|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **교과목 명** | | 디지털 시스템 설계 | | | | | | | |
| **설계 제목** | | LED / 7-Segment Animation | | | | | | | |
| **설계 기간** | | 2015년도 2학기 | | | | | | | |
| **지도교수** | | 김태환 | | | | | | | |
| **팀원** | | **이름** | 박종현 | **학번** | 2012122112 | **☎** | 010-9179-3566 | **E-mail** | cww77889@naver.com |
| **이름** | 이수연 | **학번** | 2012122200 | **☎** | 010-5129-7753 | **E-mail** | syiag@hanmail.net |
| **이름** | 김신재 | **학번** | 2012122043 | **☎** | 010-7755-4633 | **E-mail** | Johan327@naver.com |
| **목표설정** | **설계 목표** | 1. 3개의 애니메이션 패턴을 7-segment와 LEDR로 구현한다.  1) 첫 번째 애니메이션 패턴은 7-segment로 분수를 표현한다.   |  |  |  | | --- | --- | --- | |  |  |  |   첫 번째 애니메이션은 7-segment를 이용하여 분수를 표현하고자 했다. 처음에 설계할 애니메이션은 우리가 이번 프로젝트에서 어느 정도 구현할 수 있을지에 대한 잣대로 사용하기 위해 간단하지만 그렇다고 쉽지만 않은 패턴을 찾다가 이 패턴을 설계하기로 했다.  2) 두 번째 애니메이션 패턴은 LEDR로 뱀이 지나가는 것을 표현한다.   |  |  | | --- | --- | |  |  |   두 번째 애니메이션은 LEDR의 동작으로만 구성하였다. 패턴은 뱀이 왔다갔다하는 모습을 구현하고자 하는데 그 이유는 LEDR로 표현할 수 있는 애니메이션 패턴이 너무 한정적이라고 생각하여 디지털 논리회로 기초 중에 존슨 계수기를 응용해보았다.  3) 세 번째 애니메이션 패턴은 7-segment와 LEDR을 이용하여 음향막대를 표현한다.   |  |  | | --- | --- | |  |  |   세 번째 애니메이션은 7-segment와 LEDR이 같이 동작하도록 설계 해봤다. 음향막대를 구현하고자 하는 이유는 7-segment와 LEDR이 같이 동작할 때 서로의 동작이 가장 잘 어울릴 것 같아서 음향막대를 선택하였다.  4) 각각의 애니메이션 패턴은 SW[0-2]로 선택하도록 설정해준다.  2. 애니메이션의 속도를 조절할 수 있도록 한다.  1) clock이 들어가는 변수를 파라미터로 설정한다.  2) KEY3는 파라미터 값이 내려가도록 하여, 속도가 올라가도록 설정한다.  3) KEY2는 파라미터 값이 올라가도록 하여, 속도가 내려가도록 설정한다.  3. 리모컨과 연동하여 리모컨으로도 애니메이션 패턴을 바꾸고 애니메이션의 속도를 조절할 수 있도록  설정한다. | | | | | | | |
| **설계 규격** | De1-soc 보드의 7segment와 led로 애니메이션을 나타낸다. 애니메이션은 조별로 임의 것을 선정한다. 애니메이션은 DE1-SOC보드의 SW0, SW1, SW2를 사용하여 3개 중 하나를 선택한다. 설계 과정 중 이 3개 이외의 애니메이션을 추가 할 수 있으면 이 또한 역시 DE1-SOC보드의 나머지 SW를 사용하여 선택할 수 있어야 한다. 이렇게 선택된 애니메이션은 각각이 다른 모양의 형태로 7-segment와 LED를 통해 구현된다. 초기 RESET을 실행하는데 이는 DE1-SOC보드의 KEY0버튼을 이용한다. 이렇게 초기에 RESET을 실시하면 각각의 애니메이션은 초기 상태로 되돌아 갈 수 있어야 한다. 