

졸업논문청구논문

**소형 천체망원경의 자동화를 위한 GS-system
개발**

**Development of GS-system for a Small Astronomical
Telescope**

곽지성 (郭志誠 Gwak, Ji-Seong)

18008

과학영재학교 경기과학고등학교

2020

소형 천체망원경의 자동화를 위한 GS-system

개발

Development of GS-system for a Small Astronomical Telescope

[논문제출 전 체크리스트]

1. 이 논문은 내가 직접 연구하고 작성한 것이다.
2. 인용한 모든 자료(책, 논문, 인터넷자료 등)의 인용표시를 바르게 하였다.
3. 인용한 자료의 표현이나 내용을 왜곡하지 않았다.
4. 정확한 출처제시 없이 다른 사람의 글이나 아이디어를 가져오지 않았다.
5. 논문 작성 중 도표나 데이터를 조작(위조 혹은 변조)하지 않았다.
6. 다른 친구와 같은 내용의 논문을 제출하지 않았다.

Development of GS-system for a Small Astronomical Telescope

Advisor : Teacher Park, Kie Hyun

by

18008 Gwak, Ji-Seong

Gyeonggi Science High School for the gifted

A thesis submitted to the Gyeonggi Science High School in partial fulfillment of the requirements for the graduation. The study was conducted in accordance with Code of Research Ethics.*

2020. 7. 21.

Approved by
Teacher Park, Kie Hyun
[Thesis Advisor]

*Declaration of Ethical Conduct in Research: I, as a graduate student of GSHS, hereby declare that I have not committed any acts that may damage the credibility of my research. These include, but are not limited to: falsification, thesis written by someone else, distortion of research findings or plagiarism. I affirm that my thesis contains honest conclusions based on my own careful research under the guidance of my thesis advisor.

소형 천체망원경의 자동화를 위한 GS-system 개발

곽지성

위 논문은 과학영재학교 경기과학고등학교 졸업논문으로
졸업논문심사위원회에서 심사 통과하였음.

2020년 7월 21일

심사위원장 박수종 (인)

심사위원 박우진 (인)

심사위원 박기현 (인)

Development of GS-system for a Small Astronomical Telescope

Abstract

초록(요약문)은 가장 마지막에 작성한다. 연구한 내용, 즉 본론부터 요약한다. 서론 요약은 하지 않는다. 대개 첫 문장은 연구 주제 (+방법을 핵심적으로 나타낼 수 있는 문구: 실험적으로, 이론적으로, 시뮬레이션을 통해)를 쓴다. 다음으로 연구 방법을 요약한다. 선행 연구들과 구별되는 특징을 중심으로 쓴다. 뚜렷한 특징이 없다면 연구방법은 안써도 상관 없다. 다음으로 연구 결과를 쓴다. 연구 결과는 추론을 담지 않고, 객관적으로 서술한다. 마지막으로 결론을 쓴다. 이 연구를 통해 주장하고자 하는 바를 간략히 쓴다. 요약문 전체에서 연구 결과와 결론이 차지하는 비율이 절반이 넘도록 한다. 읽는 이가 요약문으로부터 얻으려는 정보는 연구 결과와 결론이기 때문이다. 연구 결과만 레포트하는 논문인 경우, 결론을 쓰지 않는 경우도 있다.

key word : Telescope

소형 천체망원경의 자동화를 위한 GS-system 개발

초 록

본 연구는 소형 천체망원경의 아크릴 바흐티노프마스크 원격제어가 가능한 제어시스
템인 GS-system을 개발하였다. GS-system은 기존에 개발한 모터포커서를 사용하되, 바흐
티노프마스크의 원격 제어 및 다른 여러 기능을 탑재하여 사용의 편의성을 증대시켰으며,
이를 사용하여 특정 소형 천체망원경에 사용할 수 있는 바흐티노프마스크 제어용 덮개를
제어할 수 있도록 하였다. GS-system을 통해 소형 천체망원경에 사용되는 바흐티노프마스
크의 제어를 편리하게 할 수 있으며, 천체 관측에 필요한 기능들 또한 ASCOM 드라이버와
호환되는 소프트웨어들을 통한 GUI (graphical user interface)를 활용해 편리하게 사용할 수
있도록 하였다.

키워드 : 자동망원경

Contents

| | |
|---|------|
| Abstract | i |
| Contents | iii |
| List of Figures | vi |
| List of Tables | viii |
| I 서론 | 1 |
| I.1 연구의 필요성 | 1 |
| I.2 연구 목적 | 4 |
| I.3 연구 문제 | 5 |
| II 연구 과정 및 방법 | 7 |
| II.1 소형 천체망원경 자동화 시스템 구성 | 7 |
| II.2 구동부 제작 | 7 |
| II.2.1 후드 제작 | 7 |
| II.2.2 아크릴 덮개 제작 | 8 |
| II.2.3 바흐티노프마스크 제작 | 10 |
| II.2.4 열선 제작 | 12 |
| II.3 제어부 제작 | 12 |
| II.3.1 모터 포커서 제어 | 12 |
| II.3.2 서보모터 제어 | 14 |
| II.3.3 열선 제어 | 15 |
| II.3.4 EEPROM(Electrically Erasable Read-Only Memory) | 16 |
| II.3.5 릴레이 스위치 | 17 |
| II.4 코딩 | 18 |

| | | |
|------------|--|-----------|
| II.4.1 | 아두이노 코딩 | 18 |
| II.4.2 | ASCOM 드라이버 및 제어 프로그램 | 18 |
| III | 연구 결과 | 19 |
| III.1 | 하드웨어 완성-구동부 | 19 |
| III.1.1 | 열선 제어 | 19 |
| III.1.2 | EEPROM(Electrically Erasable Read-Only Memory) | 19 |
| III.1.3 | 서보모터 제어 | 20 |
| III.2 | 소프트웨어 캡쳐해서 추가 | 21 |
| IV | 결론 | 23 |
| A | 부록 | 25 |
| 1 | 아두이노 코드 | 25 |
| 1 | GS-system.ino | 25 |
| 2 | Board.h | 30 |
| 3 | ButtonControl.ino | 31 |
| 4 | displayAdafruit.ino | 32 |
| 5 | EEPROM.ino | 35 |
| 6 | MainControl.ino | 36 |
| 7 | Setstep.ino | 39 |
| 2 | ASCOM driver 코드 | 40 |
| 1 | driver.cs | 40 |
| 2 | FCSH.cs | 54 |
| 3 | FCSH.Designer.cs | 55 |
| 4 | MainWindow.cs | 58 |
| 5 | ResetForm.cs | 63 |

| | | |
|-------------------|---------------------------------------|-----------|
| 6 | ResetForm.Designer.cs | 64 |
| 7 | SetupDialogForm.cs | 65 |
| 8 | SetupDialogForm.designer.cs | 68 |
| References | | 73 |

List of Figures

| | | |
|-------------------|---|----|
| Figure 1. | FocusMAX 소프트웨어를 이용하여 FWHM 값의 V-curve를 얻어 초점을 결정하는 모습 [1]. | 2 |
| Figure 2. | PP(Polypropylene) 소재의 파일 표지를 잘라서 만든 바흐티노프 마스크 . | 3 |
| Figure 3. | Takahashi사의 FSQ-106ED의 사양 [2]. | 7 |
| Figure 4. | Fusion360을 이용하여 설계한 경통 덮개 조각들 | 8 |
| Figure 5. | 제작한 후드를 천체망원경에 장착한 모습 | 8 |
| Figure 6. | 개선한 경첩 및 서보모터 | 9 |
| Figure 7. | 사용한 서보모터인 ds lx3325mg 25kg | 10 |
| Figure 8. | 서보모터를 제작한 후드에 장착한 모습. | 11 |
| Figure 9. | 후드에 서보모터를 고정하기 위해 볼트를 이용해 고정한 모습 | 12 |
| Figure 10. | 레이저에 의해 휘어진 아크릴 마스크 | 12 |
| Figure 11. | (a) 4조각으로 나누어서 출력한 FSQ106의 바흐티노프 마스크 (b) 출력된 마스크를 경통에 붙인 모습. 빛이 새지 않도록 접착부를 검은색 테이프 로 접합시켜 고정하였다. | 13 |
| Figure 12. | GS-touch의 주요 기능 | 14 |
| Figure 13. | (a) 제작한 열선의 접합부와 (b) 제작한 열선을 12V에 연결한 모습. 끝에 벨크로가 붙어있어 경통에 둘러 사용할 때 편리하다. | 15 |
| Figure 14. | EEPROM의 address별 사용 구조. 부호와 값을 절대치를 이용하여 연산 하였다. | 17 |
| Figure 15. | 실험에 사용된 릴레이스위치를 모터포커서에 연결시킨 모습 | 17 |
| Figure 16. | PWM제어를 위한 코드 | 19 |
| Figure 17. | EEPROM에서 정보를 읽어오는 과정 | 20 |

| | | |
|-------------------|---|----|
| Figure 18. | (a) 개선된 모터포커서를 이용하여 제어한 덮개 (b) 이 때 개선된 모터포 커서에 출력되는 마스크의 상태. 화면 아래의 버튼들을 이용해 상태를 제어할 수 있으며, 제어상태를 확인할 수 있다. | 21 |
| Figure 19. | 개선된 모터포커서 드라이버의 GUI 모습 | 22 |

List of Tables

Table 1. 보조관측실에 설치된 소형 천체망원경의 자동화를 위해 필요한 컨트롤 . . . 4

I. 서론

I.1 연구의 필요성

천체망원경이 대중화됨에 따라 일반인들도 소형 천체망원경과 DSLR (Digital Single Lens Replex) 카메라나 CCD 등을 이용하여 사진 관측을 하는 경우가 있다. 대부분의 천체관측은 해가 진 야간에 날씨가 맑을 경우에만 가능하다보니 시간적으로 매우 제한적이다. 또한 지상에서의 천체관측은 대기 영향에 의해 많은 어려움이 따른다. 대기의 영향을 최소화 하기 위해서는 관측 대상의 고도가 높은 시각에 관측을 하는 것이 유리하다.

또한 사진 관측에서 중요한 요소 중 하나는 초점을 정확하게 조절하는 것이다. 사진 관측시 온도 변화로 인해 경통의 팽창, 수축으로 인하여 정확하게 조절해 놓은 초점이 변하는 경우도 있다. 최근에는 이러한 문제를 해결하기 위해 컴퓨터를 이용하여 정확한 제어를 통해 짧은 시간안에 효율적인 관측을 진행하고자 노력하고 있다. Persha (2001)는 주위 온도에 따라서 초점이 변화한다는 점을 보완하고자 이를 보정할 수 있도록 온도 보상 초점 조절 방법을 연구하였다. [3]

사진 관측시 천체망원경의 초점 조절을 정밀하게 하기 위하여 다음과 같은 방법을 사용하고 있다.

첫째는 사진에서의 대기의 상태에 의한 별의 점퍼짐 함수의 FWHM (Full Width Half Maximum) 값을 활용하여 V-curve를 그리는 방법이다. Figure 1은 FocusMax 소프트웨어로 포커서를 움직이며 별의 점퍼짐 함수의 FWHM을 구하여 V-curve를 얻어 최적의 초점을 찾는 모습을 나타낸다. 이 방법을 활용할 경우 어떤 상황에서도 꽤 정확한 결과를 얻을 수 있지만, 이는 별의 플럭스가 정규분포로 퍼져있어야 정확한 결과를 얻을 수 있다. 만약 별이 포화되어 점퍼짐 함수가 정규분포와 모양이 다른 경우에는 HFD (Half Flux Diameter) 값을 이용할 수 있지만 시잉(seeing)의 영향으로 여러가 발생하여 초점을 조절하는데 어려움을 겪는다.

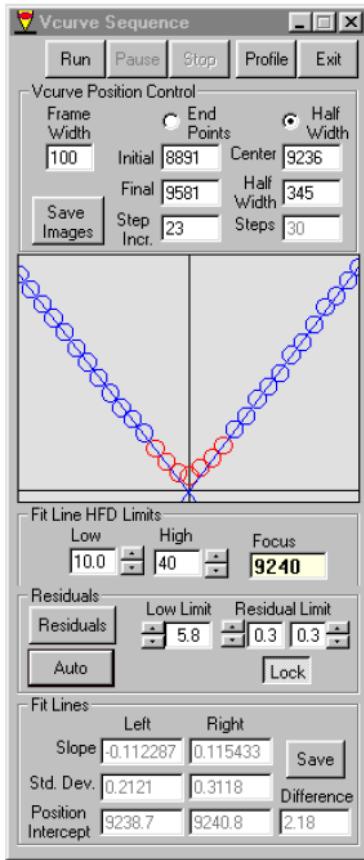


Figure 1. FocusMAX 소프트웨어를 이용하여 FWHM 값의 V-curve를 얻어 초점을 결정하는 모습 [1].

이런 어려움을 개선하기 위해서 다른 방법으로 마스크를 활용하는 방법이 주목받고 있다. 천체관측에서 초점 조절에 사용하는 하트만 마스크(Hartmann mask)와 바흐티노프 마스크(Bahtinov mask)가 있다. 바흐티노프 마스크는 러시아의 아마추어 천문학자인 바흐티노프가 고안한 마스크 중 하나로, 기존에 사용하던 하트만 마스크의 여러 단점을 보완하였기 때문에 이후 주류가 된 마스크의 종류이다. 두 마스크 모두 빛의 회절 현상을 이용한다는 공통점이 있지만, 천체관측에서는 바흐티노프 마스크가 더 포괄적으로 사용된다. 마스크를 이용하면 점퍼짐 함수의 FWHM 값을 구하여 초점을 조절하는 것에 비해 대기에 의한 영향을 덜 받는다는 것이 장점이며, 제작 및 관리에도 용이하기 때문에 천체망원경의 원격

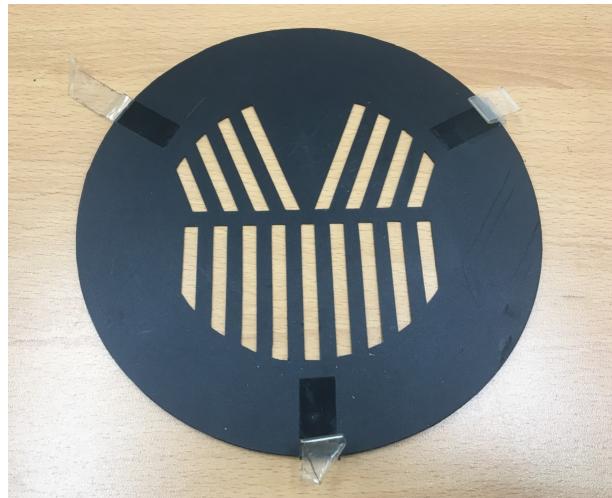


Figure 2. PP(Polypropylene) 소재의 파일 표지를 잘라서 만든 바흐티노프 마스크

제어에 알맞은 특징들을 지니고 있다.

빛의 회절은 직진하던 빛이 좁은 틈, 슬릿이나 장애물을 통과할 때 물체의 뒤편까지 빛이 나가는 현상으로, 슬릿의 폭이 좁을수록 회절이 잘 일어난다. 바흐티노프 마스크는 Figure. 2와 같이 방향이 어긋나있는 세 줄의 회절 슬릿들이 일렬로 나있는 모양을 가지고 있으며, 평행 광선인 별에서 오는 빛이 이러한 모양의 슬릿을 통과하면 빛은 좁은 틈에서 회절하게 된다. 빛은 좁은 방향으로 많이 회절하기 때문에 슬릿을 통과한 빛은 통과한 슬릿에 수평한 방향의 선 모양상을 만들게 된다. 초점이 올바르지 않은 경우에는 평행광이 한점에 모이지 않기 때문에 세 개의 선이 한 점에서 만나지 않지만, 초점이 정확하게 맞추어진 경우에는 상들이 한 점에서 모이기 때문에 세 개의 선이 한 점에서 만나게 된다.

바흐티노프 마스크는 빛의 회절을 이용하여 초점이 정확한지 여부를 쉽고 빠르게 알 수 있기 때문에 관측시간이 중요한 천체망원경의 초점을 맞추는 데 안성맞춤이다. 회절 슬릿을 이용하기 때문에 아주 밝은 별로만 초점 검출을 할 수 있다는 단점을 가지고 있지만 상대적으로 노출의 시간을 늘리는 밤에 천체관측을 할 때는 바흐티노프마스크가 좋은 선택이라고 할 수 있다.

본교 보조관측실은 자동으로 개폐되는 슬라이딩 루프(sliding roof)가 있어, 이곳에 소형 천체망원경을 고정하여 설치해 놓고 관측을 진행하고 있다. 천체망원경의 완전 자동화가 가능하다면 제한된 관측 가능 시간에 관측 성공률이 높아질 것이다. 실제로 여러 천문대에서 천체망원경 자동화를 위해 노력하고 있고, Zhang 등(2016)은 남극에 위치한 망원경의 자동관측에 대해 연구하였다 [4].

Budding (1995) 또한 뉴질랜드 카터 천체관측소(Carter)에 자동화된 소형 천체망원경을 설치하여 운영하면서 소형 천체망원경을 원격제어할 수 있도록 네트워크를 활용하면 일반적인 천체관측에 비해 비용측에서 큰 이득을 보며, 편의성또한 증대할 수 있다고 주장하였다. [5]

소형 천체망원경의 원격 제어 및 자동화를 위해서는 돔 컨트롤, 전원 컨트롤, 마운트 컨트롤, 초점 컨트롤, 망원경 덮개 컨트롤, 이슬 제거를 위한 열선 컨트롤 등이 스스로 동작하거나 컴퓨터로 제어가 가능해야 한다. 현재 본교 보조관측실에 설치된 소형 천체망원경의 자동화를 구현하기 위해서 필요한 것들을 Table 1에 나타내었다.

