|  |  |
| --- | --- |
| ADVANCED SAFETY AND BODY CONTROL MODULE  MILESTONE IV  SUBMITTED BY: DAYA MODEKAR (EMP ID: 142900)  POOJA KHANE (EMP ID: 142923)  PRATIK MALODE (EMP ID: 142924)  PRACHURJYA SHARMA (EMP ID: 142925) |  |
|  |  |



KPIT Technologies Ltd.

CONTENTS

[1 ABSTRACT 5](#_Toc37050549)

[2 INTRODUCTION 5](#_Toc37050550)

[ADVANCED AIRBAG SYSTEM 5](#_Toc37050551)

[SEAT BELT ALERT SYSTEM 6](#_Toc37050552)

[SEMI ACTIVE SUSPENSION SYSTEM 7](#_Toc37050553)

[PASSIVE KEYLESS ENTRY 8](#_Toc37050554)

[3 REQUIREMENTS 9](#_Toc37050555)

[ADVANCED AIRBAG SYSTEM 9](#_Toc37050556)

[HIGH LEVEL AND LOW LEVEL REQUIREMENTS 9](#_Toc37050557)

[MARKET ANALYSIS 9](#_Toc37050558)

[S.W.O.T. ANALYSIS 12](#_Toc37050559)

[4W&1H 12](#_Toc37050560)

[SEAT BELT ALERT SYSTEM 13](#_Toc37050561)

[HIGH LEVEL AND LOW LEVEL REQUIREMENTS 13](#_Toc37050562)

[MARKET ANALYSIS 13](#_Toc37050563)

[S.W.O.T. ANALYSIS 14](#_Toc37050564)

[4W & 1H 14](#_Toc37050565)

[SEMI ACTIVE SUSPENSION SYSTEM 15](#_Toc37050566)

[HIGH LEVEL AND LOW LEVEL REQUIREMENTS 15](#_Toc37050567)

[MARKET ANALYSIS 16](#_Toc37050568)

[SEGMENT ANALYSIS 17](#_Toc37050569)

[AGEING 17](#_Toc37050570)

[S.W.O.T. ANALYSIS 17](#_Toc37050571)

[4W & 1H 18](#_Toc37050572)

[PASSIVE KEYLESS ENTRY 18](#_Toc37050573)

[HIGH LEVEL AND LOW LEVEL REQUIREMENTS 18](#_Toc37050574)

[MARKET ANALYSIS 19](#_Toc37050575)

[S.W.O.T. ANALYSIS 22](#_Toc37050576)

[4W&1H 22](#_Toc37050577)

[4 LITERATURE REVIEW 24](#_Toc37050578)

[AIRBAG SYSTEM 24](#_Toc37050579)

[SEAT BELT ALERT SYSTEM 25](#_Toc37050580)

[SEMI ACTIVE SUSPENSION SYSTEM 25](#_Toc37050581)

[PASSIVE KEYLESS ENTRY 27](#_Toc37050582)

[5 UML DIAGRAM 28](#_Toc37050583)

[STRUCTURAL DIAGRAM 28](#_Toc37050584)

[PIN DIAGRAM 28](#_Toc37050585)

[BLOCK DIAGRAM 29](#_Toc37050586)

[BEHAVIOURAL DIAGRAM 32](#_Toc37050587)

[STATEFLOW DIAGRAM 32](#_Toc37050588)

[FLOWCHART 35](#_Toc37050589)

[6 BILL OF MATERIALS 39](#_Toc37050590)

[7 EXECUTION PLAN 40](#_Toc37050591)

[8 TEAM DESCRIPTION 40](#_Toc37050592)

[9 INTEGRATED MODELLING AND TESTING 40](#_Toc37050593)

[MODEL BASED DESIGN 40](#_Toc37050594)

[DEBUG USING SIMULINK METHODS 44](#_Toc37050595)

[TESTING USING SIMULINK TEST 45](#_Toc37050596)

[10 INTEGRATED TEST PLAN 51](#_Toc37050597)

[11 INTEGRATED COMPLIANCE 53](#_Toc37050598)

[MAAB CHECK 53](#_Toc37050599)

[CODE COVERAGE 55](#_Toc37050600)

[METRIC ANALYSIS 58](#_Toc37050601)

[12 LEARNINGS AND DIFFICULTIES OVERCOMED 59](#_Toc37050602)

[LEARNINGS 59](#_Toc37050603)

[DIFFICULTIES OVERCOMED 59](#_Toc37050604)

[TECHNICAL 59](#_Toc37050605)

[NON-TECHNICAL 59](#_Toc37050606)

[13 APPENDIX 61](#_Toc37050607)

[SIMULINK GENERATED EMBEDDED C CODE 61](#_Toc37050608)

**LIST OF FIGURES**

[Figure 1: Arduino Integrated PIN Diagram 29](#_Toc37050609)

[Figure 2: Integrated System Block Diagram 30](#_Toc37050610)

[Figure 3: Advanced Airbag Block Diagram 31](#_Toc37050611)

[Figure 4: Seat Belt Alert Block Diagram 31](#_Toc37050612)

[Figure 5: Semi-active Suspension Block Diagram 32](#_Toc37050613)

[Figure 6: Passive Keyless Entry Block Diagram 32](#_Toc37050614)

[Figure 7: Integrated System State flow Diagram 33](#_Toc37050615)

[Figure 8: Advanced Airbag State flow Diagram 34](#_Toc37050616)

[Figure 9: Seat Belt Alert State flow Diagram 34](#_Toc37050617)

[Figure 10: Semi-active Suspension State flow Diagram 35](#_Toc37050618)

[Figure 11: Passive Entry State flow Diagram 35](#_Toc37050619)

[Figure 12: Integrated System Flowchart 36](#_Toc37050620)

[Figure 13: Advanced Airbag Flowchart 37](#_Toc37050621)

[Figure 14: Seat Belt Alert Flowchart 38](#_Toc37050622)

[Figure 15: Semi-active Suspension Flowchart 39](#_Toc37050623)

[Figure 16: Passive Entry Flowchart 40](#_Toc37050624)

[Figure 17: Integrated Body and Safety Model in Simulink 41](#_Toc37050625)

[Figure 18: Main System Block 42](#_Toc37050626)

[Figure 19: Subsystem block and other feature enabled block 42](#_Toc37050627)

[Figure 20: Integrated Body and Safety Requirement Linking 43](#_Toc37050628)

[Figure 21: Main Block Signal Builder inputs 43](#_Toc37050629)

[Figure 22: Signal Builder inputs for Enabled system 44](#_Toc37050630)

[Figure 23: Integrated Output of All systems on scope 44](#_Toc37050631)

[Figure 24: Integrated Data inspector comparison 45](#_Toc37050632)

[Figure 25: Comparison of runs with Global Abs Tolerance at 10 45](#_Toc37050633)

[Figure 26: Comparison runs including increase in Abs and Rel Tolerance 46](#_Toc37050634)

[Figure 27: Toggle of input Signal switch to OFF 47](#_Toc37050635)

[Figure 28: Signal Builder Input from Excel file 47](#_Toc37050636)

[Figure 29: Signal Builder Harness Model 48](#_Toc37050637)

[Figure 30: Signal Builder Output 48](#_Toc37050638)

[Figure 31: Output Signal transition in Workspace as dataset 49](#_Toc37050639)

[Figure 32: Constant input to Main System 49](#_Toc37050640)

[Figure 33: Output signals when input is 99 50](#_Toc37050641)

[Figure 34: Output signals when input is 1000 50](#_Toc37050642)

[Figure 35: Constant inputs to Enabled system 51](#_Toc37050643)

[Figure 36: Output signals when constant input is given 51](#_Toc37050644)

[Figure 37: MAAB Result for the First Iteration 54](#_Toc37050645)

[Figure 38: MAAB Result after clearing warnings 55](#_Toc37050646)

[Figure 39: First Iteration of Code Coverage 56](#_Toc37050647)

[Figure 40: Second Iteration of Code Coverage 57](#_Toc37050648)

[Figure 41: Final Iteration of Code Coverage 58](#_Toc37050649)

[Figure 42: Metric Analysis 59](#_Toc37050650)

LIST OF TABLES

[Table 1: Advanced Airbag High and Low Level requirements 10](#_Toc37050853)

[Table 2: Advanced Airbag SWOT Analysis 13](#_Toc37050854)

[Table 3: Advanced Airbag 4W&1H 13](#_Toc37050855)

[Table 4: Seat Belt Alert High and Low Level requirements 14](#_Toc37050856)

[Table 5: Seat Belt Alert SWOT Analysis 15](#_Toc37050857)

[Table 6: Seat Belt Alert 4W&1H 15](#_Toc37050858)

[Table 7: Semi-active Suspension High and Low Level requirements 17](#_Toc37050859)

[Table 8: Semi-active Suspension Segment analysis 18](#_Toc37050860)

[Table 9: Semi-active Suspension Ageing analysis 18](#_Toc37050861)

[Table 10: Semi-active Suspension SWOT analysis 18](#_Toc37050862)

[Table 11: Semi-active Suspension 4W&1H 19](#_Toc37050863)

[Table 12: Passive Keyless Entry High and Low Level requirements 20](#_Toc37050864)

[Table 13: Passive Keyless Entry Segment analysis 23](#_Toc37050865)

[Table 14: Passive Keyless Entry Ageing analysis 23](#_Toc37050866)

[Table 15: Passive Keyless Entry SWOT analysis 23](#_Toc37050867)

[Table 16: Passive Keyless Entry 4W&1H 24](#_Toc37050868)

[Table 17: Seat Belt Alert Literature Review 26](#_Toc37050869)

[Table 18: Semi-active Suspension Literature Review 28](#_Toc37050870)

[Table 19: Bill of Materials 40](#_Toc37050871)

[Table 20: Team Execution Plan 41](#_Toc37050872)

[Table 21: Integrated Test Plan 53](#_Toc37050873)

[Table 22: Learnings 60](#_Toc37050874)

[Table 23: Technical Difficulties Overcomed 60](#_Toc37050875)

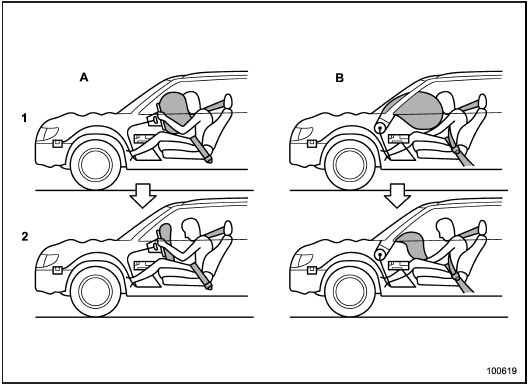
[Table 24: Non-technical Difficulties Overcomed 61](#_Toc37050876)

ADVANCED SAFETY AND BCM INTEGRAted system

# ABSTRACT

# INTRODUCTION

## ADVANCED AIRBAG SYSTEM

An airbag is an automotive safety restraint system for an occupant as well as passengers. The system consists of a flexible fabric envelope or cushion, designed to inflate rapidly during an automobile collision. Its purpose is to cushion occupants during a crash and provide protection to their bodies when they strike interior objects such as the steering wheel or a window etc. thus it lowers the number of injuries by reducing the force exerted by steering wheel, windows and the dashboard at any point on the body. continuing research and developments are going on in its module design, combustible material, air bag fabric design and material, coating etc. in making this life saving safety device further efficient. However, success of any safety restraint device depends on its correct implementation and certain safety rules to be followed.

The air bag system consists of three basic parts- an air bag module, crash sensor and a diagnosis unit. Some systems have on/off switch to deactivate air bag system. The air bag module contains both an inflator unit and the lightweight fabric air bag. The driver air bag module is located in the steering wheel hub, and the passenger air bag module is located in the instrument panel. When fully inflated, the driver air bag is approximately the diameter of a large beach ball. The passenger air bag can be two or three times larger since the distance between the right-front passenger and the instrumental panel is much larger than the distance between the driver and steering wheel. The crash sensors are located either in the front of the vehicle and/or in the passenger compartment. Vehicle can have one or more crash sensors. The sensors are activated by forces generated in significant frontal or near-frontal crashes only and not during sudden braking or while driving on rough or uneven pavement. The diagnostic unit monitors the readiness of the air bag system. The unit is activated when the vehicles ignition is turned on. If the unit identifies a problem, a warning light alerts the driver to service the air bag system before use. Most diagnostic units contain a device, which stores enough electrical energy to deploy the air bag if the vehicle battery is destroyed very early in a crash sequence. Vehicles like pickup trucks do not have rear seats; have manually operated on/off switches for the passenger air bags.

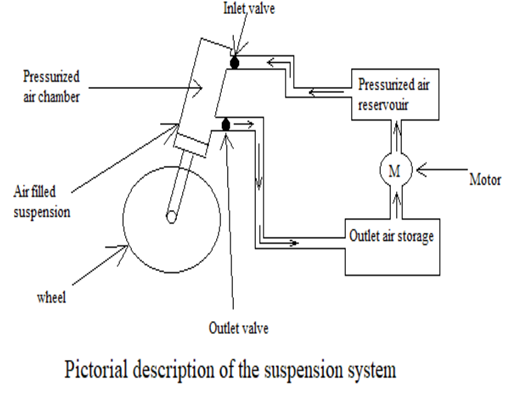
## Image result for seat belt alert display system in car imagesSEAT BELT ALERT SYSTEM

Most of the accidents are occurred because of violation of rules. Result of this major accidents happened. In our day-to-day life we are careless in our safety while driving in vehicles for this we have to introduce some techniques to do these precautions compulsory.

There are several causes of road accidents and one among the major reasons is people forget to wear seat belt. Initially if we want to solve this problem, we have to realize that to drive is not a play. We manage a machine which is very dangerous, that it could kill anybody and there are laws which we need to carry out so in order to make aware of seat belts, mandatory system has been introduced. This system basically aims at bringing the rule into act such that seat belt must be worn by every individual who drives the car.

Carelessness which has always been a part of driving system in our system and surrounding, such that people tend to defy wearing helmets in two wheelers and so in the case of car they tend not to wear seat belt. Seat belts have been designed on the basis to serve as a purpose of safety to safeguard people from accidents such as at the time of collision or during sudden braking thus wearing will always avoid people from getting injured at an adverse rate compared to not wearing them thus this topic will be a lifesaving and effective too.

## SEMI ACTIVE SUSPENSION SYSTEM

This project mainly focuses on the mid segment cars in Indian market and the motive behind developing this project is to enhance the safety in Indian cars without decreasing the overall functionality of the car. The plan is on increasing safety by controlling the body of specifically the car suspension which will in turn enhance the overall control and handling of the car.

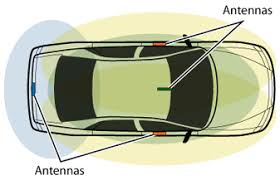
Recently the Indian consumers in automotive markets are emphasizing on safety, take the example of TATA NEXON which is the first car which is made in India who got 5 star NCAP rating and even though the car is more expensive than its competitors in its segment people are still opting for that car due to the safety provided by it.

To increase safety the focus is on control and handling of the car. This is done by softening or stiffening the suspension depending upon the mode in which the vehicle is currently operating and the acceleration or de-acceleration status.

These technologies allow car manufacturers to achieve a greater degree of ride quality and car handling by keeping the tires perpendicular to the road in corners, allowing better traction and control. An onboard computer detects body movement from sensors throughout the vehicle and, using that data, controls the action of the semi-active suspension. The system virtually eliminates body roll and pitch variation in many driving situations including cornering, accelerating, and braking.

## PASSIVE KEYLESS ENTRY

Security! A key component every customer and industry focuses on. The automobile industry deals with a lot of unauthorized access and hassle when it comes to security of an automobile. The ease of how an automobile security is dealt in this age of technology is a vast pool of technology and devices for just unlocking and doing other basic actions for ease of use.

Late 1900 has seen the use of electronics and highly new iterations of automobile show that the shift from analog and the self-control of electronics to a more sensor based and wireless ways of controlling the security and automobile lock feature. The system has its own perks and loopholes. Security is a major concern for customers and the industry has been working for more better and sophisticated systems.

The next generation saw the use of proximity sensors for the security feature of a vehicle using a button on the door called the request sensor system. This mandated the key fob be on the user whenever locking or unlocking the vehicle. The key would have to be in the vicinity for the request buttons to work for the feature it is assigned to.

The new system uses a more advanced sensor system to detect movement of the hand placement of the user on the door handle. This helps in unlocking the vehicle without any button press on the key fob or the request button system. This has a sophisticated system for security and also the ease of access to the vehicle. Also around this technology, the use of wrist band having the sensor system to lock and unlock the vehicle have been used by manufactures to attract customer showcasing the ease of access. The boot release system using the foot waving underneath the trunk also has been used by many manufacturers.