각각 구현되는 애니메이션들은 DE1-SOC보드의 KEY3 버튼과, KEY2버튼을 사용하여 재생되는 속도를 조절할 수 있어야 한다. KEY3 버튼을 눌러서 애니메이션의 재생 속도를 점진적으로 증가시킬 수 있다. KEY2버튼을 눌러서 애니메이션의 재생 속도를 점진적으로 감소 시킬 수 있다. 리모트 컨트롤의 기능을 구현할 수 있어야 한다. 리모컨을 사용하여 speed up과 speed down을 컨트롤 할 수 있어야 하면 또한 3개의 애니메이션 중 1개를 선택하는 기능을 리모컨을 사용하여 리모트 컨트롤 할 수 있어야 한다. 리셋의 기능은 리모컨을 사용한 기능에서 제외하면 보드의 버튼만을 이용한다.  애니메이션 1  SW0를 사용하여 애니메이션 1을 선택한다. 총 6개의 segment를 이용하여 분수가 나오는 모양의 애니메이션을 나타낸다. 가장 왼쪽부터 켜지기 시작하여 오른쪽으로 진행되는 단계를 나타낸다. Led는 segment의 높이에 따라 같이 켜지고 꺼지도록 한다. 애니메이션은 시작과 동시에 자동으로 반복되고 Reset 버튼은 애니메이션을 초기화 하는데 쓰인다. KEY3버튼과 KEY2버튼을 이용하여 애니메이션의 속도를 조절한다.  애니메이션 2  Led를 이용한 애니메이션으로 led가 순서대로 모두 켜진 후 켜진 순서에 따라 꺼지는 모양의 애니메이션을 나타낸다. Led는 미리 설정해 놓은 순서대로 켜지도록 한다. Led가 모두 켜지면 동시에 세 번 꺼졌다 켜지고 정지 한 후 켜진 순서대로 꺼지도록 한다. SW1 스위치를 이용하여 스위치가 올라가면 led가 켜지기 시작하고 애니메이션 실행 도중에 스위치를 다시 내리면 켜지던 led 모두 꺼지도록 한다. Led가 켜지고 꺼지는 속도는 KEY3 버튼과 KEY2 버튼을 이용하고 리모컨을 이용하여 조절할 수 있도록 한다.  애니메이션 3  음향믹서를 led로 나타낸다. Led가 켜지고 꺼지는 범위를 조절하여 음향의 크기를 나타내는 것처럼 보이도록 설정한다. | | | | | | | |
| **합성/분석** | **관련 기술** | 최근 디지털 테크놀로지가 발전하면서 디지털 기기를 사용하는 분야가 기존 분야에서 확대되기 시작했다. 이런 영역 중 애니메이션 분야와 예술 분야를 들 수 있다. 최근에는 디지털 애니메이션, 디지털 예술이라는 이름으로 기존 아날로그 방식 또는 기존의 창작 방식으로 작성되던 애니메이션 또는 예술 영역의 작품들이 컴퓨터와 같은 디지털 기기를 사용하여 작성 되기 시작하였다. 또 이러한 분야 이외에도 길거리 상점의 광고판을 보더라도 일반 그림 간판 보다는 LED등을 사용한 움직이는 로고 형태의 광고 판을 많이 접할 수 있다. 이러한 배경을 바탕으로 DE1-S0C보드의 7-segment와 LED를 사용한 애니메이션 제작이라는 프로젝트를 설계하게 되었다.  이러한 기능을 리모컨으로서 조절 할 수 있는데 이에 관련한 배경에는 'IR Receiver - 적외선 수신 모듈'이 있다. IR Receiver는 적외선을 이용한 리모트 컨트롤에 사용되는 적외선 수신 모듈로서 일반적인 TV리모컨 등에 사용되고 있다. IR Receiver의 사용 범위는 다양하여 적외선 송신부와 수신부 사이에 장애물의 유,무를 검출하는 센서로도 사용 될 수 있다. 이는 DE1-SOC보드 상에도 탑재되어 있는데 이를 이용하여 리모트 컨트롤 기능을 구현한다.  베릴로그를 이용하여 코드를 작성한다. 애니메이션1은 가장 왼쪽의 segment부터 오른쪽으로 진행하는 모양을 코드로 구성한 뒤 pin매칭을 하여 각각의 7segment에 맞게 출력되도록 한다. 애니메이션의 속도는 파라미터를 설정하고 key를 이용하여 빠르게 하거나 느리게 조절할 수 있도록 한다. | | | | | | | |