Table 1. 보조관측실에 설치된 소형 천체망원경의 자동화를 위해 필요한 컨트롤

| 컨트롤 | 구축 여부 | 제어 방법 | 비고 |
|-------------|---------|--------------------|--------------|
| 슬라이딩 루프 컨트롤 | 구축됨 | 아두이노, 릴레이스위치 | 전용 소프트웨어로 제어 |
| 마운트 컨트롤 | 구축됨 | 제조사 제공 망원경 컨트롤 시스템 | ASCOM 제어 |
| 전원 컨트롤 | 별도로 구축됨 | 아두이노, 릴레이스위치 | 전용 소프트웨어로 제어 |
| 초점 컨트롤 | 별도로 구축됨 | GS-touch | ASCOM 제어 |
| 망원경 덮개 컨트롤 | 구축안됨 | | |
| 열선 컨트롤 | 별도로 구축됨 | 수동 전기 회로 | 수동 제어 |

I.2 연구 목적

본 연구에서는 경기과학고등학교 보조관측실에 설치되어 있는 소형 천체망원경의 자동화를 위한 장치를 개발하는 것이 그 목적이다. 자동화를 위한 장치의 이름은 경기과학고등학교의 영문명인 Gyeonggi Science high school for the gifted 중에서 GS를 따와서 GS-system이라 명명하였다. GS-system은 기기의 전원 컨트롤, 마운트 컨트롤, 망원경 덮개 컨트롤, 초점

조절, 이를 제거를 위한 열선 컨트롤 등을 포함하고 있으며, 이것들을 컴퓨터로 컨트롤 하는데 성공한다면 인터넷을 통해 컴퓨터를 원격 조정하는 방법이나 컴퓨터 프로그램으로 미리 지정한 스케줄에 따라 기기들의 움직임을 설정하는 방법으로 소형 천체망원경의 자동화를 구현할 수 있게 된다.

제작한 덮개 및 시스템을 위한 펌웨어는 기존에 모터 포커서 컨트롤러로 제작하였던 GS-touch를 개량하여 사용하였으며, GS-touch와 같이 ASCOM(Astronomy Common Object Model)을 지원하여 확장성을 넓힐 수 있도록 계획하였다. GS-touch의 전용 ASCOM driver 또한 천체망원경의 원격 제어에 맞추어 개량하여 일반인들 또한 focusMAX와 같은 천체관측 소프트웨어를 통하여 쉽게 접근할 수 있도록 계획하였다.

I.3 연구 문제

본 연구에서 제시하는 연구 문제는 다음과 같다:

수정이 필요함(박기현샘)

1. 덮개는 바흐티노프 마스크를 정확하게 제어할 수 있는가?
2. 바흐티노프 마스크를 이용하여 초점을 정확하게 맞출 수 있는가?
3. 향상된 모터포커서를 이용하여 이들을 제어할 수 있는가?

이들을 통해 얻을 수 있는 결과는 다음과 같다.

1. 일반PC를 통해 천체망원경의 여러 시스템들을 제어할 수 있도록 한다. 이는 카메라의 전원, 가대의 전원 등을 포함하며, 주요 제어대상은 경통의 길이이다.
2. ASCOM에 따른 표준 초점 제어 시스템을 제공하며, FocusMAX와 같은 천체관측 소프트웨어를 통한 원활한 제어가 가능하게 한다.
3. 앞서 제시한 모든 시스템을 원격으로 조작할 수 있도록 하며 여러 센서들을 통해 정확하게 제어할 수 있도록 한다.

Godoy 등(2018)은 실험 결과와 유사한 연구 문제를 제시하였으며 [6], 이를 실제로 실험에 활용하였다. 본 결과들을 통해 앞으로 자동망원경을 제작하려는 사람들에게 도움이 되고자 한다.

II. 연구 과정 및 방법

II.1 소형 천체망원경 자동화 시스템 구성

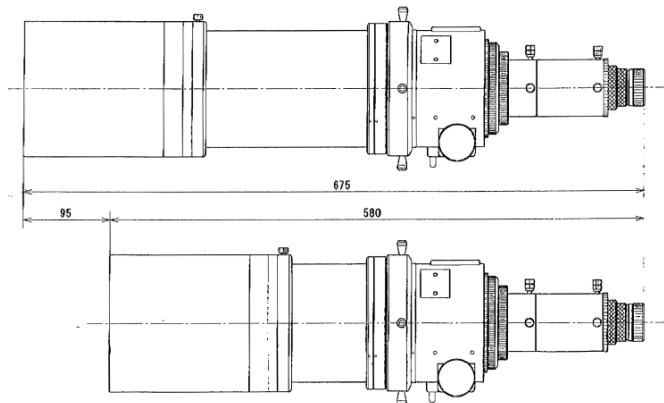


Figure 3. Takahashi사의 FSQ-106ED의 사양 [2].

II.2 구동부 제작

II.2.1 후드 제작

Fusion360(<https://www.autodesk.co.kr/products/fusion-360/students-teachers-educators>)을 이용하여 Figure 4와 같이 경통의 후드를 설계한 후, 3D 프린터로 출력하여 후드를 제작하였다. 경통의 후드는 천체망원경 별로 경통의 지름과 같은 특성이 다르고, 천체망원경 위에 고정 시킬 수 있을 만큼 견고해야한다. 때문에 경통에 씌울 수 있도록 지름을 계산하여 팔각형 모양으로 감싸는 형태로 제작하였으며, 3D 프린터에서 출력할 수 있는 크기 제한이 있고 동시에 천체망원경에 부착시킬 때 편리하게 할 수 있도록 총 네 조각으로 나누어 조립하는 방식을택하였다. 네개 조각의 렌치 볼트와 너트로 조립하여 완성할 수 있다.

본 연구에서는 부품들을 3D 프린터로 출력 후 조립하여 Figure 5와 같이 이를 천체망원경의 후드처럼 부착시키는 것에 성공하였다. 후드의 지름은 망원경에 따라 달라질 수 있으므로 다른 망원경에 대한 후드를 제작할 때에는 이에 맞추어 새로 제작하여야 한다.

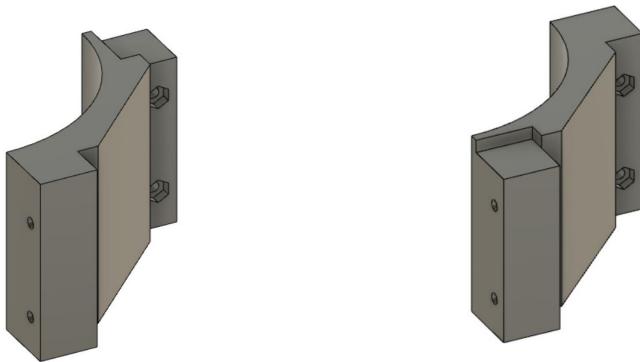


Figure 4. Fusion360을 이용하여 설계한 경통 덮개 조각들



Figure 5. 제작한 후드를 천체망원경에 장착한 모습

II.2.2 아크릴 덮개 제작

천체망원경의 자동화를 실현하기 위해서는 경통의 덮개를 열고 닫는 것이 필요한데 소형 천체망원경의 경우 이러한 기능이 개발되어 있지 않다. 천체망원경의 광학계는 사용하지 않을 때에는 먼지, 이슬 등의 피해를 최소화 하기 위하여 덮개를 덮어 두어야 하고, 관측시에만 덮개를 열고 관측하는 것이 바람직하다.

이에 제작한 후드에 꼭 맞는 크기로 아크릴을 가공하여 덮개를 제작한 경첩을 달아 서보모터로 열고 닫을 수 있도록 설계하였다. 처음에는 Figure 5와 같이 아크릴 경첩을 이용하였으나 내구성에서 문제가 있다고 판단되어 견고하게 제작하기 위하여 금속 경첩으로

교체하였다. 금속으로 제작된 경첩은 볼트로 체결하는 방식으로 후드와 안정적으로 결합할 수 있었으며, 내구성 또한 뛰어났기 때문에 서보모터로 열고 닫는데 문제가 없었다. Figure 6 와 같이 금속 재질의 경첩은 덮개와 아크릴 바흐티노프 마스크 사이를 납작볼트를 이용하여 연결할 수 있으며, 이는 접착제를 이용하여 붙이는 방법보다 간단하고 견고하게 조립되었다.

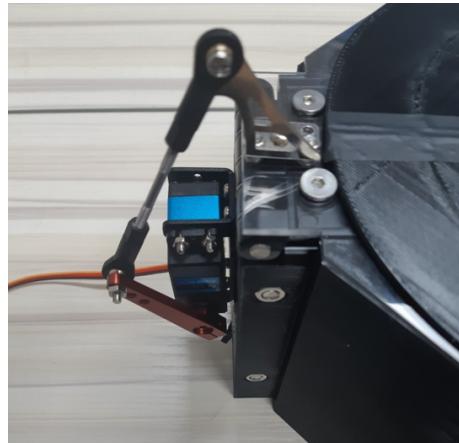


Figure 6. 개선한 경첩 및 서보모터

경통의 후드에서 바흐티노프마스크를 확실하게 제어하기 위해서는 바흐티노프마스크를 적용할 때와 적용하지 않을 때의 구분이 확실해야한다. 이를 확실하게 하기 위해서 바흐티노프마스크를 경첩을 이용하여 큰 각도로 제어하는 방법을 택하였으며, 이를 위해 덮개를 설계할 때 Figure 4와 같이 경첩의 높이를 고려하여 제작하였다.

서보모터는 일반적으로 사용하는 DC모터와 다르게 원하는 각도로 모터의 속도를 조절하여 이동시킬 수 있는 모터로, RC카의 방향제어, 로봇의 관절제어 등의 상황에서 자주 사용되곤 한다. 본 연구에서 사용되는 서보모터는 ‘ds lx3325mg 25kg’ 모델(Figure 7)으로, 서보모터이지만 360도 회전이 가능한 모델으로, 몸체가 금속으로 되어있어 내구성을 기대 할 수 있으며, 축이 톱니모양으로 되어 있기 때문에 제어가 편리하다는 장점을 가지고 있다.

서보모터의 경우 구동을 위해 필요한 핀은 3가지이며, 일반적으로 주황색, 빨간색, 갈색의 핀으로 이루어져 있다. 주황색 핀은 모터를 제어할 수 있는 핀이며, 빨간색 핀과 갈색



Figure 7. 사용한 서보모터인 ds lx3325mg 25kg

핀은 각각 5v 전원과 GND에 연결하여 서보모터를 구동할 수 있다. 서보모터의 경우 Figure 8처럼 후드의 옆면에 부착시켜 일정한 각도로 제어할 수 있게끔 설계하였다.

서보모터는 후드의 옆면에 부착시켜 제어시킨다. 이 때 정확한 위치에 부착시킬 수 있도록 일정한 간격을 두어 실험을 반복하였으며, Figure 9와 같이 적합한 위치를 찾아 고정시켰다. 서보모터로 마스크를 정확하게 제어하기 위해서는 마스크가 완전히 덮개에 고정될 수 있어야 하므로 이에 맞는 각도를 계산하여 사용하여야 한다.

II.2.3 바흐티노프마스크 제작

천체망원경에 사용되는 대부분의 바흐티노프마스크의 도안은 직접 제작이 가능하며, 본 연구의 경우 astrojargon (<http://astrojargon.net/MaskGenerator.aspx>) 사이트에서 망원경의 사양 및 마스크의 사용 목적에 맞는 적절한 바흐티노프마스크의 도안을 제작하여 사용하였다.

덮개에서 사용되는 방식을 이용한 바흐티노프마스크의 경우 기존처럼 종이나 알루미늄에 프린트된 방식일 경우 덮개에 원하는 모양으로 씌워지지 않을 가능성이 있다. 때문에 주변환경에 영향을 적게 받으면서도 그 면이 평평하여 천체관측을 진행할 때에 영향이 없어

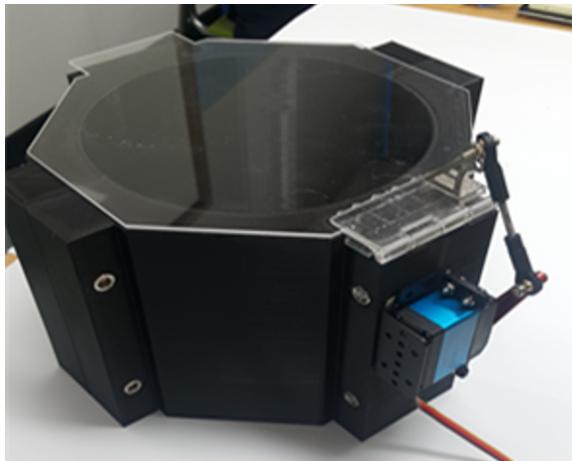


Figure 8. 서보모터를 제작한 후드에 장착한 모습.

야 한다.

연구 초기에는 아크릴이 이러한 성질을 만족하면서도 쉽게 제작할 수 있으므로 아크릴에 레이저를 써여 바흐티노프마스크 모양을 써우는 방법으로 테스트용 마스크를 제작해보았으나. 이러한 방식을 사용할 경우 Figure 10처럼 레이저로 인해 열을 받은 아크릴이 변형되어 정밀한 초점 조절이 불가능해지는 경우가 발생하였으며, 아크릴의 두께로 인해 실제로 초점을 조절할 때에 걸리는 여러 가지 변수 또한 무시할 수 없었다.

때문에 선택한 새로운 방법은 3d프린터를 이용하여 바흐티노프 마스크를 출력하는 것이다. 이 방법으로 얇으면서도 딱딱한 마스크를 사용할 수 있으므로 일반적인 아크릴을 활용한 마스크보다 적은 변수로 바흐티노프 마스크를 덮개에 부착시킬 수 있었다.

Astrojargon을 이용하여 출력한 바흐티노프마스크는 D값을 106, focal length 값을 530으로 출력하였으며, 덮개의 아크릴에 붙이기 편리하도록 지름을 아크릴과 같은 크기로 제작하였다. 출력 과정 중 바흐티노프 사이즈의 크기가 크기 때문에 4조각으로 나누어서 출력하였으며, 빛이 새는 것을 방지하기 위해 나중에 이를 검은 색 테이프로 봉합하였다.

또한, 3D 프린터의 특성상 한쪽 면은 거친 면, 다른 한쪽 면은 평평한 면을 가지고 있다.

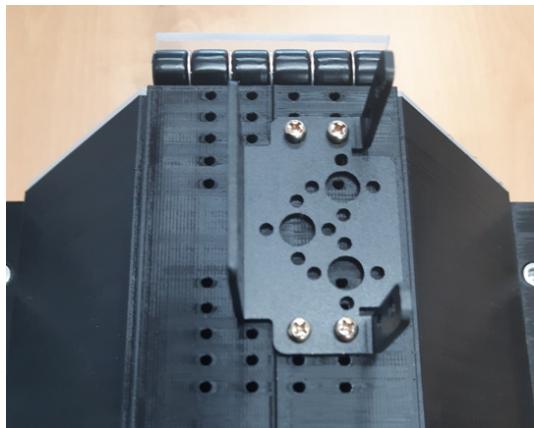


Figure 9. 후드에 서보모터를 고정하기 위해 볼트를 이용해 고정한 모습



Figure 10. 레이저에 의해 휘어진 아크릴 마스크

이 때 거리를 일정하게 하기 위해 평평한 면이 아크릴과 붙는 방향으로 고정시켰다.

II.2.4 열선 제작

열선....

II.3 제어부 제작

II.3.1 모터 포커서 제어

이전에 개발된 GS-touch(Gyeonggi Science high school for the gifted touch)는 바이폴라 스테핑 모터를 제어할 수 있는 모터 포커서로 초점 조절 기능은 이기 때문에 이를 위한 기능들이 주를 이루었다. 때문에 이를 보완하여 덮개를 제어할 수 있도록 추가로 개발할 필요가 있다.



Figure 11. (a) 4조각으로 나누어서 출력한 FSQ106의 바흐티노프 마스크 (b) 출력된 마스크를 경통에 붙인 모습. 빛이 새지 않도록 접착부를 검은색 테이프로 접합시켜 고정하였다.

Figure 12는 기존에 사용하는 GS-touch의 주요 기능들을 정리한 표이며, GS-touch는 전체적으로 두 가지 부분으로 나눌 수 있다.

첫 번째로 GS-touch의 펌웨어는 모터를 직접적으로 제어할 수 있다. DRV8825 모터 드라이버를 활용하여 최대 1/32step의 마이크로 컨트롤을 지원하기 때문에 정밀하게 경통의 길이를 조절할 수 있도록 설계되었으며, 이를 원하는 숫자로 초기화하여 얼마나 더 움직였는지 또한 알기 쉽게 하였다. 부가적인 기능으로는 현재 컨트롤러의 위치의 온도와 습도를 알 수 있도록 하여 주변 상황을 쉽게 알 수 있도록 하였다.

두 번째 부분은 GS-touch의 직접적인 제어가 가능하도록 하는 ASCOM 호환 드라이버이다. GS-touch의 드라이버는 ASCOM 홈페이지에서 제공하는 개발자용 버전을 응용하여 C# 기반으로 제작되었으며, GUI 및 애플리케이션 소프트웨어를 제공하기 때문에 펌웨어에서 제공하는 모든 기능들을 컴퓨터에서 원격으로 사용할 수 있도록 하였다.

