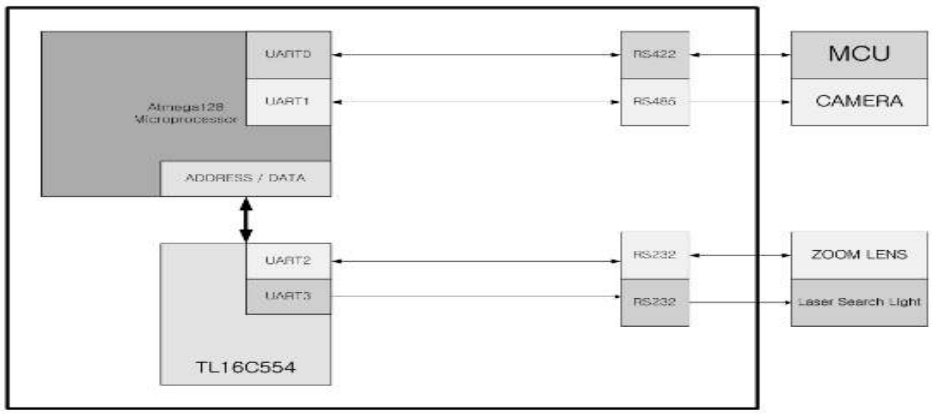


그림 3.19 ECU 블록도
Fig. 3.19 ECU Block Diagram

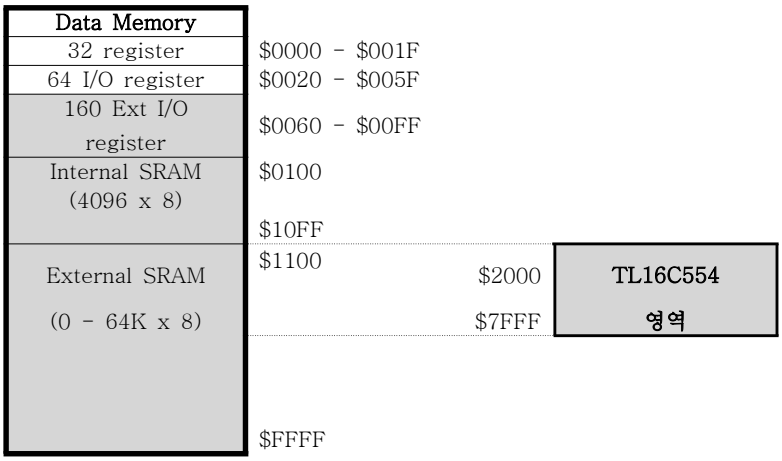


그림 3.20 ECU 메모리 맵

Fig. 3.20 ECU Memory Map

통신 채널을 확장 해주는 TL16C554가 AVR의 Address와 Data 버스를 사용하기 때문에 그림 3.20과 같이 메모리를 할당하였다.

3.3.1 UART 채널 확장 인터페이스

AVR이 제공하는 UART 채널은 2개이다. 하지만 필요한 UART 채널은 4개이기 때문에 UART 채널을 확장하는 IC(TL16C554)를 사용하여 2개의 UART 채널을 더 추가하였다. TL16C554는 4개까지 채널 확장이 가능하다.

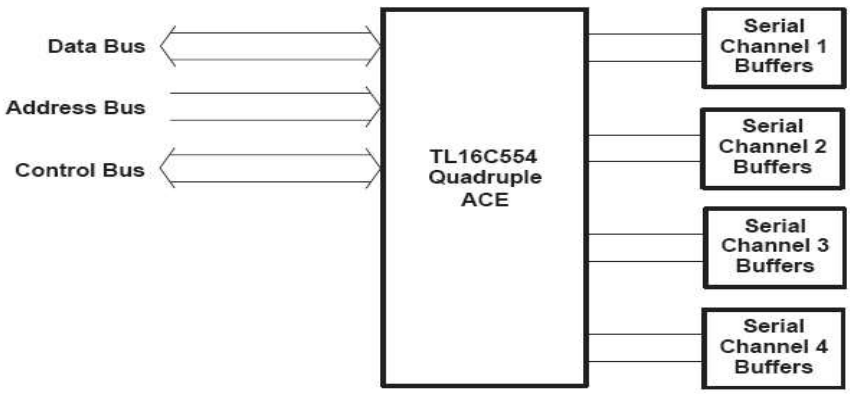


그림 3.21 TL16C554 블록도

Fig. 3.21 TL16C554 Block Diagram

그리고 TL16C554의 각 채널마다 칩 선택 신호가 달라야 하기 때문에, AVR의 어드레스 버스(A4, A5)를 사용하여 선택을 하게끔 하였다.

표 3.4 TL16C554 각 채널의 어드레스
Table 3.4 TL16C554 Address of each Channel

채널	어드레스
A 채널	0x20E0 ~ 0x20E7
B 채널	0x20D0 ~ 0x20D7

표 3.4의 어드레스 주소들은 TL16C554가 할당된 메모리 영역 내에서 A4와 A5의 값에 따라서 2개의 채널을 선택적으로 사용하게 하였다.

3.3.2 카메라 인터페이스

먼저 카메라는 RS485 방식으로 명령을 받아들이게 되어 있어서 AVR의 UART 신호에 RS485 트랜시버를 이용하여 신호를 변환할 수 있게 설계하였다. 그리고 반이중 전송 방식으로 하여 AVR에서 수신과 송신에 대한 모드를 설정 할 수 있게 하였다.

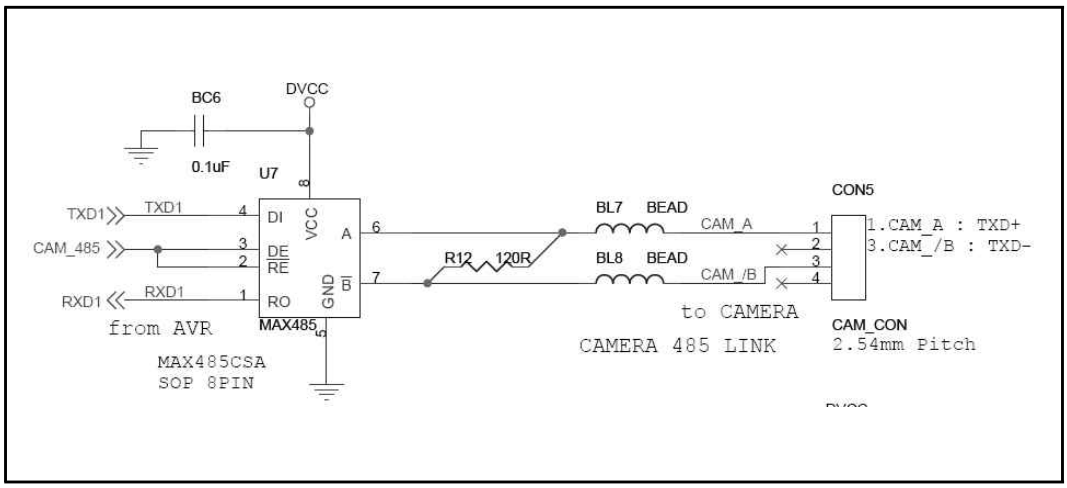


그림 3.22 ECU 카메라 RS485 인터페이스
Fig. 3.22 ECU Camera RS485 Interface

3.3.3 레이저 서치라이트와 줌 렌즈 인터페이스

레이저 서치라이트는 RS232 방식으로 명령들을 전송한다. 그래서 TL16C554의 채널 하나를 할당하여 RS232 트랜시버로 인터페이스가 된 회로를 설계하였다.