| **제작** | **설계 계획** | 1. 개요  Quartus를 사용하여 Verilog코드로서 DE1-SOC보드 상의 7-segment와 LED상에서 재생되는 애니메이션을 설계한다. 이를 구현하기 위해 DE1-SOC에서 사용할 핀에 대해서 Pin Assignment를 실행하여 코드와 보드의 연결을 만든다. Quartus를 사용하여 DE1-SOC보드의 7-segment와 LED를 제어하는 코드를 만든다. Animation은 일단 3개를 구현하되 추가적으로 첨가할 수 있다. 애니메이션을 구현하기 위해 DE1-SOC의 7-segment 6개와 LED10개를 사용한다. DE1-SOC의 SW2-0를 통해서 3개의 애니메이션 중 1개를 선택하여 결과로 출력한다. DE1-SOC의 KEY0버튼을 사용하여 Reset기능을 만든다. 추가적인 기능으로서는 DE1-SOC의 KEY3과 KEY2 버튼으로 애니메이션의 재생 속도를 조절하는 기능을 만든다. 그리고 DE1-SOC의 리모컨을 사용하여 위 기능 중 속도 조절 기능과 애니메이션 선택기능을 조작할 수 있도록 한다.  2. 애니메이션의 구현  애니메이션을 구현하기 위해 DE1-SOC의 7-segment 6개와 LED 10개를 사용한다. 기본적으로 애니메이션은 clock신호에 동기화 하여 작동하게 한다. 7-segment를 사용하는 애니메이션과 LED를 사용하는 애니메이션은 동시에 한 애니메이션을 나타내도록 구현하는 애니메이션을 1개이상 만들고, 각각 7-segment와 LED를 사용하는 애니메이션 2개를 만들 계획이다. 두 경우 모두 clock의 2진수 비트 수에 따라 해당 순간의 각 경우에 대해서 어떤 LED 또는 어떤 7-segment가 켜질 지를 결정하는 방식으로 애니메이션을 구현한다.  Clock 신호로는 DE1-SOC의 50MHz신호를 이용하여 이를 원하는 주기의 clock신호로 바꾼다. 이 신호로 동기화 되는 애니메이션을 만들기 위해 각 애니메이션에 필요한 clock신호를 각각 50MHz클럭으로부터 산출하여 각 애니메이션이 각각의 clock신호를 갖게 한다. 처음 계획은 3개의 애니메이션을 만드는 것이므로 3개의 독립된 clock신호를 만들고 이들 각각이 KEY3버튼과 KEY2버튼으로 속도 조절이 가능 하게 설계한다. 즉 clock1, clock2, clock3를 50MHz로부터 원하는 길이의 주기를 갖는 clock으로 산출하여 사용한다. 이 주기는 이후에 KEY3버튼과 KEY2버튼을 사용하여 조절 함으로서 애니메이션이 더 빠르게 동작하거나 더 느리게 동작하도록 설계한다. 위 세 개의 clock을 산출하는 계획은 다음에 기술한다.  HEX5-0의 6개의 7-segment를 사용하는 애니메이션은 decoder 모듈을 사용한다. 각각의 애니메이션의 clock은 처음 4비트의 신호로 설계하고, 해당 애니메이션이 동작할 때, 그 애니메이션의 clock신호를 decoder의 input으로 주어 16가지 경우에 대해 case문을 사용하여 7segment의 동작을 기술 한다. 이 16가지 경우를 조합하여 애니메이션을 만든다. 각 경우는 clock주기의 어느 한 순간을 나타내게 된다. 프로젝트를 진행하면서 애니메이션의 구현에 있어 좀더 많은 case의 수를 갖는, 즉 부분 동작이 많은 애니메이션을 구현할 필요가 있을 경우 clock의 비트 수를 상황에 맞게 4비트에서 좀더 확장할 수 있다. 이에 따른 decoder모듈의 case문은 clock의 각각의 case를 좀더 첨가하여 구현한다. 즉 5비트의 clock으로 바뀔 경우 decoder의 case문은 16가지에서 32가지의 case를 다루는 case문으로 바뀌고, 각 clock의 순간마다 어떤 7-segment의 어떤 부분이 켜질 지를 결정한다. 이런 식으로 clock의 비트 수를 확장하면 좀 더 많은 부분 독작을 갖는 애니메이션 구현이 가능하므로 구현하고자 하는 애니메이션의 특성에 맞게 clock의 비트 수를 결정하여 코딩한다. 또는 이후 속도 조절 모듈을 구현하면서 clock의 비트수를 KEY3버튼과 KEY2버튼을 사용하여 늘리거나 줄일 수 있다.  LED를 사용하는 애니메이션을 구현 할 때에는 코드상으로 해당 애니메이션의 clock의 2진수 값에 따라 탑모듈 상에서 always문과 case문을 사용하여 clock의 각 시간대 별로 켜지거나 꺼질 LED를 결정한다. 