1. 자동화 스크립트 실행 시 변경 부분
   1. **BasePath 설정**

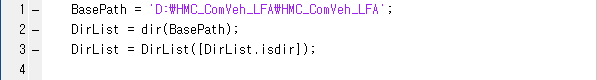


그림 . 폴더 경로 설정

automation.m의 첫 번째 줄의 BasePath에는 PreScan 시나리오 파일들이 저장된 폴더의 주

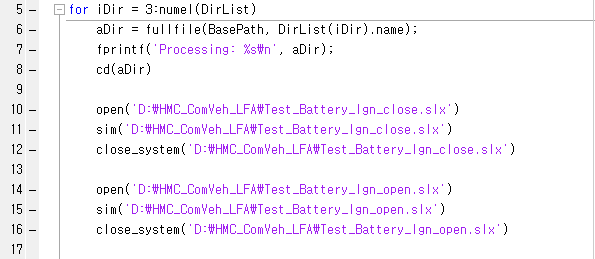
소를 입력한다.

* 1. **변수 초기화**

아래 그림 2의 ①, ②는 각각 배터리 on / off하는 시뮬링크 파일로서, 각 시나리오 폴더에 넣을 필요없이, 시나리오 폴더의 상위폴더에 파일을 두면 된다.

그 후, 10 ~ 12번째 줄에는 배터리를 꺼주는 시뮬링크 파일 (Test\_Battery\_Ign\_close.slx) 위치를 입력해준다.

마찬가지로 14 ~ 16번째 줄에는 배터리를 켜주는 시뮬링크 파일 (Test\_Battery\_Ign\_open.slx) 위치를 입력해준다.



①

②

그림 . 배터리 on / off

* 1. **시나리오 네이밍**

아래 그림 3의 24번째 줄의 ExperimentName은 맨 처음 실행시킬 시나리오의 이름을 입력한다.

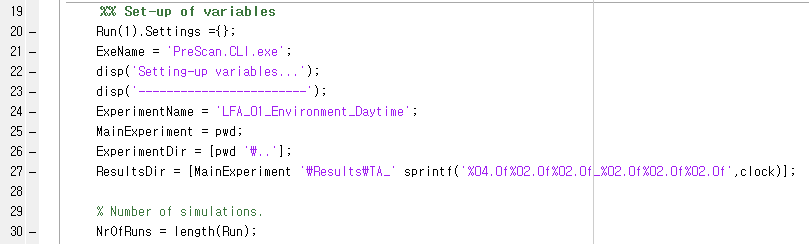


그림 . 시나리오 자동화 초기 폴더 생성

1. 자동화 스크립트 코드 상세 설명
   1. **BasePath 설정 (Automation.m 1 ~ 3 line)**

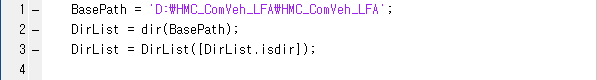
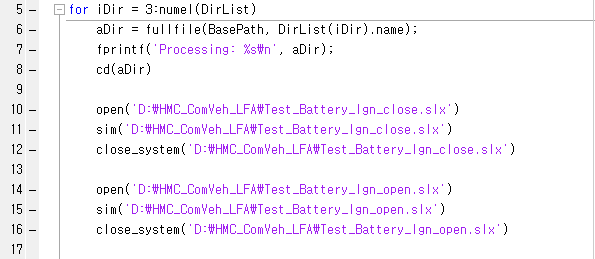


그림 . 폴더 경로 설정

automation.m의 첫 번째 줄의 BasePath에는 PreScan 시나리오 파일들이 저장된 폴더의 주소를 입력한다.

* 1. **시나리오 시뮬링크 변수값 초기화 (Automation.m 10 ~ 16 line)**

매 시나리오 시작 전마다 ‘Test\_Battery\_Ign.slx’ 시뮬링크 파일을 실행함으로써 B+와 IGN을 껐다 켬으로서, 시나리오 시작 전 매트랩 Workspace의 변수에 해당하는 값을 초기화를 해주는 역할이다.