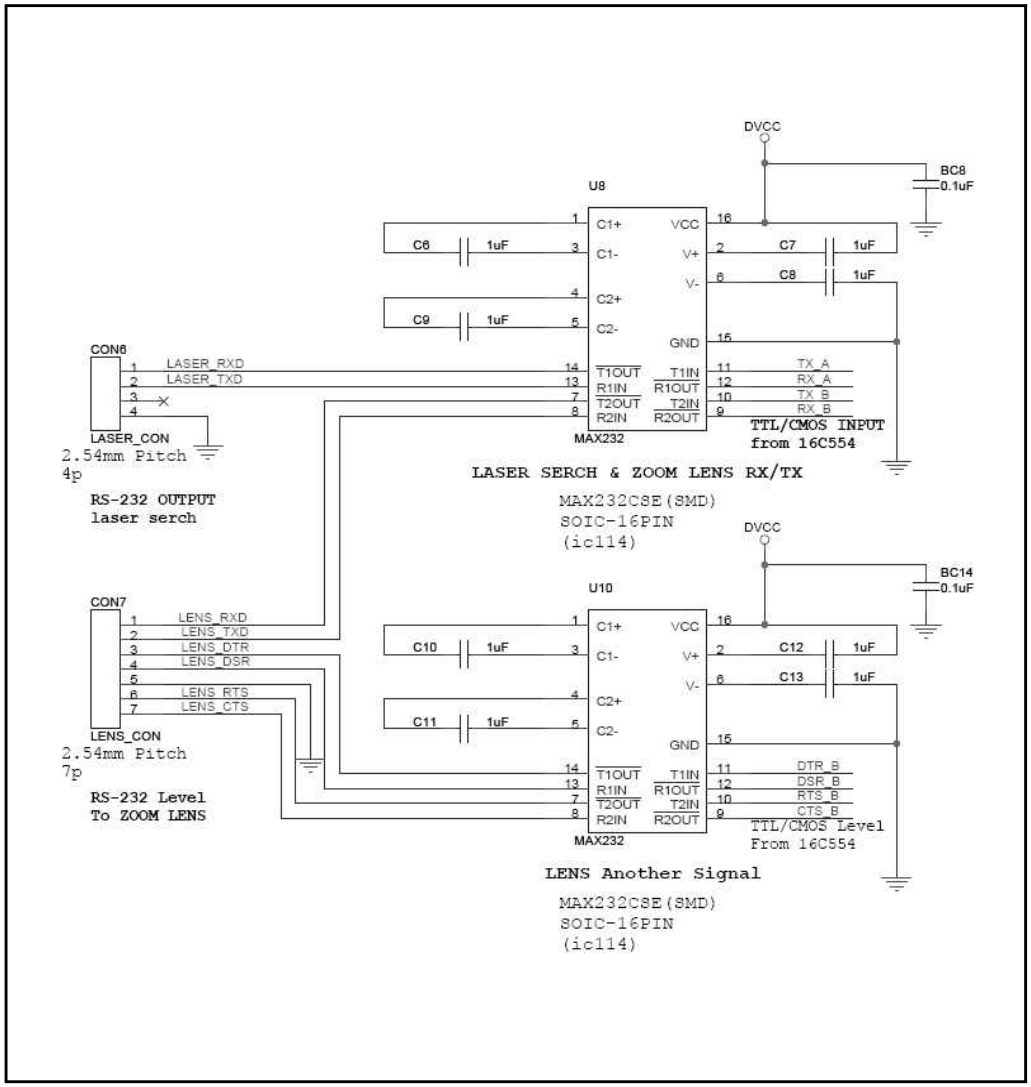


그림 3.23 레이저 서치라이트와 줌 렌즈 통신 인터페이스

Fig. 3.23 Laser Search Light & Zoom Lens Communication Interface

줌 렌즈는 RS232 통신이긴 하지만 Full Modem 방식이다(그림 3.24). 일반적으로 사용되는 RXD/TXD 외에도 DTD/DSR/RTS/CTS 신호도 사용한다. AVR에서는 Full Modem 을 지원하지 않기 때문에 TL16C554의 채널 하나를 할당하였다.

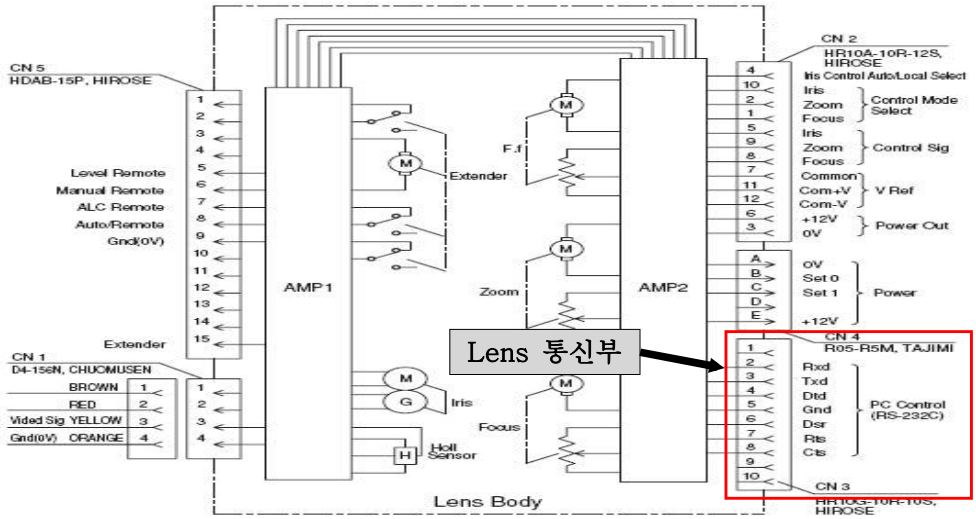


그림 3.24 줌 렌즈 결선
Fig. 3.24 Zoom Lens Wiring

TL16C554에서 나오는 TTL/CMOS 로직 레벨을 RS232 로직 레벨로 변환해주기 위해서, 사용되는 모든 신호가 RS232 트랜시버를 그림 3.25와 같이 통과하게 설계하였다.

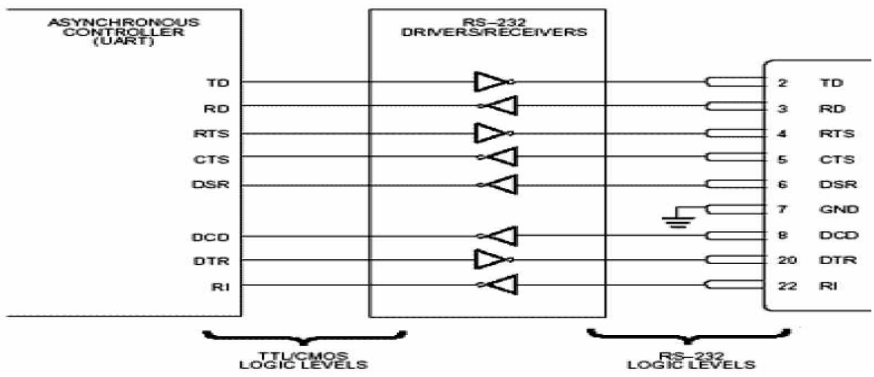
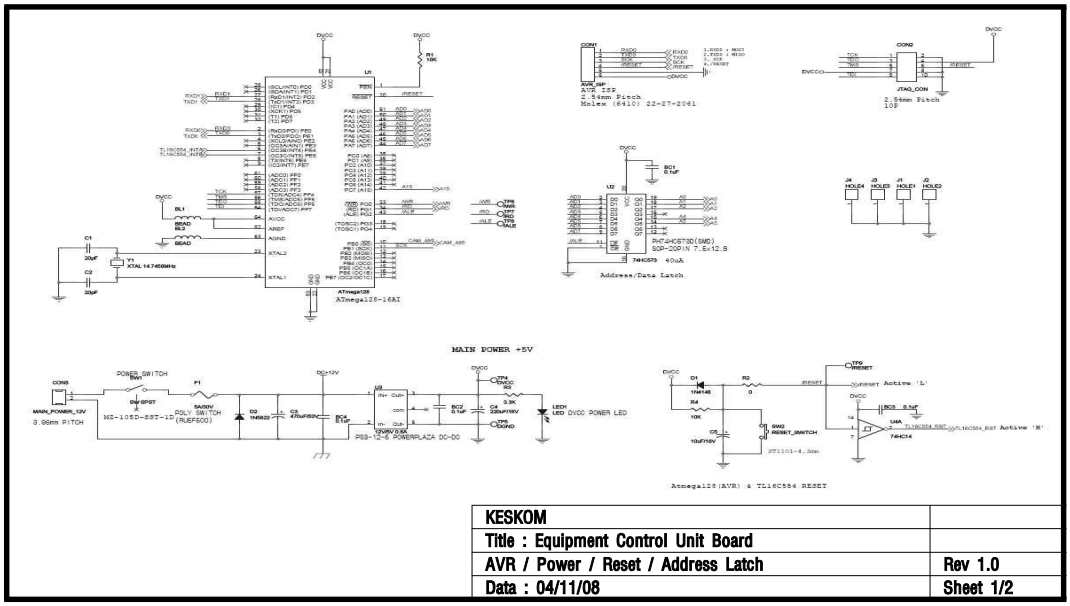
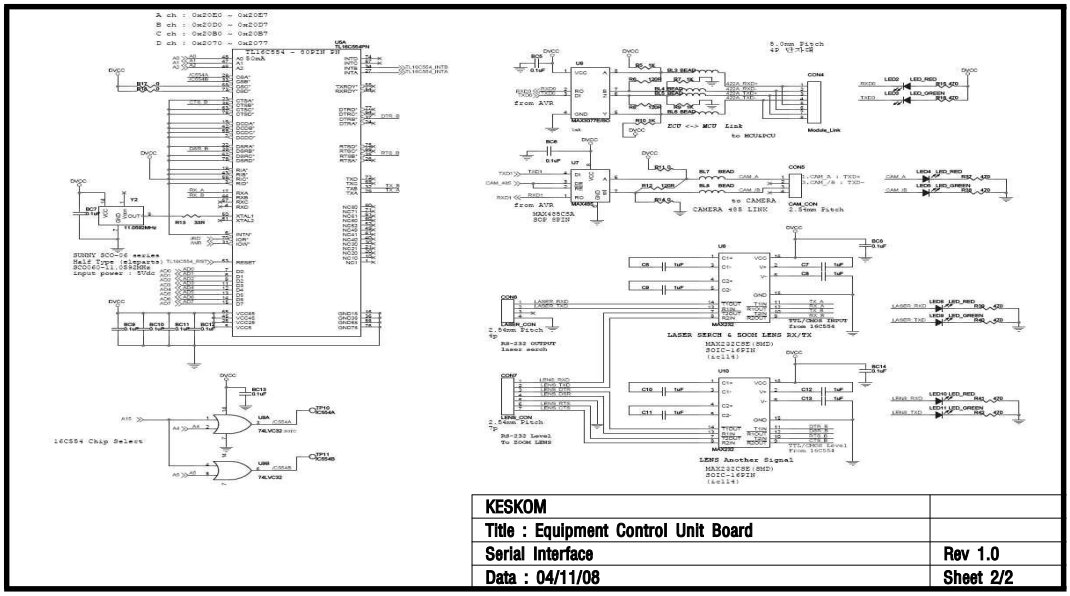