# REQUIREMENTS

## ADVANCED AIRBAG SYSTEM

### HIGH LEVEL AND LOW LEVEL REQUIREMENTS

|  |  |  |  |
| --- | --- | --- | --- |
| S.no | Stakeholder Requirements | System Requirements | Software Requirements |
| 1 | Alarm after 5sec when Seatbelt system integrated at seat back rest is not applied after ignition. | Buzzer, Atmega 328p, LED, weight sensor. | Arduino API,Matlab(Simulink) |
| 2 | When front airbag is triggered during the manual driving mode, the steering is retracted automatically near dash board. | Servo motor, crash detection sensor, accelerometer, airbag(led) | Arduino API,Matlab(Simulink) |
| 3 | Laterally positioned wing shaped airbags to also be triggered depending on impact level. | crash detection sensor,accelerometer,airbag(led) | Arduino API,Matlab(Simulink) |
| 4 | Facilitate seat belt tensioning during any crash. | crash detection sensor, accelerometer, seatbelt tensioner(led) | Arduino API,Matlab(Simulink) |

Table 1: Advanced Airbag High and Low Level requirements

### MARKET ANALYSIS

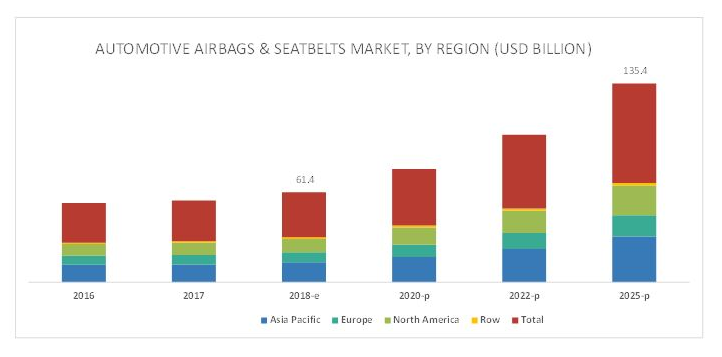
The automotive airbags & seatbelts market is estimated to be USD 55.77 billion in 2017 and is projected to reach 135.43 billion by 2025, at a CAGR of 11.97% during the forecast period. Increasing demand for safe, more efficient driving experiences, and growing stringency of safety regulations across the globe are playing a key role for driving airbags & seatbelts market. The base year considered for the study is 2017, and the forecast has been provided for the period between 2018 and 2025. The report analyses and forecasts the market size, in terms of volume (000’/million units) and value (USD million/ billion), of the automotive airbags & seatbelts market.

The automotive industry is witnessing a rapid evolution in safety features. The main objective behind the implementation of these features in vehicles is to provide a safer and more efficient and convenient driving experience. The increase in road accidents is a major concern for automobile manufacturers and governments. According to NHTSA, the total number of fatalities due to road accidents in the US in 2016 was 37,461, which grew by 5.6% from 2015. These statistics plainly illustrate the often-underestimated dangers inherent in driving a car. While these numbers are staggeringly high, they are fortunately in decline. Much of this decline in the frequency of vehicular crashes can be attributed to the widespread adoption of automotive safety systems, such as seatbelts and airbags.

Passenger car segment is estimated to be the fastest growing segment of the automotive airbags & seatbelts market, by vehicle. The number of passenger cars is growing at a significant rate in the emerging economies of the Asia pacific region. This can be attributed to the rise in GDP and the population of these countries, resulting in improved lifestyle, increased purchasing power of consumers, and development of infrastructure. Airbags and seatbelts are provided as a standard feature in passenger cars in most of the countries for preventing fatalities due to accidents.

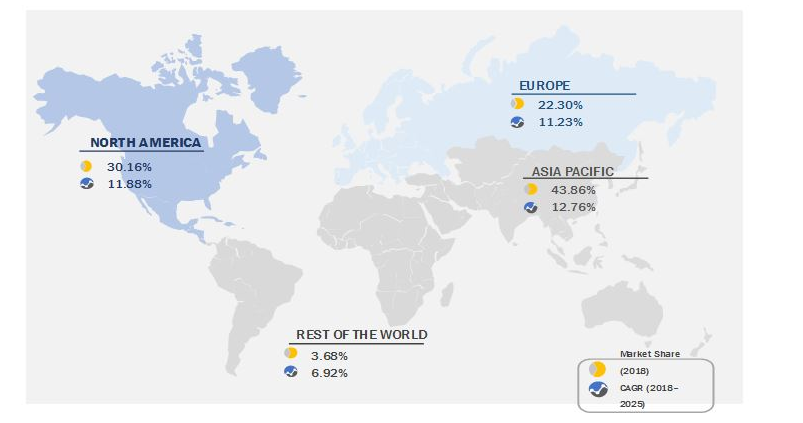
**“side & curtain airbags: the fastest growing segment of the automotive airbags & seatbelts market, by airbags type “**

Side & curtain airbags are the fastest growing segment of the automotive airbags & seatbelts market, by airbags type. Side airbags are designed to help protect an adult's chest in a serious side-impact crash. These are mounted on the side of the seat or on the door, usually on the roof rail above the side windows. In the event of a side-impact crash, curtain airbags are used to protect an adult's head. According to insurance institute for highway safety, head-protecting side airbags reduce driver fatality risk by 45%. This effectiveness will help to grow the side & curtain airbags market in near future.



the Asia pacific region is estimated to dominate the automotive airbags & seatbelts market, in terms of value, in 2018.the rising consumer income levels, increased vehicle production in developing countries such as India and china, and increase in the number of luxury vehicles in countries such as japan fuel the growth of the automotive airbags & seatbelts market in the Asia pacific region.

The Asia pacific region is estimated to hold the largest share, by value, of the automotive airbags & seatbelts market in 2018. The demand for automotive airbags & seatbelts in this region is triggered by increasing vehicle production in developing nations such as china and India and the increasing number of luxury vehicles in countries such as china and japan. This region is also the leading producer of automobiles in the world.



### S.W.O.T. ANALYSIS

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * Retractable steering to ensure max inflation of airbag * System ensures that occupant is Properly restraint with applied Seat belt tensioners | * During a crash, the airbag deployment can be harmful any child/occupant who is in close proximity to airbag deployment |
| OPPORTUNITY | **THREAT** |
| * Efficient ways can be taken in account for child occupant | * Fatal Injuries can occur when someone is very close to or direct contact with air bag module when it deploys |

Table 2: Advanced Airbag SWOT Analysis

### 4W&1H

|  |  |  |
| --- | --- | --- |
| S.no | Question | Description |
| 1 | What | Advanced Airbag System |
| 2 | Where | The system is situated in front of driver and as well other passengers |
| 3 | Why | To provide Safety to driver and other passenger in crash situations |
| 4 | When | External accidents leading to severe crash |
| 5 | How | Using various sensors to detect various occupants in vehicle along with crash detection sensors to make a decision on impact level and accordingly trigger the airbags and retract the steering |

Table 3: Advanced Airbag 4W&1H

## SEAT BELT ALERT SYSTEM

### HIGH LEVEL AND LOW LEVEL REQUIREMENTS

|  |  |  |  |
| --- | --- | --- | --- |
| STAKEHOLDER’S  REQUIREMENTS | DESCRIPTION | HARDWARE REQUIREMENTS | SOFTWARE REQUIREMENTS |
| Seatbelt warning and ignition control | System will turn off the ignition whenever the seat belts are not buckled and alert message will be displayed. | 1. ATMEGA 328p  2. Seat belt sensor (Reed sensor)  3. LCD display | 1.Embeeded C  2.Code blocks  3.SimilIDE |

Table 4: Seat Belt Alert High and Low Level requirements

### MARKET ANALYSIS

Growing awareness regarding the importance of implementing passive safety systems, such as airbags and seatbelts, among consumers is also likely to augment market expansion. Governments, especially in emerging nations including India, and China, are increasingly making amendments and introducing new regulations to ensure enhanced road safety in 2017, the Union Road Transport Ministry of India announced that cars manufactured from 1st July 2019 onwards must install essential safety features to curb the issues pertaining to over speeding and its consequences. These safety features include front airbags, seat belt reminders, and speed alert, and parking systems. On the other hand, factors such as production of counterfeit goods may hinder market growth as these pose a great threat to the sales, revenue, and brand image of key companies. According to the U.S. Federal Trade Commission, the global automotive parts industry claims to have lost billions of dollars due to the growing availability of counterfeit products.

### S.W.O.T. ANALYSIS

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * It reduces the likelihood of death done by the car accidents | * Installation of system is compulsory |
| OPPORTUNITY | **THREAT** |
| * Detection of seat belt closure of aircraft seat belt detection where ever used | * Any failure in seat belt sensor causes the entire system fails |

Table 5: Seat Belt Alert SWOT Analysis

### 4W & 1H

|  |  |
| --- | --- |
| QUESTION | DESCRIPTION |
| What | What we have made is mandatory seat belts in order to prevent accidents. The innovation is to provide seat belts using sensor circuits which are called as Ignition Inducing Seat belts using relay and micro controlling programming structure for automobiles |
| Where | Installation can be done in all the vehicles in which we require seat belt safety |
| Why | While driving car wearing seat belt is important that can save our life during accident periods but most of us are careless to wear seatbelt. Driver will be unable to start vehicle without wearing seat belt |
| When | When driver wants to start the engine |
| How | By buckling the seatbelt reed sensor detects presence of seatbelt. Now seat belt reed sensor transmits the signal to the Microcontroller |

Table 6: Seat Belt Alert 4W&1H

## SEMI ACTIVE SUSPENSION SYSTEM

### HIGH LEVEL AND LOW LEVEL REQUIREMENTS

|  |  |  |
| --- | --- | --- |
| STAKEHOLDERS REQUIREMENTS | HIGH LEVEL REQUIREMENTS | LOW LEVEL REQUIREMENTS |
| |  | | --- | | * The car should have two modes of driving. | | * Continuous monitoring of pressure should be there. | | * The car should have better comfort over city roads and should have better control over bumps and potholes. | | * The car should have better handling over highways. | | * It should continuously monitor the status of mode. | | * The driver should have a the total control in which mode he wants the car to be in. | | * The stiffness of the suspension should not change over time. | | * If in any case the system fails the driver should get a warning informing about it. | | * The transition between modes should be fast. | | * The car should have lesser braking distance over highways. | | * The car should have better acceleration over highways. | | |  | | --- | | * Mode select switch * Microcontroller | | * Pressure sensor * Logic level shifter for each solenoid to convert o/p from MC to 12V | |  | | * Solenoid switch one way valve * Pressurized gas reservoir | | * Gas chambered suspension * DC Motor * Indicator to indicate the current mode * Outlet gas storage * Power supply * Knob to select mode | | * Use of Atmega 328p to interact with the sensors. * Pressure sensor and accelerometer sensor. * Use of lamp or led to indicate the Ignition status and the mode status. |

Table 7: Semi-active Suspension High and Low Level requirements

### MARKET ANALYSIS

The low segment vehicles uses solid suspension with unsprang differential which is also known as LEAF SPRING suspension. They are cheap but at a cost of having lower spring rate also lower ride height adjustability.

The medium segment cars offer more versatility in spring rate or suspension rate and ride height adjustability by adding features like Hydro pneumatic suspension which was introduced in Citroën BX car. Also in cars belonging to a segment slightly lower than Citroën BX they have Coil over suspension also advanced coil over suspension such as nitrox suspension. These have better spring rate and height adjustability and reliability but at the cost of increased cost.

The highest segment cars offers suspensions with systems like active suspension control like in Ferrari 458 Italia which is an electromagnetic suspension which offers active suspension control. They also have added features like race coil over suspension, anti-roll over bar, wishbone suspension, sway bars, etc. These increases the handling and control of the car which ultimately makes the car of race specs but they are very expensive and the added components increases the overall maintenance cost.

In general as we go upwards from lower segment cars which has cheaper suspension system but has lower suspension rate or versatility to higher segment cars which has much more versatility the main factor which plays an important role is COST

### **SEGMENT ANALYSIS**

|  |  |  |
| --- | --- | --- |
| LOW SEGMENT | MID SEGMENT | HIGHER SEGMENT |
| * Coil springs are used which has fixed stiffness. | * Suspension having dampers whose stiffness can be changes are used. * Semi active suspension system is used here. | * Airbag suspension system with variable stiffness and height is present. * Active suspension system is used here. |

Table 8: Semi-active Suspension Segment analysis

### **AGEING**

|  |  |  |
| --- | --- | --- |
| BEFORE 10 YEARS | AFTER 5 YEARS | CURRENT TECHNOLOGY |
| * Vehicles had only coil over suspension which had poor performance and lacked versatility. | * Adaptive suspension which can change the vehicle height and suspension stiffness by image processing of road ahead. | * Suspension which can change height of vehicle and its stiffness is changed according to the mode it is currently operating is present. |

Table 9: Semi-active Suspension Ageing analysis

### S.W.O.T. ANALYSIS

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * It improves the ride quality increases vehicle handling improves traction. | * The vehicle cost increases. |
| OPPORTUNITY | **THREAT** |
| * By usage of image processing the suspension characteristics can be changed according to the road ahead. | * If the sensors fail or if the pressurized air reservoir gets empty it can lead to undesired results. |

Table 10: Semi-active Suspension SWOT analysis

### 4W & 1H

|  |  |
| --- | --- |
| WHAT *is the feature?* | WHERE *is the system used?* |
| This feature is the Semi active suspension system for improving the handling of the vehicle by changing suspension stiffness. | This feature is present in all vehicles and is one of the most important feature of a vehicle. |
| WHY *is it used?* | **WHEN** *was this technology introduced?* |
| To improve the ride quality, increase the handling and improve the traction of the vehicle. | This technology is present in market since the birth of vehicles. |
| HOW *does the feature work?* | |
| * The feature uses Pressure and accelerometer sensors to determine the pressure inside the suspension and the accelerometer reading. * Depending upon the pressure inside the suspension the pressure is regulated and brought to a fixed threshold valve. * And according to the input from the accelerometer valve if the vehicle is accelerating or de-accelerating the rear or the front suspension if stiffened respectively. | |

Table 11: Semi-active Suspension 4W&1H

## PASSIVE KEYLESS ENTRY

### HIGH LEVEL AND LOW LEVEL REQUIREMENTS

|  |  |  |  |
| --- | --- | --- | --- |
| SL.NO | STAKEHOLDER REQUIREMENTS | SYSTEM REQUIREMENTS | SOFTWARE REQUIREMENTS |
| 1 | The vehicle signal range should not set the lock unless the receiver goes out of range | Servo motor,Atmega 328p,Status change, Radio Transceiver | Arduino API,Matlab(Simulink) |
| 2 | The door signal goes on after the vehicle status changes | Servo motor, Signal sensor(led) | Arduino API,Matlab(Simulink) |
| 3 | When receiver comes from vehicle to door, the status changes after it reaches door signal | Door servo, led, buzzer | Arduino API,Matlab(Simulink) |

Table 12: Passive Keyless Entry High and Low Level requirements

### MARKET ANALYSIS

EARLY SYSTEMS

Early implementation used a remote function having the lock, unlock buttons. This technology was featured in early automobiles as a luxury feature for ease of access to the vehicle. Only high end vehicles of automobile industry at those times provided this wireless remote entry system. The system had a transmitter circuit, which was inside the vehicle, and a receiver end in the key fob. A button press would allow the radio signal from the transmitter to do the locking or unlocking feature.

The two most common remote keyless-entry devices are:

The fob that goes on user key ring to [lock and unlock](https://auto.howstuffworks.com/power-door-lock.htm) user car doors (Many of these fobs also arm and disarm a [car alarm system](https://auto.howstuffworks.com/car-alarm.htm)). The small controller that hangs off user [car's](https://auto.howstuffworks.com/car.htm) sun visor to open and close the garage door. Some [home security systems](https://auto.howstuffworks.com/burglar-alarm.htm) also have [remote controls](https://electronics.howstuffworks.com/remote-control.htm), but these are not so common. The fob that user carry on user keychain or use to open the garage door is actually a small radio transmitter. When user push a button on the fob, user turn on the transmitter and it sends a code to the receiver (either in the car or in the garage). Inside the car or garage is a radio receiver tuned to the frequency that the transmitter is using (300 or 400 MHz is typical for modern systems). The transmitter is similar to the one in a [radio-controlled toy](https://science.howstuffworks.com/rc-toy.htm).

In the very early days of garage door openers, around the 1950s, the transmitters were extremely simple. They sent out a single signal, and the garage door opener responded by opening or closing. As garage door openers became common, the simplicity of this system created a big problem -- anyone could drive down the street with a transmitter and open any garage door! They all used the same frequency and there was no security.

By the 1970s, garage door openers had gotten slightly more sophisticated and had a controller chip and a DIP switch. A DIP switch has eight tiny switches arranged in a small package and [soldered](https://science.howstuffworks.com/cold-heat.htm) to the circuit board. By setting the DIP switches inside the transmitter, user controlled the code that the transmitter sent. The garage door would only open if the receiver's DIP switch were set to the same pattern. This provided some level of security, but not much. Eight DIP switches provide only 256 possible combinations. The transmitters in these circa-1970 garage door openers were also very simple where the transmitter consisted of two transistors and a couple of resistors, and not much else. A two-transistor transmitter like this, powered by a 9-volt [battery](https://auto.howstuffworks.com/battery.htm), is as simple as a radio transmitter gets. It's the same transmitter that user find in a pair of low-power walkie-talkies. Remote-entry transmitters have gotten a lot more sophisticated since then.