①

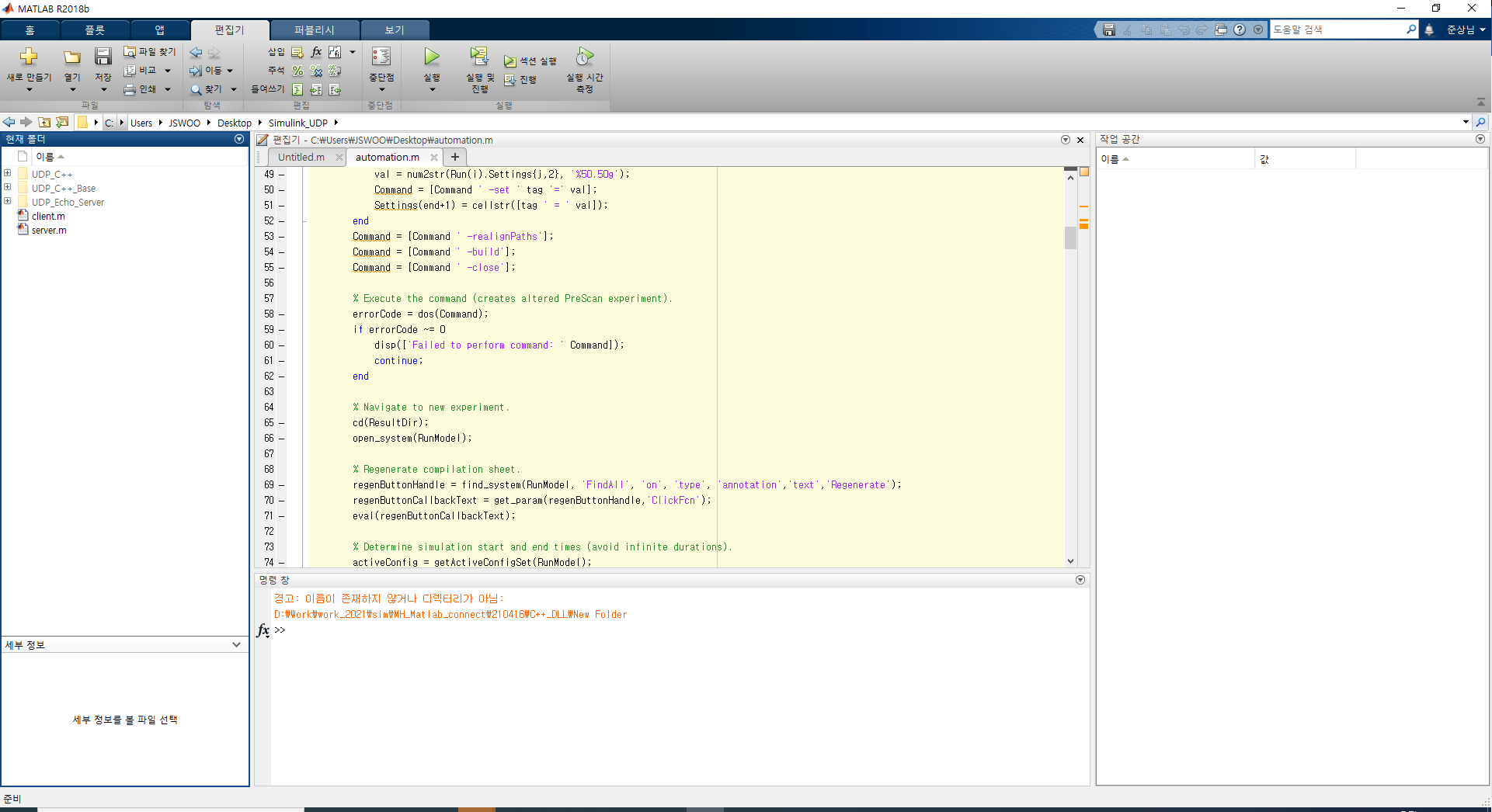
②

그림 . 배터리 on / off

1. 시나리오 시작 전 배터리를 꺼주며 기존 시나리오 데이터 삭제
2. 배터리를 켜주며 변수 초기화

|  |
| --- |
| **Test\_Battery\_Ign.slx**  open(‘…\Test\_Battery\_Ign.slx’) : ‘Test\_Battery\_Ign.slx’ 시뮬링크 파일을 열어준다.  sim(‘…\Test\_Battery\_Ign.slx’) : ‘Test\_Battery\_Ign.slx’ 시뮬링크 파일을 실행하여 B+, IGN을 껐다가 켜주는 역할을 한다.  close\_system(‘…\Test\_Battery\_Ign.slx’) : ‘Test\_Battery\_Ign.slx’ 실행 후 닫아준다.  아래 그림의 사각 박스에서 ‘Test\_Battery\_Ign.slx’의 실행 시간 설정을 해줄 수 있다. |

* 1. **시뮬레이션 파일 저장 폴더 설정 (Automation.m 20 ~ 27 line)**
     1. **기본 설정**



작업공간 (Workspace)

그림 . 작업공간 (Workspace) 위치

위 그림 6은 시나리오 시뮬링크 파일 실행 후 저장된 변수 값들을 확인할 수 있는 작업공간을 캡처한 것이다.

스크립트 3번째줄의 DirList는 첫 번째 줄의 BasePath로 설정한 폴더의 하위 폴더 및 파일들을 의미한다.

작업공간에 저장되는 DirList의 내용은 아래 그림 7과 같다.