그림 3.25 일반적인 RS-232 모뎀 응용
Fig. 3.25 Typical RS-232 modem application

3.3.4 ECU 회로 및 레이아웃



(a) ECU AVR / Power / Reset / Address Latch circuit



(b) ECU Serial Interface circuit

그림 3.26 ECU 회로도

Fig. 3.26 ECU Schematic

MCU와의 통신, 카메라, 줌 렌즈, 레이저 서치라이트를 제어할 통신 인터페이스의 사양을 조사하여 그림 3.26과 같은 회로를 설계하였고, 그림 3.27과 같이 레이아웃을 하면서 커넥터와 칩들의 배치를 하였다. 그림 3.28은 레이아웃을 통하여 제작된 ECU 보드이다.

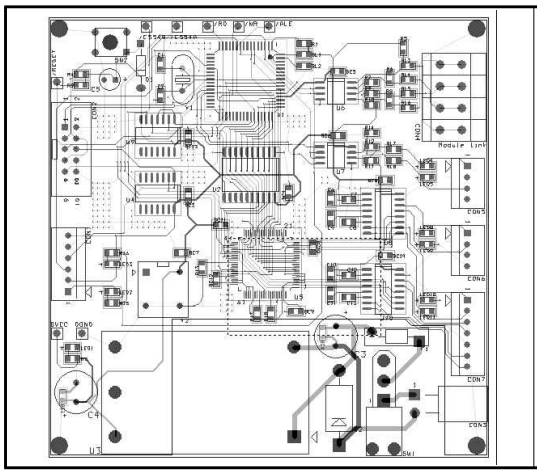


그림 3.27 ECU PCB LAYOUT
Fig. 3.27 ECU PCB LAYOUT

그림 3.28 제작된 ECU 보드
Fig. 3.28 Produced ECU Board

제 4 장 실험 및 결과

4.1 Unit 제어 실험

4.1.1 PCU 모션 콘트롤러 제어 실험

그림 4.1과 4.2는 모션 콘트롤러에서 Motor Driver로 보내는 신호 중 일부를 캡쳐한 것이다. XCW+는 속도를 나타내고, XCCW+는 방향을 나타낸다. 기본적으로 속도는 XCW+ 의 주파수 변경으로 정의 되고, 방향은 모션 콘트롤러의 모드에 따라 다르지만 실험에서는 그림 4.3과 같이 XCCW+의 High/Low로 방향을 정의한다. 그림 4.4는 그림 4.3과 같은 신호를 출력할 수 있게 해주는 프로그램 코드이다.

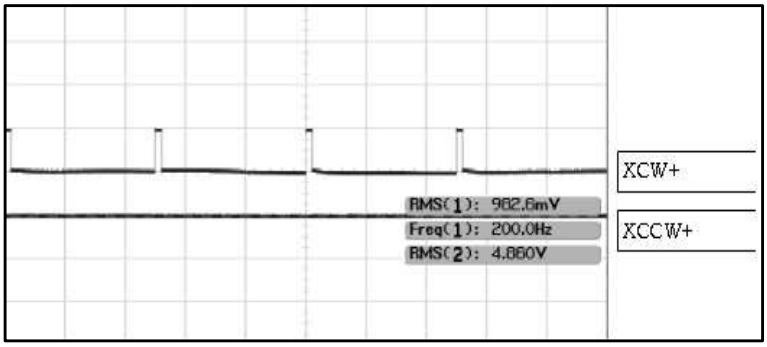


그림 4.1 속도 : 200pps, 방향 : +
Fig 4.1 Velocity : 200pps, Direction : +

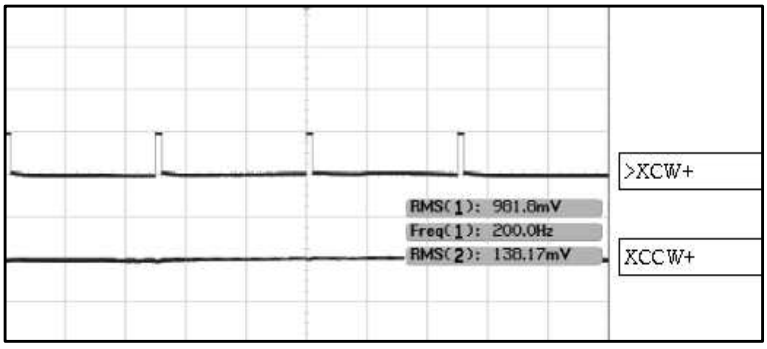


그림 4.2 속도 : 200pps, 방향 : -
Fig. 4.2 Velocity : 200pps, Direction : -

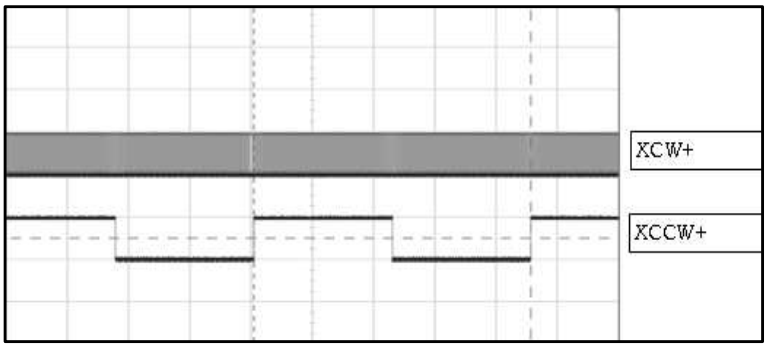


그림 4.3 방향(XCCW+) 변환
Fig. 4.3 Direction(XCCW+) Change

```
p645_wreg(AXS_AX,WPRMD, 0x00002000); // MOD continuous, DIR : +
p645_wreg(AXS_AX,WRENV5,0x00000088);//bit 7~0 10(7:6) 001(5:3) 000(2:0)
                                // 10(7:6): 조건 성립시 감속 정지
                                // 001(5:3): 비교 방법 RCMP5 Data = 비교 counter(방향 무관계)
                                // 000(2:0): COUNTER1(지령 위치)
p645_vset(AXS_AX,100L,200L,100,0,0,0,'L',0);        //100pps to 200pps,1000ms
p645_wreg(AXS_AX,WRCMP5,p645_rreg(AXS_AX,RRCUN1)+0x000000C8);
                                        //current value + 200pulse(+180 degree)
p645_wcom(AXS_AX,STAUD);
p645_wait(AXS_AX);
p645_wcom(AXS_AX,CUN1R);                //counter1 value reset
p645_wreg(AXS_AX,WPRMD, 0x00002008);            // MOD continuous, DIR : -
p645_wreg(AXS_AX,WRENV5,0x00000088);            //bit 7~0 10(7:6) 001(5:3) 000(2:0)
                                // 10(7:6): 조건 성립시 감속 정지
                                // 001(5:3): 비교 방법 RCMP5 Data = 비교 counter(방향 무관계)
                                // 000(2:0): COUNTER1(지령 위치)
p645_vset(AXS_AX,100L,200L,100,0,0,0,'L',0);        //100pps to 200pps,1000ms
p645_wreg(AXS_AX,WRCMP5,p645_rreg(AXS_AX,RRCUN1)-0x000000C8);
                                //current value - 200pulse(-180 degree)
p645_wcom(AXS_AX,STAUD);
p645_wait(AXS_AX);
p645_wcom(AXS_AX,CUN1R);                //counter1 value reset
```