다시 말해서 탑 모듈에 led0~led9까지 10개의 신호를 선언하여 output으로 설정한 뒤에 이들의 값을 해당 애니메에션의 전용 clock의 이진수 값에 따른 case로 나누어 해당 순간 켜질 LED와 꺼질 LED를 assign함으로서 애니메이션의 부분 동작을 결정한다. 이렇게 함으로서 매 순간 10개의 LED의 상태가 어떻게 될지 결정하게 된다. 여기에서도 마찬가지로 부분 동작이 많은 애니메이션을 설계하게 될 경우는 해당 애니메이션 전용 clock의 비트 수를 확장하고 탑 모듈 상의 LED를 기술하는 case문의 case를 추가 함으로서 좀더 자연스러운 애니메이션을 설계할 수 있다.  3. 속도의 조절  50MHz의 DE1-SOC 클럭 신호로부터 애니메이션의 수만큼 새로운 클럭 신호를 만든다. 이 각각의 클럭신호는 해당 애니메이션을 동기화 시킨다. 이러한 애니메이션의 전용 클럭은 주기가 고정 된 것이 아니라 KEY3버튼과 KEY2버튼을 사용하여 주기를 변화 시켜 애니메이션의 재생 속도를 조절한다. 애니메이션은 그 애니메이션 전용의 클럭을 갖게 되므로 애니메에션 재생시간 동안 한 애니메이션의 속도를 조절해도 다른 애니메이션의 클럭 신호에 영향을 주지 않게 된다. KEY3버튼과 KEY2버튼을 사용하여 주기를 변화시키는 방법은 n개의 비트를 갖는 클럭의 와이어 신호 counter0를 선언하고 50MHz 클럭의 상승 엣지 마다(20ns) 1씩더하여 n개의 비트를 모두 채워 overflow가 발생하면 counter0가 1이 되도록 하는 방법을 선택한다. 이 때 overflow를 인식하기 위해 NOR게이트로서 출력을 산출한다. Overflow인 경우는 n개의 비트가 꽉 차서 다시 0000…….0000와 같이 n개의 0이 되는 순간을 말한다. KEY3버튼과 KEY2버튼을 사용하여 n의 값의 크기를 변경하므로서 클럭 주기의 조절이 가능하다. 즉 20ns마다(50MHz클럭의 주기) 채워야 할 비트수를 KEY3버튼과 KEY2버튼을 사용하여 정해 줌으로서 overflow발생까지 얼마가 걸릴지에 따라 여러가지 새로운 주기를 갖는 클럭 신호를 여러 개 만들 수 있게 된다.  4. 애니메이션의 선택  애니메이션은 3개중 1개를 선택하여 보여주게 설계한다. 이러한 애니메이션의 선택은 DE1-SOC보드 상의 스위치인 SW0, SW1, SW2의 3개의 스위치를 사용한다. Quartus의 Assignment Editor상에서 pin assignment를 하여 해당 스위치를 사용할 수 있는 핀으로 선정한다. 코드 상에서 이러한 애니매에션의 선택은 다음과 같이 기술 할 수 있다.  always@\*  case({sw1,sw2,sw3})  ({1'b1,1'b0,1b'0}): //…………………..  ({1'b0,1'b1,1b'0}): //…………………..  ({1'b0,1'b0,1b'1}): ; //…………………..  default : //…………………..  endcase  즉 always문과 case문을 사용하여 3개의 스위치의 상태에 따른 decoder에 전달될 입력을 선택하는 것이다. SW1이 1, 즉 올라가 있고 나머지 SW2와 SW3 스위치가 내려가 있는 상태라면 1번 애니메이션을 선택하는 경우이므로 7-segment의 출력을 담당하는 decoder모듈에 1번 애니메이션의 clock신호를 입력으로 주면 된다. decoder모듈에서는 이 1번 애니메이션의 클럭 신호를 각 순간마다 해석하여 7-segment에 출력할 결과를 산출 하게 된다. 다시 말해서 1번 애니메이션의 동작을 표현하게 된다. SW1와 SW3가 0, 즉 내려가 있고 나머지 SW2가 1, 즉 올라가 있는 상태라면 바로 2번 애니메이션을 실행하겠다는 신호이므로 7-segment를 나타낼 decoder모듈의 입력에는 2번 애니메이션의 클럭 신호가 입력 되게 된다. 