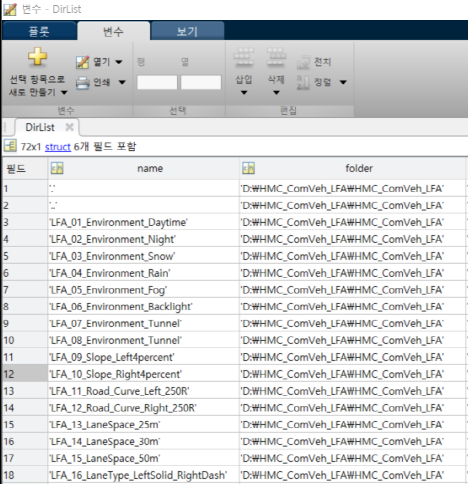


그림 . 변수 DirList (BasePath 하위 시나리오 폴더)

DirList의 항목들은, 작업공간에 저장되어 있는 DirList 변수를 더블클릭하면 확인 가능하다.

자동화 스크립트 5번째 줄 for iDir = 3:numel(DirList)의 3:numel(DirList)는 DirList의 처음으로 시작되는 하위폴더인 ‘LFA\_01\_Environment\_Daytime’부터 마지막 폴더까지 지정하는 범위이다.

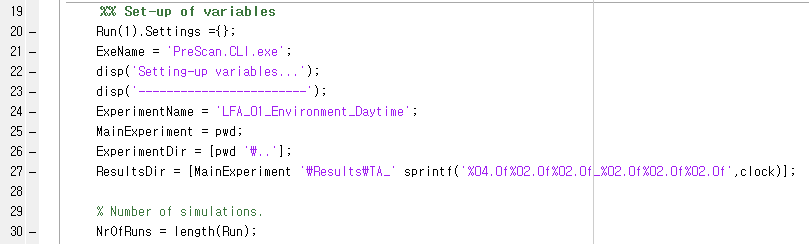


그림 . 시나리오 자동화 시 생성되는 폴더 및 폴더 네이밍 구조

자동화 스크립트 실행시키기 위해서는 20번째 줄의 ExperimentName은 여러 개의 시나리오 중 제일 먼저 실행시킬 시나리오 파일 명을 적어준다.

여기서는 LFA를 기준으로 했으므로 아래 그림과 같이 ExperimentName은 ‘LFA\_01\_Environment\_Daytime’으로 적어주었다.

그림 8의 스크립트 27번째 줄과 같이 ResultsDir은 해당 시나리오 폴더 안, ‘…\Results\TA\_시뮬레이터 실행 시간’에 해당하는 폴더가 생성된다.

예시) D:\HMC\_ComVeh\_LFA\LFA\_01\_Environment\_Daytime\Results\TA\_20210607\_144633과 같이 시나리오 실행 시간 (분초 단위) 폴더가 생성된다. 아래 그림 10 참조.

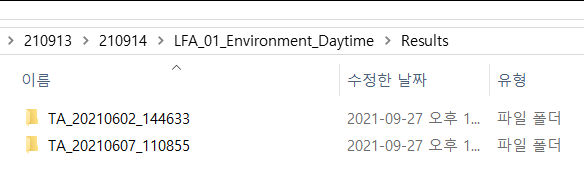


그림 . 해당 시나리오 실행 시 결과 저장되는 폴더 경로 및 이름

* + 1. **추가 설정 1 – 실행했던 시나리오 다음 시나리오부터 실행 설정하기**

만약, ‘LFA\_12\_Road\_Curve\_Right\_250R’부터 마지막까지 순차 실행하고 싶다면

1. 위 그림 6의 작업공간에서 DirList 변수 이름을 더블클릭하여 변수 내용을 확인한다.
2. 그림 7과 같이 DirList 변수 내용 창이 활성화되면 ‘LFA\_12\_Road\_Curve\_Right\_250R’에 해당하는 필드 열의 숫자인 12를 그림 2와 같이 자동화 스크립트의 5번째 줄 for iDir = 12:numel(DirList)과 같이 숫자 변경을 해준다.
3. 그림 8의 자동화 스크립트 24번째 줄의 ExperimentName을 LFA\_12\_Road\_Curve\_Right\_250R로 변경해준다.
   * 1. **추가 설정 2 – 특정 시나리오만 간소화 후 실행 설정하기**

순차적으로 자동화 실행이 아닌 특정 몇 개를 실행하고 싶다면,

1. 특정 시나리오 폴더를 넣은 상위 폴더를 생성한다.
2. Test\_Battery\_Ign\_close.slx, Test\_Battery\_Ign\_open.slx 시뮬링크 파일을 특정 시나리오 폴더들이 복사되어 있는 폴더로 복사한다

예) D:\HMC\_ComVeh\_LFA\LFA\_211013과 같이 폴더를 생성 후 특정 시나리오 폴더, Test\_Battery\_Ign\_close.slx, Test\_Battery\_Ign\_open.slx를 복사해서 넣는다.