그림 4.4 모터 구동 기본 프로그램 코드
Fig. 4.4 Base Program Code of Motor Operation

4.1.2 MCU의 LCD 실험

LCD는 먼저 초기화를 시켜줘야 한다. 초기화 과정은 다음과 같다(그림 4.5).

```
void init_lcd(void)
{
        writeControl(SYSTEM_SET); // Window size, and select the LCD interface format
        writeData(SYS_P1);      // 0x10 | (IV << 5) | (WS << 3) | (M2 << 2) | (M1 << 1) | M0
        writeData(SYS_P2);      // 0x00 | (WF << 7) | FX
        writeData(FY);          // 0x07
        writeData(CR);          // 한 라인에 표시하는 어드레스의 수   320/8-1 = 39 = 0x27
        writeData(TCR);         // 한 라인의 길이 CR + 4 = 43
        writeData(LF);          // 한 프레임의 높이 240-1 line
        writeData(APL);         // 한 라인의 총 어드레스, 40
        writeData(APH);         //
        writeControl(SCROLL); //
        writeData(SAD1L);       // 첫 번째 레이어의 시작 어드레스
        writeData(SAD1H);
        writeData(SL1);         // 첫 번째 스크롤 화면에 표시하는 라인의 수
        writeData(SAD2L);       // 두 번째 레이어의 시작 어드레스
        writeData(SAD2H);
        writeData(SL2);         // 두 번째 스크롤 화면에 표시하는 라인의 수
        writeData(0x00);        // p7 Don't care
        writeData(0x00);        // p8 Don't care
        writeData(0x00);        // p9 Don't care
        writeData(0x00);        // p10 Don't care
        writeControl(HDOT_SCR); // Set horizontal scroll position
        writeData(SCRD);
        writeControl(OVLAY);  // 텍스트/그래픽 모드에서 각 레이어가 겹쳐질 때 표시 방법
        writeData(OVLAY_P1); // 0 0 0 OV DM2 DM1 MX1 MX0
        writeControl(DISP_OFF); // Turn display off
        clearScreen_text();       // 텍스트 화면 지움
        clearScreen_graphics();   // 그래픽 화면 지움
        writeControl(CSRW);       // 커서 위치에 대한 디스플레이 주소 정보
        writeData(0x00);
        writeData(0x00);
        writeControl(CSRFORM); // 커서의 크기와 형태 설정.
        writeData(CRX);          //0000 0100
        writeData(CSRF_P2);    //1000 0110
        writeControl(CGRAM_ADR); // CGRAM 의 시작 주소
        writeData(SAGL);
        writeData(SAGH);
        writeControl(DISP_ON); // Turn display on
        writeData(FLASH);       // 커서의 동작 설정과 플레쉬 빈도를 설정.
                                // 0x16 Layer 1 (text) on, Layer 2 (graphics) on   0001 0110
        writeControl(CSRDIR_RIGHT); //  커서 이동 방향 설정.
}
```

그림 4.5 그래픽 LCD 초기화 함수

Fig. 4.5 Graphic LCD Initialize Function

LCD의 초기화는 LCD 모듈에 인터페이스 되어 있는 LCD 콘트롤러의 레지스터의 값들을 AVR의 어드레스와 데이터 버스를 이용하여 변경한다. 그림 4.5는 MCU의 초기화를 위해 정해준 순서에 따라 그 값들을 정의 한 것이다.

LCD의 초기화가 끝난 후 원하는 정보를 그림 4.6과 같이 LCD에 디스플레이 하였다. ARPA RADAR로부터 원하는 Target의 정보(거리, 방위각)를 확인 할 수 있고, 현재 페데스털의 이동각도(AZ, EL)와 모드(MODE) 상태(RADAR, Joystick), 안정화(Stabilization)기능의 ON/OFF 상태, Laser Searchlight의 ON/OFF 상태들을 모니터링 할 수 있다.

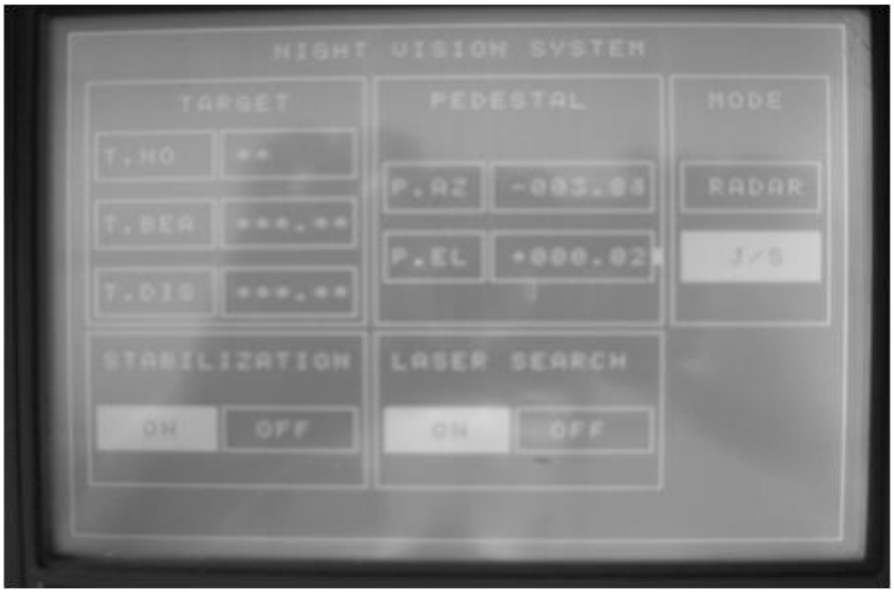


그림 4.6 LCD 화면

Fig. 4.6 LCD Display

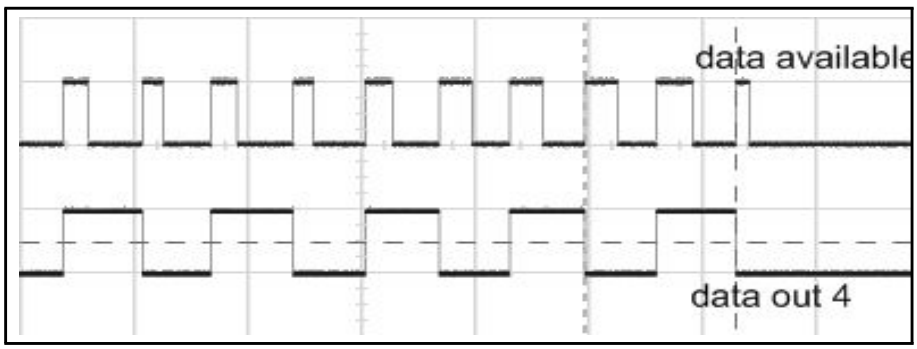
4.1.3 MCU의 키보드 실험

실험에 사용한 키보드는 Target의 번호, 모드 선택, Camera, Zoom Lens, Laser Searchlight등의 구성요소를 가진다. 키보드는 AVR의 외부 인터럽트 핀에 Keypad Controller의 data available 핀을 연결하여 Rising Edge에 인터럽트가 발생하여 Keypad Controller의 Data가 AVR로 입력되어 지게 되어 있다. 그림 4.7과 같이 외부 인터럽트 서비스 루틴 내에 Keypad Controller의 Chip Select를 해주고 Data(4bit)를 전역변수에 저장함으로써 루틴이 끝난다.