PRESENT SYSTEM

With the remote keyless-entry systems that user finds on [cars](https://auto.howstuffworks.com/car.htm) today, [security](https://auto.howstuffworks.com/car-alarm.htm) is a big issue. If people could easily open other people's cars in a crowded parking lot at the mall, it would be a real problem. And with the proliferation of [radio scanners](https://electronics.howstuffworks.com/radio-scanner.htm), user also need to prevent people from "capturing" the code that user transmitter sends. Once they have user code, they can simply re-transmit it to open user car.

The controller chip in any modern controller uses something called a hopping code or a rolling code to provide security. Some systems uses a [40-bit](https://computer.howstuffworks.com/bytes.htm) rolling code. Forty bits provide 240 (about 1 trillion) possible codes.

Working:

* The transmitter's controller chip has a [memory](https://computer.howstuffworks.com/computer-memory.htm) location that holds the current 40-bit code. When user push a button on user key fob, it sends that 40-bit code along with a function code that tells the car what user want to do (lock the doors, unlock the doors, open the trunk, etc.).
* The receiver's controller chip also has a memory location that holds the current 40-bit code. If the receiver gets the 40-bit code it expects, then it performs the requested function. If not, it does nothing.
* Both the transmitter and the receiver use the same [pseudo-random number generator](https://computer.howstuffworks.com/question697.htm). When the transmitter sends a 40-bit code, it uses the pseudo-random number generator to pick a new code, which it stores in memory. On the other end, when the receiver receives a valid code, it uses the same pseudo-random number generator to pick a new one. In this way, the transmitter and the receiver are synchronized. The receiver only opens the door if it receives the code it expects.
* If user are a mile away from user car and accidentally push the button on the transmitter, the transmitter and receiver are no longer synchronized. The receiver solves this problem by accepting any of the next 256 possible valid codes in the pseudo-random number sequence. This way, user (or user three-year-old child) could "accidentally" push a button on the transmitter up to 256 times and it would be okay -- the receiver would still accept the transmission and perform the requested function. However, if user accidentally push the button 257 times, the receiver will totally ignore user transmitter. It won't work anymore.
* So, what do user do if user three-year-old child DOES desynchronize user transmitter by pushing the button on it 300 times, so that the receiver no longer recognizes it? Most cars give user a way to resynchronize. Here is a typical procedure:
* Turn the [ignition](https://auto.howstuffworks.com/ignition-system.htm) key on and off eight times in less than 10 seconds. This tells the security system in the car to switch over to programming mode.
* Press a button on all of the transmitter user wants the car to recognize. Most cars allow at least four transmitters.
* Switch the ignition off.
* Given a 40-bit code, four transmitters and up to 256 levels of look-ahead in the pseudo-random number generator to avoid de-synchronization, there is a one-in-a-billion chance of user transmitter opening another car's doors. When user take into account the fact that all car manufacturers use different systems and that the newest systems use many more bits, user can see that it is nearly impossible for any given key fob to open any other car door.
* User can also see that code capturing will not work with a rolling code transmitter like this. Older garage door transmitters sent the same 8-bit code based on the pattern set on the DIP switches. Someone could capture the code with a radio scanner and easily re-transmit it to open the door. With a rolling code, capturing the transmission is useless. There is no way to predict which [random number](https://computer.howstuffworks.com/question697.htm) the transmitter and receiver have chosen to use as the next code, so re-transmitting the captured code has no effect. With trillions of possibilities, there is also no way to scan through all the codes because it would take years to do that.

**SEGMENT ANALYSIS**

|  |  |  |
| --- | --- | --- |
| LOW SEGMENT | MID SEGMENT | HIGHER SEGMENT |
| * Central Locking from driver’s side using physical key * Remote control for door lock/unlock | * Request sensor buttons on door handles * Proximity detection for the system to get connected to the beacon | * Use of keycard for lock/unlock * Passive entry when user’s hand is detected near the door handle |

Table 13: Passive Keyless Entry Segment analysis

AGEING

|  |  |  |
| --- | --- | --- |
| BEFORE 10 YEARS | AFTER 5 YEARS | CURRENT TECHNOLOGY |
| * Central locking using the physical locking from the driver side * Key fab having only lock and unlock button * Hands-free entry using code in the window panel | * Tags inserted in human body using nanotechnology * AI recognizing user’s action for automatic lock/unlock | * Request sensor buttons * Contact based cards for lock/unlock mechanisms * Using mobile devices over connected services |

Table 14: Passive Keyless Entry Ageing analysis

### S.W.O.T. ANALYSIS

|  |  |
| --- | --- |
| STRENGTH | WEAKNESS |
| * More secure system as the beacon sensitivity is required | * Proximity sensor may not be able to comprehend sometimes |
| OPPORTUNITY | **THREAT** |
| * Motion sensor can be used to detect even the slightest movement | * If remote is within the proximity vicinity, security issues of hacking and robbing can happen |

Table 15: Passive Keyless Entry SWOT analysis

### 4W&1H

|  |  |
| --- | --- |
| WHAT *is the feature?* | WHERE *is the system used?* |
| This feature is the Passive entry system for entry into a locked system | This feature is primarily used in vehicles to lock/unlock doors without physical key |
| WHY *is it used?* | **WHEN** *was this technology introduced?* |
| To increase the ease of access and security of a vehicle using wireless communication | This was introduced in early 1900 when automobile industry competed for excellence to increase the luxury feature of their vehicles |
| HOW *does the feature work?* | |
| * The feature uses proximity sensors to transmit and receive radio signals from the beacon (i.e. the key) * Transfer of data over a controlled code is used for enhanced security * Approaching the sensor unlocks while moving away from the range locks the vehicle | |

Table 16: Passive Keyless Entry 4W&1H

# LITERATURE REVIEW

## AIRBAG SYSTEM

[1] Air Bag: A Safety Restraint System of an Automobile (Tasnim N. Shaikh, Satyajeet Chaudhari and Hiren Rasania):

An Airbag is an automotive safety restraint system for an occupant as well as passengers. The system consists of a flexible fabric envelope or cushion, designed to inflate rapidly during an automobile collision. Its purpose is to cushion occupants during a crash and provide protection to their bodies when they strike interior objects such as the steering wheel or a window etc. Thus it lowers the number of injuries by reducing the force exerted by steering wheel, windows and the dashboard at any point on the body. This is accomplished in two ways, viz;

By increasing the interval over the force being applied or

By spreading the force over a large area of the body.

Air bag fabric has to keep a balance between two extreme conditions. It has to be sufficiently flexible to fold into relatively small volumes. At the same time it should be sufficiently strong to withstand the deployment at high speed, e.g. under the influence of an explosive charge, and the impact of passengers or other influences when inflated. Air bags must inflate very rapidly to be effective, and therefore come out of the steering wheel hub or instrumental panel with considerable force, generally at a speed over 100 mph.

Decision Algorithm for Smart Airbag Deployment Safety Issues (Aini Hussain, Member, IEEE, M A Hannan, Azah Mohamed, Senior Member, IEEE, Hilmi Sanusi and Burhanuddin Yeop Majlis, Member, IEEE):

Airbag deployment has been known to be responsible for huge death, incidental injuries and broken bones due to low crash severity and wrong deployment decisions. Therefore, the authorities and industries have been looking for more innovative and intelligent products to be realized for future enhancements in the vehicle safety systems (vsss). Although the vsss technologies have advanced considerably, they still face challenges such as how to avoid unnecessary and untimely airbag deployments that can be hazardous and fatal. Currently, most of the existing airbag systems deploy without regard to occupant size and position. It intends to provide a thorough discussion relating to the occupancy detection, occupant size classification, occupant off-position detection to determine safe distance zone for airbag deployment, crash-severity analysis and airbag decision algorithms via a computer modeling. The proposed system model consists of three main modules namely, occupant sensing, crash severity analysis and decision fusion.

## SEAT BELT ALERT SYSTEM

|  |  |  |  |
| --- | --- | --- | --- |
| Sr No. | Features | Title | Author |
| 1. | Seat belt | Seat belt alert system with ignition control in car | 1 Akshay Vetal, 2 Gaurav Sakurde,  3 Pradip Salunke,  4 Krinshna Barskar,  5 Prof. M. B. Bankar |
| 2. | Seat belt | Seat belt safety features using sensors to protect occupants | S.D Rahul,  Bhardwaj,  ShraddhaR,  Jogdhankar |

Table 17: Seat Belt Alert Literature Review

## SEMI ACTIVE SUSPENSION SYSTEM

|  |  |  |  |
| --- | --- | --- | --- |
| Project Name | Topic Name | Author | Description |
| Semi active suspension system | Improvement of vehicle roll stability by varying suspension properties. | Taehyun Shim and Pradheep C. Velusamy | Some suspension properties such as roll centre height, roll steer, and roll chamber these strongly influence the vehicle roll dynamics. In this paper, the effects of suspension properties on vehicle roll response have benn investigated using a multi-body vehicle dynamics programme. Design of experiments has been used for identifying critical hardpoints affecting ths suspension parameter optimization. This approach provides a viable alternative to costlier active control systems for economy class vehicles. |
|  | Variable Stiffness Suspension System | Joseph Jerz, Jr. 131 Wylerhorn St., Crestline, Calif.  Reference: UNITED STATES PATENTS | In this paper they discussed about the variable suspension stiffness system they designed using two springs. The springs are connected in series with one of the spring being stiffer than the other and the other one is softer.  Under normal road conditions the softer spring is actuated giving a more comfortable ride but the vehicle suspension system is supported by the stronger stiffer spring under heavy load conditions. The conversion between these two conditions may be effected automatically, by engagement under heavy load conditions of a pair of stop shoulders acting to limit compression of the light spring. Similarly, upon excessive extension of the springs, an additional set of stop shoulders may automatically become effective to limit the amount of extension of the softer spring and cause the stiffer spring to resist further extension. |

Table 18: Semi-active Suspension Literature Review

## PASSIVE KEYLESS ENTRY

[1] Hands-free Remote Entry System

This paper shows the hands-free system used in automobiles for increased security and ease of access to the vehicle without any physical entry on system to identify the signal and unlock which button is pressed. Using a control unit to provide a code for every request and to use it for authentication purpose to follow the action.

[2]Passive remote

The purpose is to help understand the basic technology involved in the system to create an area of sensors where in the main receiver when inside, could be used only for accessing the various functioned it is programmed to.

[3] Keyless remote system

The use of a mobile device having Bluetooth connectivity is used to connect to the vehicle and perform the security actions from anywhere in the world. The use of only the mobile device to connect to the vehicle having a ranged frequency.

[4]Study on vehicle and garage Remote entry

This information helped to understand the old system of using a radio signal of specific frequency to perform actions and the required lock/unlock functions. The direct connection when in direct sight provided a convenience feature which was at times only implemented in luxury cars. Having a key fob to do such actions without physical effort helped a lot and changed the present iterations on a whole new level.