1. 그림 4의 첫 번째 줄의 BasePath를 D:\HMC\_ComVeh\_LFA\LFA\_211013로 변경한다.
2. 특정 몇 개의 시나리오이므로 그림 5의 5번째 줄의 내용은 for iDir = 3:numel(DirList)과 같이 적어준다.
3. 그림 5와 같이 자동화 스크립트 24번째 줄의 ExperimentName을 특정 시나리오 중 맨 처음 시나리오 이름으로 변경해준다.
   1. **시뮬레이션 실행 (Automation.m 32 ~ 100 line)**

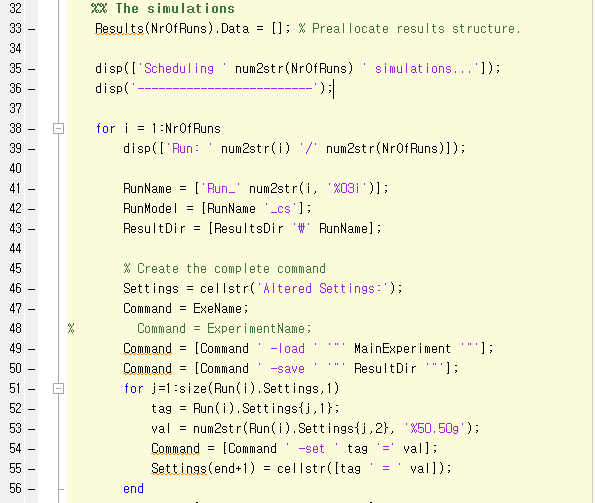


그림 . 저장되는 프리스캔 파일 및 시뮬링크 파일 네이밍

그림 7의 39 ~ 43 번째 줄의 명령어를 통해 해당 시나리오의 시뮬링크 파일을 ‘Run\_01\_cs’ 이름의 시뮬링크 파일로 Results 폴더 안에 그림 6의 폴더 이름과 같이 생성된 폴더 -> Run\_001 폴더에 PreScan 시나리오 파일 (.pex), 시나리오 시뮬링크 파일 (.slx), 시뮬링크 변수 파일 (.mat) 파일 등이 저장된다 (그림 14 참조).

그림 5의 17 번째 줄의 ExeName = ‘PreScan.CLI.exe’ 명령어를 통해 PreScan GUI가 실행되며, 아래 그림 8의 과정으로 해당 시나리오 PreScan 파일이 새로 Build 과정을 거치게 된다.

그림 9의 69 ~ 70번째 줄과 같이 새로 생성된 PreScan 파일이 있는 경로로 변경되며, 같은 경로에 PreScan 파일 빌드 후 생기는 시뮬링크 파일이 실행 및 업데이트가 진행된다 (73 ~ 80번째 줄).

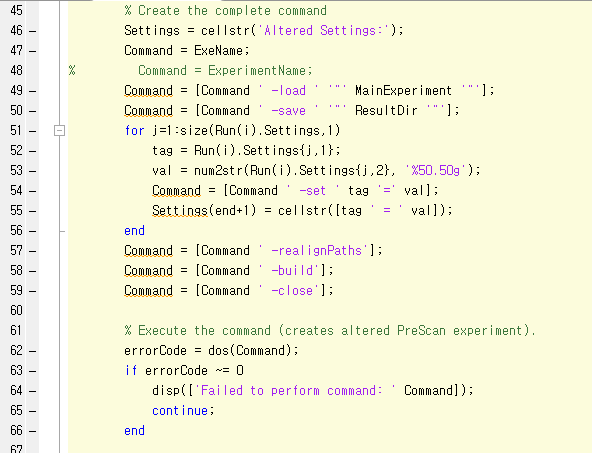


그림 . 자동화 실행 시, 해당 시나리오 PreScan 파일 새로 생성

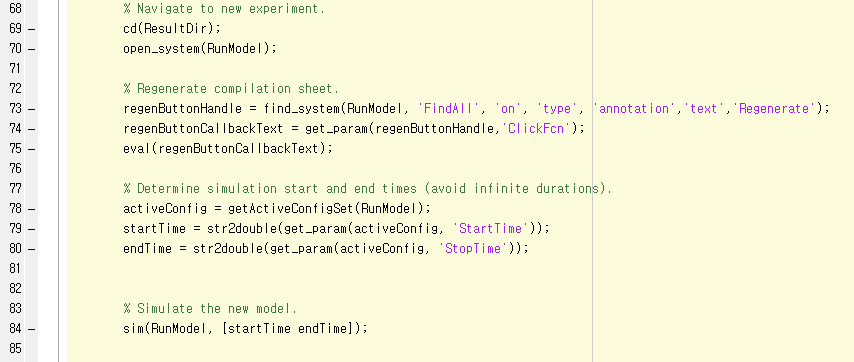


그림 . 자동화 실행 시, 해당 시나리오 시뮬링크 파일 새로 생성

이후 새로 생성된 ‘Run\_01\_cs.slx’ 시뮬링크 파일이 있는 디렉토리로 현재 실행 위치가 변경 (69번째 줄)되며, …\TA\_...\Run\_001 폴더안의 ‘Run\_01\_cs.slx’ 시뮬링크 파일을 실행 (84번째 줄)된다.