비록 GS-touch는 여러 가지 기능을 가지고 있지만 천체망원경의 원격조작에 필요한 여러 기능들을 완벽하게 갖추지는 못하였다. 대표적으로, EEPROM을 사용하지 않았기 때문에 원하는 길이에 step값을 저장할 수 없으며, 온도 및 습도 측정 외의 편의기능 또한 없기

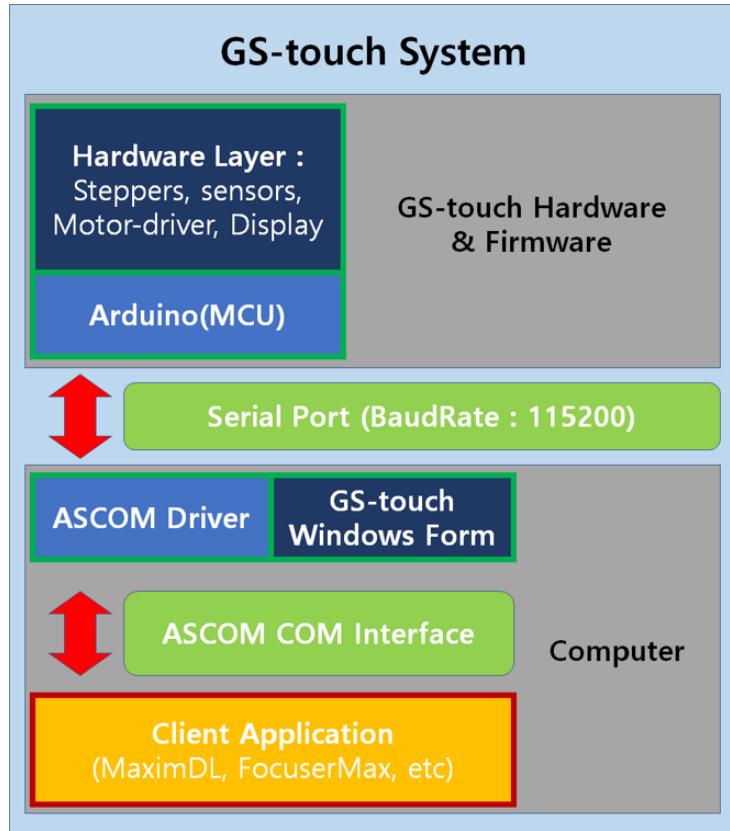


Figure 12. GS-touch의 주요 기능

때문에 실제로 사용할 때 많은 불편함이 있다. 본 연구에서는 이러한 단점을 보완하여 EEPROM을 적용시키고 열선을 사용 가능하도록 하는 등의 여러 편의 기능들을 개선하였으며, 이를 사용하여 천체망원경용 덮개를 제어할 수 있도록 하였다. 앞으로 실제 천체관측에서의 활용성은 기존에 없었던 천체망원경용 덮개와 더불어 천체망원경의 원격 조작에 큰 도움을 줄 수 있을 것이다.

II.3.2 서보모터 제어

바흐티노프마스크를 제어하기 위해서 필요한 조건을 여러 가지가 있겠지만, 가장 중요한 조건 중 하나는 마스크를 사용하는 때와 그렇지 않을 때를 정확히 구분해야하며, 특히 마스크

를 사용할 때에는 모든 상황에서 같은 환경을 제공할 수 있어야 한다. 본 연구에서는 경첩을 이용한 각도의 차이로 마스크를 제어하기 때문에 정확한 각도의 이동이 가장 중요하기 때문에 원하는 각도로 정확히 이동할 수 있는 서보모터를 사용하여 정확한 제어가 가능하도록 하였다.

II.3.3 열선 제어

열선또한 정확한 천체관측에 필요한 장비 중 하나이다. 관측을 진행할 때에 경통의 온도가 내려가면 렌즈에 이슬이 맷하게 되는데, 이는 정확한 상을 맷하게 하는데 크게 어려움을 주게 된다. 열선을 경통에 감아놓으면 경통의 온도에 따라서 열선을 제어하여 깨끗한 렌즈를 사용할 수 있게되므로 관측을 편리하게 할 수 있으며, 때문에 천체망원경을 원격제어하기 위해서는 온도를 측정할 수 있는 장비와 열선이 필수적이다. 열선은 히팅케이블을 이용하여 제작하였으며, 망원경의 크기와 열의 필요세기 등을 고려하여 히킹케이블의 저항을 선택한다.

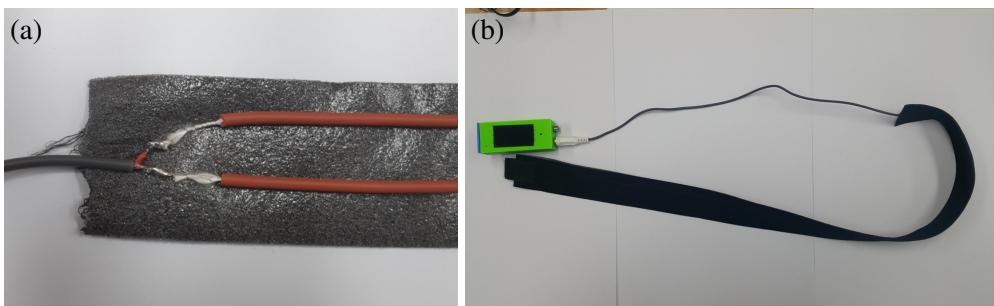


Figure 13. (a) 제작한 열선의 접합부와 (b) 제작한 열선을 12V에 연결한 모습. 끝에 벨크로가 붙어있어 경통에 둘러 사용할 때 편리하다.

열선을 제작하는 방법은 간단하다. 먼저 Figure 13a처럼 끈끈한 면에 히팅케이블을 부착한 뒤에 알맞는 12V 어댑터를 분해하여 극을 선과 연결한다. 그리고 열선을 다시 덮어주게 되면 12V 전원을 통해 작동하는 열선을 제작할 수 있다. 열선을 실제로 사용할 시에는 그 편의성을 증대시키기 위해 열선의 끝부분을 벨크로로 연결할 수 있도록 만들게 되며, 이렇게

제작한 열선은 렌즈가 있는 부분을 찾아 둘러주면 벨크로로 끝을 고정시켜 쉽게 망원경에 부착시킬 수 있다.

열선은 PWM(Pulse Width Modulation)을 이용하여 세기를 조절할 수 있다. PWM은 디지털 신호의 밀도로 그 세기를 조절하는 방식으로 디지털 신호인 0과 1이 출력되는 시간의 비율을 조절하여 원하는 밀도로 제어할 수 있도록 한다. 본 연구에서 직접 제작한 열선을 포함한 대부분의 열선은 12V의 전압을 이용해 제어해야하기 때문에 아두이노의 출력 전원인 5V로는 신호만 제어하고 스위칭 트랜ジ스터를 이용하여 12V의 전압을 PWM으로 제어할 수 있도록 설계하였다.

II.3.4 EEPROM(Electrically Erasable Read-Only Memory)

MCU(Micro Controller Unit)내에 원하는 값을 저장하는 방법은 여러 가지가 있다. 일반적으로 펌웨어가 실행되기 전에 변수를 선언하고, loop 문 속에 들어있는 여러 함수들을 통해 그 값을 바꾸는 방법을 사용하곤 한다. 하지만 원격 천체관측을 위한 펌웨어인 만큼, MCU내의 변수만으로 값을 저장하는 것은 상당히 위험하며, 그 전원을 계속 유지할 수 없기 때문에 다른 방법으로 값을 저장할 필요성을 느꼈으며, EEPROM은 기존 방법에 비해 안전하게 값을 저장할 수 있기 때문에 사용하게 되었다.

EEPROM은 대표적인 룸(ROM - read only memory)의 한 종류로서, 전원을 차단해도 저장된 정보를 유지하는 비휘발성 메모리이다. EEPROM은 address를 가지고 있어서 각각의 address 안에 지정된 bytes의 값을 저장할 수 있다. 본 연구에서 사용된 MCU인 Teensy 3.2는 0에서 1023까지의 address지를 가지고 있고 하나의 address에 2048byte, 즉 0부터 255까지의 수를 저장할 수 있다. 모터포커서의 step을 저장하기 위해서는 약 100000범위의 수를 저장할 필요가 있다. GS-touch는 약 -50000에서 50000까지의 수를 저장할 수 있기 때문에 256진법을 활용하여 수를 저장할 수 있도록 설계하였다.

| | | | |
|-------------|------------|------------|----------|
| (Address=)0 | 1 | 2 | 3 |
| Step 자릿수 | Step 부호 | 256의 자리 | 1의 자리 |

Figure 14. EEPROM의 address별 사용 구조. 부호와 값을 절대치를 이용하여 연산하였다.

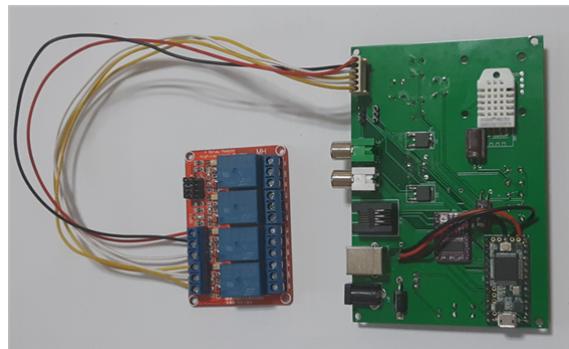


Figure 15. 실험에 사용된 릴레이스위치를 모터포커서에 연결시킨 모습

II.3.5 릴레이 스위치

릴레이 스위치는 전자적으로 이루어진 여러 스위치를 한번에 제어할 수 있도록 제작된 장비이다. 천체관측을 시작 및 종료할 때 가대, CCD, 카메라 등 한번에 여러 가지의 전원을 제어해야 하는데, 릴레이 스위치를 이용하면 한번에 제어할 수 있어 편리하다.

Figure 15와 같이 각각의 스위치를 선으로 연결시켜 mcu의 각 핀에 연결시켜 작동시킬 수 있으며, 함수로 지정하여 이들을 한번에 제어할 수 있도록 하였다.

릴레이 스위치는 6개의 핀으로 이루어져 있으며, 순서대로 +5v, GND, 1,2,3,4의 Relay이다. 때문에 각 핀들을 모두 연결시킨 뒤에 MCU를 이용하여 적절하게 제어할 수 있다. 각각의 기호는 P, Q, R, S이며, 상태만을 나타내기 때문에 서보모터 제어와 마찬가지로 오직 0과 1만을 입력받는다.

II.4 코딩

II.4.1 아두이노 코딩

작성중

II.4.2 ASCOM 드라이버 및 제어 프로그램

작성중

III. 연구 결과

III.1 하드웨어 완성-구동부

제작한 모터포커서는 제어키와 숫자로 이루어져 있는 기호를 통해 통신을 한다. 예를 들어, a에 모터 시계방향 회전을 할당하고 a:100이라는 기호를 입력하면 모터는 시계방향으로 100step만큼 이동하게 되는 것이다. 보강한 모터포커서 또한 이러한 방법을 응용하였으며, 어떤 방법으로 제어에 성공하였는지 서술한다.

이 완성품이 어떻게 동작하는지 (박기현샘W)

III.1.1 열선 제어

앞서 설명하였듯 열선의 제어는 전압의 PWM을 이용하여 실행된다. Figure 16와 같이シリ얼 포트를 통한 입력을 할 때 필요한 기호는 A와 D이며, 0 100의 값을 입력받아 500ms 주기로 값을 변화시킬 수 있도록 설계하였다.

```
if(delayMillis > 0 && delayMillis <500)
{
    //Serial.println(delayMillis);
    currentMillis = millis();
    if(currentMillis > previousMillis + delayMillis)
    {
        previousMillis = currentMillis;
        currentState = !currentState;
        delayMillis = 500-delayMillis;
    }
    if(currentState == true) {digitalWrite(PWMPin1,HIGH); digitalWrite(LED,HIGH);}
    else {digitalWrite(PWMPin1,LOW); digitalWrite(LED,LOW);}
}
//test OK
```

Figure 16. PWM제어를 위한 코드

III.1.2 EEPROM(Electrically Erasable Read-Only Memory)

모터포커서의 원활한 사용을 위해 저장되어야 하는 값은 모터의 위치를 저장할 수 있는 Position 값이다. Figure17은 값을 EEPROM에서 읽어오는 과정을 나타낸 것으로, 모터포커서

```

void eepRead()
{
    int N = EEPROM.read(0); // 0은 자릿수 저장
    int dir = EEPROM.read(1); // 1은 부호저장 (0이상이면 1, 아니면 0)
    int v = 0;

    if(N==0) v = EEPROM.read(2);
    else v = EEPROM.read(2)*256+EEPROM.read(3);
    if(!dir) v *= (-1); //dir이 0이면 음수이므로 -1을 곱한다

    EEPcurrentPosition = v;
    //Serial.println(EEPcurrentPosition);
}

```

Figure 17. EEPROM에서 정보를 읽어오는 과정

가 최초로 실행될 때 EEPROM에서 값을 불러오고, 모터를 움직여 값을 변화시킨 직후에 적용된 값을 EEPROM으로 입력시키면 EEPROM값과 모터포커서의 Position 값을 항상 동기화시킬 수 있다.

III.1.3 서보모터 제어

일반적인 서보모터 또한 PWM을 응용하여 제어할 수 있으나, 대부분의 라이브러리에서 이미 그 값을 적용시켜놓은 최적화 함수가 존재한다. 이를 활용하여 서보모터를 원하는 각도로 움직일 수 있도록 펌웨어 상에 코드를 제작하였다. 특히나 OLED를 활용하여 덮개에 부착된 마스크를 정확하게 제어할 수 있도록 하였다.

시리얼 포트를 통한 입력을 할 때 필요한 기호는 N이며, 오직 0과 1만을 입력받아 각각의 상태로 서보모터를 제어한다.

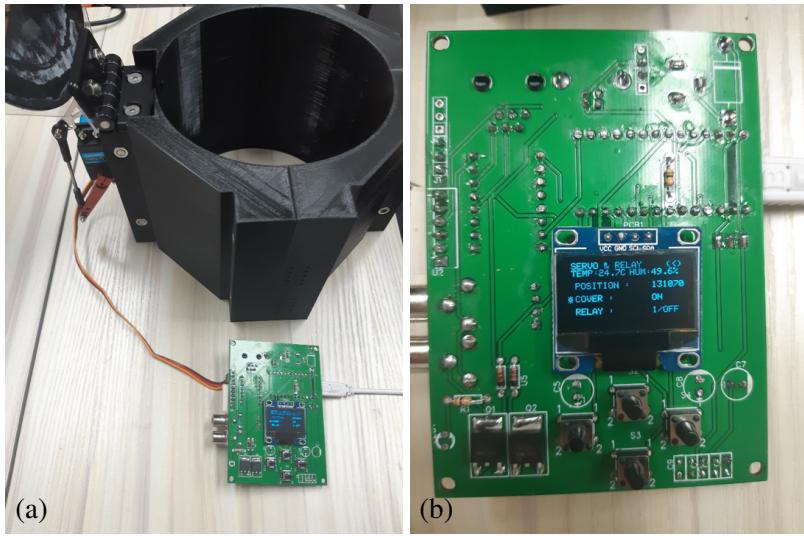


Figure 18. (a) 개선된 모터포커서를 이용하여 제어한 덮개 (b) 이 때 개선된 모터포커서에 출력되는 마스크의 상태. 화면 아래의 버튼들을 이용해 상태를 제어할 수 있으며, 제어상태를 확인할 수 있다.

III.2 소프트웨어 캡쳐해서 추가

기존의 모터포커서의 여러 기능들이 추가됨에 따라, 기존에 만들었던 ASCOM 호환 드라이버 또한 업데이트를 할 필요성이 있었다. 때문에 여러가지 기능들을 추가하여 기존의 GUI를 발전시켰으며, Fig19은 발전시킨 ASCOM 드라이버의 GUI의 모습을 나타내었다.

기존의 모터포커서에서 추가된 기능인 덮개 제어기능, 열선 제어기능, Relay Control을 포함하고 있으며 각각의 Relay가 제어하는 기능들을 적었다.

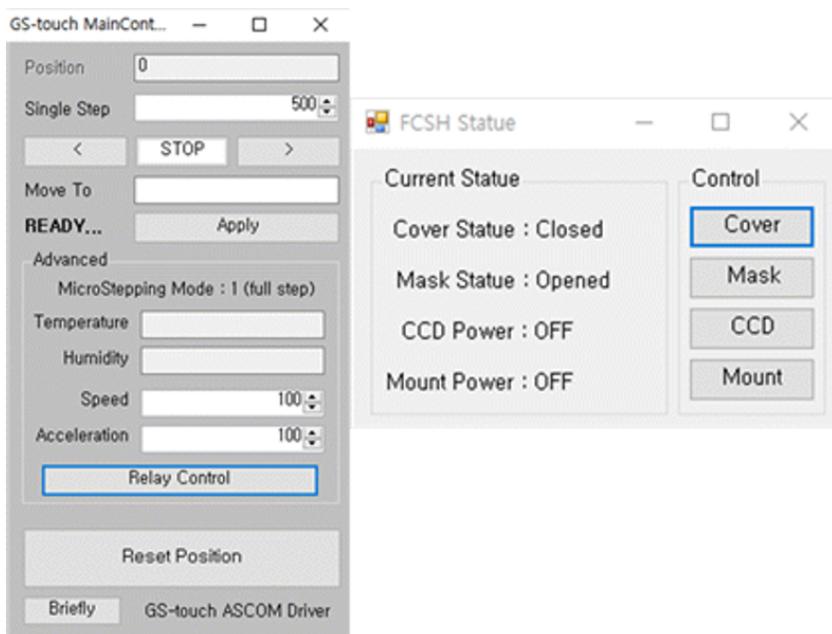


Figure 19. 개선된 모터포커서 드라이버의 GUI 모습

IV. 결론

본 연구에서는 소형 천체망원경의 자동화를 위한 GS-system을 개발하였다.

관측 시스템원격 천체관측을 편리하게 하기 위해 필요한 여러 시스템들을 기존에 개발된 모터포커서를 이용하여 개발하여 원격 천체관측에 도움이 되는 것을 목적으로 하였으며, 이를 위한 바흐티노프 마스크 제어용 망원경 덮개 및 시험 동작을 시행하였다. 본 제어시스템을 개발하기 위해 사용한 툴은 EasyEDA, Fusion 360, Arduino IDE, Visual Studio 2017, ASCOM flatform 등을 포함하며 이들의 효율적인 사용을 위한 여러 라이브러리가 사용되었다.

본 연구의 결과물을 요약하면 다음과 같다.

첫째, 3D프린터와 망원경의 경통 구조를 적절히 활용하여 바흐티노프 마스크의 제어가 가능한 덮개를 개발하였다. 덮개의 크기 때문에 일반적인 3D프린터로 출력하기 힘들어 네 부분으로 나누어 출력하였으며, 경첩을 이용하여 마스크를 제어하기 때문에 경첩이 들어갈 부분을 만들어주었다. 경첩은 같은 방향에 부착된 서보모터를 이용하여 제어되며, 서보모터는 항상 같은 각도로 움직일 수 있기 때문에 경첩과 마스크를 동작시킬때의 안정성이 매우 뛰어나다.