# UML DIAGRAM

## STRUCTURAL DIAGRAM

### PIN DIAGRAM

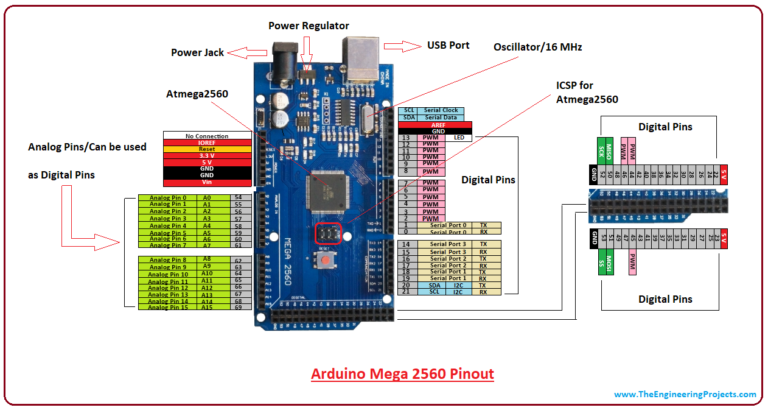


Figure 1: Arduino Integrated PIN Diagram

### BLOCK DIAGRAM

#### **INTEGRATED SYSTEM**

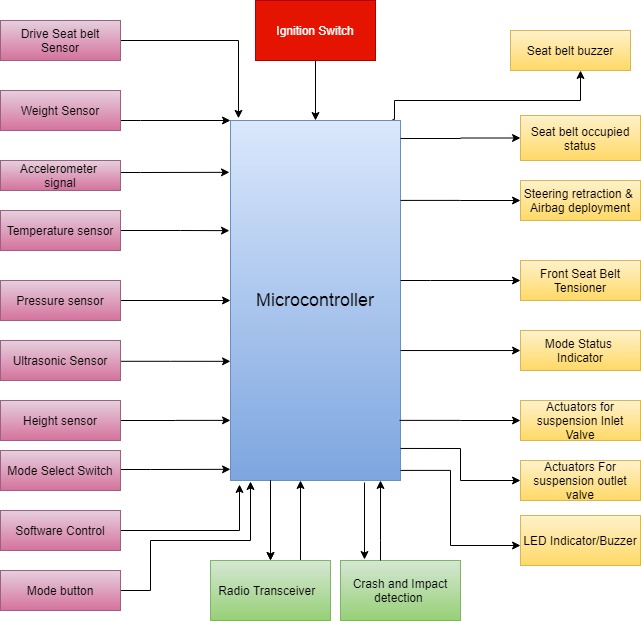
********

Figure 2: Integrated System Block Diagram

#### ADVANCED AIRBAG SYSTEM

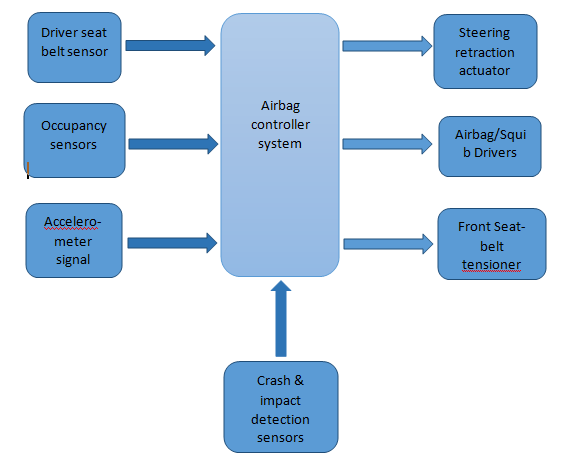
****

Figure 3: Advanced Airbag Block Diagram

#### SEAT BELT ALERT SYSTEM

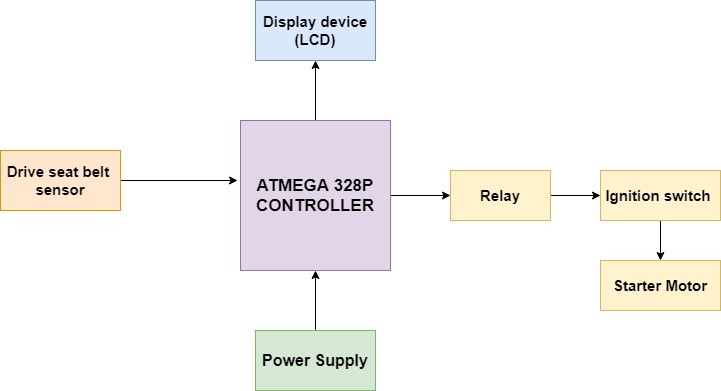


Figure 4: Seat Belt Alert Block Diagram

#### SEMI ACTIVE SUSPENSION SYSTEM

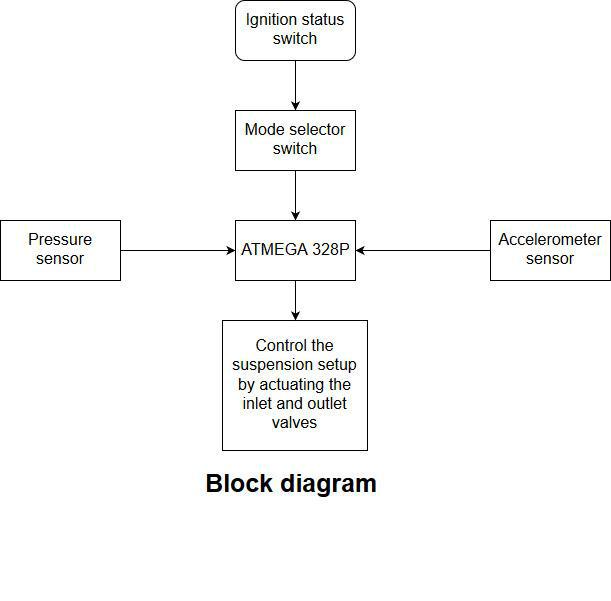


Figure 5: Semi-active Suspension Block Diagram

#### PASSIVE KEYLESS SYSTEM

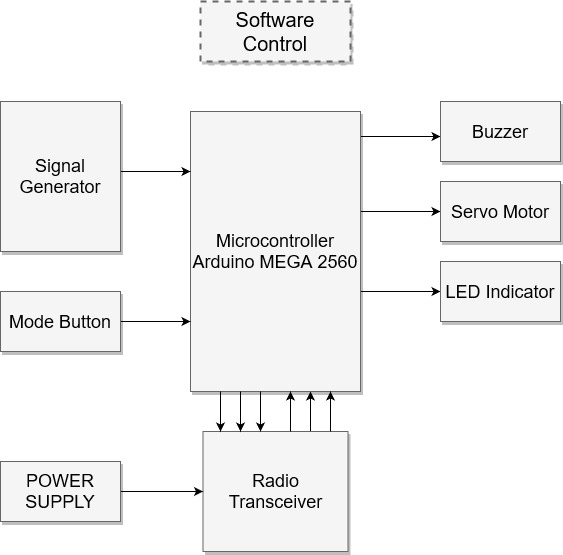


Figure 6: Passive Keyless Entry Block Diagram

## BEHAVIOURAL DIAGRAM

### STATEFLOW DIAGRAM

#### INTEGRATED SYSTEM

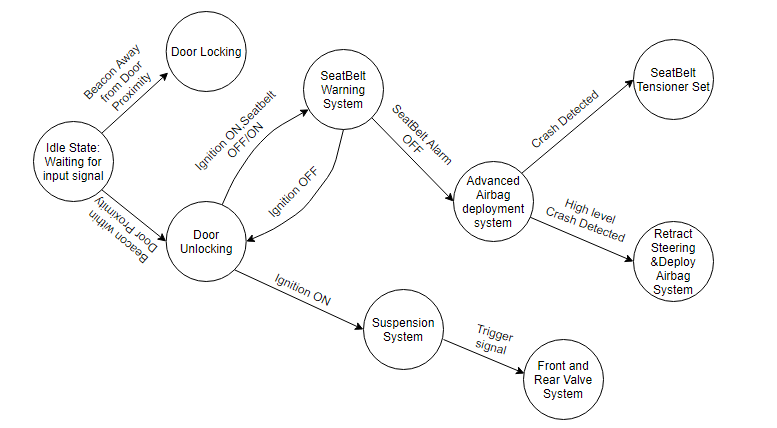


Figure 7: Integrated System State flow Diagram

#### ADVANCED AIRBAG SYSTEM

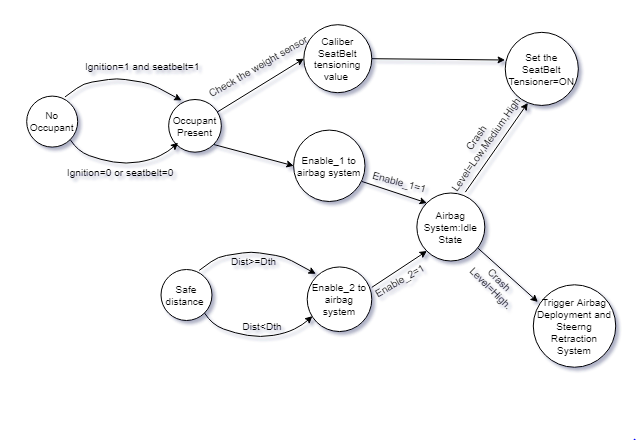


Figure 8: Advanced Airbag State flow Diagram

#### SEAT BELT WARNING SYSTEM



Figure 9: Seat Belt Alert State flow Diagram

#### SEMI ACTIVE SUSPENSION SYSTEM

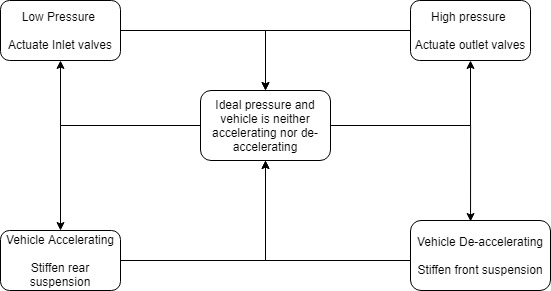


Figure 10: Semi-active Suspension State flow Diagram

#### PASSIVE KEYLESS ENTRY

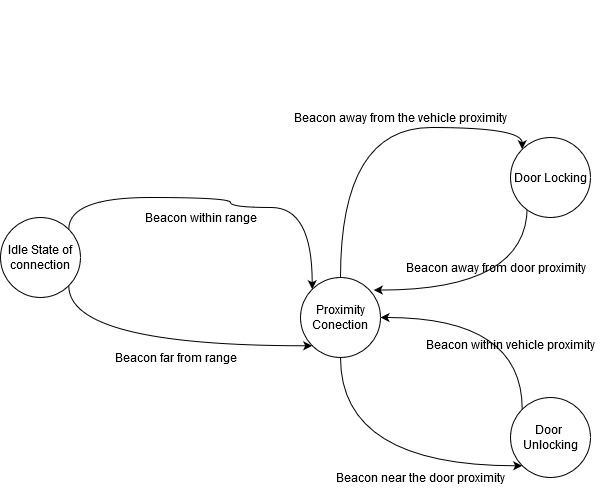


Figure 11: Passive Entry State flow Diagram

### FLOWCHART

#### INTEGRATED FLOWCHART

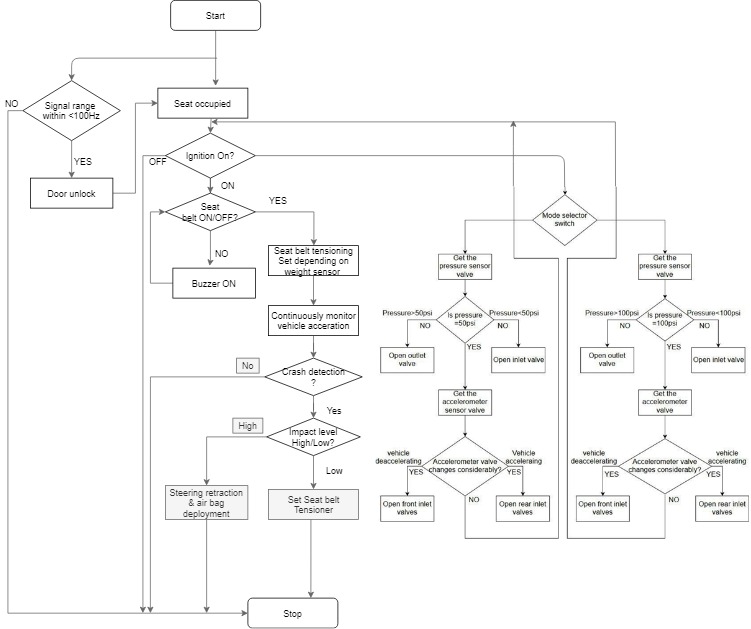


Figure 12: Integrated System Flowchart

#### ADVANCED AIRBAG SYSTEM

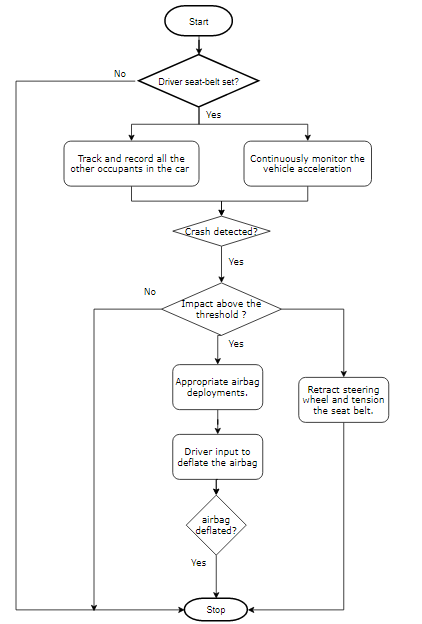


Figure 13: Advanced Airbag Flowchart

#### SEAT BELT ALERT SYSTEM

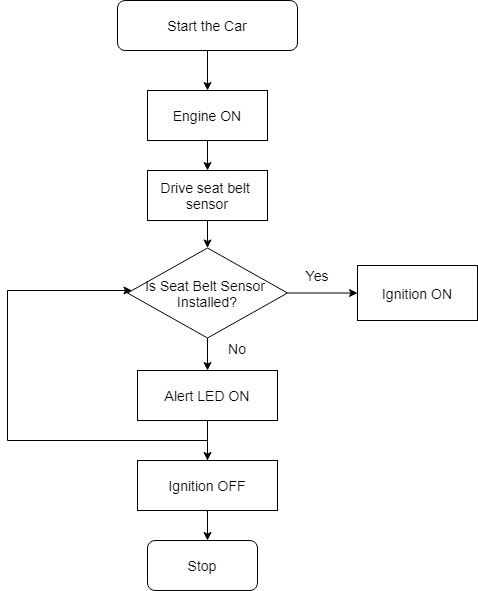


Figure 14: Seat Belt Alert Flowchart

#### SEMI ACTIVE SUSPENSION SYSTEM

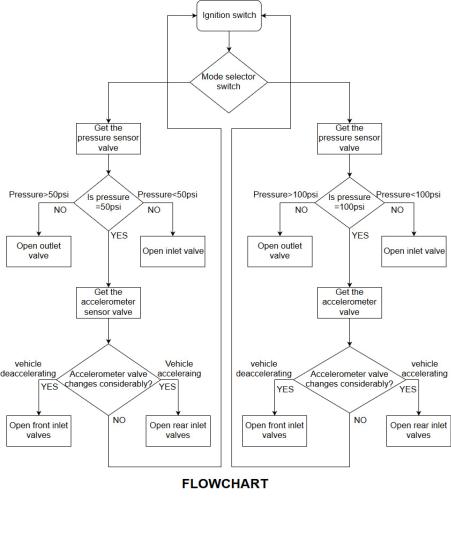


Figure 15: Semi-active Suspension Flowchart

#### PASSIVE KEYLESS ENTRY

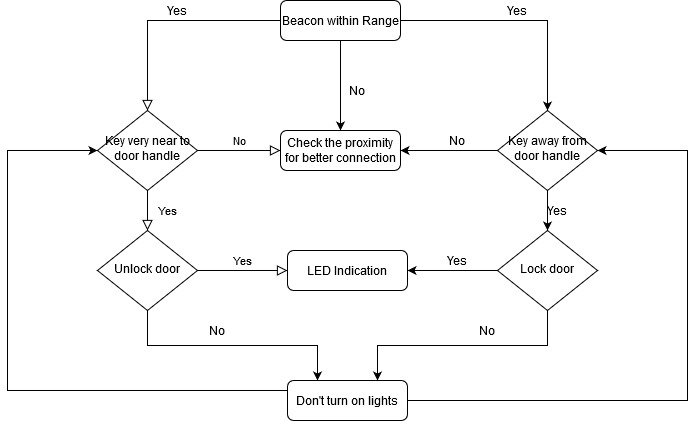


Figure 16: Passive Entry Flowchart

# BILL OF MATERIALS

|  |  |  |
| --- | --- | --- |
| COMPONENTS | QUANTITY | PRICE |
| Arduino MEGA | 1 | 1500 |
| Proximity sensor | 4 | 200 |
| Buzzer | 1 | 100 |
| LED | 1 | 80 |
| Servo motor | 1 | 400 |

Table 19: Bill of Materials

# EXECUTION PLAN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SL. NO. | TASK | STARTING DATE | COMPLETION DATE | HOURS ALLOTED |
| 1 | Problem Statement and Requirement Gathering | 07-02-2020 | 10-02-2020 | 18 hours |
| 2 | Hardware Design | 07-02-2020 | 10-02-2020 | 10 hours |
| 3 | Software Coding | 14—02-2020 | 17-02-2020 | 24 hours |
| 4 | Implementations and Simulations | 14-02-2020 | 17-02-2020 | 15 hours |
| 5 | Integrations |  |  | 20 hours |
| 6 | Testing |  |  | 10 hours |
| 7 | Documentation |  |  | 9 hours |