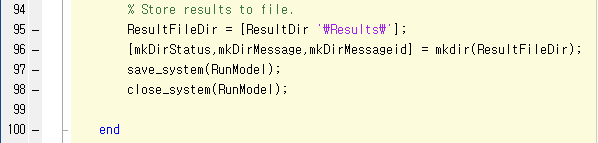


그림 . 시나리오 종료 후 결과 파일이 저장되는 위치

시나리오 종료 후 그림 10의 95번째 줄의 경로 (..\TA\_...\Run\_001) 폴더 안에 결과파일들이 저장된다.

그 후 97 ~ 98번째 줄의 내용과 같이 시나리오 실행 시뮬링크 파일의 결괏값들을 저장하고 종료한다.

그림 11은 결괏값들이 저장되는 폴더 안 구성 요소들이다.

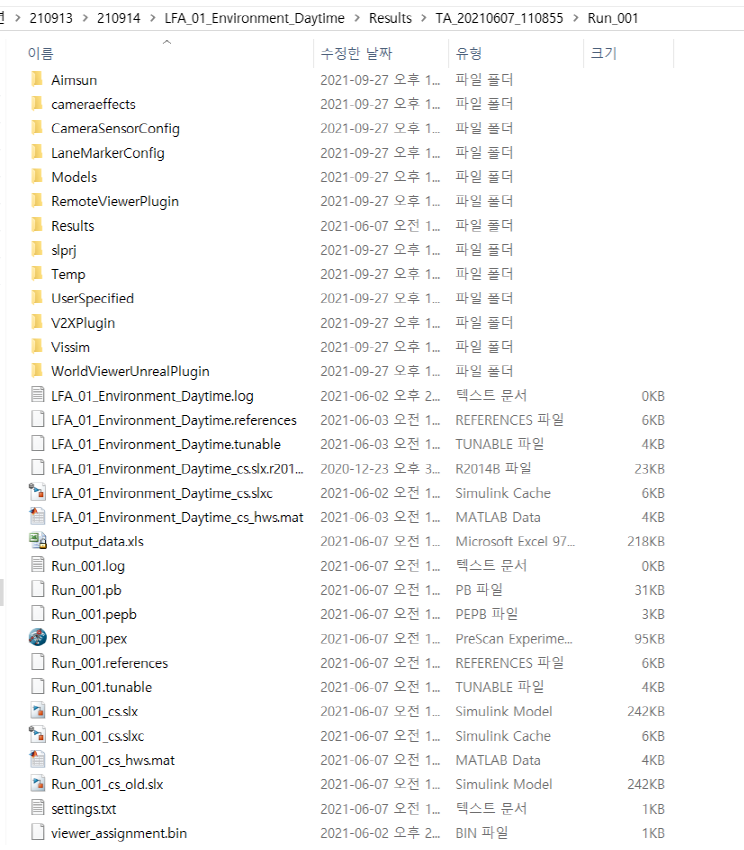


그림 . 자동화 스크립트로 시나리오 실행 시 저장되는 최종 구성요소들

* 1. **데이터 후처리 (Automation.m 105 ~ 636 line)**

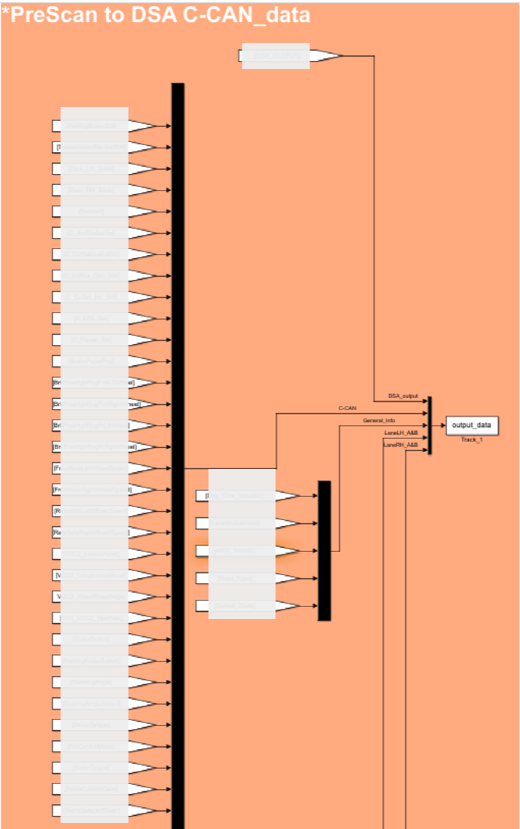


그림 . 데이터 저장 시그널 및 저장 관련 시뮬링크 블록

시나리오 시뮬링크와 DSA 제어기 간의 통신으로 출력되는 P-CAN 및 C-CAN의 데이터들을 엑셀로 변환하는 과정이다.