둘째, 천체망원경의 원격 제어 및 덮개의 원활한 동작을 위해 모터포커서를 개선하였다. 개선한 부분은 크게 네 가지로. 열선의 제어, EEPROM 적용, 덮개의 제어를 위한 서보모터 제어, 전원 동작의 편리함을 더하는 릴레이스위치의 제어로 나눌 수 있다. 또한 새로 생긴 기능들을 적절히 활용할 수 있도록 포커서 컨트롤 소프트웨어의 기능들 또한 개선하였다.

현재 망원경을 사용하기 위해서는 미리 덮개를 열고 있어야 한다. 하지만 덮개를 열고 있는 상태에서 면지가 경통의 속으로 들어가게 되는 단점이 존재하게 되어 이는 원격 천체망원경의 제어에 큰 어려움이 된다. 하지만 이번 연구를 통해서 덮개를 미리 열고 있어야 하는 단점을 없애 면지가 최대한 안 들어오도록 할 수 있게 되었고, 덮개로 바흐티노프마스크를

제어할 수 있게 되어 포커서 컨트롤러를 만들어 초점을 맞출 때 v-curve를 그릴 필요 없이 초점을 맞출 수 있게 되었으며, 열선을 설치하여 렌즈에 이슬이 맺히는 것을 방지하여 좀 더 좋은 별상을 얻을 수 있게 될 것으로 예상된다.

A. 부록

A.1. 아두이노 코드

A.1.1. GS-system.ino

```
1 // https :// www.airspayce.com/mikem/arduino/AccelStepper/classAccelStepper .html#a608b2395b64ac15451d16d0371fe13ce
2
3 #include "Board.h"
4
5 // for menu control
6 short subm = 0;
7 short menu = 1;
8 short power = 1;
9 short rm = 0;
10
11 // microstepping
12 short stepmode = 1;
13
14 // controlling PWM (0~100)
15 int PWM_value = 0;
16 bool currentstate = false;
17 unsigned long currentMillis = 0;
18 unsigned long previousMillis = 0;
19 unsigned long delayMillis = 0;
20
21 // for EEPROM and PCMODE
22 bool PCMODE = false;
23 int EEPcurrentPosition = 0;
24
25 #include <AccelStepper.h>
26 #include <DHT.h>
27 #include <Servo.h>
28
29 // for Stepper motor control
30 AccelStepper stepper(motorInterfaceType, STEP, DIR);
31
32 // for the temperature and humidity sensor
33 #define DHTTYPE DHT22
34 DHT dht(DHT22_PIN,DHTTYPE);
35 int chkSensor;
36 String Temperature;
37 String Humidity;
38
39 // for Servo motor control
40 Servo servo;
41 bool BMask1 = false;
42 bool BMask2 = false;
43
44 // for Relay control
45 bool Relay1 = false;
46 bool Relay2 = false;
47 bool Relay3 = false;
48 bool Relay4 = false;
49
50 String inputString = "";
51
```

```

52 void setup() {
53
54     Serial .begin(115200);
55     Serial .println ("GStouch#");
56     dis_start ();
57 //   Servo_start ();
58
59     eepRead();
60     stepper .setCurrentPosition (EEPcurrentPosition );
61
62     stepper .setMaxSpeed(2000.0);
63     stepper .setAcceleration (300.0);
64     stepper .setSpeed(100);
65
66     inputString .reserve (200);
67
68     pinset ();
69 }
70
71
72 void loop() {
73
74     if( stepper .distanceToGo() == 0 || subm!=2)
75     {
76         Temperature = String (dht.readTemperature () ,1);
77         Humidity = String (dht.readHumidity(),1);
78         humidityTemperatureReport();
79         buttonRead();
80         draw();
81     }
82
83     if( delayMillis >0 && delayMillis <500)
84     {
85         // Serial .println (delayMillis );
86         currentMillis = millis ();
87         if( currentMillis > previousMillis + delayMillis )
88         {
89             previousMillis = currentMillis ;
90             currentstate = ! currentstate ;
91             delayMillis = 500-delayMillis;
92         }
93         if( currentstate == true ) { digitalWrite (PWMPin1,HIGH); digitalWrite(LED,HIGH);}
94         else { digitalWrite (PWMPin1,LOW); digitalWrite(LED,LOW);}
95 // test OK
96     }
97 }
98
99 void reportPosition () {
100    Serial .print ("POSITION:");
101    Serial .print ( stepper .currentPosition ());
102    Serial .println ("#");
103 }
104
105
106 /**
107 * process the command we received from the client
108 * command format is <Letter><Space><Integer>
109 * i.e. A 500 ---- Fast Rewind with 500 steps
110 */

```

```

111 void serialCommand(String commandString) {
112     char _command = commandString.charAt(0);
113     int _value = commandString.substring(2).toInt ();
114     String _answer = "";
115     int _currentPosition = stepper . currentPosition () ;
116     int _newPosition = _currentPosition ;
117
118
119
120     switch (_command) {
121
122         case 'A': // SET PWM
123             case 'a': _newPosition = _currentPosition ; // non move command
124                 // PWM test
125                 delayMillis = 5 * _value;
126                 previousMillis = millis ();
127                 currentMillis = millis ();
128                 currentstate = false ;
129                 if ( delayMillis != 0) currentstate = true ;
130                 if ( delayMillis == 500) { digitalWrite (PWMPin1,HIGH);digitalWrite(LED,HIGH);}
131                 break;
132
133         case 'B': // REVERSE "<"
134             case 'b': _newPosition = _currentPosition - _value;
135                 break;
136
137         case 'C': // FORWARD ">"
138             case 'c': _newPosition = _currentPosition + _value;
139                 break;
140
141         case 'E': // MOVE TO POSITION
142             case 'e': _newPosition = _value;
143                 break;
144
145         case 'F': // GET CURRENT POSITION
146             case 'f': _answer += _currentPosition ;
147                 break;
148
149         case 'G': // SET CURRENT POSITION
150             case 'g': _newPosition = _value;
151                 _currentPosition = _value;
152                 stepper . setCurrentPosition (_value);
153                 break;
154
155         case 'H': // SET ACCELERATION
156             case 'h': _newPosition = _currentPosition ; // non move command
157                 stepper . setAcceleration (_value);
158                 _answer += "SET-ACCELERATION:" ;
159                 _answer += _value;
160                 break;
161
162         case 'I': // SET SPEED
163             _newPosition = _currentPosition ; // non move command
164                 stepper . setSpeed(_value);
165                 _answer += "SET-SPEED:" ;
166                 _answer += _value;
167                 break;
168
169         case 'i': // GET SPEED

```

```

170 _newPosition = _currentPosition ; // non move command
171 _answer += "GET-SPEED:";
172 _answer += stepper.speed();
173 break;
174
175 case 'k': // GET TEMPERATURE & HUMIDITY
176 _newPosition = _currentPosition ; // non move command
177 humidityTemperatureReport();
178 break;
179
180 case 'N': // Mask Set
181 case 'n': _newPosition = _currentPosition ; // non move command
182 BMask1 = !BMask1;
183 servo.attach(ServoPin1); // 5도 동시제어
184 if (BMask1) servo.write(_value);
185 else if (!BMask1) servo.write(0);
186 delay(500);
187 servo.detach();
188 delay(500);
189
190 case 'O': // Mask Set
191 case 'o': _newPosition = _currentPosition ; // non move command
192 BMask2 = !BMask2;
193 servo.attach(ServoPin2); // 5도 동시제어
194 if (BMask2) servo.write(_value);
195 else if (!BMask2) servo.write(0);
196 delay(500);
197 servo.detach();
198 delay(500);
199
200 case 'P': // RelayPin1
201 case 'p': _newPosition = _currentPosition ;
202 Relay1 = !Relay1;
203 if (Relay1) digitalWrite(RelayPin1,HIGH);
204 else digitalWrite(RelayPin1,LOW);
205
206 case 'Q': // RelayPin2
207 case 'q': _newPosition = _currentPosition ;
208 Relay2 = !Relay2;
209 if (Relay2) digitalWrite(RelayPin2,HIGH);
210 else digitalWrite(RelayPin2,LOW);
211
212 case 'R': // RelayPin
213 case 'r': _newPosition = _currentPosition ;
214 Relay3 = !Relay3;
215 if (Relay3) digitalWrite(RelayPin3,HIGH);
216 else digitalWrite(RelayPin3,LOW);
217
218 case 'S': // RelayPin
219 case 's': _newPosition = _currentPosition ;
220 Relay4 = !Relay4;
221 if (Relay4) digitalWrite(RelayPin4,HIGH);
222 else digitalWrite(RelayPin4,LOW);
223
224 case 'X': // GET STATUS – may not be needed
225 case 'x':
226     stepper.stop();
227     break;
228 case 'Z': // IDENTIFY

```

```

229     case 'z': _answer += "GStouch";
230     PCMODE = true;
231     break;
232
233     default :
234     _answer += "GStouch";
235     break;
236 }
237
238 if (_ newPosition != _ currentPosition) {
239     // a move command was issued
240     Serial . print ("MOVING:");
241     Serial . print (_ newPosition);
242     Serial . println ("#");
243     //
244     stepper . moveTo(_ newPosition);
245     stepper . runToPosition ();
246     _ answer += "POSITION:";
247     _ answer += stepper . currentPosition ();
248 }
249
250 eepWrite(_ newPosition);
251 eepRead();
252 Serial . print (_ answer);
253 Serial . println ("#");
254 }
255
256 /**
257 * handler for the serial communications
258 * calls the SerialCommand whenever a new command is received
259 */
260 void serialEvent () {
261     while ( Serial . available ()) {
262         char inChar = (char) Serial . read ();
263         inputString += inChar;
264         if (inChar == '\n') {
265             serialCommand(inputString);
266             inputString = "";
267         }
268     }
269 }
270
271 /**
272 * for DHT routine
273 */
274 void humidityTemperatureReport() {
275     chkSensor = digitalRead (DHT22_PIN);
276     Temperature = String (dht . readTemperature () ,1);
277     Humidity = String (dht . readHumidity () ,1);
278     switch (chkSensor) {
279     case 1:
280         Serial . print ("TEMPERATURE:");
281         Serial . print (Temperature);
282         Serial . println ("#");
283         delay(50);
284         Serial . print ("HUMIDITY:");
285         Serial . print (Humidity);
286         Serial . println ("#");
287         delay(50);

```

```

288     break;
289 case 0:
290     Serial . print ("TEMPERATURE:");
291     Serial . print ("CHECKSUMERROR");
292     Serial . println ("#");
293     Serial . print ("HUMIDITY:");
294     Serial . print ("CHECKSUMERROR");
295     Serial . println ("#");
296     break;
297 default :
298     Serial . print ("TEMPERATURE:");
299     Serial . print ("UNKNOWNERROR");
300     Serial . println ("#");
301     Serial . print ("HUMIDITY:");
302     Serial . print ("UNKNOWNERROR");
303     Serial . println ("#");
304     break;
305 }
306 }
307
308 void Servo_start ()
309 {
310     servo . attach (ServoPin1); // 5 도    동시제어
311     servo . write (0);
312     delay (500);
313     servo . detach ();
314 }
```

A.1.2. Board.h

```

1 #ifndef Board_h
2 #define Board_h
3
4 // #define Teensy
5 #ifndef Teensy
6 #define Arduino_GStouch 1
7 #endif
8
9
10 #ifdef Teensy
11 #endif
12
13 #ifdef Arduino_GStouch
14
15 #define motorInterfaceType 1
16 #define DIR 0
17 #define STEP 13
18 #define MS0 15
19 #define MS1 16
20 #define MS2 17
21 #define MOTOR_STEPS 200
22 #define DHT22_PIN 2
23 #define UPpin 12
24 #define DOWNpin 9
25 #define RIGHTpin 10
26 #define LEFTpin 11
27 #define ServoPin1 3
28 #define ServoPin2 5
29 #define LED 13
```

```

30 #define PWMPin1 20
31 #define PWMPin2 22
32 #define MAX_POSITION 65535
33
34 // need to set
35
36 #define RelayPin1 7
37 #define RelayPin2 21
38 #define RelayPin3 8
39 #define RelayPin4 23
40
41 #endif
42
43 #endif

```

A.1.3. ButtonControl.ino

```

1
2
3 // about button switch
4 short S[4]={0}; short now[4]={1}; short was[4]={0};
5 //
6 int _resetPosition = 0;
7
8 // 어떻게 입력이 되었나 ?
9 void controljudge ()
10 {
11     now[0]=digitalRead(UPpin);
12     now[1]=digitalRead(DOWNpin);
13     now[2]=digitalRead(RIGHTpin);
14     now[3]=digitalRead(LEFTpin);
15     for( int i=0;i<4;i++)
16     {
17         if (now[i]!=was[i]&&now[i]==0) S[i]=1;
18         else S[i]=0;
19         was[i]=now[i];
20     }
21 }
22
23 // 설정된 핀번호 풀업저항으로 선언
24 void pinset ()
25 {
26     pinMode(UPpin,INPUT_PULLUP);
27     pinMode(DOWNpin,INPUT_PULLUP);
28     pinMode(RIGHTpin,INPUT_PULLUP);
29     pinMode(LEFTpin,INPUT_PULLUP);
30
31     pinMode(PWMPin1,OUTPUT);
32     pinMode(LED,OUTPUT);
33     digitalWrite (LED,LOW);
34     digitalWrite (PWMPin1,LOW);
35
36     pinMode(PWMPin2,OUTPUT);
37     digitalWrite (PWMPin2,LOW);
38
39     pinMode(RelayPin1,OUTPUT);
40     pinMode(RelayPin2,OUTPUT);
41     pinMode(RelayPin3,OUTPUT);
42     pinMode(RelayPin4,OUTPUT);

```

```

43     digitalWrite (RelayPin1,LOW);
44     digitalWrite (RelayPin2,LOW);
45     digitalWrite (RelayPin3,LOW);
46     digitalWrite (RelayPin4,LOW);
47 }
48
49 void buttonRead()
50 {
51     controljudge ();
52     if (subm < 2)
53     {
54         if (S[2]) subm++;
55         else if (S[3]) subm--;
56
57         else if (S[0]&&menu>1) menu--;
58         else if (S[1]&&menu==2&&subm==1) menu++;
59         else if (S[1]&&menu<2) menu++;
60     }
61     else
62     {
63         switch(menu)
64     {
65         case 1: // POSITION CONTROL
66             motorControl();
67             break;
68
69         case 2: // STEPMODE SET & RESET POSITION
70             MSmodeControl();
71             break;
72
73         case 3: // SERVO & RELAY
74             Servo_Relay_Control();
75     }
76 }
77 }
```

A.1.4. displayAdafruit.ino

```

1 // https :// github .com/olikraus /u8glib/wiki/ fontsize
2 // https :// github .com/olikraus /u8glib/wiki/ userreference # setprintpos
3
4
5 #include <Wire.h>
6 #include <Adafruit_GFX.h>
7 #include <Adafruit_SSD1306.h>
8 #include <Fonts/FreeSansBold6pt7b.h>
9 #define OLED_RESET 4
10 Adafruit_SSD1306 display(OLED_RESET);
11 #define LOGO16_GLCD_HEIGHT 16
12 #define LOGO16_GLCD_WIDTH 16
13 #define XPOS 0
14 #define YPOS 1
15 #define DELTAY 2
16
17
18 // #include<U8Glib.h>
19 // U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0); // I2C / TWI
20
21
```

```

22 void dis_start ()
23 {
24     display.begin(SSD1306_SWITCHCAPVCC);
25     display.display();
26     delay(1000);
27     display.clearDisplay();
28     display.setTextColor(WHITE);
29     display.setTextSize(1);
30 }
31
32 void draw()
33 {
34
35     display.clearDisplay();
36
37     Serial.print(subm);
38     Serial.print(menu);
39     Serial.println(rm);
40
41     display.setCursor(4,8);
42     display.print("TEMP:");
43     display.print(Temperature);
44     display.print("C");
45
46     display.setCursor(68,8);
47     display.print("HUM:");
48     display.print(Humidity);
49     display.print("%");
50
51     display.setCursor(8,22);
52     display.print("POSITION:_____");
53     display.print(stepper.currentPosition());
54
55     switch(subm)
56     {
57         case 0:
58             display.setCursor(4,0);
59             display.print("STATUS");
60             display.setCursor(8,36);
61             display.print("MSMODE:_____");
62             display.print(stepmode);
63             display.setCursor(8,50);
64             if (PCMODE) display.print("PCMODE:_____ON");
65             else display.print("PCMODE:_____OFF");
66             display.setCursor(108,0);
67             display.print(">");
68             break;
69
70         case 1:
71             //u8g.setFont(u8g_font_unifont); u8g.setFontPosTop();
72             display.setCursor(4,0);
73             display.print("MENU");
74             display.setCursor(8,36);
75             display.print("MOTOR_CONTROL");
76             display.setCursor(8,46);
77             display.print("MOTOR_SETTING");
78             display.setCursor(8,56);
79             display.print("SERVO_&_RELAY");
80

```

```

81     display . setCursor (102,0) ;
82     display . print ("(<>)") ;
83     display . setCursor(0,26+10*menu);
84     display . print (">");
85     break;
86
87     default :
88         switch(menu)
89     {
90         case 1:
91             //u8g.setFont ( u8g_font_unifont ); u8g.setFontPosTop();
92             display . setCursor (4,0) ;
93             display . print ("POSITION_CONTROL");
94             display . setCursor (8,36) ;
95             display . print ("POWER:");
96             display . setCursor (92,36) ;
97             display . print (power);
98             display . setCursor (8,50) ;
99             display . print ("MICROSTEPPING:");
100            display . setCursor (92,50) ;
101            display . print (stepmode);
102            display . setCursor (108,0) ;
103            display . print ("(<)");
104            display . setCursor(0,22+rm*14);
105            if (subm==2)display . print (">");
106            else display . write (15);
107            break;
108
109        case 2:
110            //u8g.setFont ( u8g_font_unifont ); u8g.setFontPosTop();
111            display . setCursor (4,0) ;
112            display . print ("STEPMODE_CHANGE");
113            //u8g.setFont(u8g_font_5x8);
114            display . setCursor (8,36) ;
115            display . print ("STEPMODE:~~~~");
116            display . print (stepmode);
117            display . setCursor (8,46) ;
118            display . print ("POSITION_TO:~");
119            display . print ( _resetPosition );
120            display . setCursor (8,56) ;
121            display . print ("APPLY_SETTINGS");
122            display . setCursor (108,0) ;
123            display . print ("(<)");
124            display . setCursor(0,36+rm*10);
125            if (subm==2)display . print (">");
126            else display . write (15);
127            break;
128
129        case 3:
130            //u8g.setFont ( u8g_font_unifont ); u8g.setFontPosTop();
131            display . setCursor (4,0) ;
132            display . print ("SERVO_&_RELAY");
133            //u8g.setFont(u8g_font_5x8);
134            display . setCursor (8,36) ;
135            display . print ("COVER:~~~~~");
136            if (BMask1) display . print ("ON");
137            else display . print ("OFF");
138            display . setCursor (8,50) ;
139