Table 20: Team Execution Plan

# TEAM DESCRIPTION

# INTEGRATED MODELLING AND TESTING

## MODEL BASED DESIGN

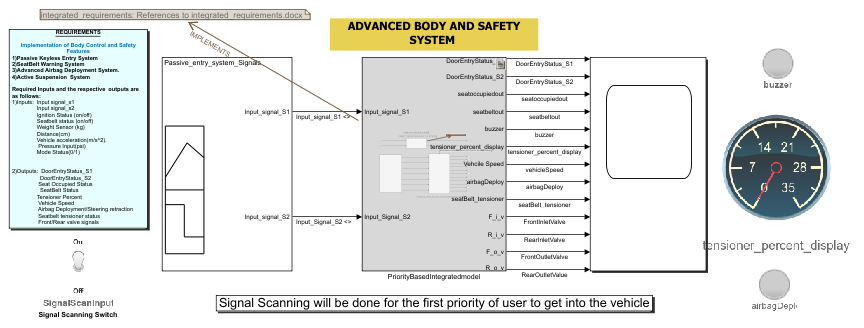


Figure 17: Integrated Body and Safety Model in Simulink

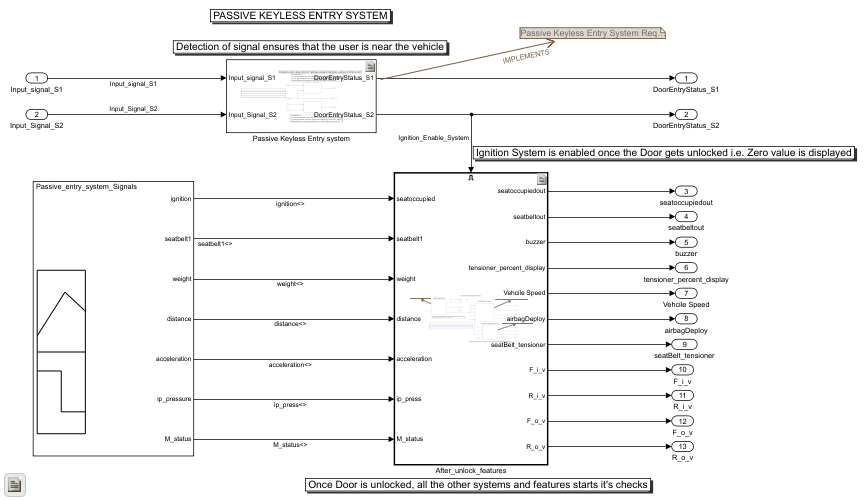


Figure 18: Main System Block

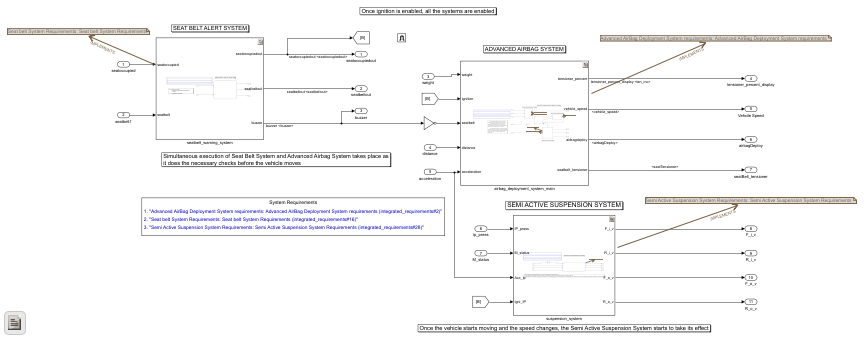


Figure 19: Subsystem block and other feature enabled block

REQUIREMENT LINKING USING EDITOR



Figure 20: Integrated Body and Safety Requirement Linking

INTEGRATED SIGNAL BUILDER

**Signal builder for the main block system**

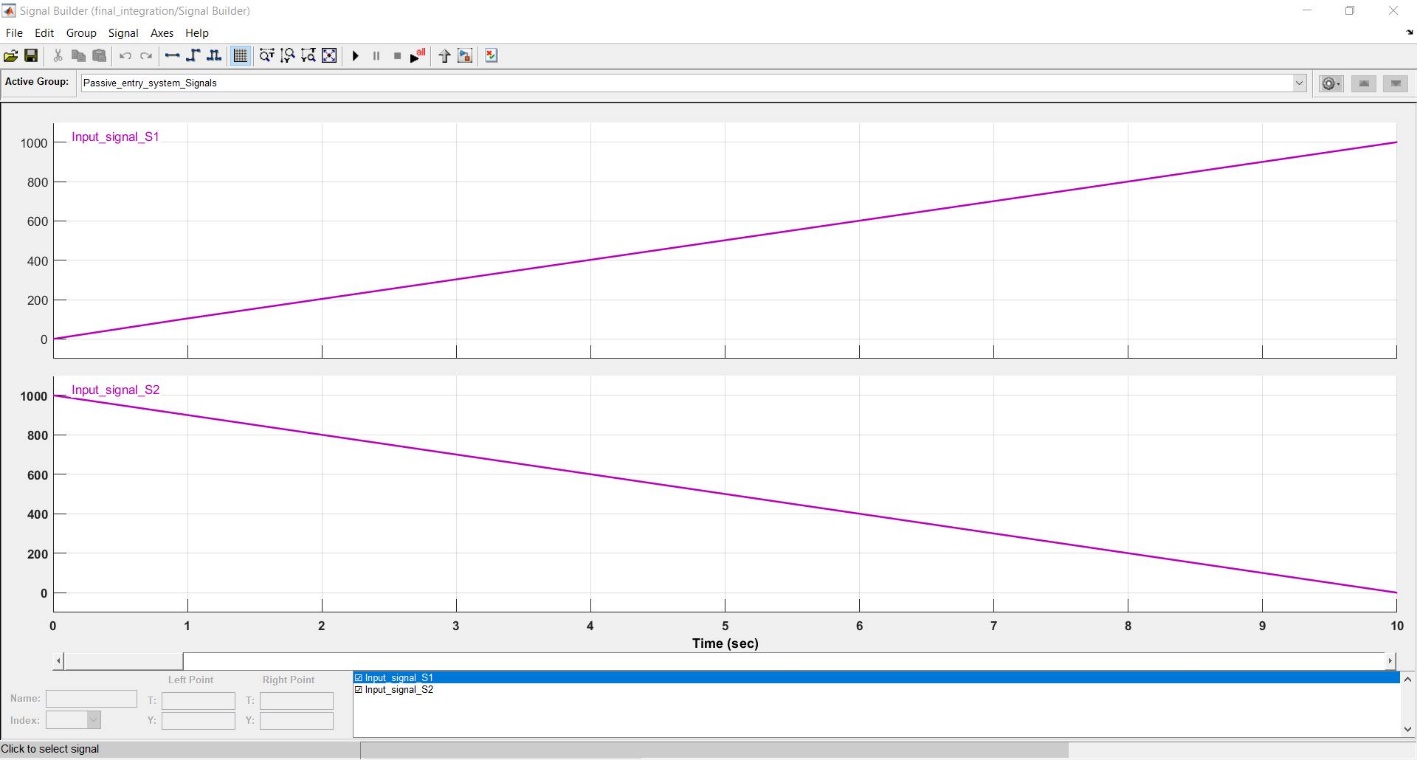


Figure 21: Main Block Signal Builder inputs

**SIGNAL BUILDER FOR THE ENABLED SYSTEM**

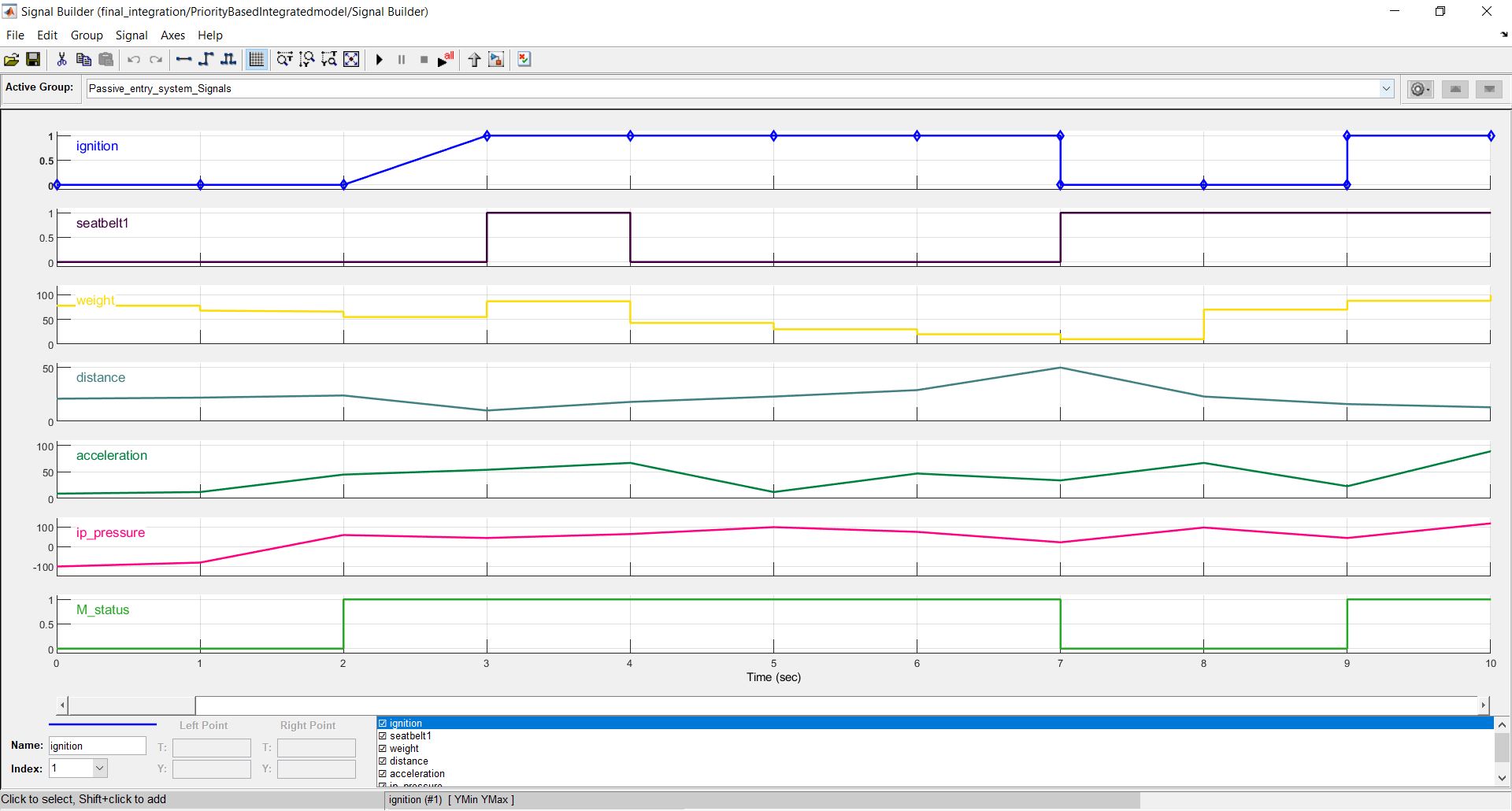


Figure 22: Signal Builder inputs for Enabled system

INTEGRATED OUTPUT on SCOPE

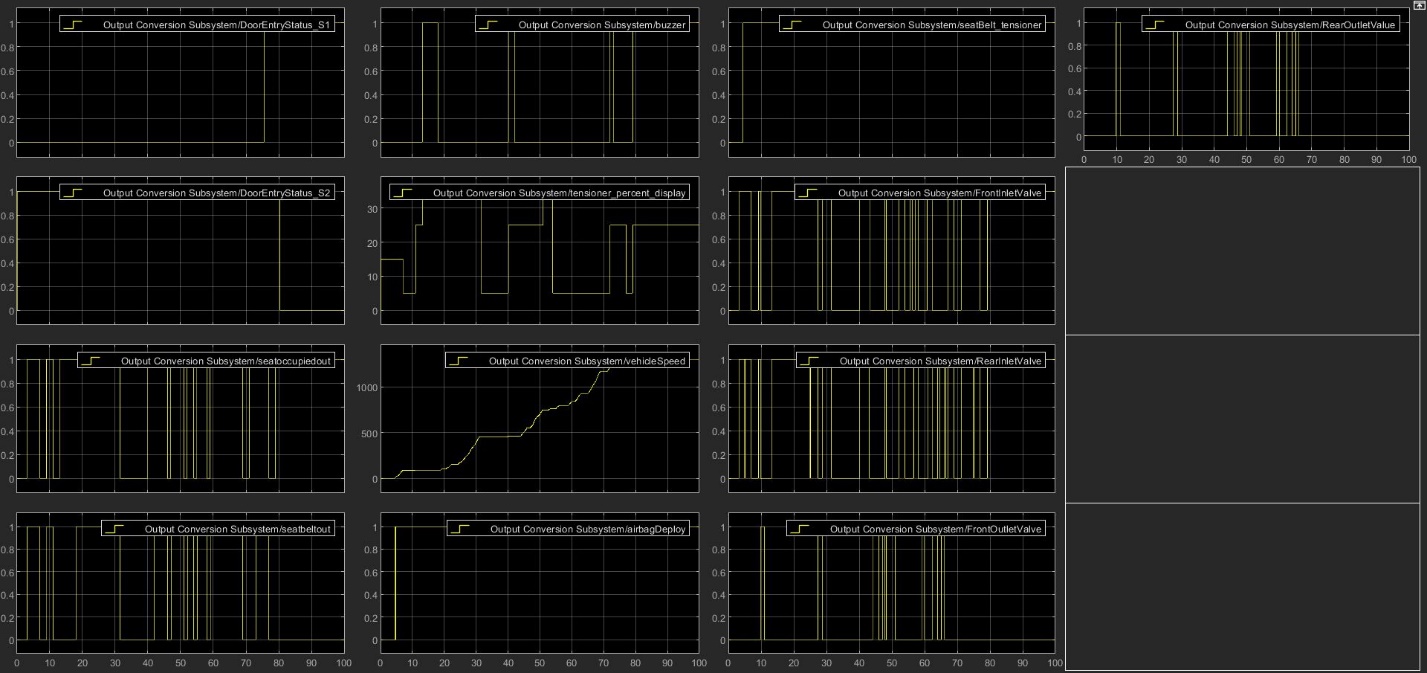


Figure 23: Integrated Output of All systems on scope

## DEBUGGING

DATA INSPECTOR

**Initial comparison gave no signals to be in tolerance level**

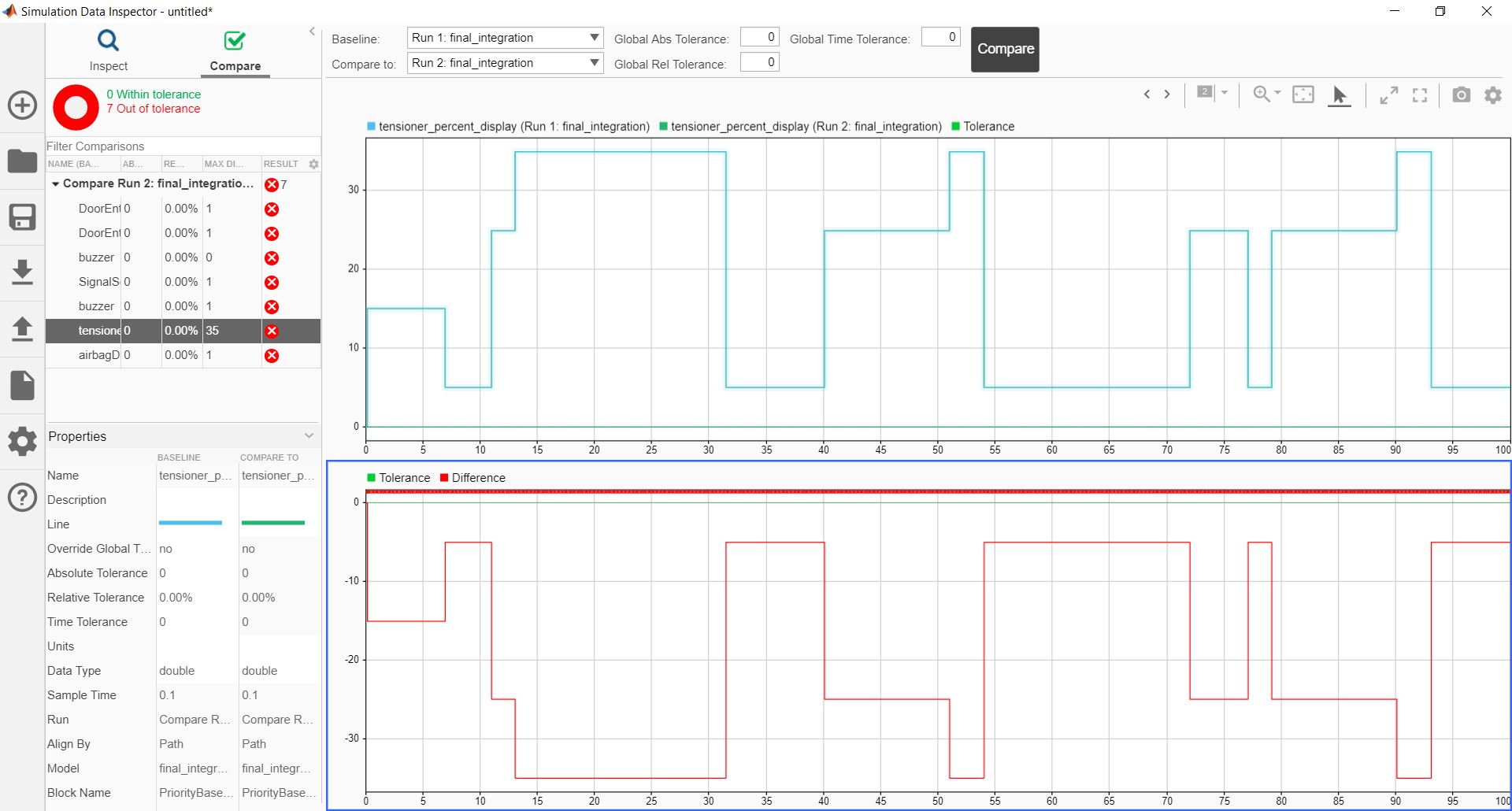


Figure 24: Integrated Data inspector comparison

**When comparing signals using Global Abs Tolerance at 10**

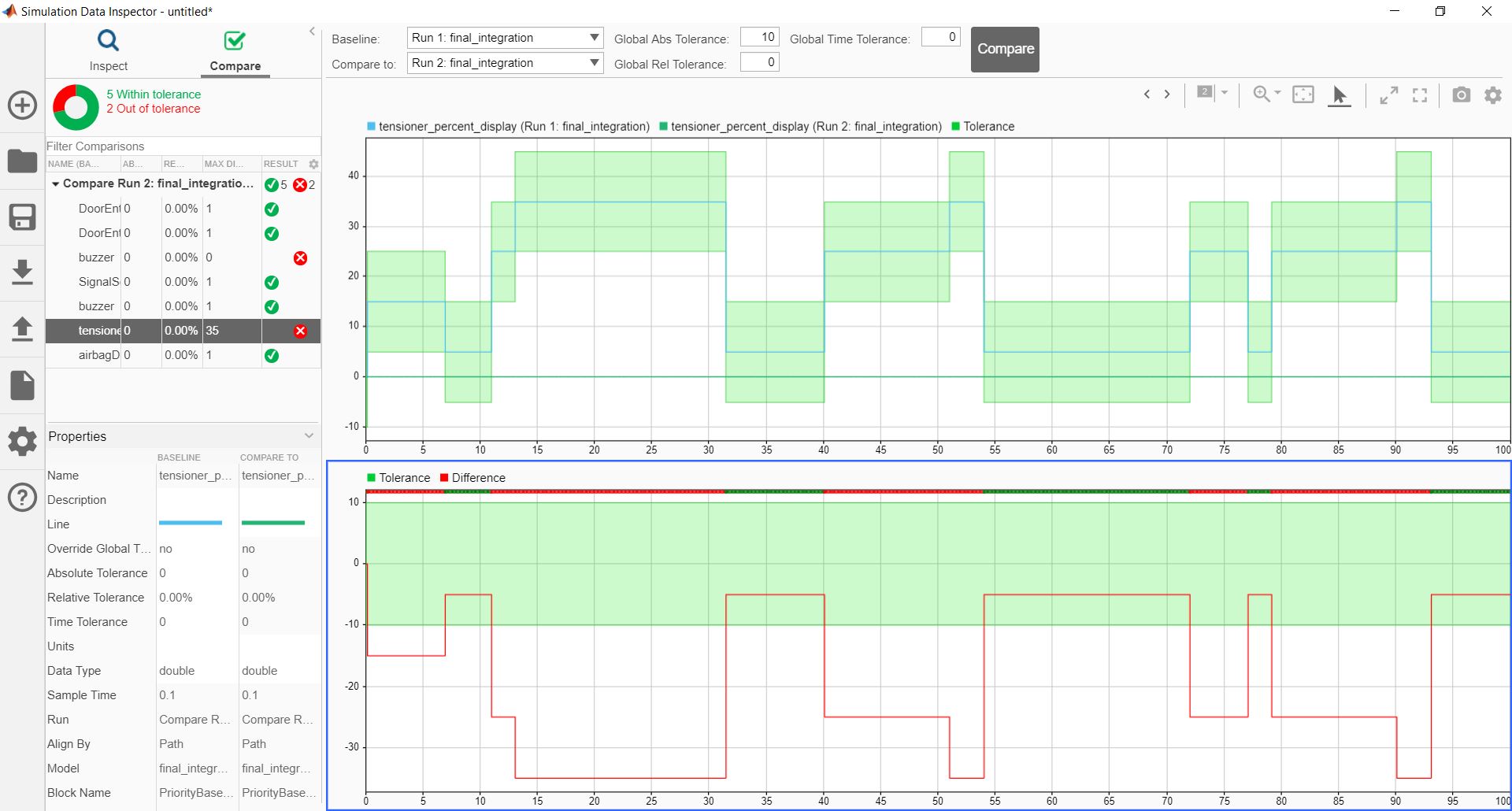
****

Figure 25: Comparison of runs with Global Abs Tolerance at 10

* Signals for the Buzzer and Tensioner values were not within tolerance

**Again comparing signals using Global Abs Tolerance at 10 and Global Rel Tolerance at 2**

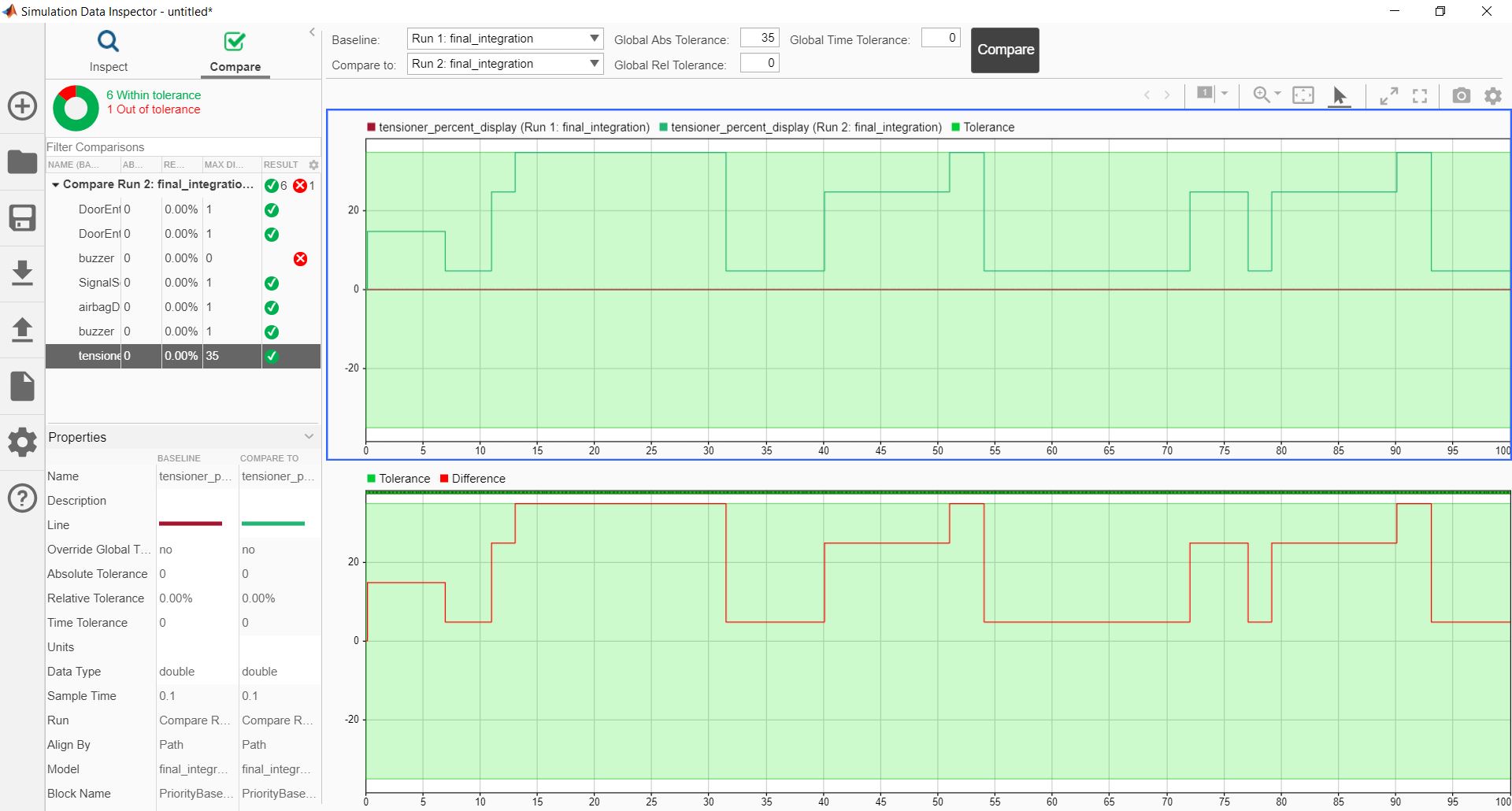
****

Figure 26: Comparison runs including increase in Abs and Rel Tolerance

* The Tensioner value was set inside tolerance. However, the buzzer was not executed so it was not inside any tolerance level

## TESTING USING SIMULINK TEST

TESTING BY TOGGLING THE SIGNAL SCAN SWITCH

**OUTPUT ON SCOPE**

* Signal Scan switch acts as the main enabler signal for the whole system
* If there is no user, to detect the signal then no feature will get activated