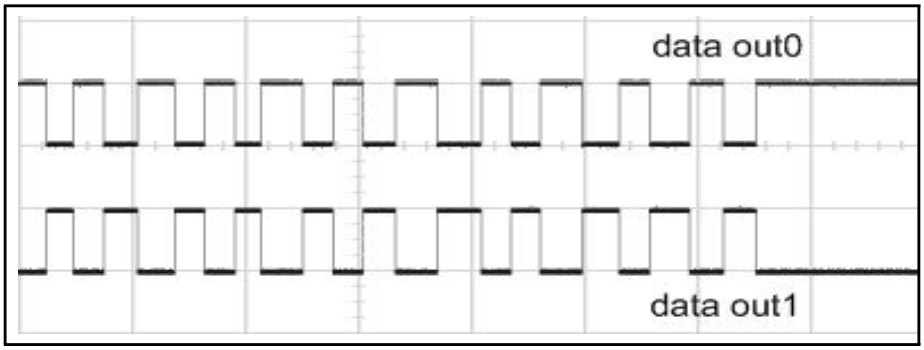
```
void int0_isr(void)
{
            fECU_flag1= 1;
            fECU_KEY1 = 1;

            // 74C922 Chip1 Select
            PORTE = 0x3F;                   //1011 1111
            delay(100);
            PORTE = 0xFF;
            KeyData1 = PIND&0xF0;           // Save key value to KeyData1
}
void int1_isr(void)
{

            fECU_KEY2 =1;
            fECU_flag2= 1;
            key2_num = 1;
            fst_flag = 1;
            fst_key = 1;

            // 74C922 Chip2 Select
            PORTE = 0x3F;                   //0111 1111
            delay(100);
            PORTE = 0xFF;
            KeyData2 = PINB&0xF0;           // Save key value to KeyData2
}
```

그림 4.7 Key 인터럽트 Code

Fig. 4.7 Key Interrupt Code



(a) data available 과 data out4



(b) data out0 와 data out1

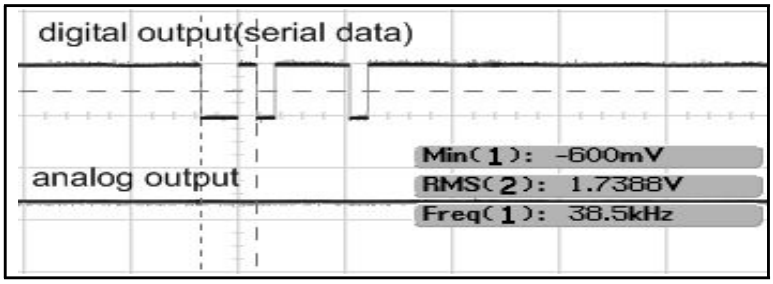
그림 4.8 16-키 엔코더 출력

Fig. 4.8 16-key Encoder Output

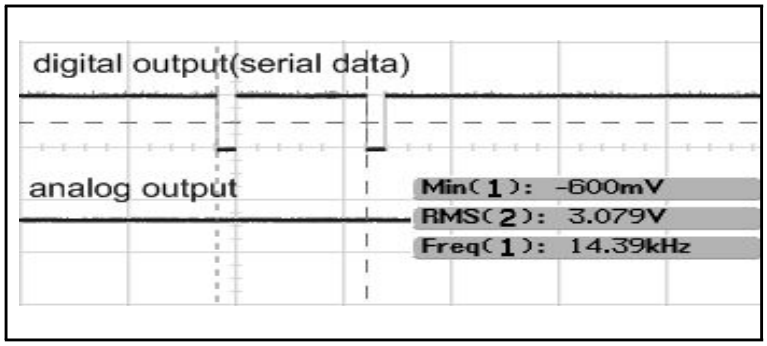
키보드를 눌렀을 때 16-키 엔코더의 출력이 어떻게 나오고 있는지 계측기를 통하여 그림 4.8과 같은 결과를 확인했다. data available 의 상승 에지에서 인터럽트가 발생하게 하였다. 인터럽트가 발생할 때 마다 data out4의 값이 변하는 것을 그림 4.8(a)를 통하여 알 수 있다. 그림 4.8(b)는 다른 출력도 정상적으로 변화하는지를 확인한 것이다.

4.1.4 MCU Joystick 실험

실험에 사용한 조이스틱은 포텐셔미터를 기반으로 up, down, left, right의 기능을 가지고 있다. Analog인 포텐셔미터로부터 나오는 전압 값을 AVR의 A/D Convertor를 통하여 Digital로 변환하여 그 Data를 PCU로 전송을 하게 된다. 그림 4.10과 같이 A/D convertor 값은 인터럽트를 통하여 그 값들을 전역변수에 저장하게 된다.



(a) 포텐셔미터 전압 1.7388V



(b) 포텐셔미터 전압 3.079V

그림 4.9 조이스틱 포텐셔미터 아날로그 출력과 디지털 출력

Fig. 4.9 Joystick Potentiometer analog output and digital output

```
void adc_isr(void)
{
        ADCSRA = 0x00;
        switch(AdcCurrentChannel())
        {
                case ADC0       : AdcData0 = AdcGetData(); break;
                case ADC1       : AdcData1 = AdcGetData(); break;
                case ADC2       : AdcData2 = AdcGetData(); break;
                case ADC3       : AdcData3 = AdcGetData(); break;
                case ADC4       : AdcData4 = AdcGetData(); break;
                case ADC5       : AdcData5 = AdcGetData(); break;
                case ADC6       : AdcData6 = AdcGetData(); break;
                case ADC7       : AdcData7 = AdcGetData(); break;
                default:
                        break;
        }
        AdcNextChannel();
        ADCSRA = 0xEF;
}
```

그림 4.10 ADC 인터럽트 서비스 루틴

Fig. 4.10 ADC Interrupt Service Routine

조이스틱의 Analog 출력에 따라 A/D 컨버터를 지난 후의 Digital 출력이 어떻게 달라지는 그림 4.9로 확인했다. 그림 4.9(a)는 아날로그 출력이 1.7388V이고 그림 4.9(b)는 3.079V인데, 그림 4.10의 인터럽트 서비스 루틴에서 저장된 A/D 값을 시리얼 통신으로 출력한 값이 디지털 출력이다.

4.2 MCU 통신 실험

4.2.1 MCU와 PCU 간 통신 실험

MCU는 PCU로부터 페데스털의 위치 정보를 받고, 또 PCU로 조이스틱과 레이더 정보를 보낸다. 표4.1은 MCU가 PCU로부터 받는 페데스털의 정보의 프로토콜 구조이고, 표 4.2는 MCU가 PCU로 보내는 정보의 프로토콜 구조이다.

표4.1 PCU에서 MCU로 보내는 데이터 프로토콜 구조

Table 4.1 Transmit PCU to MCU data protocol instruction

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Sync	Address	Azimuth High data	Azimuth2 Low data	Elevation1 High data	Elevation2 Low data	Checksum

표 4.2 MCU에서 PCU로 보내는 데이터 프로토콜 구조

Table 4.1 Transmit MCU to PCU data protocol instruction

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
Sync	Address	Command 1	Command 2	Data 1	Data 2	Checksum

•Byte 1 (Sync) : 0xFF로 고정

•Byte 2 (Address) : 각 Unit의 지정된 주소

•Byte 3 & 4 (Command 1 & 2) : 표 4.3 참조

•Byte 5 & 6 (Data 1 & 2) : 표 4.4 참조

•Byte 7 (Checksum) : 체크섬, Sync는 제외하여 Byte2~Byte6을 합산한 값

표 4.3 Command 비트 정의

Table 4.3 Command bit definition

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Command 1	Special Command	Radar Bearing	Radar Distance	0	0	0	0	0
Command 2	0	0	0	Elevation Down	Elevation Up	Azimuth Left	Azimuth Right	0

표 4.4 Data 정의

Table 4.4 Data definition

	Command	Data 1	Data 2	비고
Special Command	0x8000	0x00	0x02	Stabilization on
		0x00	0x04	stabilization off
Radar Bearing	0x4000	Signed Int type (High byte)	Signed Int type (Low byte)	
Radar Distance	0x2000	Signed Int type (High byte)	Signed Int type (Low byte)	
Elevation Down	0x0010	Elevation Speed (0x00~0x3F)		0x00 : Stop ~0x3F : High
Elevation Up	0x0008	Elevation Speed (0x00~0x3F)		0x00 : Stop ~0x3F : High
Azimuth Left	0x0004		Azimuth Speed (0x00~0x3F)	0x00 : Stop ~0x3F : High
Azimuth Right	0x0002		Azimuth Speed (0x00~0x3F)	0x00 : Stop ~0x3F : High

표적의 레이더 정보(그림 4.11)를 MCU는 받아서 표 4.2에 정의된 구조에 따라 PCU로 레이더 정보를 보낸다. 단 표적의 레이더 정보에는 방위각과 거리가 있는데, 표 4.4에서 보는 것과 같이 한 프레임에 두 개의 정보를 다 보내지 못하므로 그림 4.12와 같이 두 개의 프레임을 한 번(주기 20msec)에 보낸다.