```

```

140     display . print ("RELAY_");
141     if (!rm) display . print ("1");
142     else display . print (rm);
143     display . print ("/");
144     switch(rm)
145     {
146         case 0:
147             if (Relay1) display . print ("ON");
148             else display . print ("OFF");
149             break;
150         case 1:
151             if (Relay1) display . print ("ON");
152             else display . print ("OFF");
153             break;
154
155         case 2:
156             if (Relay2) display . print ("ON");
157             else display . print ("OFF");
158             break;
159         case 3:
160             if (Relay3) display . print ("ON");
161             else display . print ("OFF");
162             break;
163         case 4:
164             if (Relay4) display . print ("ON");
165             else display . print ("OFF");
166             break;
167     }
168
169
170     display . setCursor (8,64);
171     display . print ("APPLY_SETTNGS(");
172     display . write (15);
173     display . print (")");
174     display . setCursor (108,0);
175     display . print ("(<)");
176     if (rm) display . setCursor (0,50);
177     else display . setCursor (0,36);
178     if (subm==2)display . print (">");
179     else display . write (15);
180     break;
181
182
183     }
184
185     break;
186 }
187 display . display () ;
188 }
```

A.1.5. EEPROM.ino

```

1 #include <EEPROM.h> // 부호고려
2
3 void eepRead()
4 {
5     int bias = 65535;
6     int N = EEPROM.read(0); // 0은 자릿수 저장
7     int dir = EEPROM.read(1); // 1은 부호저장 (0 이상이면 1, 아니면 0)
```

```

8 int v = 0;
9
10 if (N==0) v = EEPROM.read(2);
11 else v = EEPROM.read(2)*256+EEPROM.read(3);
12 if (! dir) v *= (-1); // dir이 0 이면 음수이므로 -1을 곱한다
13 v = v + bias;
14 EEPROM currentPosition = v;
15 // Serial . println (EEP currentPosition );
16 }
17
18 void eepWrite(int value_ )
19 {
20     int bias = 65535;
21     value_ = value_ - bias;
22     int v = value_ ;
23     if (v>=0) EEPROM.write(1,1);
24     else {EEPROM.write(1,0); v=v*(-1);}
25
26     if (v>255)
27     {
28         EEPROM.write(0,1);
29         EEPROM.write(2,v/256);
30         EEPROM.write(3,v%256);
31     }
32     else
33     {
34         EEPROM.write(0,0);
35         EEPROM.write(2,v);
36         EEPROM.write(3,0);
37     }
38 }
39 }
```

A.1.6. MainControl.ino

```

1
2 int ct=0;
3 int pt=0;
4 void motorControl()
5 {
6     int stp = 0;
7     switch(power)
8     {
9         case 1: stp=1; break;
10        case 2: stp=10; break;
11        case 3: stp=100; break;
12        case 4: stp=300; break;
13        case 5: stp=1000; break;
14    }
15
16    switch(subm)
17    {
18        case 2:
19            if (S[0] && rm) rm--;
20            else if(S[1] && !rm) rm++;
21            else if(S[2]) subm++;
22            else if(S[3]) subm--;
23        break;
24    }
```

```

25     case 3:
26         switch(rm)
27     {
28         case 0: // POSITION
29
30             if (!now[0]||! now[1])
31             {
32                 int _nextPosition = stepper . currentPosition ();
33                 if (!now[0]) _nextPosition += stp;
34                 else _nextPosition -= stp;
35                 if (_nextPosition != stepper . currentPosition ())
36                 {
37                     stepper .moveTo(_nextPosition);
38                     stepper .runToPosition ();
39                 }
40             }
41             else if (S[3]) subm--;
42         break;
43
44         case 1: // POWER
45             if (S[0] && power<5) power++;
46             else if (S[1] && power>1) power--;
47             else if (S[3])
48             {
49                 subm--;
50                 switch(power)
51                 {
52                     case 1: stp=1; break;
53                     case 2: stp=10; break;
54                     case 3: stp=100; break;
55                     case 4: stp=300; break;
56                     case 5: stp=1000; break;
57                 }
58             }
59             stepper .setSpeed(300*stp);
60         break;
61     }
62     break;
63 }
64 }
65
66 void MSmodeControl()
67 {
68     switch(subm)
69     {
70         case 2:
71             if (S[0] && rm--) rm--;
72             else if (S[1] && rm != 2) rm++;
73             else if (S[2]) subm++;
74             else if (S[3]) subm--;
75         break;
76
77         case 3:
78             switch(rm)
79             {
80                 case 0:
81                     if (S[0] && stepmode<4) stepmode++;
82                     else if (S[1] && stepmode>1) stepmode--;
83                     else if (S[3]) subm--;

```

```

84     break;
85
86 case 1:
87     if (S[0] && _resetPosition < 500000 && rm==1) _resetPosition = ( _resetPosition +100)/100*100;
88     else if (S[1] && _resetPosition > 0 && rm==1) _resetPosition = ( _resetPosition -100)/100*100;
89     else if (S[3]) subm--;
90     break;
91
92 case 2:
93     if (S[2])
94     {
95         rm=0; subm=0;
96         stepper . setCurrentPosition ( _resetPosition ); // reset position setting
97         _resetPosition = 0;
98
99         setstep (); // microstepping setting
100    }
101    else if (S[3]) subm--;
102   }
103 }
104
105 }
106
107 // Main_Control로 이동
108 void Servo_Relay_Control()
109 {
110     switch(subm)
111     {
112     case 2:
113         if (S[0] && rm--) rm--;
114         else if (S[1] && rm!=4) rm++;
115         else if (S[2]) subm++;
116         else if (S[3]) subm--;
117         break;
118
119     case 3:
120         switch(rm)
121         {
122             case 0:
123                 if (S[0])
124                 {
125                     BMask1 = true;
126                     servo . attach (ServoPin1); // 5도 동시제어
127                     servo . write (150);
128                     delay(500);
129                     servo . detach ();
130                     delay(500);
131                 }
132                 else if (S[1])
133                 {
134                     BMask1 = false;
135                     servo . attach (ServoPin1); // 5도 동시제어
136                     servo . write (30);
137                     delay(500);
138                     servo . detach ();
139                     delay(500);
140                 }
141                 else if (S[3]) subm--;
142             break;

```

```

143
144     case 1:
145         if(S[0] && !Relay1)
146             {Relay1 = !Relay1; digitalWrite (RelayPin1,HIGH);}
147         else if(S[1] && Relay1)
148             {Relay1 = !Relay1; digitalWrite (RelayPin1,LOW);}
149         else if(S[3]) subm--;
150     break;
151
152     case 2:
153         if(S[0] && !Relay2)
154             {Relay2 = !Relay2; digitalWrite (RelayPin2,HIGH);}
155         else if(S[1] && Relay2)
156             {Relay2 = !Relay2; digitalWrite (RelayPin2,LOW);}
157         else if(S[3]) subm--;
158     break;
159
160     case 3:
161         if(S[0] && !Relay3)
162             {Relay3 = !Relay3; digitalWrite (RelayPin3,HIGH);}
163         else if(S[1] && Relay3)
164             {Relay3 = !Relay3; digitalWrite (RelayPin3,LOW);}
165         else if(S[3]) subm--;
166     break;
167
168     case 4:
169         if(S[0] && !Relay4)
170             {Relay4 = !Relay4; digitalWrite (RelayPin4,HIGH);}
171         else if(S[1] && Relay4)
172             {Relay4 = !Relay4; digitalWrite (RelayPin4,LOW);}
173         else if(S[3]) subm--;
174     break;
175 }
176 }
177
178 }
179 }
```

A.1.7. Setstep.ino

```

1 void setstep ()
2 {
3     switch(stepmode)
4     {
5         case 1:
6             digitalWrite (MS0,LOW);
7             digitalWrite (MS1,LOW);
8             digitalWrite (MS2,LOW);
9             break;
10
11        case 2:
12            digitalWrite (MS0,HIGH);
13            digitalWrite (MS1,LOW);
14            digitalWrite (MS2,LOW);
15            break;
16
17        case 3:
18            digitalWrite (MS0,LOW);
19            digitalWrite (MS1,HIGH);
```

```

20     digitalWrite (MS2,LOW);
21     break;
22
23     case 4:
24     digitalWrite (MS0,HIGH);
25     digitalWrite (MS1,HIGH);
26     digitalWrite (MS2,LOW);
27     break;
28
29     default :
30     Serial . println ("Microstepping_mode_should_be_from_1_to_4");
31   }
32   Serial . print ("Microstepping_mode:"); Serial . print (stepmode);
33 }
```

A.2. ASCOM driver 코드

A.2.1. driver.cs

```

1 // tabs=4
2 //
-----  

3 // TODO fill in this information for your driver, then remove this line !
4 //
5 // ASCOM Focuser driver for GS_touch
6 //
7 // Description : Lorem ipsum dolor sit amet, consetetur sadipscing elitr , sed diam
8 // nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam
9 // erat, sed diam voluptua. At vero eos et accusam et justo duo
10 // dolores et ea rebum. Stet clita kasd gubergren, no sea takimata
11 // sanctus est Lorem ipsum dolor sit amet.
12 //
13 // Implements: ASCOM Focuser interface version: <To be completed by driver developer>
14 // Author: (XXX) Your N. Here <your@email.here>
15 //
16 // Edit Log:
17 //
18 // Date      Who Vers  Description
19 // -----  -----
20 // dd-mmm-yyyy XXX 6.0.0 Initial edit , created from ASCOM driver template
21 //
-----  

22 //
23
24
25 // This is used to define code in the template that is specific to one class implementation
26 // unused code canbe deleted and this definition removed.
27 #define Focuser
28
29
30 using System;
31 using System.Collections.Generic;
32 using System.Diagnostics;
33 using System.Linq;
34 using System.Text;
```

```

35
36 using System.Windows.Forms;
37 using System.Runtime.InteropServices ;
38
39 using ASCOM;
40 using ASCOM.Astrometry;
41 using ASCOM.Astrometry.AstroUtils;
42 using ASCOM.Utilities;
43 using ASCOM.DeviceInterface;
44 using System.Globalization ;
45 using System.Collections ;
46 using System.IO.Ports ;
47
48 namespace ASCOM.GS_touch
49 {
50     #region Event Handling Libraries
51
52     public class FocuserHumidityChangedEventArgs : EventArgs
53     {
54         public readonly string Humidity;
55
56         public FocuserHumidityChangedEventArgs(string humidity)
57         {
58             this.Humidity = humidity;
59         }
60     }
61
62
63     public class FocuserStateChangedEventArgs : EventArgs
64     {
65         public readonly bool IsMoving;
66
67         public FocuserStateChangedEventArgs(bool isMoving)
68         {
69             this.IsMoving = isMoving;
70         }
71     }
72
73     public class FocuserTemperatureChangedEventArgs : EventArgs
74     {
75         public readonly string Temperature;
76
77         public FocuserTemperatureChangedEventArgs(string temperature )
78         {
79             this.Temperature = temperature ;
80         }
81     }
82
83     public class FocuserValueChangedEventArgs : EventArgs
84     {
85         public readonly int LastValue;
86         public readonly int NewValue;
87
88         public FocuserValueChangedEventArgs(int LastValue , int NewValue)
89         {
90             this.LastValue = LastValue;
91             this.NewValue = NewValue;
92         }
93     }

```

```

94     #endregion
95
96     // Your driver's DeviceID is ASCOM.GS_touch.Focuser
97
98     // The Guid attribute sets the CLSID for ASCOM.GS_touch.Focuser
99     // The ClassInterface /None attribute prevents an empty interface called
100    // _GS_touch from being created and used as the [ default ] interface
101
102   // TODO Replace the not implemented exceptions with code to implement the function or
103   // throw the appropriate ASCOM exception.
104
105
106  /// <summary>
107  /// ASCOM Focuser Driver for GS_touch.
108  /// </summary>
109  [Guid("92161BAA-FB3B-4A67-A120-0948F98AB1CB")]
110  [ ClassInterface ( ClassInterfaceType .None)]
111  public class Focuser : IFocuserV3
112 {
113
114     /// <summary>
115     /// ASCOM DeviceID (COM ProgID) for this driver.
116     /// The DeviceID is used by ASCOM applications to load the driver at runtime.
117
118     internal static string driverID = "ASCOM.GS_touch.Focuser";
119
120     // TODO Change the descriptive string for your driver then remove this line
121
122     /// <summary>
123     /// Driver description that displays in the ASCOM Chooser.
124
125     private static string driverDescription = "ASCOM_Focuser_Driver_for_GS-touch";
126
127
128     internal static string comPortProfileName = "COM_Port"; // Constants used for Profile persistence
129     internal static string comPortDefault = "COM4";
130     internal static string traceStateProfileName = "Trace_Level";
131     internal static string traceStateDefault = "false";
132
133     internal static string comPort; // Variables to hold the current device configuration
134
135     internal static string showUIProfileName = "Show_Controller";
136     internal static string showUIDefault = "true";
137     internal static string stepperMotor = "stepper";
138     internal static bool traceState;
139     internal static bool showUI;
140
141     internal static string motorDriver;
142
143     private bool connectedState;
144     private Util utilities;
145     private AstroUtils astroUtilities;
146     internal static TraceLogger tl;
147
148     private SerialPort serialPort;
149     private bool isMoving = false;
150     private double temperature;
151     private double humidity;
152     private string message;
153     private string existingMessage;

```

```

153
154     private int focuserPosition = 0; // Class level variable to hold the current focuser position
155     private const int focuserSteps = 10000;
156     private double stepSize = 500;
157     private int maxIncrement = 1000;
158
159     /// <summary>
160     /// Initializes a new instance of the <see cref="GS_touch"/> class.
161     /// Must be public for COM registration.
162     /// </summary>
163     public Focuser()
164     {
165         ReadProfile();
166         tl = new TraceLogger("", "GS_touch");
167         tl.Enabled = traceState;
168         tl.LogMessage("Focuser", "Starting „initialisation „");
169
170         connectedState = false; // Initialise connected to false
171         utilities = new Util(); // Initialise util object
172         astroUtilities = new AstroUtils(); // Initialise astro utilities object
173         //TODO: Implement your additional construction here
174
175         tl.LogMessage("Focuser", "Completed „initialisation „");
176     }
177
178     #region Event Handling
179     #region Event Handling Declaration + Mainwindow Declaration
180     private MainWindow mainWindow;
181     public event EventHandler<FocuserValueChangedEventArgs> FocuserValueChanged;
182     public event EventHandler<FocuserStateChangedEventArgs> FocuserStateChanged;
183     public event EventHandler<FocuserTemperatureChangedEventArgs> FocuserTemperatureChanged;
184     public event EventHandler<FocuserHumidityChangedEventArgs> FocuserHumidityChanged;
185     #endregion //need
186     #region Event Handling Functions
187     public virtual void OnFocuserValueChanged(FocuserValueChangedEventArgs e)
188     {
189         if (FocuserValueChanged != null)
190         {
191             FocuserValueChanged(this, e);
192         }
193     }
194
195     public virtual void OnFocuserStateChanged(FocuserStateChangedEventArgs e)
196     {
197         if (FocuserStateChanged != null)
198         {
199             FocuserStateChanged(this, e);
200         }
201     }
202
203     public virtual void OnFocuserTemperatureChanged(FocuserTemperatureChangedEventArgs e)
204     {
205         if (FocuserTemperatureChanged != null)
206         {
207             FocuserTemperatureChanged(this, e);
208         }
209     }
210
211     public virtual void OnFocuserHumidityChanged(FocuserHumidityChangedEventArgs e)

```

```

212     {
213         if (FocuserHumidityChanged != null)
214         {
215             FocuserHumidityChanged(this, e);
216         }
217     }
218
219 #endregion
220 #endregion
221
222 #region Common properties and methods.
223
224 // <summary>
225 // Displays the Setup Dialog form.
226 // If the user clicks the OK button to dismiss the form, then
227 // the new settings are saved, otherwise the old values are reloaded.
228 // THIS IS THE ONLY PLACE WHERE SHOWING USER INTERFACE IS ALLOWED!
229 // </summary>
230 public void SetupDialog()
231 {
232     // consider only showing the setup dialog if not connected
233     // or call a different dialog if connected
234     if (IsConnected)
235         System.Windows.MessageBox.Show("Already_connected,_just_press_OK");
236
237     using (SetupDialogForm F = new SetupDialogForm())
238     {
239         var result = F.ShowDialog();
240         if (result == System.Windows.Forms.DialogResult.OK)
241         {
242             WriteProfile(); // Persist device configuration values to the ASCOM Profile store
243         }
244     }
245 }
246
247 public ArrayList SupportedActions
248 {
249     get
250     {
251         tl.LogMessage("SupportedActions_Get", "Returning_empty_arraylist");
252         return new ArrayList();
253     }
254 }
255
256 public string Action( string actionName, string actionParameters )
257 {
258     CheckConnected("Action");
259     if (IsConnected)
260     {
261         serialPort.WriteLine(actionName + ":" + actionParameters );
262     }
263     return "";
264 }
265
266 public void CommandBlind(string command, bool raw)
267 {
268     CheckConnected("CommandBlind");
269     // Call CommandString and return as soon as it finishes
270     this.CommandString(command, raw);