****

Figure 27: Toggle of input Signal switch to OFF

TESTING USING SIGNAL BUILDER

**SIGNAL INPUT FROM EXCEL FILE**

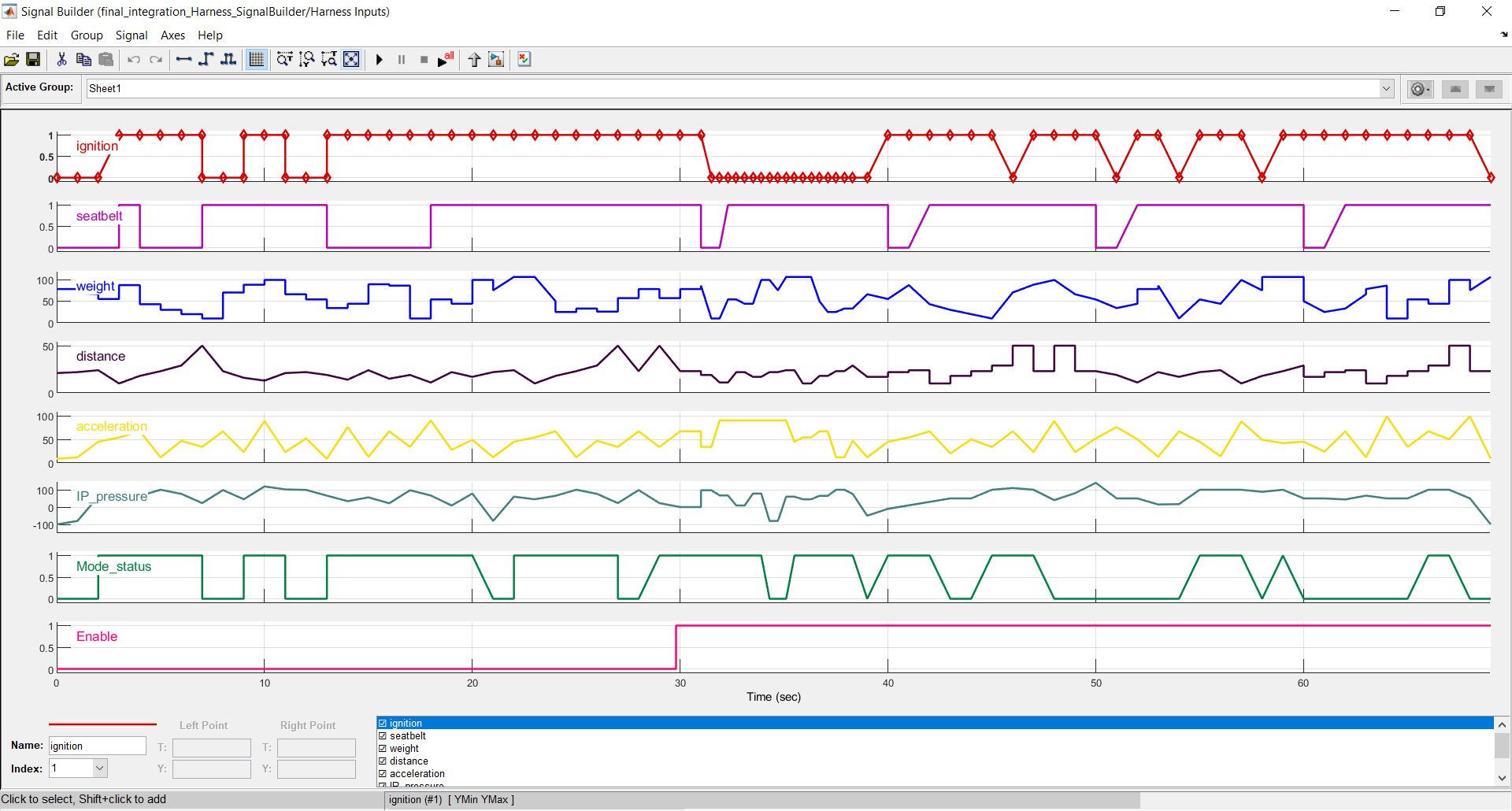


Figure 28: Signal Builder Input from Excel file

**HARNESS MODEL**

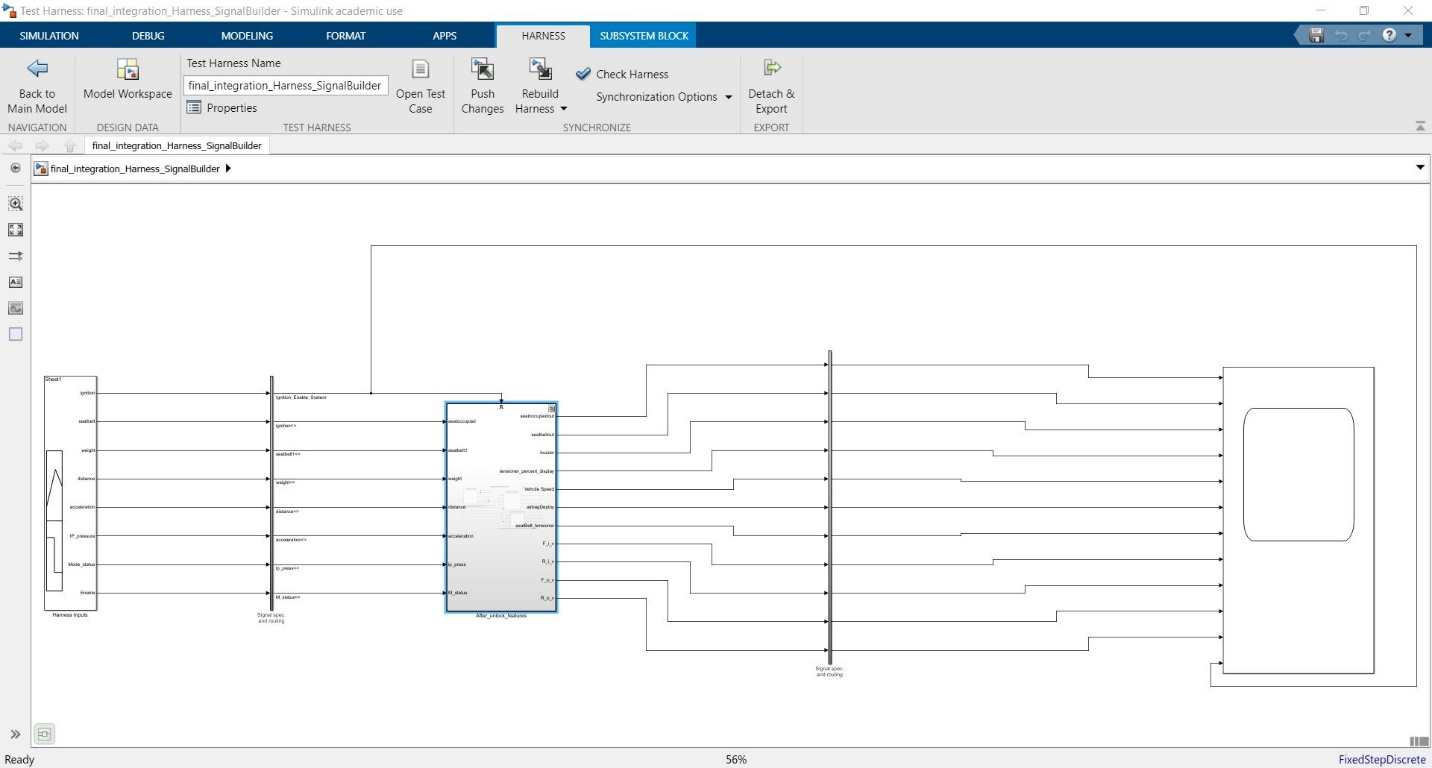


Figure 29: Signal Builder Harness Model

**OUTPUT SIGNAL ON SCOPE**

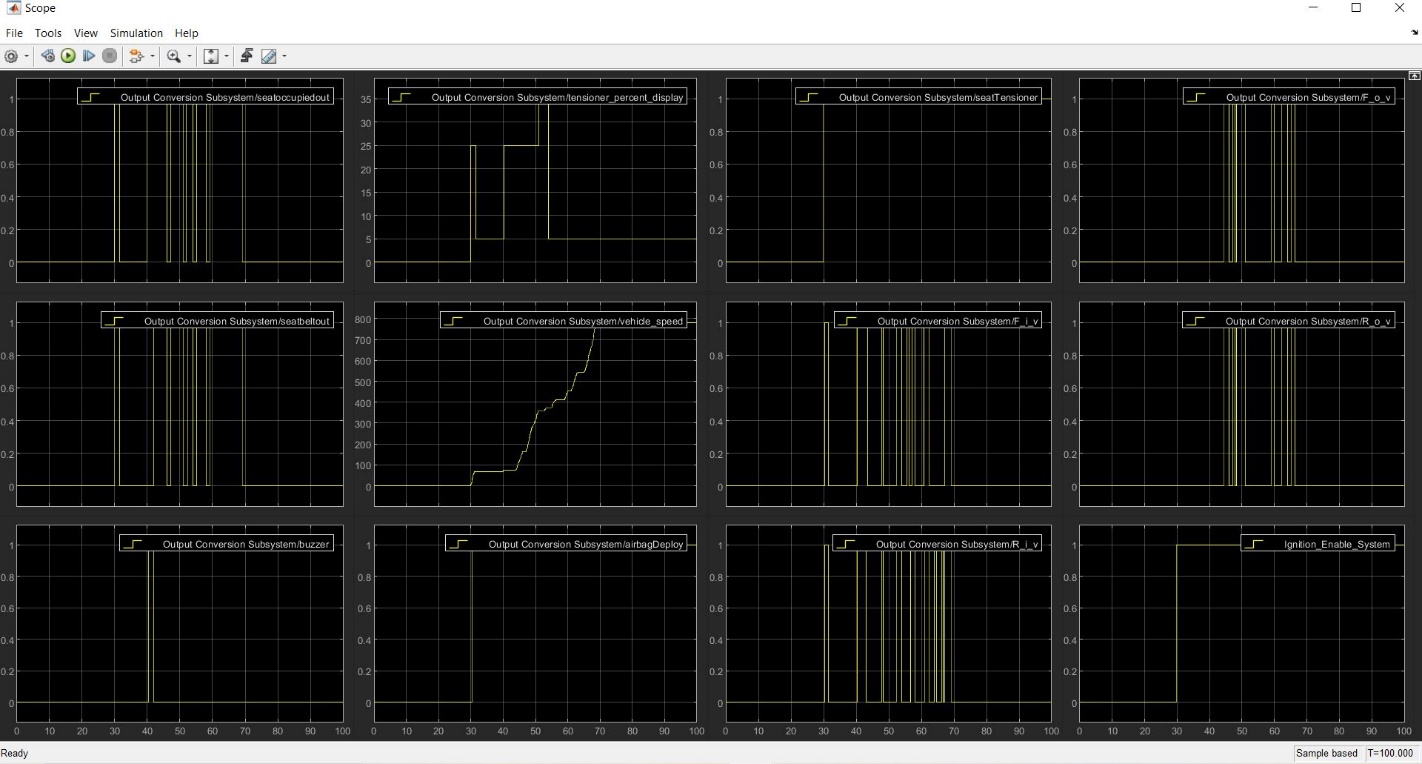


Figure 30: Signal Builder Output

**OUTPUT USING OUTPORT TO DATASET**

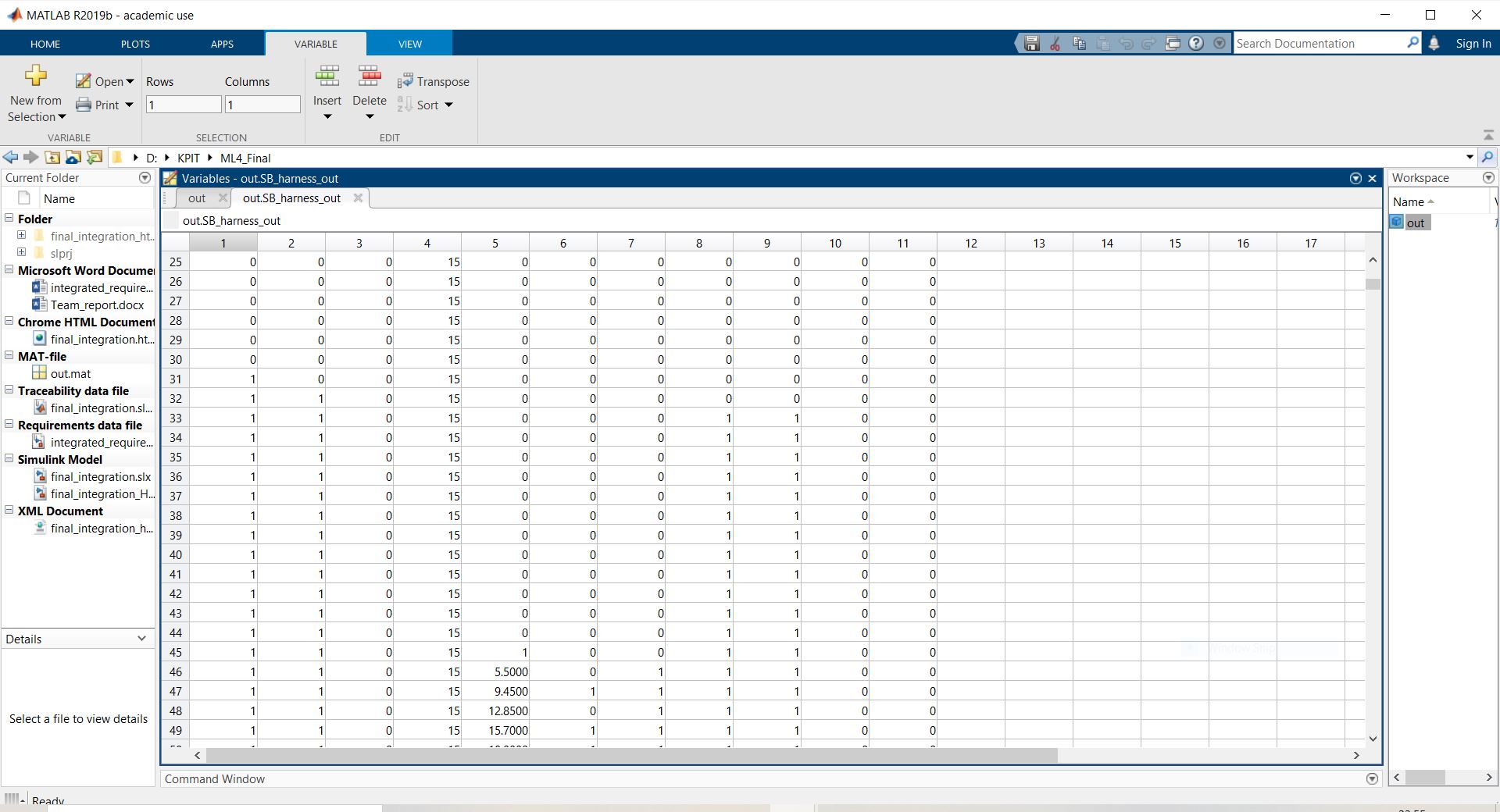
****

Figure 31: Output Signal transition in Workspace as dataset

TESTING USING CONSTANT INPUTS TO MAIN SYSTEM

**HARNESS MODEL**

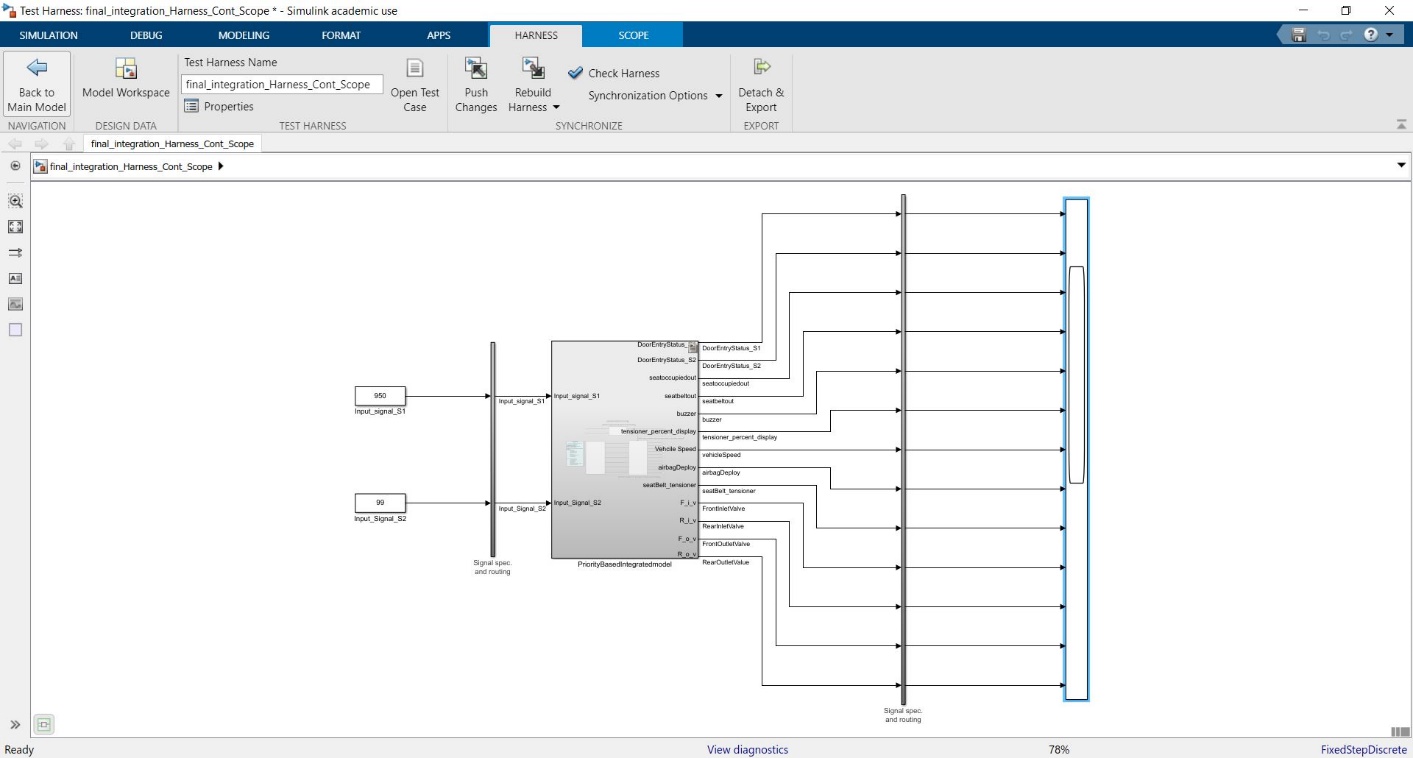
****

Figure 32: Constant input to Main System

**OUTPUT ON SCOPE WHEN INPUT SIGNAL IS 99**

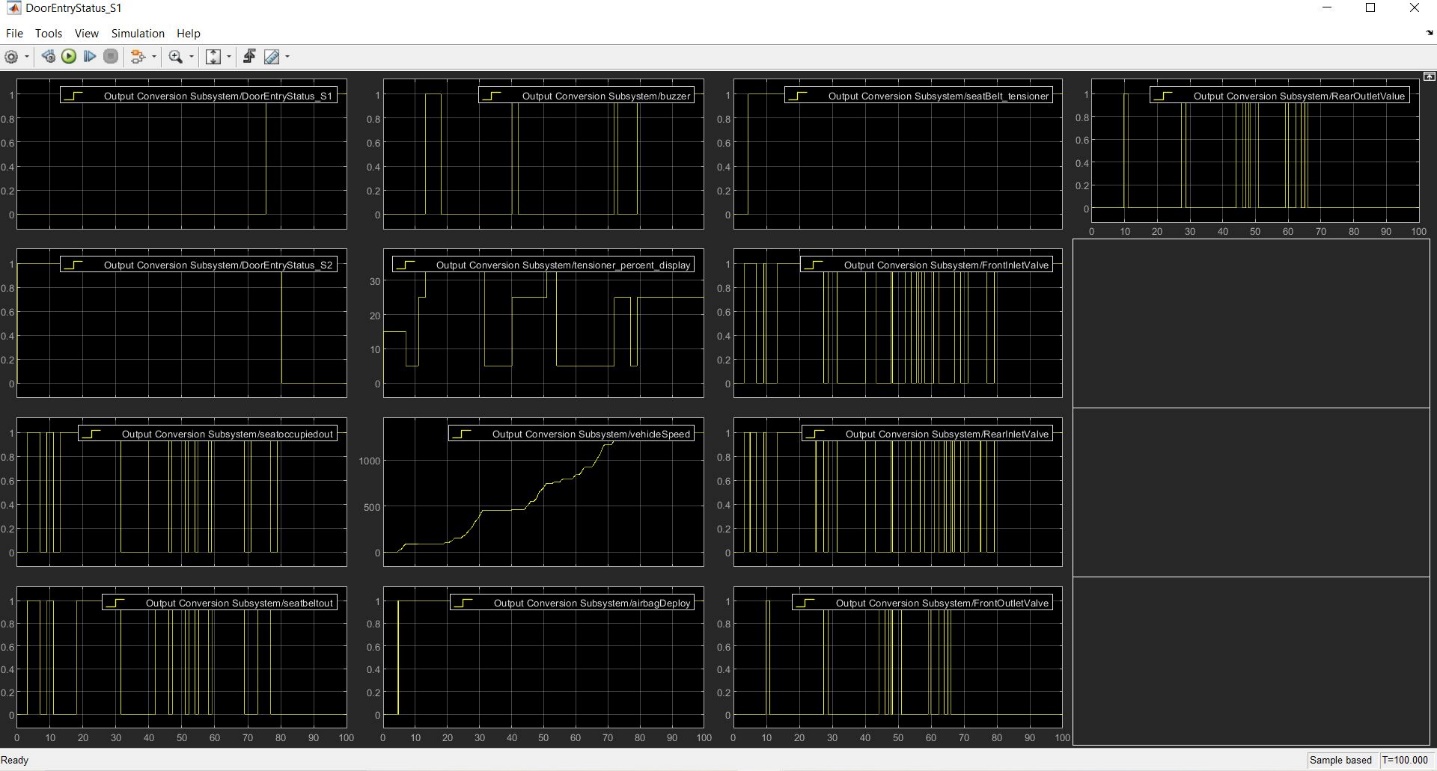
****

Figure 33: Output signals when input is 99

**OUTPUT ON SCOPE WHEN INPUT SIGNAL IS 1000**

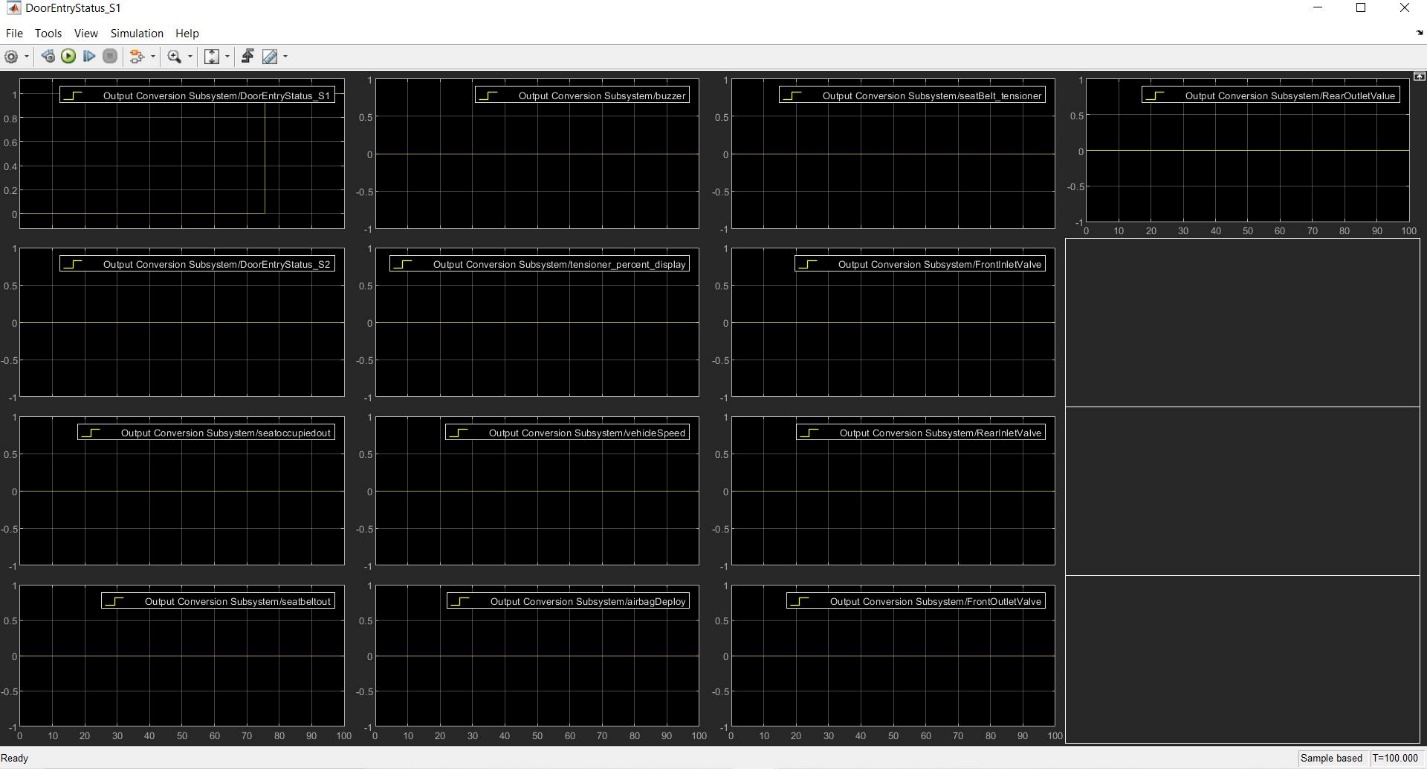
****

Figure 34: Output signals when input is 1000

TESTING USING CONSTANT INPUTS TO ENABLED SYSTEM

**HARNESS MODEL**

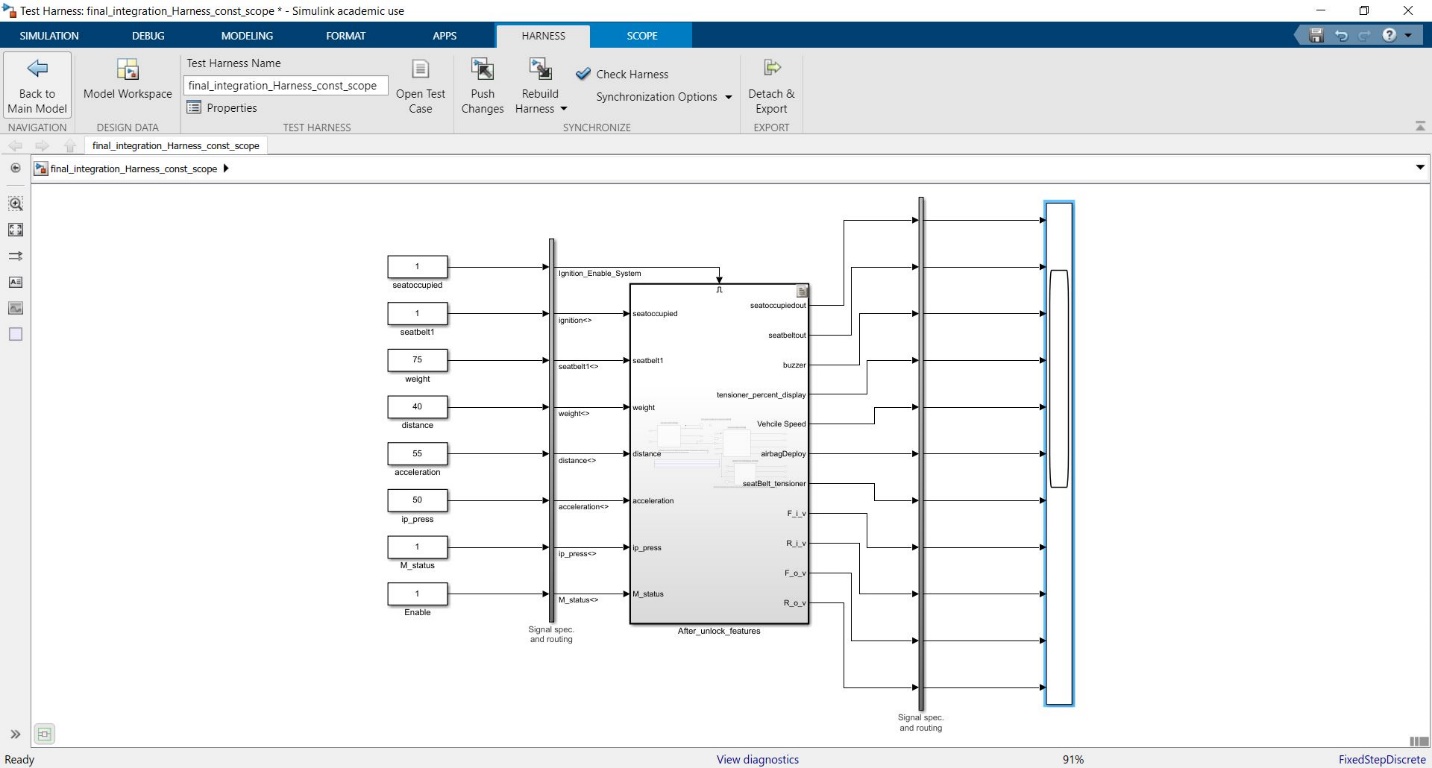


Figure 35: Constant inputs to Enabled system

**OUTPUT ON SCOPE**

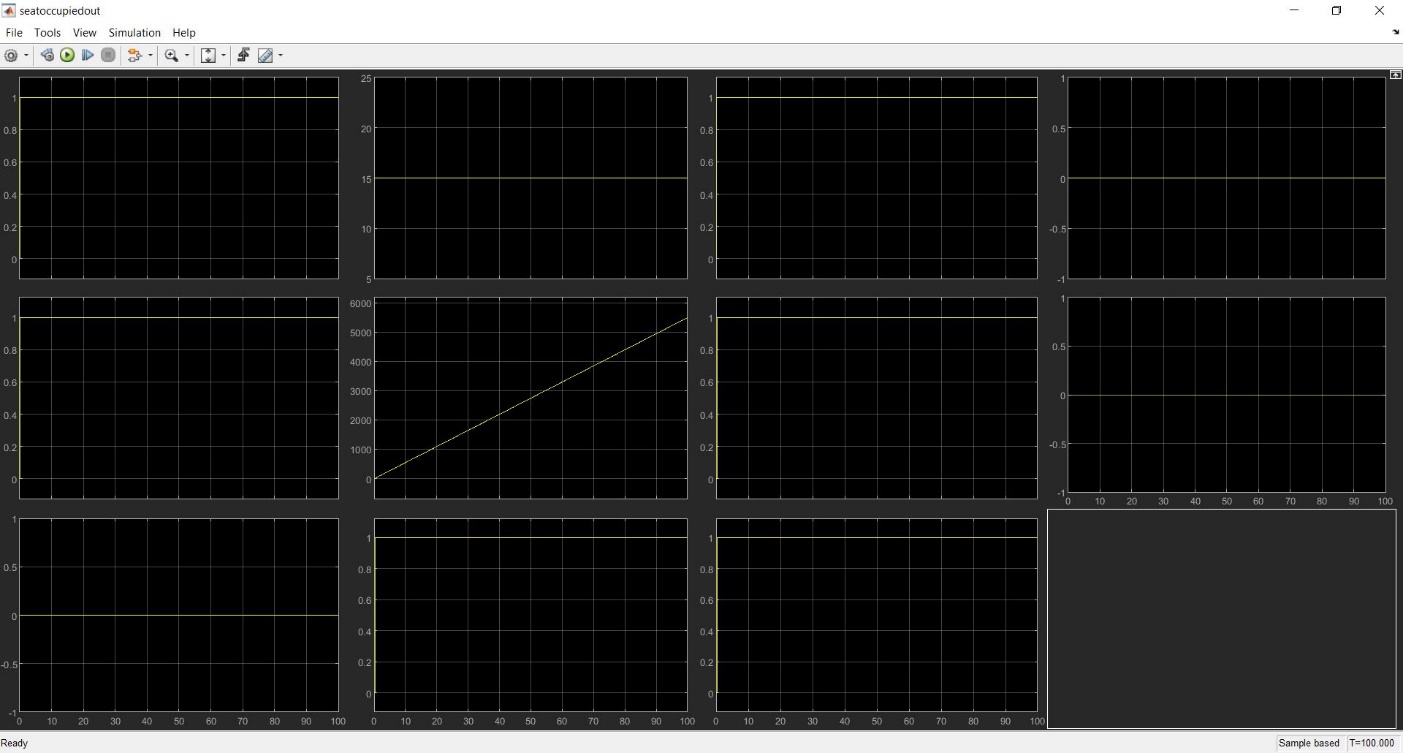


Figure 36: Output signals when constant input is given