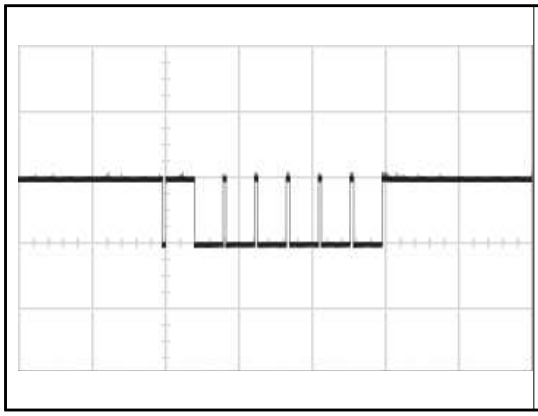


그림 4.11 레이더 정보 수신

Fig. 4.11 Receive Radar Data

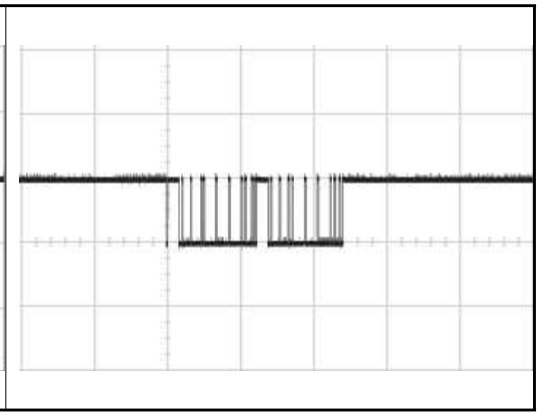
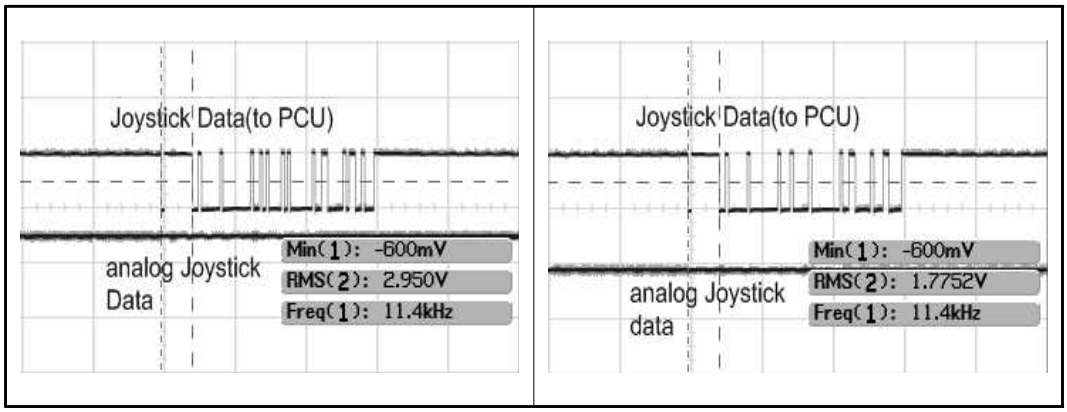


그림 4.12 레이더 정보 송신

Fig. 4.12 Transmit Radar Data

그림 4.13은 조이스틱의 아날로그 값에 따라 MCU가 PCU로 송신하는 프레임의 변화를 계측한 것이다. 그림 4.13의 (a)와 (b)에서 Analog Joystick Data가 변화함에 따라 MCU가 송신 하는 정보(Joystick Data)도 달라지는 것을 볼 수 있다. PCU는 이 Data를 수신하여 페데스털의 2축 DD-Motor를 구동 시킨다.



(a) (b)

그림 4.13 조이스틱 정보 송신

Fig. 4.13 Transmit Joystick Data

4.2.2 MCU와 ECU간 통신 실험

MCU와 ECU간 통신에서 전달되는 정보는 단순히 카메라, 레이저 서치라이트, 줌렌즈가 특정한 동작을 하게하는 명령뿐이다. 그래서 프로토콜 구조도 PCU와의 통신 구조보다도 간단하다. 표 4.5는 그 구조를 나타내고 있다.

표 4.5 MCU에서ECU로 가는 데이터 프로토콜 구조

Table 4.5 Transmit MCU to ECU data protocol instruction

Byte 1	Byte 2	Byte 3	Byte 5
Sync	Address	Command	Checksum

•Byte 1 (Sync) : 0xFF로 고정

•Byte 2 (Address) : 0x01로 고정

•Byte 3 (Command) : 표 4.6 참조

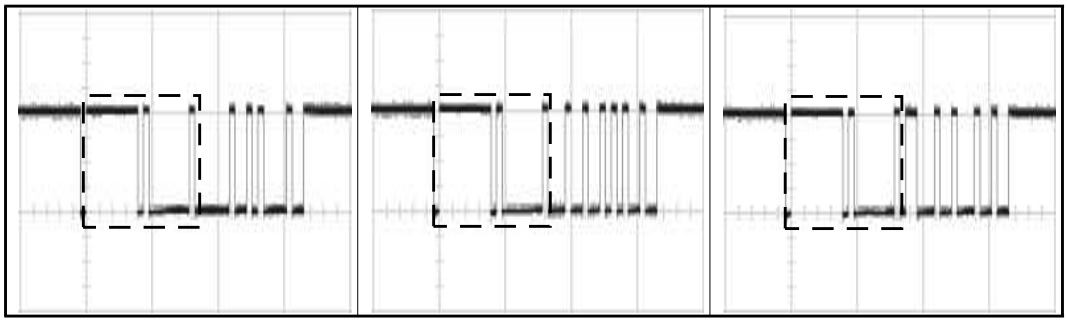
•Byte 4 (Checksum) : 체크섬, Sync를 제외하여 Byte2~3을 합산한 값

표 4.6는 각각의 장치들에 대한 MCU가 ECU로 보내는 명령의 값을 정의한 것이다.

표 4.6 명령과 값 정의
Table 4.6 Command and Value definition

Camera		Lens		Laser	
Command	Value	Command	Value	Command	Value
Menu Setup	0x10	Iris Close	0x20	LD Power On	0x30
Menu Up	0x11	Iris Open	0x21	LD Power Off	0x31
Menu Down	0x12	Zoom Wide	0x22	Power 1 step Up	0x32
Menu Left	0x13	Zoom Tele	0x23	Power 1 step Down	0x33
Menu Right	0x14	Focus Near	0x24	Beam Size Up	0x34
		Focus Far	0x25	Beam Size Down	0x35

MCU에서 ECU로 가는 정보는 대부분이 키보드의 명령들이다. 카메라, 줌 렌즈, 레이저 서치라이트가 모두 키보드로 제어되기 때문이다. 그림 4.14는 서로 다른 키를 눌렀을 때, MCU에서 어떠한 정보가 송신되는지 계측기로 측정한 것이다. 표 4.5에 나와 있는 것처럼 Sync와 Address는 고정이기 때문에 점선으로 표시된 사각형의 데이터는 다 동일하지만 그 이후부터의 데이터가 서로 다른 것을 확인하였다.



(a) Iris Close (b) Focus Near (c) Zoom Tele

그림 4.14 MCU에서 ECU로 가는 키보드 명령
Fig. 4.16 Transmit MCU to ECU keyboard command

그림 4.15는 키보드를 눌렀을 때, 그 명령을 ECU로 전송하는 프로그램 함수이다.

```
void ECU_COMMAND_SEND(void)
{
	U8 ecu_buffer[4];
	U8 ecu_cmd=0;
	U8 ecu_check=0;
	I16 i;

	if(fECU_flag1 == 1)
	{
		ecu_cmd = ECU_Key1();
		// ECU Command 1 < ZOOM LENS COMMAND to ECU >
		// SYNC | ADDRESS | COMMAND1 | COMMAND2 | CHECKSUM
		if(ecu_cmd >= 0x20 && ecu_cmd <= 0x25)
		{
			ecu_buffer[0] = SYNC;
			ecu_buffer[1] = ECU_ADDRESS;
			ecu_buffer[2] = ecu_cmd;

			for(i=1, ecu_check = 0 ; i <= 2 ; i++)
			{
				ecu_check += ecu_buffer[i];
			}
			ecu_buffer[3] = ecu_check;
			TX0_string(ecu_buffer, 4);
			fECU_flag1 =0;
		}
		else if(ecu_cmd == 0xEB)
		{
			fECU_flag1 =0;
		}
	}

	else if(fECU_flag2 == 1)
	{
		ecu_cmd = ECU_Key2();
		// ECU Command 2 < CAMERA & LASER SEARCH COMMAND to ECU >
		// SYNC | ADDRESS | COMMAND | CHECKSUM

		if(ecu_cmd >= 0x10 && ecu_cmd <= 0x35)
		{
			ecu_buffer[0] = SYNC;
			ecu_buffer[1] = ECU_ADDRESS;
			ecu_buffer[2] = ecu_cmd;

			for(i=1, ecu_check = 0 ; i <= 2 ; i++)
			{
				ecu_check += ecu_buffer[i];
			}
			ecu_buffer[3] = ecu_check;
			TX0_string(ecu_buffer,4);
			fECU_flag2 = 0;
		}
		else if(ecu_cmd == 0xEA)		// ANOTHER KEY press
		{
			fECU_flag2 = 0;
		}
	}

}
```