그림 12와 같이 여러 시그널들을 모아 output\_data 블록을 통해 데이터를 한 곳으로 통합 할 수 있다.

신호 블록을 연결한 순서대로 output\_data.signal.value에 저장된다.

변수 이름은 Goto  / From  블록으로 설정 가능하다.

* + 1. **데이터 샘플 타임 변경**

시나리오 주행 데이터의 저장 주기를 바꾸고 싶다면, 시나리오 시뮬링크 파일의  아이콘(To Workspace)을 찾아 더블클릭 해준다.

아래 그림 12와 같이 활성화되면 Sample time 란의 값을 원하는 시간으로 변경해줄 수 있다.

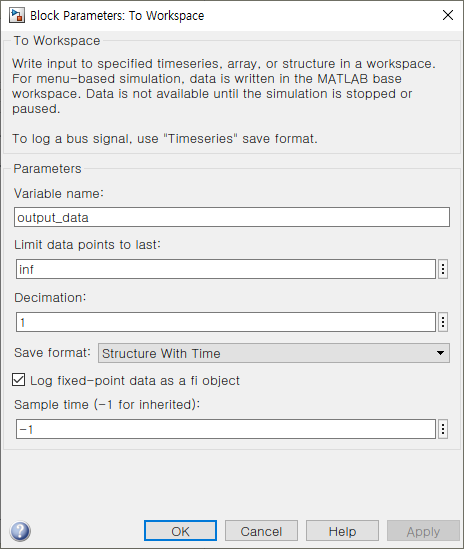


그림 . 데이터 저장 주기 설정

예시) 0.5초 단위로 데이터 저장을 하고싶다면, Sample time에 0.5로 입력하면 된다.

|  |
| --- |
| **주의사항**  DSA 제어기로 송출하는 PreScan의 데이터를 출력해주는 To Workspace에서 Sample time이 위의 그림과 같이 -1 (시나리오 시스템과 동일한 Sample time)일 경우 작업공간 (Workspace)에 쌓이는 데이터 양이 65,565 행 이상이 되어 엑셀로 변환이 안되는 현상 발생.  LFA 시나리오는 카메라 frame rate (16Hz)로 인해 샘플링 타임이 1/320 (0.003125)이나 엑셀로 데이터 출력을 위해 샘플링 타임의 10배인 0.03125로 적용  SCC\_일반 / 컷인 시나리오는 레이더의 Sweep rate (10Hz)로 인해 샘플링 타임이 1/300 (0.00333)이나 엑셀로 데이터 출력을 위해 샘플링 타임의 10배인 0.03으로 적용  시나리오 시작 전 받고자 하는 샘플링 타임 설정 후 시나리오 실행해야한다. |

* + 1. **데이터 저장 및 매트랩에서 확인하는 방법**

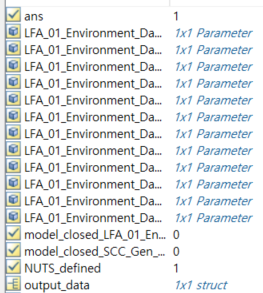


그림 . 작업공간에서의 output\_data 변수

저장된 output\_data는 그림 14에서와 같이 작업공간에 output\_data 변수를 더블클릭하여 변수 창을 활성화한다.

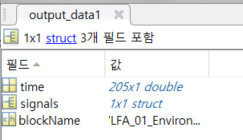


그림 . 활성화된 output\_data 변수 창

시뮬레이션 시간은 위 그림 15의 time 변수, 각 신호의 내용은 signals를 더블클릭하여 확인할 수 있다.

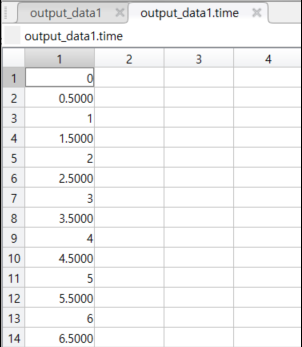


그림 . output\_data의 시간 변수

그림 16은 output\_data 변수 안 time 필드 값을 더블클릭하면 나타나는 창이다.

데이터 저장 주기는 그림 13에서 sample time 값을 바꿔줄 수 있다.

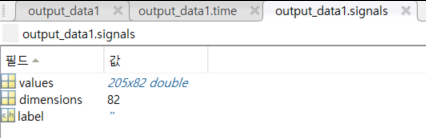


그림 . output\_data의 시그널 변수

각 신호에 대한 값을 확인하기 위해 그림 15의 signals를 더블클릭하면 그림 17처럼 value, dimensions, label 항목이 있다.