따라서 7-segment에는 2번 클럭에 입력에 따른 시간대별로의 애니메이션의 부분동작이 표현 됨으로써 2번 애니메이션의 동작이 7-segment에 표현되게 된다. 3번 애니메이션을 동작 시키는 3개의 스위치 입력 조합 즉 SW1=0, SW2=0, SW3=1의 case에 대해서도 같은 원리도 동작하게 된다. 위 3개의 스위치의 조합 즉 1번, 2번, 3번 애니메이션 중 하나를 고른 경우가 아니라 예를 들어 SW1과 SW3이 동시에 올라간 SW1=1, SW2=0, SW3=1인 경우 등은 4비트의 수 0을 decoder의 입력으로 줌으로서 7-segment에는 아무것도 출력 되지 않게 설계한다. 코드 상에서는 case문의 default구문을 사용하여 이러한 예외 입력을 처리한다.  5. 리모트 컨트롤  DE1-SOC보드 상의 3개의 스위치와 speed up버튼, speed down버튼을 DE1-SOC의 리모컨을 사용하여 기능을 구현한다. 즉 3개의 애니메이션 중 1개를 선택하는 기능과 속도를 조절하는 기능을 보드상의 스위치와 버튼뿐만 아니라 리모컨으로도 조작이 가능하도록 설계한다. 이를 구현하기 위해서 DE1-SOC보드의 IR Receiver핀을 사용한다. Quartus의 Assignment Editor에서 이 IR Receiver의 핀을 assign후에 리모컨을 통해 IR Receiver로 전달 되는 신호를 case별로 해석하여 각 경우 별로 어떤 기능의 신호인지를 판단하여 기존 코드 상에 기존 기능과 같은 역할을 하도록 코드를 추가한다.  각 기능 별 신호를 구별해 내기 위해 oversampling을 이용한다. DE1-SOC의 50MHz신호로 계속해서 IR Receiver를 통해 들어오는 신호를 계속 sampling함으로서 신호가 0 또는 1로 지속되는 시간을 detect한다. 리모컨으로 전달되는 신호는 NEC IR TRANMISSON PROTOCOL을 따라 전송되기 때문에 이를 참고하여 순서대로 들어오게 되는 신호의 형태에 따른 state별 의미의 순서를 생각 하기 위해 FSM을 사용하여 전달 되는 신호를 해석한다. | | | | | | | |
| **시험/평가** | **검증 계획** | 먼저 DE1-SOC보드에 출력 상태가 원하는 애니메이션을 출력하는지 확인하는 것이 첫 번째이다. 만약에 출력상태가 원하는 대로 나오지 않는다면 코드를 잘못 설계한 건 아닌지, pin mapping에 문제가 없는지, DE1-SOC보드가 고장난건 아닌지 확인하도록 한다. DE1-SOC보드의 출력 상태가 원하는 대로 나온다면 KEY를 이용한 speed up과 speed down이 제대로 이루어지는지 확인한다.  Speed up/down의 경우도 작동이 제대로 안 된다면 코드상의 문제가 아닌지 확인한다. 50MHz clock이 26bit counter를 다 채우게 되면 보드에 clock이 들어오는 형식의 코드를 쓰기 때문에 speed up/down의 경우 코드상의 문제일 확률이 높다.  다음 애니메이션 패턴을 바꾸는 경우, SW의 입력으로 패턴이 바뀌기 때문에 제대로 작동이 안될 경우 speed up/down과 마찬가지로 코드상의 문제일 확률이 높다. 따라서 KEY와 SW입력은 설계된 코드를 잘 봐줘야 한다.  리모컨의 연동은 pin mapping과 코드 설계상황을 동시에 봐줘야 한다.  애니메이션1은 7-segement를 이용한 분수 패턴이여서, DE1-SOC보드에서의 출력 상태가 물처럼 부드럽게 이어지는지 확인해야 한다. 애니메이션2는 LEDR을 이용한 존슨 계수기 패턴이다. 따라서 뱀이 왔다갔다 하는 것처럼 보여지는지 확인해야 한다. 애니메이션3은 7-segment와 LEDR을 이용한 가장 복잡한 설계이다. 라디오나 컴퓨터에 나오는 음향 막대처럼 보여져야 하고, 아무래도 가장 복잡한 설계이기 때문에 코드가 가장 복잡할 것 이다. | | | | | | | |