```

```

271         // or
272         throw new ASCOM.MethodNotImplementedException("CommandBlind");
273         // DO NOT have both these sections ! One or the other
274     }
275
276     public bool CommandBool(string command, bool raw)
277     {
278         CheckConnected("CommandBool");
279         string ret = CommandString(command, raw);
280         // TODO decode the return string and return true or false
281         // or
282         throw new ASCOM.MethodNotImplementedException("CommandBool");
283         // DO NOT have both these sections ! One or the other
284     }
285
286     public string CommandString(string command, bool raw)
287     {
288         CheckConnected("CommandString");
289         // it's a good idea to put all the low level communication with the device here,
290         // then all communication calls this function
291         // you need something to ensure that only one command is in progress at a time
292
293         if (IsConnected)
294         {
295             serialPort.WriteLine(command + ":" + stepSize );
296             //System.Threading.Thread.Sleep(10);
297         }
298
299         string message = "sent_" + command + ":" + stepSize ;
300
301         tl.LogMessage("command_", message);
302
303         System.Diagnostics.Debug.WriteLine("messaaage_from_arduino_" + message);
304         return message;
305     }
306
307     public void Dispose()
308     {
309         // Clean up the tracelogger and util objects
310         tl.Enabled = false ;
311         tl.Dispose();
312         tl = null ;
313         utilities .Dispose();
314         utilities = null ;
315         astroUtilities .Dispose();
316         astroUtilities = null ;
317     }
318
319     public bool Connected
320     {
321         get
322         {
323             tl.LogMessage("Connected_Get", IsConnected.ToString());
324             return IsConnected;
325         }
326         set
327         {
328             tl.LogMessage("Connected_Set", value.ToString ());
329         }

```

```

330     ReadProfile();
331
332     if (value == IsConnected)
333     {
334         if (!mainWindow.Visible && showUI)
335         {
336             mainWindow = new MainWindow(this);
337             mainWindow.Show();
338         }
339     }
340
341     if (value)
342     {
343         connectedState = true;
344         tl.LogMessage("Connected_Set", "Connecting_to_port_" + comPort);
345         // TODO connect to the device
346         try
347         {
348             serialPort = new SerialPort(comPort, 115200);
349             serialPort.DataReceived += new SerialDataReceivedEventHandler(this.serialPort_DataReceived
350             );
351             if (!serialPort.IsOpen)
352             {
353                 serialPort.Open();
354             }
355             Action("F", ""); //GET POSITION
356             Action("k", ""); //GET TEMPERATURE AND HUMIDITY
357
358             // when we establish connection, set up the increment, step and speed
359             Action("I", "100"); // SET SPEED
360             Action("J", "200"); // SET MAXSPEED
361             Action("H", "200"); // SET ACCELERATION
362
363
364             if (showUI)
365             {
366                 mainWindow = new MainWindow(this);
367                 mainWindow.Show();
368                 System.Diagnostics.Debug.WriteLine("Connected_" + IsConnected.ToString());
369             }
370         }
371         catch (Exception e)
372         {
373             connectedState = false;
374             tl.LogMessage("Cannot_Open_Serial_Port", comPort);
375             if (serialPort.IsOpen)
376             {
377                 serialPort.Close();
378             }
379         }
380     }
381     else
382     {
383         connectedState = false;
384         tl.LogMessage("Connected_Set", "Disconnecting_from_port_" + comPort);
385         // TODO disconnect from the device
386
387

```

```

388             if (showUI)
389             {
390                 mainWindow.Close();
391             }
392             if ( serialPort .IsOpen)
393             {
394                 serialPort .Close();
395             }
396         }
397     }
398 }
399
400 public string Description
401 {
402     // TODO customise this device description
403     get
404     {
405         tl .LogMessage("Description_Get", driverDescription );
406         return driverDescription ;
407     }
408 }
409
410 public string DriverInfo
411 {
412     get
413     {
414         Version version = System.Reflection.Assembly.GetExecutingAssembly().GetName().Version;
415         // TODO customise this driver description
416         string driverInfo = "Information about the driver itself . Version: " + String.Format(CultureInfo .
417             InvariantCulture , "{0}.{1}", version .Major, version .Minor);
418         tl .LogMessage("DriverInfo_Get", driverInfo );
419         return driverInfo ;
420     }
421 }
422
423 public string DriverVersion
424 {
425     get
426     {
427         Version version = System.Reflection.Assembly.GetExecutingAssembly().GetName().Version;
428         string driverVersion = String.Format(CultureInfo . InvariantCulture , "{0}.{1}", version .Major,
429             version .Minor);
430         tl .LogMessage("DriverVersion_Get", driverVersion );
431         return driverVersion ;
432     }
433 }
434
435 public short InterfaceVersion
436 {
437     // set by the driver wizard
438     get
439     {
440         tl .LogMessage("InterfaceVersion_Get", "3");
441         return Convert.ToInt16("3");
442     }
443 }
444
445 public string Name
446 {

```

```

445         get
446     {
447         string name = "Short_driver_name - please customise";
448         tl .LogMessage("Name_Get", name);
449         return name;
450     }
451 }
452
453     public SerialPort SerialPort
454     {
455         get
456         {
457             return serialPort ;
458         }
459     }
460
461     private void serialPort_DataReceived ( object sender , SerialDataReceivedEventArgs e)
462     {
463         try
464         {
465             message = SerialPort .ReadTo("#");
466             System.Diagnostics.Debug.WriteLine(message);
467             existingMessage = SerialPort .ReadExisting();
468             System.Diagnostics.Debug.WriteLine("Existing " + existingMessage);
469
470             if (message.Contains("POSITION"))
471             {
472                 focuserPosition = Convert.ToInt16(message.Split (':') [1]);
473                 OnFocuserValueChanged(new FocuserValueChangedEventArgs(focuserPosition, focuserPosition));
474                 OnFocuserStateChanged(new FocuserStateChangedEventArgs(false));
475                 this .isMoving = false ;
476             }
477
478             if (message.Contains("MOVING"))
479             {
480                 OnFocuserStateChanged(new FocuserStateChangedEventArgs(true));
481                 this .isMoving = true ;
482             }
483
484             if (message.Contains("TEMPERATURE"))
485             {
486                 this .temperature = Convert.ToDouble(message.Split (':') [1]);
487                 OnFocuserTemperatureChanged(
488                     new FocuserTemperatureChangedEventArgs(message.Split(':')[1] + "C"));
489             }
490
491             if (message.Contains("HUMIDITY"))
492             {
493                 this .humidity = Convert.ToDouble(message.Split (':') [1]);
494                 OnFocuserHumidityChanged(
495                     new FocuserHumidityChangedEventArgs(message.Split(':')[1] + "%"));
496             }
497         }
498         catch (Exception ex)
499         {
500             tl .LogMessage("Encountered_an_Exeption", ex.Message);
501         }
502     }
503

```

```

504     }
505 #endregion
506
507 #region IFocuser Implementation
508
509     public bool Absolute
510     {
511         get
512         {
513             tl.LogMessage("Absolute_Get", true.ToString());
514             return true; // This is an absolute focuser
515         }
516     }
517
518     public void Halt()
519     {
520         tl.LogMessage("Halt", "Called");
521         SerialPort.WriteLine("X");
522     }
523
524     public bool IsMoving
525     {
526         get
527         {
528             tl.LogMessage("IsMoving_Get", false.ToString());
529             return this.isMoving; // This focuser always moves instantaneously so no need for IsMoving ever to
530             be True
531         }
532     }
533
534     public bool Link
535     {
536         get
537         {
538             tl.LogMessage("Link_Get", this.Connected.ToString());
539             return this.Connected; // Direct function to the connected method, the Link method is just here
540             for backwards compatibility
541         }
542         set
543         {
544             tl.LogMessage("Link_Set", value.ToString());
545             this.Connected = value; // Direct function to the connected method, the Link method is just here
546             for backwards compatibility
547         }
548     }
549     public int MaxIncrement
550     {
551         get
552         {
553             return maxIncrement; // Maximum change in one move
554         }
555         set
556         {
557             maxIncrement = value;
558         }
559     }

```

```

560     {
561         get
562         {
563             tl.LogMessage("MaxStep_Get", focuserSteps.ToString());
564             return focuserSteps; // Maximum extent of the focuser, so position range is 0 to 10,000
565         }
566     }
567
568     public void Move(int Position)
569     {
570         tl.LogMessage("Move", Position.ToString());
571         // focuserPosition = Position;
572         Action("E", Position.ToString()); // Set the focuser position
573     }
574
575     public int Position
576     {
577         get
578         {
579             return focuserPosition; // Return the focuser position
580         }
581     }
582
583     public int MicroStepMode
584     {
585         get
586         {
587             return MicroSteppingMode; // Return microstepping mode
588         }
589         set
590         {
591             MicroSteppingMode = value;
592         }
593     }
594     public int Fcsh
595     {
596         get
597         {
598             return FCSH; // Return FCSH value (0~16)
599         }
600         set
601         {
602             FCSH = value;
603         }
604     }
605     public double StepSize
606     {
607         get
608         {
609             return stepSize;
610         }
611         set
612         {
613             stepSize = value;
614         }
615     }
616
617     public bool TempComp
618     {

```

```

619         get
620     {
621         tl.LogMessage("TempComp_Get", false.ToString());
622         return false ;
623     }
624     set
625     {
626         tl.LogMessage("TempComp_Set", "Not_implemented");
627         throw new ASCOM.PropertyNotImplementedException("TempComp", false);
628     }
629 }
630
631     public bool TempCompAvailable
632     {
633         get
634     {
635         tl.LogMessage("TempCompAvailable_Get", false.ToString());
636         return false ; // Temperature compensation is not available in this driver
637     }
638 }
639
640     public double Temperature
641     {
642         get
643     {
644         tl.LogMessage("Temperature_Get", "Not_implemented");
645         return temperature ;
646     }
647 }
648
649     public string MotorDriver
650     {
651         get
652     {
653         return motorDriver;
654     }
655     set
656     {
657         motorDriver = value;
658     }
659 }
660 #endregion
661
662 #region Private properties and methods
663 // here are some useful properties and methods that can be used as required
664 // to help with driver development
665
666 #region ASCOM Registration
667
668 // Register or unregister driver for ASCOM. This is harmless if already
669 // registered or unregistered .
670 //
671 /// <summary>
672 /// Register or unregister the driver with the ASCOM Platform.
673 /// This is harmless if the driver is already registered / unregistered .
674 /// </summary>
675 /// <param name="bRegister">If <c>true</c>, registers the driver , otherwise unregisters it .</param>
676 private static void RegUnregASCOM(bool bRegister)
677 {

```

```

678     using ( var P = new ASCOM.Utilities.Profile () )
679     {
680         P.DeviceType = "Focuser";
681         if ( bRegister )
682         {
683             P.Register ( driverID ,  driverDescription );
684         }
685         else
686         {
687             P.Unregister ( driverID );
688         }
689     }
690 }
691
692 /// <summary>
693 /// This function registers the driver with the ASCOM Chooser and
694 /// is called automatically whenever this class is registered for COM Interop.
695 /// </summary>
696 /// <param name="t">Type of the class being registered , not used.</param>
697 /// <remarks>
698 /// This method typically runs in two distinct situations :
699 /// <list type="numbered">
700 /// <item>
701 /// In Visual Studio, when the project is successfully built .
702 /// For this to work correctly , the option <c>Register for COM Interop</c>
703 /// must be enabled in the project settings .
704 /// </item>
705 /// <item>During setup, when the installer registers the assembly for COM Interop.</item>
706 /// </list >
707 /// This technique should mean that it is never necessary to manually register a driver with ASCOM.
708 /// </remarks>
709 [ComRegisterFunction]
710 public static void RegisterASCOM(Type t)
711 {
712     RegUnregASCOM(true);
713 }
714
715 /// <summary>
716 /// This function unregisters the driver from the ASCOM Chooser and
717 /// is called automatically whenever this class is unregistered from COM Interop.
718 /// </summary>
719 /// <param name="t">Type of the class being registered , not used.</param>
720 /// <remarks>
721 /// This method typically runs in two distinct situations :
722 /// <list type="numbered">
723 /// <item>
724 /// In Visual Studio, when the project is cleaned or prior to rebuilding .
725 /// For this to work correctly , the option <c>Register for COM Interop</c>
726 /// must be enabled in the project settings .
727 /// </item>
728 /// <item>During uninstall , when the installer unregisters the assembly from COM Interop.</item>
729 /// </list >
730 /// This technique should mean that it is never necessary to manually unregister a driver from ASCOM.
731 /// </remarks>
732 [ComUnregisterFunction]
733 public static void UnregisterASCOM(Type t)
734 {
735     RegUnregASCOM(false);
736 }

```

```

737
738 #endregion
739
740 /// <summary>
741 /// Returns true if there is a valid connection to the driver hardware
742 /// </summary>
743 private bool IsConnected
744 {
745     get
746     {
747         // TODO check that the driver hardware connection exists and is connected to the hardware
748         return connectedState;
749     }
750 }
751
752 /// <summary>
753 /// Use this function to throw an exception if we aren't connected to the hardware
754 /// </summary>
755 /// <param name="message"></param>
756 private void CheckConnected(string message)
757 {
758     if (!IsConnected)
759     {
760         throw new ASCOM.NotConnectedException(message);
761     }
762 }
763
764 /// <summary>
765 /// Read the device configuration from the ASCOM Profile store
766 /// </summary>
767 internal void ReadProfile()
768 {
769     using (Profile driverProfile = new Profile())
770     {
771         driverProfile.DeviceType = "Focuser";
772         traceState = Convert.ToBoolean(driverProfile.GetValue(driverID, traceStateProfileName, string.Empty, traceStateDefault));
773         tl.Enabled = Convert.ToBoolean(driverProfile.GetValue(driverID, traceStateProfileName, string.Empty, traceStateDefault));
774         comPort = driverProfile.GetValue(driverID, comPortProfileName, string.Empty, comPortDefault);
775         showUI = Convert.ToBoolean(driverProfile.GetValue(driverID, showUIProfileName, string.Empty, showUIDefault));
776     }
777 }
778
779 /// <summary>
780 /// Write the device configuration to the ASCOM Profile store
781 /// </summary>
782 internal void WriteProfile()
783 {
784     using (Profile driverProfile = new Profile())
785     {
786         driverProfile.DeviceType = "Focuser";
787         try
788         {
789             driverProfile.WriteLine(driverID, traceStateProfileName, traceState.ToString());
790             driverProfile.WriteLine(driverID, comPortProfileName, comPort.ToString());
791             driverProfile.WriteLine(driverID, showUIProfileName, showUI.ToString());
792         }

```

```

793         }
794     catch (Exception ex)
795     {
796         tl .LogMessage("Cannot_write_to_driveProfile", ex.Message);
797     }
798     // showUI = Convert.ToBoolean( driverProfile .GetValue(driverID , showUIProfileName, string .Empty,
799                             showUIDefault));
800     // driverProfile .WriteValue(driverID , traceStateProfileName , tl .Enabled.ToString ());
801     // driverProfile .WriteValue(driverID , comPortProfileName, comPort.ToString ());
802 }
803
804 /// <summary>
805 /// Log helper function that takes formatted strings and arguments
806 /// </summary>
807 /// <param name=" identifier "></param>
808 /// <param name="message"></param>
809 /// <param name="args"></param>
810 #endregion
811 }
812 }
```

A.2.2. FCSH.cs

```

1   using System;
2   using System.Collections.Generic;
3   using System.ComponentModel;
4   using System.Data;
5   using System.Drawing;
6   using System.Linq;
7   using System.Text;
8   using System.Windows.Forms;
9
10  namespace ASCOM.GS_touch
11  {
12      public partial class FCSH : Form
13  {
14      private ASCOM.GS_touch.MainWindow mainWindow;
15      private int FCSH_Value;
16      public FCSH(MainWindow mainWindow)
17      {
18          this .mainWindow = mainWindow;
19          InitializeComponent ();
20          InitStatue ();
21          FCSH_Value = new int();
22          FCSH_Value = mainWindow.askFCSH();
23      }
24      private void InitStatue ()
25      {
26          if (FCSH_Value / 2 % 2 == 1) label_C.Text = "Cover_Status:_Opened";
27          else label_C.Text = "Cover_Status:_Closed";
28
29          if (FCSH_Value / 4 % 2 == 1) label_M.Text = "Mask_Status:_Opened";
30          else label_M.Text = "Mask_Status:_Closed";
31
32          if (FCSH_Value / 8 % 2 == 1) label_CC.Text = "CCD_Power:_ON";
33          else label_CC.Text = "CCD_Power:_OFF";
34
35          if (FCSH_Value / 16 % 2 == 1) label_M.Text = "Mount_Power:_ON";

```

```

36         else label_Mo.Text = "Mount\u201d;OFF";
37     }
38
39     private void FCSH_Load(object sender, EventArgs e)
40     {
41
42     }
43
44     private void button_Cover_Click(object sender, EventArgs e)
45     {
46         mainWindow.applyFCSH(1);
47         if (label_C.Text == "Cover\u201dStatue\u201d;Closed") label_C.Text = "Cover\u201dStatue\u201d;Opened";
48         else label_C.Text = "Cover\u201dStatue\u201d;Closed";
49     }
50
51
52     private void button_Mask_Click(object sender, EventArgs e)
53     {
54         mainWindow.applyFCSH(2);
55         if (label_M.Text == "Mask\u201dStatue\u201d;Closed") label_M.Text = "Mask\u201dStatue\u201d;Opened";
56         else label_M.Text = "Mask\u201dStatue\u201d;Closed";
57     }
58
59     private void button_CCD_Click(object sender, EventArgs e)
60     {
61         mainWindow.applyFCSH(3);
62         if (label_CC.Text == "CCD\u201dPower\u201d;OFF") label_CC.Text = "CCD\u201dPower\u201d;ON";
63         else label_CC.Text = "CCD\u201dPower\u201d;OFF";
64     }
65
66     private void button_Mount_Click(object sender, EventArgs e)
67     {
68         mainWindow.applyFCSH(4);
69         if (label_Mo.Text == "Mount\u201dPower\u201d;OFF") label_Mo.Text = "Mount\u201dPower\u201d;ON";
70         else label_Mo.Text = "Mount\u201dPower\u201d;OFF";
71     }
72 }
73 }
```