그림 4.15 ECU 명령 송신 함수

Fig. 4.15 ECU Command Send Function

4.3 실선 실험(Sea Trial)

4.3.1 실험 환경

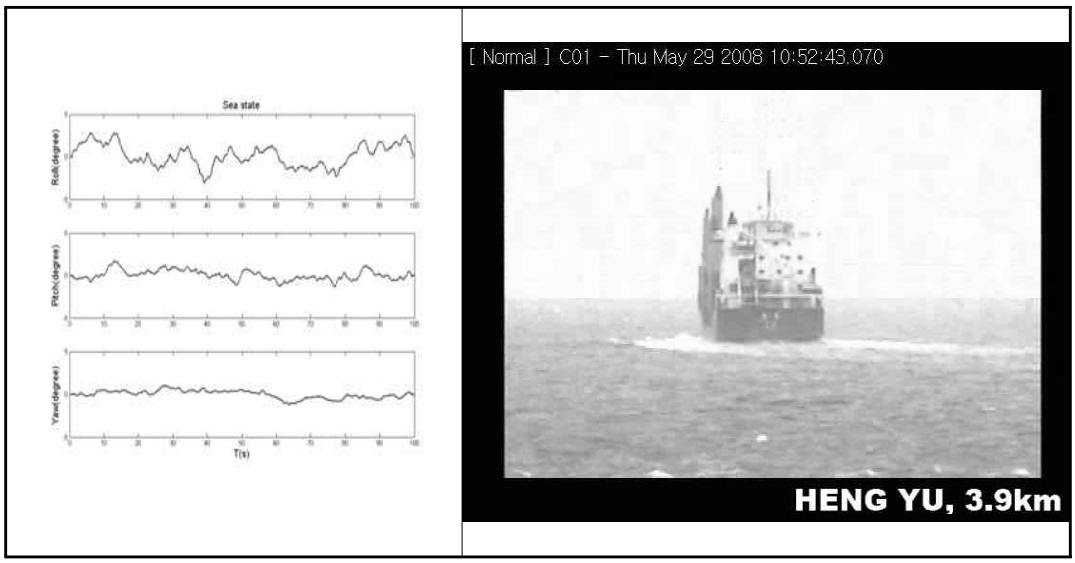
앞선 각 Unit의 개별적인 실험 후, 실제 선박에 NVS를 장착하여 실선 테스트(Sea Trial)을 수행하였다. 그림 4.16은 실험 환경이다. 브리지(Bridge) 랙의 키보드 내부에 MCU를 설치하고, 갑판 페데스털의 하단 내부에 PCU, 그리고 페데스털의 팔 부분에 설치되어 있는 Case 내부에 ECU를 설치하였다.

(a) 브리지 랙 (b) 갑판 페데스털

그림 4.16 실험 환경

Fig. 4.16 Test environment

4.3.2 페데스털 제어 실험



(a) 해상 상태 (b) 카메라 촬영 영상

그림 4.17 실험 결과 1

Fig. 4.17 Test result 1

그림 4.17(a)는 페데스털의 카메라에서 그림 4.17(b)를 촬영할 당시 해상의 상태에서의 선박 운동을 광자이로 센서를 이용해 측정한 것이다. 이때 롤링은 5.2도, 피칭은 3.6도, 요잉은 1.5도 정도였고, 표적물과 페데스털이 설치된 선박과의 거리는 항해 장비를 통해 확인한 결과 3.9km 정도였다.

먼저 ARPA 레이더를 이용하여 표적을 지정하면 페데스털은 표적을 지향하며 추적을 시작한다. 이 때 ECU는 MCU로부터 표적의 거리를 받아서 줌 렌즈의 줌과 포커서의 파라미터를 조정하여 표적이 브리지의 랙에 있는 모니터에 표시되는 정도를 조정한다. 하지만 ARPA 레이더의 정보만으로 표적을 추적하면 작은 오차가 발생한다. 그래서 유저가 조이스틱과 키보드를 이용하여 원하는 영상을 얻을 수 있게 페데스털과 줌 렌즈를 수동으로 조정을 한다.

그림 4.18은 그림 4.17(b)의 표적을 사용자가 원하는 곳을 지정하여 촬영하기 위해 수동으로 페데스털을 조정하는 것이다.

(a) 수동 조정 중 (b) 수동 조정 후

그림 4.18 실험 결과 2

Fig. 4.18 Test result 2

그림 4.17(b)은 ARPA 레이더를 통하여 표적의 전체적인 모습을 촬영하고, 그 뒤에 표적의 선명을 확인하기 위하여 MCU의 조이스틱과 키보드를 이용하여 페데스털 과 줌 렌즈를 조정하고 있는 것이 그림 4.20의 두 그림이다.

그림 4.18(a)에서는 표적의 선명이 모니터의 중앙에 있지 않다. 또 선명의 초점이 맞지 않아 흐리게 보인다. 하지만 조이스틱으로 페데스털을 조정하여 선명을 모니터의 중앙에 맞춘 뒤, 줌 렌즈의 초점을 조정하여 그림 4.18(b)의 영상을 획득하였다.