# INTEGRATED TEST PLAN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEST ID | PURPOSE OF TEST | INPUT | EXPECTED OUTPUT | OBSERVED OUTPUT | RESULT |
| T\_01 | To test working of features when switch is OFF | Signal Scan switch is in OFF state | All features should not show any value | All features’ value is zero | **PASS** |
| T\_02 | To test working of features when switch is ON | Signal Scan switch is in ON state | All features should work and show values | All the features are providing outputs | **PASS** |
| T\_03 | To test Door\_Status\_S2 enabling system | Output from Door\_Status\_S2 is given as Zero | The enabled subsystem should activate the other features | All the systems in enabled subsystems are working as required | **PASS** |
| T\_04 | Test enabled subsystem when enabled transition happens midway of execution time, i.e. 40 sec | Enabled system input is transitioning from 1 to 0,  i.e. Door\_Status\_S2 changes from Lock to Unlock | All systems should continue execution and show normal output | **A)** Features are executing only after the transition  **B)** Airbag, Seat Belt and Suspension system is executing much faster rate and the idle output is not shown  **C)**Tension\_disp, veh\_speed, F\_o\_v, F\_i\_v, R\_o\_v, R\_i\_v all accounts show that the signals increase in value rapidly just after the transition happens for the system enabling. Output should not show rapid change of data and continue its flow of signal execution | **A) PASS**  **B) FAIL**  **C)**  **FAIL** |