A.2.3. FCSH.Designer.cs

```

1  namespace ASCOM.GS_touch
2  {
3      partial class FCSH
4      {
5          /// <summary>
6          /// Required designer variable.
7          /// </summary>
8          private System.ComponentModel.IContainer components = null;
9
10         /// <summary>
11         /// Clean up any resources being used.
12         /// </summary>
13         /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
14         protected override void Dispose(bool disposing)
15         {
16             if (disposing && (components != null))
17             {
18                 components.Dispose();
19             }
20         }
21     }
22 }
```

```

19         }
20         base.Dispose(disposing);
21     }
22
23 #region Windows Form Designer generated code
24
25 // <summary>
26 // Required method for Designer support – do not modify
27 // the contents of this method with the code editor .
28 // </summary>
29 private void InitializeComponent()
30 {
31     this.groupBox_Status = new System.Windows.Forms.GroupBox();
32     this.label_Mo = new System.Windows.Forms.Label();
33     this.label_CC = new System.Windows.Forms.Label();
34     this.label_M = new System.Windows.Forms.Label();
35     this.label_C = new System.Windows.Forms.Label();
36     this.button_Cover = new System.Windows.Forms.Button();
37     this.button_Mask = new System.Windows.Forms.Button();
38     this.button_CCD = new System.Windows.Forms.Button();
39     this.button_Mount = new System.Windows.Forms.Button();
40     this.groupBox_Control = new System.Windows.Forms.GroupBox();
41     this.groupBox_Status.SuspendLayout();
42     this.groupBox_Control.SuspendLayout();
43     this.SuspendLayout();
44
45     // groupBox_Status
46     //
47     this.groupBox_Status.Controls.Add(this.label_Mo);
48     this.groupBox_Status.Controls.Add(this.label_CC);
49     this.groupBox_Status.Controls.Add(this.label_M);
50     this.groupBox_Status.Controls.Add(this.label_C);
51     this.groupBox_Status.Location = new System.Drawing.Point(12, 12);
52     this.groupBox_Status.Name = "groupBox_Status";
53     this.groupBox_Status.Size = new System.Drawing.Size(175, 146);
54     this.groupBox_Status.TabIndex = 0;
55     this.groupBox_Status.TabStop = false;
56     this.groupBox_Status.Text = "Current_Status";
57
58     // label_Mo
59     //
60     this.label_Mo.AutoSize = true;
61     this.label_Mo.Location = new System.Drawing.Point(7, 120);
62     this.label_Mo.Name = "label_Mo";
63     this.label_Mo.Size = new System.Drawing.Size(92, 12);
64     this.label_Mo.TabIndex = 3;
65     this.label_Mo.Text = "Mount_Power:_";
66
67     // label_CC
68     //
69     this.label_CC.AutoSize = true;
70     this.label_CC.Location = new System.Drawing.Point(16, 90);
71     this.label_CC.Name = "label_CC";
72     this.label_CC.Size = new System.Drawing.Size(79, 12);
73     this.label_CC.TabIndex = 1;
74     this.label_CC.Text = "CCD_Power:_";
75
76     // label_M
77     //

```

```

78     this .label_M.AutoSize = true ;
79     this .label_M.Location = new System.Drawing.Point(13, 60);
80     this .label_M.Name = "label_M";
81     this .label_M.Size = new System.Drawing.Size(87, 12);
82     this .label_M.TabIndex = 2;
83     this .label_M.Text = "Mask\u00a0Statue\u00a0";
84     //
85     // label_C
86     //
87     this .label_C.AutoSize = true ;
88     this .label_C.Location = new System.Drawing.Point(11, 30);
89     this .label_C.Name = "label_C";
90     this .label_C.Size = new System.Drawing.Size(89, 12);
91     this .label_C.TabIndex = 1;
92     this .label_C.Text = "Cover\u00a0Statue\u00a0";
93     //
94     // button_Cover
95     //
96     this .button_Cover.Location = new System.Drawing.Point(6, 21);
97     this .button_Cover.Name = "button_Cover";
98     this .button_Cover.Size = new System.Drawing.Size(75, 26);
99     this .button_Cover.TabIndex = 2;
100    this .button_Cover.Text = "Cover";
101    this .button_Cover.UseVisualStyleBackColor = true ;
102    this .button_Cover.Click += new System.EventHandler(this .button_Cover_Click);
103    //
104    // button_Mask
105    //
106    this .button_Mask.Location = new System.Drawing.Point(6, 51);
107    this .button_Mask.Name = "button_Mask";
108    this .button_Mask.Size = new System.Drawing.Size(75, 26);
109    this .button_Mask.TabIndex = 3;
110    this .button_Mask.Text = "Mask";
111    this .button_Mask.UseVisualStyleBackColor = true ;
112    this .button_Mask.Click += new System.EventHandler(this .button_Mask_Click);
113    //
114    // button_CCD
115    //
116    this .button_CCD.Location = new System.Drawing.Point(6, 81);
117    this .button_CCD.Name = "button_CCD";
118    this .button_CCD.Size = new System.Drawing.Size(75, 26);
119    this .button_CCD.TabIndex = 4;
120    this .button_CCD.Text = "CCD";
121    this .button_CCD.UseVisualStyleBackColor = true ;
122    this .button_CCD.Click += new System.EventHandler(this .button_CCD_Click);
123    //
124    // button_Mount
125    //
126    this .button_Mount.Location = new System.Drawing.Point(6, 111);
127    this .button_Mount.Name = "button_Mount";
128    this .button_Mount.Size = new System.Drawing.Size(75, 26);
129    this .button_Mount.TabIndex = 5;
130    this .button_Mount.Text = "Mount";
131    this .button_Mount.UseVisualStyleBackColor = true ;
132    this .button_Mount.Click += new System.EventHandler(this .button_Mount_Click);
133    //
134    // groupBox_Control
135    //
136    this .groupBox_Control.Controls.Add(this .button_Cover);

```

```

137     this .groupBox_Control.Controls.Add(this .button_Mount);
138     this .groupBox_Control.Controls.Add(this .button_Mask);
139     this .groupBox_Control.Controls.Add(this .button_CCD);
140     this .groupBox_Control.Location = new System.Drawing.Point(193, 12);
141     this .groupBox_Control.Name = "groupBox_Control";
142     this .groupBox_Control.Size = new System.Drawing.Size(87, 146);
143     this .groupBox_Control.TabIndex = 6;
144     this .groupBox_Control.TabStop = false ;
145     this .groupBox_Control.Text = "Control";
146     //
147 // FCSH
148 //
149     this .AutoScaleDimensions = new System.Drawing.SizeF(7F, 12F);
150     this .AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
151     this .ClientSize = new System.Drawing.Size(287, 165);
152     this .Controls .Add(this .groupBox_Control);
153     this .Controls .Add(this .groupBox_Status);
154     this .Name = "FCSH";
155     this .Text = "FCSH_Status";
156     this .Load += new System.EventHandler(this .FCSH_Load);
157     this .groupBox_Status.ResumeLayout(false );
158     this .groupBox_Status.PerformLayout();
159     this .groupBox_Control.ResumeLayout(false );
160     this .ResumeLayout(false );
161 }
162 }
163 #endregion
164
165 private System.Windows.Forms.GroupBox groupBox_Status;
166 private System.Windows.Forms.Label label_Mo;
167 private System.Windows.Forms.Label label_CC;
168 private System.Windows.Forms.Label label_M;
169 private System.Windows.Forms.Label label_C;
170 private System.Windows.Forms.Button button_Cover;
171 private System.Windows.Forms.Button button_Mask;
172 private System.Windows.Forms.Button button_CCD;
173 private System.Windows.Forms.Button button_Mount;
174 private System.Windows.Forms.GroupBox groupBox_Control;
175
176 }
177 }
```

A.2.4. MainWindow.cs

```

1  using System;
2  using System.Collections .Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Windows.Forms;
9
10 namespace ASCOM.GS_touch
11 {
12     public partial class MainWindow : Form
13     {
14         ASCOM.GS_touch.Focuser focuser;
```

```

16
17     public MainWindow(Focuser focuser)
18     {
19         this .focuser = focuser ;
20         this .focuser .FocuserValueChanged += FocuserValueChanged;
21         this .focuser .FocuserStateChanged += FocuserStateChanged;
22         this .focuser .FocuserHumidityChanged += FocuserHumidityChanged;
23         this .focuser .FocuserTemperatureChanged += FocuserTemperatureChanged;
24         InitializeComponent () ;
25         InitControls () ;
26
27         this .FormBorderStyle = FormBorderStyle.FixedSingle ;
28     }
29
30     delegate void SetCurrentPositionCallBack (int position );
31     delegate void SetCurrentStateCallBack (bool isMoving);
32     delegate void SetCurrentTemperatureCallBack( string temperature );
33     delegate void SetCurrentHumidityCallBack( string humidity );
34
35     private void SetCurrentTemperature( string temperature )
36     {
37         textBox_Temperature.InvokeIfRequired(textBox_Temperature => { textBox_Temperature.Text = temperature ;
38             });
39     }
40
41     private void SetCurrentHumidity( string humidity )
42     {
43
44         textBox_Humidity.InvokeIfRequired(textBox_Humidity => { textBox_Humidity.Text = humidity ; });
45     }
46
47     public void SetCurrentPosition (int position )
48     {
49
50         textBox_Position .InvokeIfRequired( textBox_Position => { textBox_Position .Text = position .ToString () ;
51             });
52
53     private void FocuserValueChanged(object sender , FocuserValueChangedEventArgs e)
54     {
55         // this .textBoxCursorPosition .Text = e.NewValue.ToString();
56         SetCurrentPosition (e.NewValue);
57     }
58
59     private void SetCurrentState (bool isMoving)
60     {
61         if (label_Moving.InvokeRequired)
62         {
63             SetCurrentStateCallBack d = new SetCurrentStateCallBack( SetCurrentState );
64             this .Invoke(d, new object [] { isMoving });
65         }
66         else
67         {
68             if (isMoving)
69             {
70                 label_Moving.Text = "MOVING...";
71             }
72             else
73             {

```

```

73             label_Moving.Text = "READY...";
74         }
75     }
76 }
77
78 #region Changed EventArg Declaration
79
80 private void FocuserHumidityChanged(object sender, FocuserHumidityChangedEventArgs e)
81 {
82     SetCurrentHumidity(e.Humidity);
83 }
84
85 private void FocuserTemperatureChanged(object sender, FocuserTemperatureChangedEventArgs e)
86 {
87     SetCurrentTemperature(e.Temperature);
88 }
89
90 private void FocuserStateChanged(object sender, FocuserStateChangedEventArgs e)
91 {
92     SetCurrentState(e.IsMoving);
93 }
94 #endregion
95
96 private void InitControls()
97 {
98     textBox_Position.Text = focuser.Position.ToString();
99     label_MS.Text = "MicroStepping_Mode:_";
100    switch (focuser.MicroStepMode)
101    {
102        case 1:
103            label_MS.Text += "1_(full_step)";
104            focuser.Action("M", Convert.ToString(1));
105            break;
106
107        case 2:
108            label_MS.Text += "2_(half_step)";
109            focuser.Action("M", Convert.ToString(2));
110            break;
111
112        case 3:
113            label_MS.Text += "3_(1/4_step)";
114            focuser.Action("M", Convert.ToString(3));
115            break;
116
117        case 4:
118            label_MS.Text += "4_(1/8_step)";
119            focuser.Action("M", Convert.ToString(4));
120            break;
121    }
122 }
123
124 private ResetForm resetForm;
125 private void button_Reset_Click(object sender, EventArgs e)
126 {
127     resetForm = new ResetForm(this);
128     resetForm.Show();
129     InitControls();
130 }
131

```

```

132     private void button_Advanced_Click(object sender, EventArgs e)
133     {
134         panel_Advanced.Visible = !panel_Advanced.Visible;
135
136         if (panel_Advanced.Visible)
137         {
138             this .Height = 505;
139             button_Advanced.Text = " Briefly ";
140         }
141         else
142         {
143             this .Height = 280;
144             button_Advanced.Text = "Advanced";
145         }
146     }
147
148     private void button_Apply_Click(object sender, EventArgs e)
149     {
150         focuser .Move(Convert.ToInt16(textBox_MoveTo.Text));
151     }
152
153     private void button_stop_Click (object sender, EventArgs e)
154     {
155         focuser .Halt();
156     }
157
158     private void button_moveleft_Click (object sender, EventArgs e)
159     {
160         focuser .CommandString("B", true);
161     }
162
163     private void button_moveright_Click (object sender, EventArgs e)
164     {
165         focuser .CommandString("C", true);
166     }
167
168     private void numericUpDown_SingleStep_ValueChanged(object sender, EventArgs e)
169     {
170         focuser .StepSize = Convert.ToDouble(numericUpDown_SingleStep.Value);
171     }
172
173     private void numericUpDown_Speed_ValueChanged(object sender, EventArgs e)
174     {
175         focuser .Action("J", numericUpDown_Speed.Value.ToString());
176     }
177
178     private void numericUpDown_Acceleration_ValueChanged(object sender, EventArgs e)
179     {
180         focuser .Action("H", numericUpDown_Acceleration.Value.ToString());
181     }
182
183     private void numericUpDown_PWM_ValueChanged(object sender, EventArgs e)
184     {
185         focuser .Action("A", numericUpDown_PWM.Value.ToString());
186     }
187 // 일단 A에 PWM 할당
188
189     private void timer1_Tick(object sender, EventArgs e)
190     {

```

```

191         // do not disrupt the motor when it is moving
192         if (!focuser.IsMoving)
193         {
194             System.Diagnostics.Debug.WriteLine("Getting_Temperature");
195             focuser.Action("k", "");
196         }
197     }
198
199     private void groupBox_Advanced_Enter(object sender, EventArgs e)
200     {
201
202     }
203
204     public String askPosition()
205     {
206         return focuser.Position.ToString();
207     }
208     public void applyPosition(String resetPosition)
209     {
210         focuser.Action("G", resetPosition);
211         SetCurrentPosition(Convert.ToInt32(resetPosition));
212         return;
213     }
214
215     private void MainWindow_Load(object sender, EventArgs e)
216     {
217
218     }
219
220     private FCSH fCSH;
221     private void button_maskControl_Click(object sender, EventArgs e)
222     {
223         fCSH = new FCSH(this);
224         fCSH.Show();
225     }
226     public void applyFCSH(int type)
227     {
228         if (type==1) focuser.Action("N", "");
229         else if (type == 2) focuser.Action("O", "");
230         else if (type == 3) focuser.Action("P", "");
231         else if (type == 4) focuser.Action("Q", "");
232         else if (type == 5) focuser.Action("R", "");
233         else if (type == 6) focuser.Action("S", "");
234         return;
235     }
236     public int askFCSH()
237     {
238         return focuser.Fcsh;
239         // 일단 이렇게 해놓자
240     }
241
242 }
243
244 #region ExtensionMethods
245
246     public static class ExtensionMethods
247     {
248         public static void InvokeIfRequired(this TextBox control, Action<TextBox> action)
249         {

```

```

250         if ( control .InvokeRequired)
251     {
252         control .Invoke(new Action(() => action ( control )));
253     }
254     else
255     {
256         action ( control );
257     }
258   }
259 }
260 #endregion
261 }
```

A.2.5. ResetForm.cs

```

1   using  System;
2  using System. Collections .Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Windows.Forms;
9
10 namespace ASCOM.GS_touch
11 {
12     public  partial  class ResetForm : Form
13     {
14         private ASCOM.GS_touch.MainWindow mainWindow;
15         public ResetForm(MainWindow mainWindow)
16     {
17         this .mainWindow = mainWindow;
18         InitializeComponent ();
19         textBox_ResetPosition .Text = mainWindow.askPosition();
20     }
21
22     internal  const  int MaxStep = 65535;
23
24     private void button_Reset_Click (object  sender , EventArgs e)
25     {
26         if (Convert.ToInt32( textBox_ResetPosition .Text) > MaxStep || Convert.ToInt32( textBox_ResetPosition .
27             Text) < -1*MaxStep)
28         {
29             System.Windows.MessageBox.Show("Reset_Position_Out_of_Range");
30         }
31         else
32         {
33             mainWindow.applyPosition(textBox_ResetPosition .Text);
34             //mainWindow.SetCurrentPosition(Convert.ToInt32( textBox_ResetPosition ));
35             this .Close();
36         }
37     }
38     private void button_Cancel_Click (object  sender , EventArgs e)
39     {
40         this .Close();
41     }
42 }
43 }
```