| **일정** | | |  |  |  | | --- | --- | --- | | 주차 | 진행 계획 | 예상 산출물 및 Demo 내용 | | 1 | 1번 애니메이션을 완성한다. 스위치 SW0를 사용해 애니메이션 1을 선택한다. reset버튼 기능을 구현한다. | SW0를 사용하여 애니메이션 1을 선택한다. 총 6개의 segment를 이용하여 분수가 나오는 모양의 애니메이션을 나타낸다. 가장 왼쪽부터 켜지기 시작하여 오른쪽으로 진행되는 단계를 나타낸다. Led는 segment의 높이에 따라 같이 켜지고 꺼지도록 한다. 애니메이션은 SW1스위치를 올림과 동시에 자동으로 반복되고 Reset 버튼은 애니메이션을 초기화 하는데 쓰인다. | | 2 | 2번 애니메이션과 3번 애니메이션을 완성한다. 완성된 3개의 애니메이션을 SW0, SW1, SW2를 사용하여 선택하는 기능을 완성한다. | 2번 애니메이션은 led가 순서대로 모두 켜진 후 켜진 순서에 따라 꺼지는 모양의 애니메이션을 나타낸다. Led가 모두 켜지면 동시에 세 번 꺼졌다 켜지고 정지 한 후 켜진 순서대로 꺼지도록 한다. SW1 스위치를 이용하여 스위치가 올라가면 led가 켜지기 시작하고 애니메이션 실행 도중에 스위치를 다시 내리면 켜지던 led를 정지 시킨 후 켜진 순서대로 꺼지도록 한다.  애니메이션 3은 음향믹서를 7segment와 led로 나타낸다. 7segment와 Led가 켜지고 꺼지는 범위를 조절하여 음향의 크기를 나타내는 것처럼 보이도록 설정한다. SW2를 사용하여 애니메이션을 선택 한다. SW2가 올라간 상태이면 애니메이션 3이 실행되다가 SW2를 내리면 모든 애니메이션이 종료되고 아무것도 표시하지 않는다. | | 3 | 속도 조절 기능과, 리모트 컨트롤 기능을 완성한다. | 3개의 애니메이션 모두 KEY3버튼을 누르면 점진적으로 재생 속도가 증가한다. KEY2버튼을 누르면 3개의 애니메이션 모두 점진적으로 재생 속도가 감소한다. 리모트 컨트롤 기능을 구현하여 speed up, speed down, 3개의 애니메이션 중 1개를 선택 하는 기능을 확인한다. | | | | | | | | |
| **역할 분담** | | 1. 김신재  - project의 애니매이션 선택 기능, 속도 조절 기능, 리모트 컨트롤 기능 등 각 기능을 verilog코드로 구현할 방법을 모색하고 아이디어를 산출하는 역할을 담당한다. 애니메이션 선택의 기능, 속도 조절의 기능, 리모트 컨트롤을 project의 기능으로 삽입할 방법 등의 아이디어를 고안한다. 소스코드 작성 작성이 과중 할 경우 다른 조원을 도와 직접 코드를 작성하는 역할을 담당한다. 보드의 실장 결과를 시험하며 작성 된 목표에 부합 되는지 분석하는 역할을 담당한다.  2. 이수연  - 7-segment와 LED상에 나타낼 애니메이션의 구상을 담당한다. 여러 가지 경우의 애니메이션 동작을 제시함으로서 실현 가능한 애니메이션을 산출한다. 시뮬레이션과 DE1-SOC보드에 실장하여 실험한 결과를 기반으로 애니메에선의 보완할 점과 잘 된 점을 분석하고 다음 애니메이션 작성시 이를 참고 할 수 있도록 정리하는 역할을 담당한다. 최종 시연 결과 동영상을 촬영하고 편집을 담당한다. 매주 project의 구현을 위해 조별로 실시하는 회의진행 후 작성되는 회의록의 작성을 담당한다. 소스코드 작성 작성이 과중 할 경우 다른 조원을 도와 직접 코드를 작성하는 역할을 담당한다.  3. 박종현  - 코드의 작성과 모델심을 통한 시뮬레이션 시행을 담당한다. 구현 방법의 아이디어를 바탕으로 이를 실재 코드로 기술하는 역할을 담당한다. 이렇게 작성 된 코드를 모델심을 통해 시뮬레이션 해 봄으로서 결점이나 보완해야 할 부분 또는 실수한 부분 등을 찾아내어 개선 할 방법을 모색 하는 역할을 담당한다. 매주 실시하는 project의 기능 구현 및 개선을 위한 회의를 주최하고 주관하는 역할을 담당한다. 속도 조절 기능, 리모트 컨트롤 기능 등을 추가 함에 있어 다른 조원의 업무과 과중 할 시 다른 조원을 도와 기능을 코드화 하는 아이디어 구상 작업에 참여한다. | | | | | | | |