Values는 각 신호에 대한 결괏값을 의미하며, dimensions는 현재 시뮬레이션 결과 받았던 시그널의 갯수를 의미한다.

그러므로 그림 17에서 확인 가능 정보는 받은 시그널의 갯수는 82개의 시그널을 받았고, 시뮬레이션 시간은 102초임을 알 수 있다. (204개의 행 (0s인 1행 제외) \* 0.5s = 102s )

Values를 더블클릭하면 아래 그림 18번과 같이 결괏값을 확인할 수 있다.

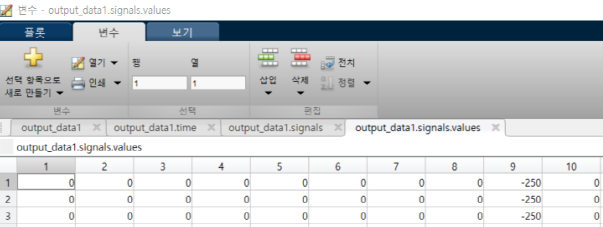


그림 . 각 신호에 대한 변수 결괏값

그림 18의 각 시그널은 그림 19의 108 ~ 110번째 줄에 해당하는 변수의 값이다.

1번 열 : ACCDistanceAlertSignal

2번 열 : ACCMode

3번 열 : ACCSetDistanceMode

주의할 점

output\_data.signals.values의 1열은 시간 열이 아니라, 첫 번째로 받는 시그널이다.

여기서는 DSA 제어기에서 받는 값들을 나타낸다.

즉, DSA 제어기로부터 받는 신호 중 첫번째 신호인 ACCDistanceAlertSignals 신호의 결괏값을 의미한다.

* + 1. **저장 데이터 엑셀로 변환**
       1. **저장 데이터 엑셀로 변환 시 결괏값 변수화**

그림 19의 105번째 줄과 같이 filename에다가 원하는 이름을 넣고 .xls를 붙여준다.

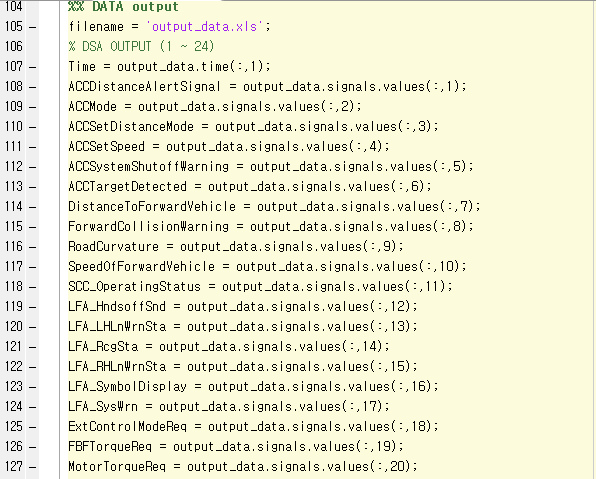


그림 . 데이터 저장 시그널 목록

파일 이름은 문자열 형태 (string)로 인식하므로 반드시 ‘ ’ (작은따옴표)를 붙여주어야 한다.

그림 19의 107번째 줄은 output\_data의 시간에 해당하는 열, 107번째 줄부터 마지막 줄까지는 그림 12에서 설정된 시그널들의 결괏값이다.

output\_data.signals.values(:, 1)의 콜론은 해당 시그널의 값을 전부, 1에 해당하는 값은 받고자 하는 시그널을 의미한다.

* + - 1. **엑셀로 변환하기 위한 각 신호들의 이름 설정**

Results\_Name에는 각 시그널에 해당하는 이름을 지정할 수 있다.

문자로 받아들일 수 있도록 변수 이름 앞 뒤에 반드시 ‘변수이름’과 같이 (‘) 작은 따옴표를 붙여주어야 한다.

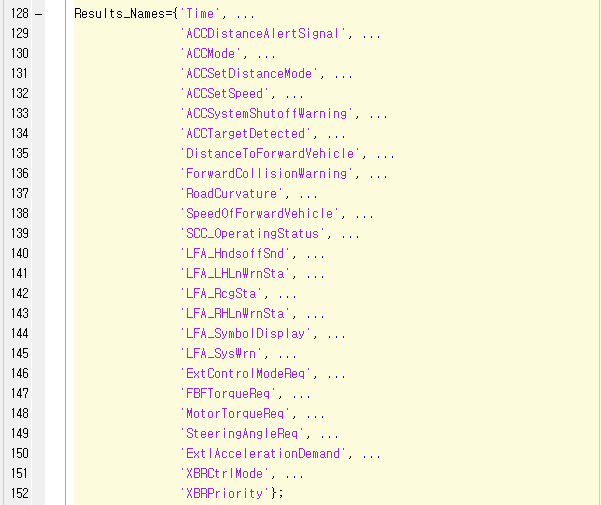


그림 . 엑셀에 저장할 때 1행에 저장될 변수 이름 목록

* + - 1. **엑셀로 변환하기 위한 각 신호들의 변수 설정**

Results\_Values에는 엑셀로 저장될 시그널을 설정해줄 수 있다.