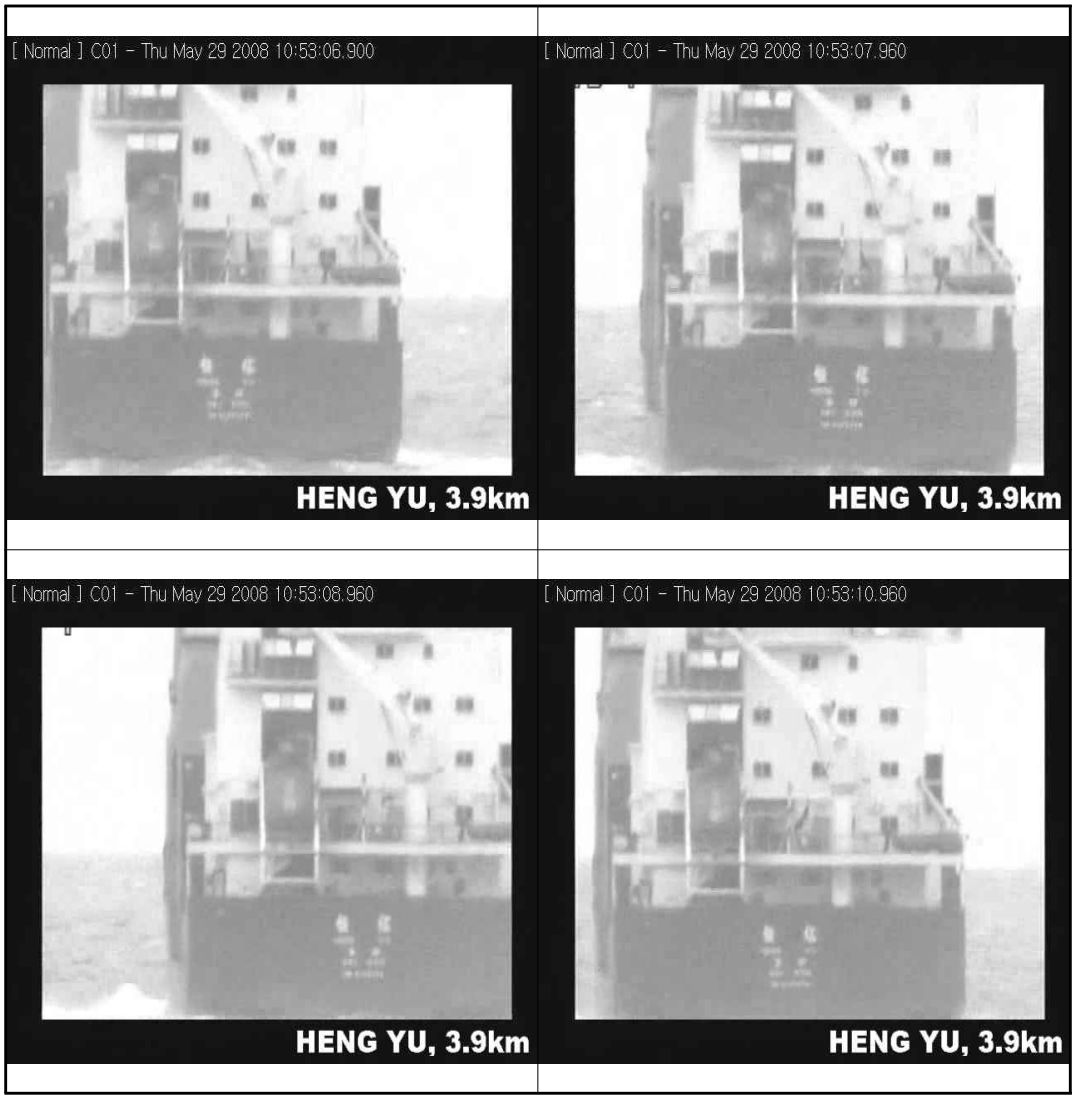


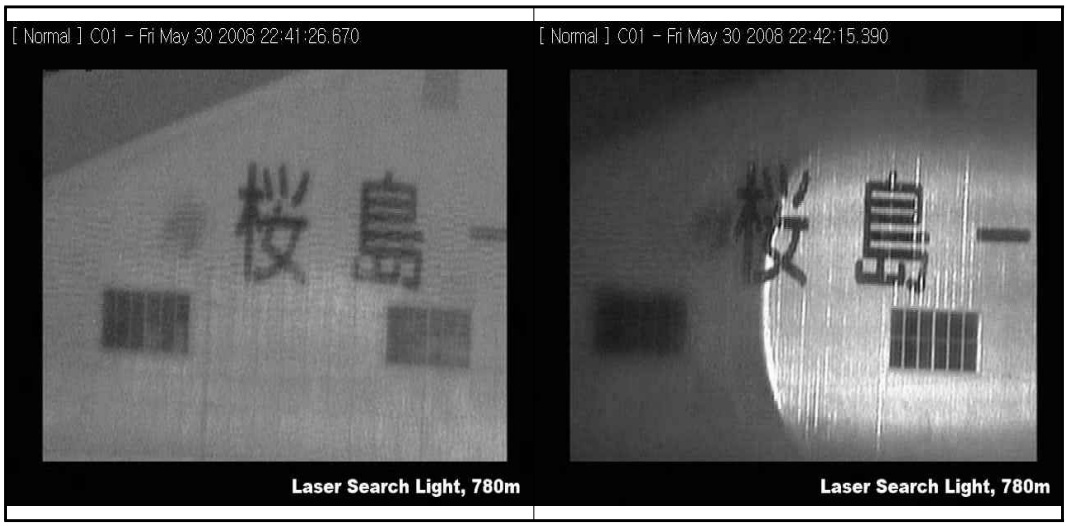
그림 4.19 선박 동요 안정화 실험 영상

Fig. 4.19 Ship stabilization test image

그림 4.19는 표적을 4초간 추적한 영상이다. 파도에 의해 선박이 롤링과 피칭이 있음에도 페데스털은 그림 4.21과 같이 큰 흔들림 없이 표적의 영상을 획득하고 있는 것을 확인했다. 표적까지의 거리가 3.9km라는 것을 감안하면 그림 4.19와 같은 정도의 오차는 아주 미세한 방위각의 이동과 ARPA 레이더의 오차이다.

4.3.3 레이저 서치라이트 실험

이번 실험은 야간에 빛이 부족한 곳에서의 카메라의 성능과 레이저 서치라이트를 광원으로 했을 때의 그 성능에 대한 실험이다.



(a) 레이저 서치라이트 OFF　　(b) 레이저 서치라이트 ON

그림 4.20 레이저 서치라이트 실험

Fig. 4.20 Laser searchlight test

그림 4.20(a)에서와 같이 빛이 부족한 곳에서도 카메라가 어느 정도 성능을 내어 표적의 문자를 카메라를 통하여 사용자가 식별이 가능하다. 하지만 문자가 있는 곳에 그림 4.20(b)처럼 레이저 서치라이트를 비췄을 때 문자가 더 선명해지는 것을 실험을 통하여 확인했다.

제 5 장 결 론

본 논문은 NVS의 제어를 위한 MCU, ECU, PCU 등의 하드웨어의 설계와 구현, 그리고 구현된 하드웨어들을 이용한 시스템 통합 및 NVS의 실선 테스트(Sea Trial)에 대한 내용이다.

먼저 선박의 6 자유도 운동을 축 단위로 분해 검출하여 선박의 운동을 실시간으로 보상할 수 있는 2축 구조의 고정도 안정화 페데스털을 제어하는 PCU를 설계 및 제작하였다. PCU는 페데스털의 동요안정화 및 표적추적 기능을 수행하게 하는 제어장치로써, ARM 코어 프로세서, PCL6045 모션 콘트롤 IC[1], 광 자이로 센서인터페이스로 구성하였다.

또한 NVS의 페데스털에 탑재되어 있는 카메라, 렌즈, 레이저 서치라이트를 원격 제어하는 ECU를 설계 및 제작하였다. ECU는 AVR 프로세서 인터페이스, 장비 제어를 위한 시리얼 통신 인터페이스로 구성하였다.

그리고 사용자가 키보드와 조이스틱을 조작함으로써 PCU와 ECU로 각종 정보와 명령을 시리얼 통신으로 전송하는 MCU를 설계 및 제작하였다. 또 MCU는 PCU와 ECU로의 정보와 명령 전송 외에도 그래픽 LCD를 통하여 사용자가 현재 NVS의 상태를 알 수 있게 하였다.

고성능 프로세서를 탑재한 PCU를 사용하게 됨으로써, 페데스털 제어의 주요 기능인 축 변환 알고리즘, 안정화 알고리즘, 표적추종 알고리즘 및 모터제어 알고리즘 등을 효율적으로 구동시킬 수 있었으며, MCU 및 ECU의 구현으로 페데스털, 카메라와 렌즈, 그리고 레이저 서치라이트도 동시에 효율적인 원격제어가 가능하였다.

실선 테스트를 통하여 하드웨어 설계 및 구현 과정에서 몇 가지의 보완사항이 발견되었으며, 다음 설계에는 문제점을 보완할 예정이다. 페데스털과 탑재장비의 제어에는 문제가 없었지만 보다 사용자가 편리하게 사용할 수 있는 사용자 위주의 유저 인터페이스가 필요하다는 것을 확인했다.

참 고 문 헌

[1] 김정근, “선박용 지향성 페데스털의 추종 및 동요 안정화 제어에 관한 연구,” 한국해양대학교 박사 논문, 2008.

[2] 진강규, 황승욱, “선박용 동요 안정식(Stabilization) Night Vision System개발,” 산업자원부 산업기술개발 보고서, 2007.

[3] 김정근, 송세훈, 황승욱, 진강규, “선박 Night Vision 시스템용 페데스털의 제어부 개발,” 2006 한국마린엔지니어링학회 전기학술대회 논문집, pp.107-108

[4] JK. KIM, SH. SONG, SH. BAEK, SW. HWANG and GG. JIN, "Design of a 페데스털 Part for the Marine Surveillance Night Vision System," 2006 Asia Navigation Conference, pp. 123-126

[5] 안양근, “선박용 위성안테나의 Stabilized 페데스털 구조와 제어알고리즘의 설계 및 구현에 관한 연구,” 한국해양대학교 석사학위 논문, 1997.

[6] 고운용, 황승욱, 진강규, “선박용 위성 안테나 페데스털의 안정화 제어,” ’98 한국자동제어학술회의 논문집, pp. 188-191, 1998.

[7] 고운용, “Stabilized 위성안테나용 페데스털 제어장치의 설계 및 구현에 관한 연구,” 한국해양대학교 석사학위 논문, 1998.

[8] J. Uffenbeck, The 80x86 Family Design, Programing, and Interfacing, Prentice Hall Inc., 1992.

[9] Nippon Pulse Motor Co., Ltd, User's Manual For PCL6045B Pulse Control LSI, MNAL.MO.PCL-6045B-1, 2003.

[10] E. Auzas, "Design Considerations for the Embedded PC," Embedded System Conference West San Jose, California, 1995.

[11] 송세훈, “2DOF PID 제어기의 RCGA기반 동조에 관한 연구,” 한국해양대학교 석사학위 논문, 2008.

[12] Mituhiko Araki and Hidefumi Taguchi. (2003). Two-Degree-of-Freedom PID Controllers. *International Journal of Control, Automation, and Systems* Vol. 1, No. 4.

[13] M. Kanda and K. Hiroi. (1991). Super two-degree-of-freedom PID algorithm. *Proc. 30th SICE Annual Conference*, pp. 465-466.

[14] 홍건표, 하동호, 개발자들을 위한 ARM 프로세서, Ohm사, 2006.

[15] 윤덕용, AVR ATmega128 마스터, Ohm사, 2004.

[16] SEIKO EPSON CORPORATION, LCD Controller ICs S1D13305 Series Technical Manual, MF1167-02, 2001.