A.2.6. ResetForm.Designer.cs

```
1  namespace ASCOM.GS_touch
2 {
3     partial class ResetForm
4     {
5         /// <summary>
6         /// Required designer variable.
7         /// </summary>
8         private System.ComponentModel.IContainer components = null;
9
10        /// <summary>
11        /// Clean up any resources being used.
12        /// </summary>
13        /// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
14        protected override void Dispose(bool disposing)
15        {
16            if (disposing && (components != null))
17            {
18                components.Dispose();
19            }
20            base.Dispose(disposing);
21        }
22
23        #region Windows Form Designer generated code
24
25        /// <summary>
26        /// Required method for Designer support – do not modify
27        /// the contents of this method with the code editor.
28        /// </summary>
29        private void InitializeComponent()
30        {
31            this.label_ResetPosition = new System.Windows.Forms.Label();
32            this.textBox_ResetPosition = new System.Windows.Forms.TextBox();
33            this.button_Reset = new System.Windows.Forms.Button();
34            this.button_Cancel = new System.Windows.Forms.Button();
35            this.SuspendLayout();
36            //
37            // label_ResetPosition
38            //
39            this.label_ResetPosition.AutoSize = true;
40            this.label_ResetPosition.Location = new System.Drawing.Point(26, 36);
41            this.label_ResetPosition.Name = "label_ResetPosition";
42            this.label_ResetPosition.Size = new System.Drawing.Size(81, 12);
43            this.label_ResetPosition.TabIndex = 0;
44            this.label_ResetPosition.Text = "Position _want";
45            //
46            // textBox_ResetPosition
47            //
48            this.textBox_ResetPosition.Location = new System.Drawing.Point(128, 30);
49            this.textBox_ResetPosition.Name = "textBox_ResetPosition";
50            this.textBox_ResetPosition.Size = new System.Drawing.Size(150, 21);
51            this.textBox_ResetPosition.TabIndex = 2;
52            //
53            // button_Reset
54            //
55            this.button_Reset.Location = new System.Drawing.Point(128, 65);
56            this.button_Reset.Name = "button_Reset";
57            this.button_Reset.Size = new System.Drawing.Size(70, 25);
```

```

58     this.button_Reset.TabIndex = 4;
59     this.button_Reset.Text = "Apply";
60     this.button_Reset.UseVisualStyleBackColor = true ;
61     this.button_Reset.Click += new System.EventHandler(this.button_Reset_Click);
62     //
63     // button_Cancel
64     //
65     this.button_Cancel.Location = new System.Drawing.Point(208, 65);
66     this.button_Cancel.Name = "button_Cancel";
67     this.button_Cancel.Size = new System.Drawing.Size(70, 25);
68     this.button_Cancel.TabIndex = 5;
69     this.button_Cancel.Text = "Cancel";
70     this.button_Cancel.UseVisualStyleBackColor = true ;
71     this.button_Cancel.Click += new System.EventHandler(this.button_Cancel_Click);
72     //
73     // ResetForm
74     //
75     this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 12F);
76     this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
77     this.ClientSize = new System.Drawing.Size(304, 111);
78     this.Controls.Add(this.button_Cancel);
79     this.Controls.Add(this.button_Reset);
80     this.Controls.Add(this.textBox_ResetPosition );
81     this.Controls.Add(this.label_ResetPosition );
82     this.Name = "ResetForm";
83     this.Text = "Reset_Settings";
84     this.ResumeLayout(false);
85     this.PerformLayout();
86     this.PerformLayout();
87 }
88
89 #endregion
90
91 private System.Windows.Forms.Label label_ResetPosition ;
92 private System.Windows.Forms.TextBox textBox_ResetPosition;
93 private System.Windows.Forms.Button button_Reset;
94 private System.Windows.Forms.Button button_Cancel;
95 }
96 }
```

A.2.7. SetupDialogForm.cs

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Drawing;
5  using System.Runtime.InteropServices ;
6  using System.Text;
7  using System.Windows.Forms;
8  using ASCOM.Utilities;
9  using ASCOM.GS_touch;
10
11 using System.IO.Ports ;
12 using System.Diagnostics ;
13 using System.Threading;
14 using System.Threading.Tasks;
15 using System.Linq;
16
17 namespace ASCOM.GS_touch
```

```

18 {
19     [ComVisible(false)] // Form not registered for COM!
20     public partial class SetupDialogForm : Form
21     {
22         public SetupDialogForm()
23         {
24             InitializeComponent();
25             // Initialise current values of user settings from the ASCOM Profile
26             InitUI();
27         }
28
29         private void cmdOK_Click(object sender, EventArgs e) // OK button event handler
30         {
31             // Place any validation constraint checks here
32             // Update the state variables with results from the dialogue
33             Focuser.comPort = (string)comboBoxComPort.SelectedItem;
34             Focuser.showUI = showUI.Checked;
35
36             // String modeString = comboBoxMS.SelectedIndex.ToString().Substring(0,0);
37             int mode = Convert.ToInt32(comboBoxMS.SelectedIndex.ToString().Split(':')[0]);
38             Focuser.MicroSteppingMode = mode;
39         }
40
41         private void cmdCancel_Click(object sender, EventArgs e) // Cancel button event handler
42         {
43             Close();
44         }
45
46         private void BrowseToAscom(object sender, EventArgs e) // Click on ASCOM logo event handler
47         {
48             try
49             {
50                 System.Diagnostics.Process.Start("http://ascom-standards.org/");
51             }
52             catch (System.ComponentModel.Win32Exception noBrowser)
53             {
54                 if (noBrowser.ErrorCode == -2147467259)
55                     MessageBox.Show(noBrowser.Message);
56             }
57             catch (System.Exception other)
58             {
59                 MessageBox.Show(other.Message);
60             }
61         }
62
63         private void InitUI()
64         {
65             chkTrace.Checked = Focuser.traceState;
66             showUI.Checked = Focuser.showUI;
67
68             // set the list of com ports to those that are currently available
69             comboBoxComPort.Items.Clear();
70
71             String[] ports = SerialPort.GetPortNames();
72             foreach (string port in ports)
73             {
74                 Debug.WriteLine("Port_here_" + port);
75                 if (Detect_TFocuser(port))
76                 {

```

```

77             comboBoxComPort.Items.Add(port);
78         }
79     }
80
81     comboBoxComPort.Items.AddRange(System.IO.Ports.SerialPort.GetPortNames()); // use System.IO
82         because it's static
83     // select the current port if possible
84     if (comboBoxComPort.Items.Contains(Focuser.comPort))
85     {
86         comboBoxComPort.SelectedItem = Focuser.comPort;
87     }
88
89     comboBoxMS.Items.Add("1_(full_step)");
90     comboBoxMS.Items.Add("2_(1/2_step)");
91     comboBoxMS.Items.Add("3_(1/4_step)");
92     comboBoxMS.Items.Add("4_(1/8_step)");
93     comboBoxMS.SelectedIndex = Focuser.MicroSteppingMode;
94 }
95
96 private bool Detect_TFocuser( string portName)
97 {
98     SerialPort testPort = new SerialPort (portName, 115200);
99     try
100    {
101        testPort .Open();
102        testPort .WriteLine("Z");
103
104        Thread.Sleep(100);
105        string returnMessage = testPort .ReadExisting() .ToString () ;
106
107        testPort .Close();
108        Debug.WriteLine(returnMessage);
109
110        if (returnMessage.Contains ("EQEQFOCUSER_STEPPER") || returnMessage.Contains("POSITION"))
111        {
112            Focuser.motorDriver = Focuser.stepperMotor;
113            return true ;
114        }
115        else return false ;
116    }
117    catch (Exception e)
118    {
119        Debug.WriteLine(e.Message);
120        return false ;
121    }
122 }
123
124 private void chkTrace_CheckedChanged(object sender, EventArgs e)
125 {
126
127 }
128
129 private void showUI_CheckedChanged(object sender, EventArgs e)
130 {
131
132 }
133
134 private void Reconnect_Click(object sender, EventArgs e)

```

```

135     {
136         chkTrace.Checked = Focuser.traceState ;
137         showUI.Checked = Focuser.showUI;
138
139         // set the list of com ports to those that are currently available
140         comboBoxComPort.Items.Clear();
141
142         String [] ports = SerialPort .GetPortNames();
143         foreach ( string port in ports )
144         {
145             Debug.WriteLine("Port_here_" + port);
146             if (Detect_TFocuser(port))
147             {
148                 comboBoxComPort.Items.Add(port);
149             }
150         }
151
152         comboBoxComPort.Items.AddRange(System.IO.Ports.SerialPort.GetPortNames()); // use System.IO
153         // because it's static
154         // select the current port if possible
155         if (comboBoxComPort.Items.Contains(Focuser.comPort))
156         {
157             comboBoxComPort.SelectedItem = Focuser.comPort;
158         }
159
160         private void MS_SelectedIndexChanged(object sender, EventArgs e)
161     {
162
163     }
164
165
166     }
167 }
```

A.2.8. SetupDialogForm.designer.cs

```

1 namespace ASCOM.GS_touch
2 {
3     partial class SetupDialogForm
4     {
5         /// <summary>
6         /// Required designer variable.
7         /// </summary>
8         private System.ComponentModel.IContainer components = null;
9
10        /// <summary>
11        /// Clean up any resources being used.
12        /// </summary>
13        // <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>
14        protected override void Dispose(bool disposing)
15        {
16            if (disposing && (components != null))
17            {
18                components.Dispose();
19            }
20            base.Dispose(disposing);
21        }
22    }
```

```

23 #region Windows Form Designer generated code
24
25 // <summary>
26 // Required method for Designer support – do not modify
27 // the contents of this method with the code editor .
28 // </summary>
29 private void InitializeComponent ()
30 {
31     this .cmdOK = new System.Windows.Forms.Button();
32     this .cmdCancel = new System.Windows.Forms.Button();
33     this . Title = new System.Windows.Forms.Label();
34     this .picASCOM = new System.Windows.Forms.PictureBox();
35     this .Comport = new System.Windows.Forms.Label();
36     this .comboBoxComPort = new System.Windows.Forms.ComboBox();
37     this .showUI = new System.Windows.Forms.CheckBox();
38     this .MS_mode = new System.Windows.Forms.Label();
39     this .comboBoxMS = new System.Windows.Forms.ComboBox();
40     this .chkTrace = new System.Windows.Forms.CheckBox();
41     this .Reconnect = new System.Windows.Forms.Button();
42     ((System.ComponentModel.ISupportInitialize)(this .picASCOM)).BeginInit();
43     this .SuspendLayout();
44     //
45     // cmdOK
46     //
47     this .cmdOK.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
48     this .cmdOK.DialogResult = System.Windows.Forms.DialogResult.OK;
49     this .cmdOK.Location = new System.Drawing.Point(328, 112);
50     this .cmdOK.Name = "cmdOK";
51     this .cmdOK.Size = new System.Drawing.Size(69, 22);
52     this .cmdOK.TabIndex = 0;
53     this .cmdOK.Text = "OK";
54     this .cmdOK.UseVisualStyleBackColor = true;
55     this .cmdOK.Click += new System.EventHandler(this.cmdOK_Click);
56     //
57     // cmdCancel
58     //
59     this .cmdCancel.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Right)));
60     this .cmdCancel.DialogResult = System.Windows.Forms.DialogResult.Cancel;
61     this .cmdCancel.Location = new System.Drawing.Point(328, 140);
62     this .cmdCancel.Name = "cmdCancel";
63     this .cmdCancel.Size = new System.Drawing.Size(69, 23);
64     this .cmdCancel.TabIndex = 1;
65     this .cmdCancel.Text = "Cancel";
66     this .cmdCancel.UseVisualStyleBackColor = true ;
67     this .cmdCancel.Click += new System.EventHandler(this.cmdCancel_Click);
68     //
69     // Title
70     //
71     this . Title .Font = new System.Drawing.Font("Microsoft_Sans_Serif", 15F);
72     this . Title .Location = new System.Drawing.Point(12, 8);
73     this . Title .Name = "Title";
74     this . Title .Size = new System.Drawing.Size(335, 31);
75     this . Title .TabIndex = 2;
76     this . Title .Text = "GS-touch_ASCOM_Driver_(v_1.0.0b)";
77     //
78     // picASCOM
79     //

```

```

80    this.picASCOM.Anchor = ((System.Windows.Forms.AnchorStyles)((System.Windows.Forms.AnchorStyles.Top | 
81        System.Windows.Forms.AnchorStyles.Right)));
82    this.picASCOM.Cursor = System.Windows.Forms.Cursors.Hand;
83    this.picASCOM.Image = global::ASCOM.GS_touch.Properties.Resources.ASCOM;
84    this.picASCOM.Location = new System.Drawing.Point(17, 42);
85    this.picASCOM.Name = "picASCOM";
86    this.picASCOM.Size = new System.Drawing.Size(48, 56);
87    this.picASCOM.SizeMode = System.Windows.Forms.PictureBoxSizeMode.AutoSize;
88    this.picASCOM.TabIndex = 3;
89    this.picASCOM.TabStop = false;
90    this.picASCOM.Click += new System.EventHandler(this.BrowseToAscom);
91    this.picASCOM.DoubleClick += new System.EventHandler(this.BrowseToAscom);
92    // 
93    // Comport
94    // 
95    this.Comport.AutoSize = true;
96    this.Comport.Location = new System.Drawing.Point(93, 53);
97    this.Comport.Name = "Comport";
98    this.Comport.Size = new System.Drawing.Size(161, 12);
99    this.Comport.TabIndex = 5;
100   this.Comport.Text = "Baudrate\u00b7115200\u00b7COM_port";
101   // 
102   // comboBoxComPort
103   // 
104   this.comboBoxComPort.FormattingEnabled = true;
105   this.comboBoxComPort.Location = new System.Drawing.Point(260, 50);
106   this.comboBoxComPort.Name = "comboBoxComPort";
107   this.comboBoxComPort.Size = new System.Drawing.Size(137, 20);
108   this.comboBoxComPort.TabIndex = 7;
109   // 
110   // showUI
111   // 
112   this.showUI.AutoSize = true;
113   this.showUI.Location = new System.Drawing.Point(20, 120);
114   this.showUI.Name = "showUI";
115   this.showUI.Size = new System.Drawing.Size(114, 16);
116   this.showUI.TabIndex = 8;
117   this.showUI.Text = "Show_Controller";
118   this.showUI.UseVisualStyleBackColor = true;
119   this.showUI.CheckedChanged += new System.EventHandler(this.showUI_CheckedChanged);
120   // 
121   // MS_mode
122   // 
123   this.MS_mode.AutoSize = true;
124   this.MS_mode.Location = new System.Drawing.Point(133, 78);
125   this.MS_mode.Name = "MS_mode";
126   this.MS_mode.Size = new System.Drawing.Size(121, 12);
127   this.MS_mode.TabIndex = 9;
128   this.MS_mode.Text = "Microstepping_Mode";
129   // 
130   // comboBoxMS
131   // 
132   this.comboBoxMS.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;
133   this.comboBoxMS.FormattingEnabled = true;
134   this.comboBoxMS.Location = new System.Drawing.Point(260, 76);
135   this.comboBoxMS.Name = "comboBoxMS";
136   this.comboBoxMS.Size = new System.Drawing.Size(137, 20);
137   this.comboBoxMS.TabIndex = 10;
138   this.comboBoxMS.SelectedIndexChanged += new System.EventHandler(this.MS_SelectedIndexChanged);

```

```

138    //  

139    // chkTrace  

140    //  

141    this.chkTrace.AutoSize = true;  

142    this.chkTrace.Location = new System.Drawing.Point(20, 142);  

143    this.chkTrace.Name = "chkTrace";  

144    this.chkTrace.Size = new System.Drawing.Size(75, 16);  

145    this.chkTrace.TabIndex = 6;  

146    this.chkTrace.Text = "Trace_on";  

147    this.chkTrace.UseVisualStyleBackColor = true;  

148    this.chkTrace.CheckedChanged += new System.EventHandler(this.chkTrace_CheckedChanged);  

149    //  

150    // Reconnect  

151    //  

152    this.Reconnect.Location = new System.Drawing.Point(233, 112);  

153    this.Reconnect.Name = "Reconnect";  

154    this.Reconnect.Size = new System.Drawing.Size(89, 52);  

155    this.Reconnect.TabIndex = 11;  

156    this.Reconnect.Text = "Reconnect";  

157    this.Reconnect.UseVisualStyleBackColor = true;  

158    this.Reconnect.Click += new System.EventHandler(this.Reconnect_Click);  

159    //  

160    // SetupDialogForm  

161    //  

162    this.AutoScaleDimensions = new System.Drawing.SizeF(7F, 12F);  

163    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;  

164    this.ClientSize = new System.Drawing.Size(408, 169);  

165    this.Controls.Add(this.Reconnect);  

166    this.Controls.Add(this.comboBoxMS);  

167    this.Controls.Add(this.MS_mode);  

168    this.Controls.Add(this.showUI);  

169    this.Controls.Add(this.comboBoxComPort);  

170    this.Controls.Add(this.chkTrace);  

171    this.Controls.Add(this.Comport);  

172    this.Controls.Add(this.picASCOM);  

173    this.Controls.Add(this.Title);  

174    this.Controls.Add(this.cmdCancel);  

175    this.Controls.Add(this.cmdOK);  

176    this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;  

177    this.MaximizeBox = false;  

178    this.MinimizeBox = false;  

179    this.Name = "SetupDialogForm";  

180    this.SizeGripStyle = System.Windows.Forms.SizeGripStyle.Hide;  

181    this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;  

182    this.Text = "GS-touch_Setup";  

183    ((System.ComponentModel.ISupportInitialize)(this.picASCOM)).EndInit();  

184    this.ResumeLayout(false);
185    this.PerformLayout();
186
187 }
188
189 #endregion
190
191 private System.Windows.Forms.PictureBox picASCOM;
192 private System.Windows.Forms.Button cmdOK;
193 private System.Windows.Forms.Button cmdCancel;
194 private System.Windows.Forms.Button Reconnect;
195 private System.Windows.Forms.Label Title;
196 private System.Windows.Forms.Label Comport;

```

```
197     private System.Windows.Forms.Label MS_mode;
198     private System.Windows.Forms.ComboBox comboBoxComPort;
199     private System.Windows.Forms.ComboBox comboBoxMS;
200     private System.Windows.Forms.CheckBox chkTrace;
201     private System.Windows.Forms.CheckBox showUI;
202 }
203 }
```

References

- [1] Weber, L., & Brady, S. (2001). Fast auto-focus method and software for ccd-based telescopes. In *Minor Planet Amateur/Professional Workshop*, (pp. 104–113).
- [2] Specifications of the takahashi fsq-106ed refractor. <https://www.takahashi-europe.com/en/FSQ-106ED.specifications.php>. Accessed: 2020-07-26.
- [3] Persha, G. C. (2001). Temperature compensating focuser for telescope. US Patent 6,327,081.
- [4] Zhang, G.-y., Wang, J., Tang, P.-y., Jia, M.-h., Chen, J., Dong, S.-c., . . . & Zhang, H.-f. (2016). An autonomous observation and control system based on EPICS and RTS2 for Antarctic telescopes. *Monthly Notices of the Royal Astronomical Society*, 455(2), 1654–1664.
- [5] Budding, E. (1995). A global network of small automated telescopes. *Astrophysics and Space Science*, 228(1-2), 299–307.
- [6] Godoy, R., Fernández, G., Aballay, J., Collado, O., Fernández, C., & Ruartes, H. (2018). Control remoto de telescopios. *60*, 62.