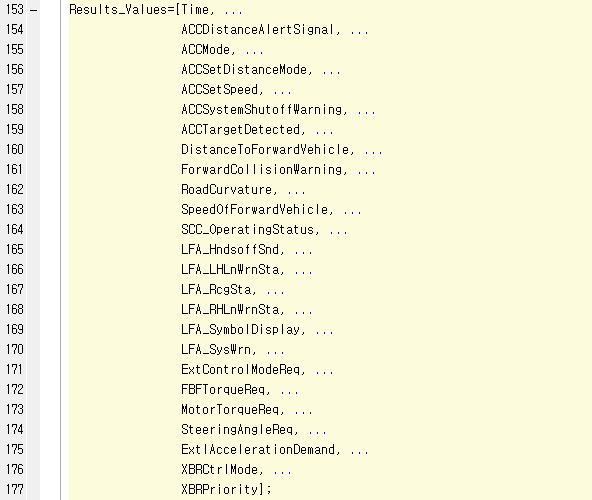


그림 . 엑셀에 저장할 때 2행부터 저장될 변수의 결괏값 목록

* + - 1. **작업공간의 결괏값 엑셀로 변환**

sheet=1; - 엑셀 시트의 몇 번째 탭에 저장할 것인지를 입력할 수 있다.

xlRange=’A2’; – 지정한 엑셀 시트의 원하는 셀부터 값을 저장할 수 있다.

아래 그림에서는 엑셀 시트 첫 번째 탭, 1행에는 각 변수들의 이름을, 2행부터는 각 변수들의 값을 저장한다.

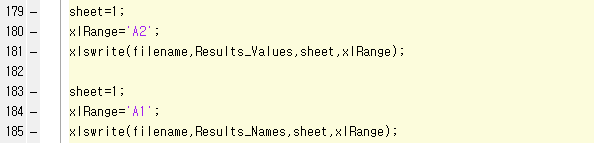


그림 . 엑셀 시트에 결괏값 저장

엑셀로 변환 후

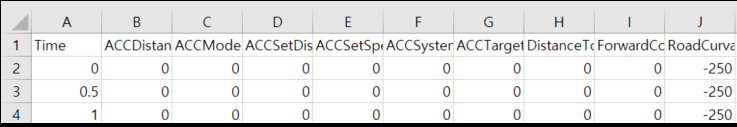


그림 . 엑셀에서 출력되는 결괏값들

그림 23의 Results\_Names 코드는 위 그림 26의 1행에 해당하는 각 시그널들의 이름이다.

그림 24의 Results\_Values 코드는 위 그림 26의 2행부터 시작되는 각 시그널들의 결괏값이다.

* + 1. **특정 시그널만 추출해서 저장하는 방법**

현재 자동화 스크립트에는 현 과제에서 받아야하는 시그널 전부 저장하도록 설정되어있다.

특정 시그널만 추출하고자 하는 경우,

1. 그림 23의 6 ~ 8번째 줄과 같이 해당 시그널에 해당하는 시그널 이름, 열의 번호를 확인한다.
2. 그림 23의 10 ~ 12번째 줄과 같이 Result\_Names에는 {} (셀 형태) 안에 시간과 받고자 하는 시그널의 이름을 문자열 형태로 입력한다.
3. 그림 23의 14 ~ 16번째 줄과 같이 Result\_Values에는 [] (배열형태) 안에 시간과 받고자 하는 시그널의 이름을 입력한다.
4. 그림 23의 19, 23번째 줄과 같이 엑셀에 저장할 시트 번호를 입력한다.

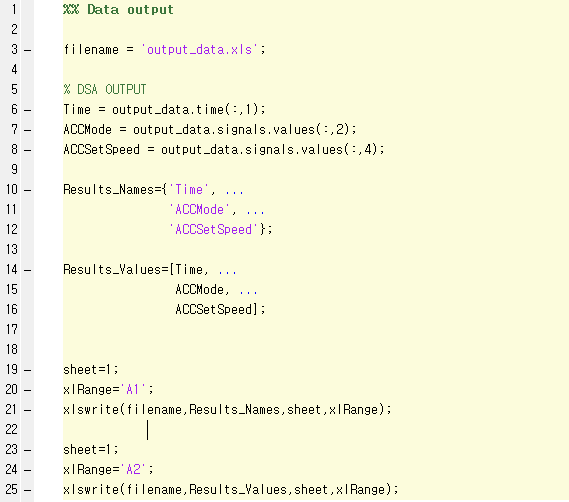


그림 . 특정 시그널 추출하는 경우