| T\_05 | Testing at particular value of Constant if the output value matches | seatoccupied and seatbelt1=1, weight=75, distance=40, acceleration=55, ip\_press=50, M\_status=1, Enable=1 | All the systems in the subsystems should show the at point output in the scope | The exact value for the given input constant values are shown. seatoccupout=1,  seatbeltout=1,  buzzer=0,  tension\_disp=15,  veh\_speed=5500,  sirbagdeploy=1,  belt\_tesnion=1,  F\_i\_v=1,R\_i\_v=1,  F\_o\_v=0,R\_o\_v=0 | **PASS** |

Table 21: Integrated Test Plan

# INTEGRATED COMPLIANCE

## MAAB CHECK

BEFORE CLEARING OF WARNINGS

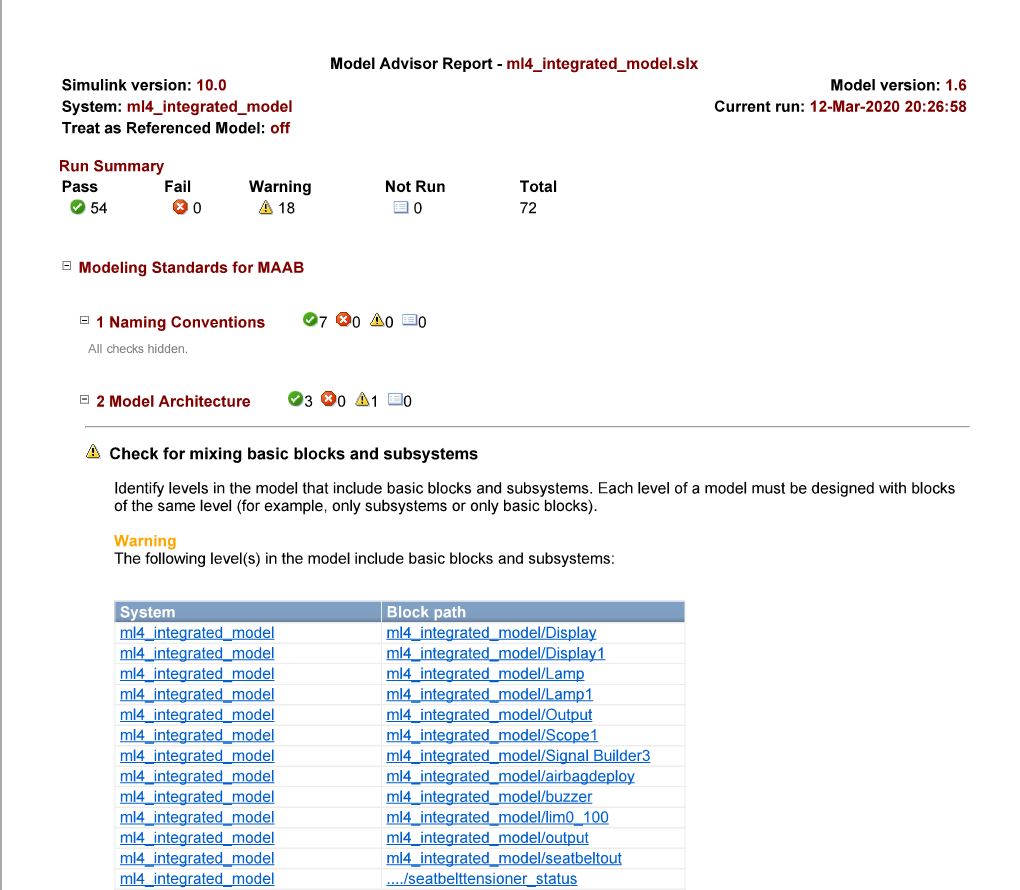


Figure 37: MAAB Result for the First Iteration

AFTER WARNING CLEARANCE AND CHECK

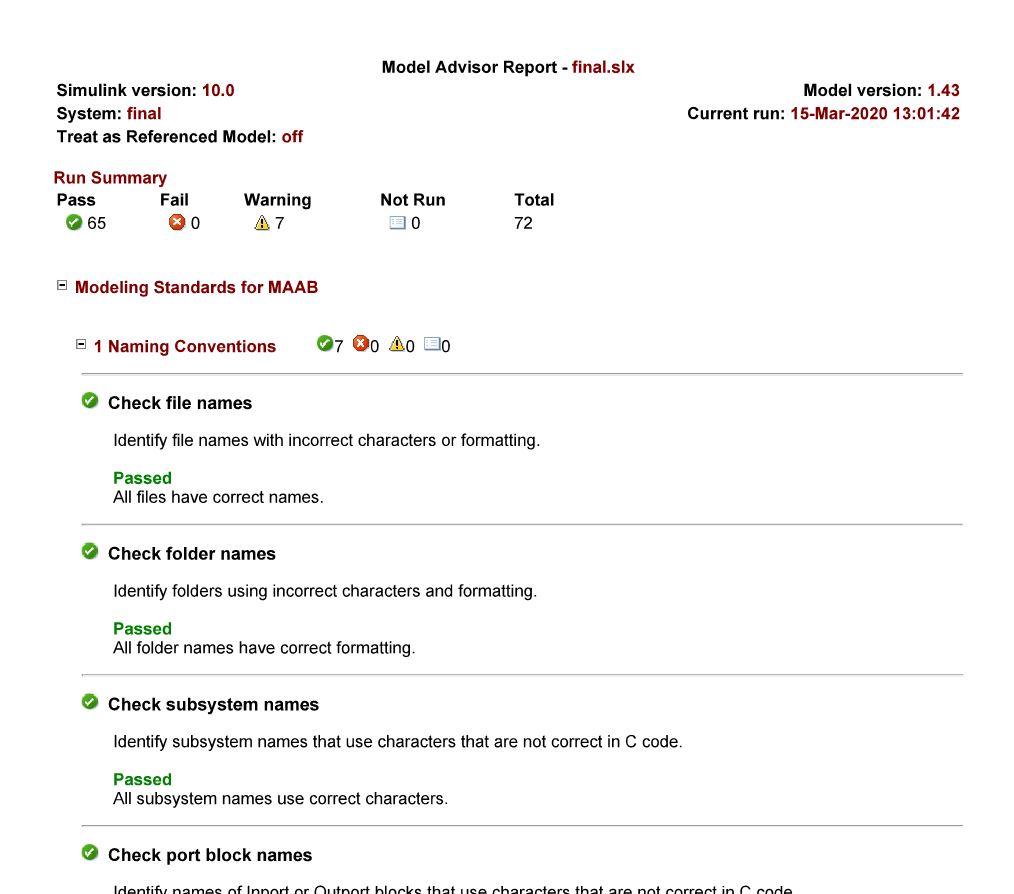


Figure 38: MAAB Result after clearing warnings

## CODE COVERAGE

CHECKING COVERAGE USING MCDC OPTION FOR ENTIRE MODEL

FIRST ITERATION OF CODE COVERAGE

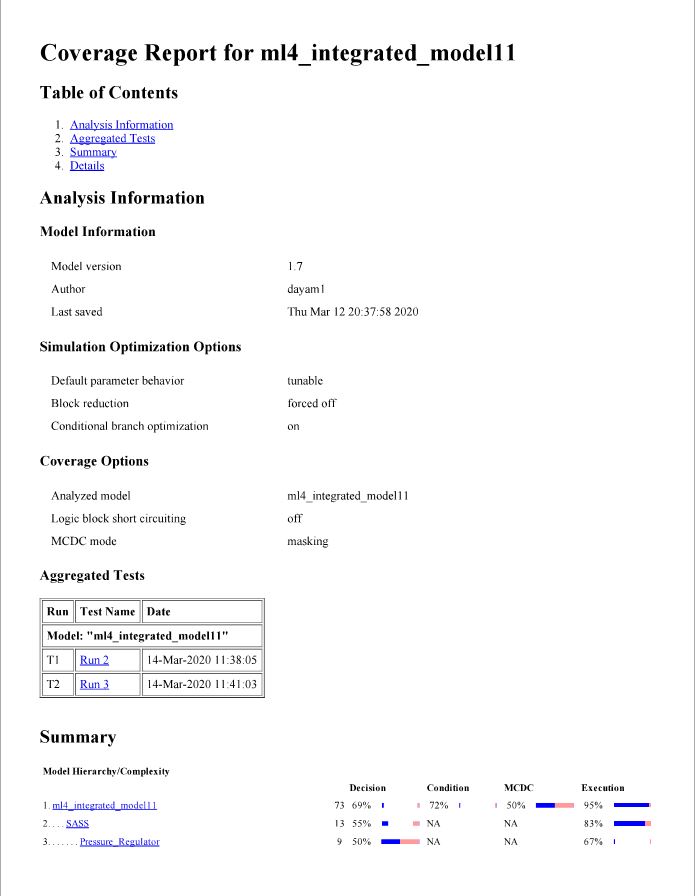


Figure 39: First Iteration of Code Coverage

SECOND ITERATION OF CODE COVERAGE

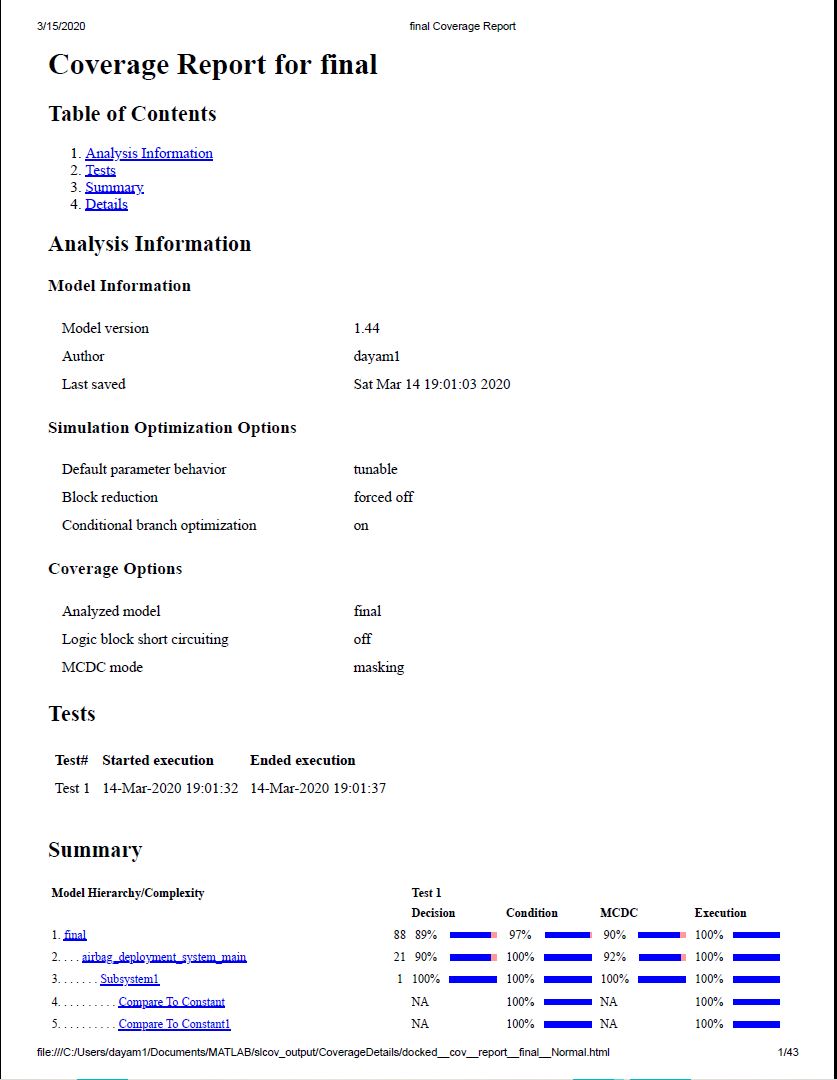


Figure 40: Second Iteration of Code Coverage

FINAL ITERATION OF CODE COVERAGE

* The coverage difference of -2% is found in the final model as the final system was created based on priority of each feature
* Moreover the decision chances also decreases making it less possible for certain paths to be used for all times

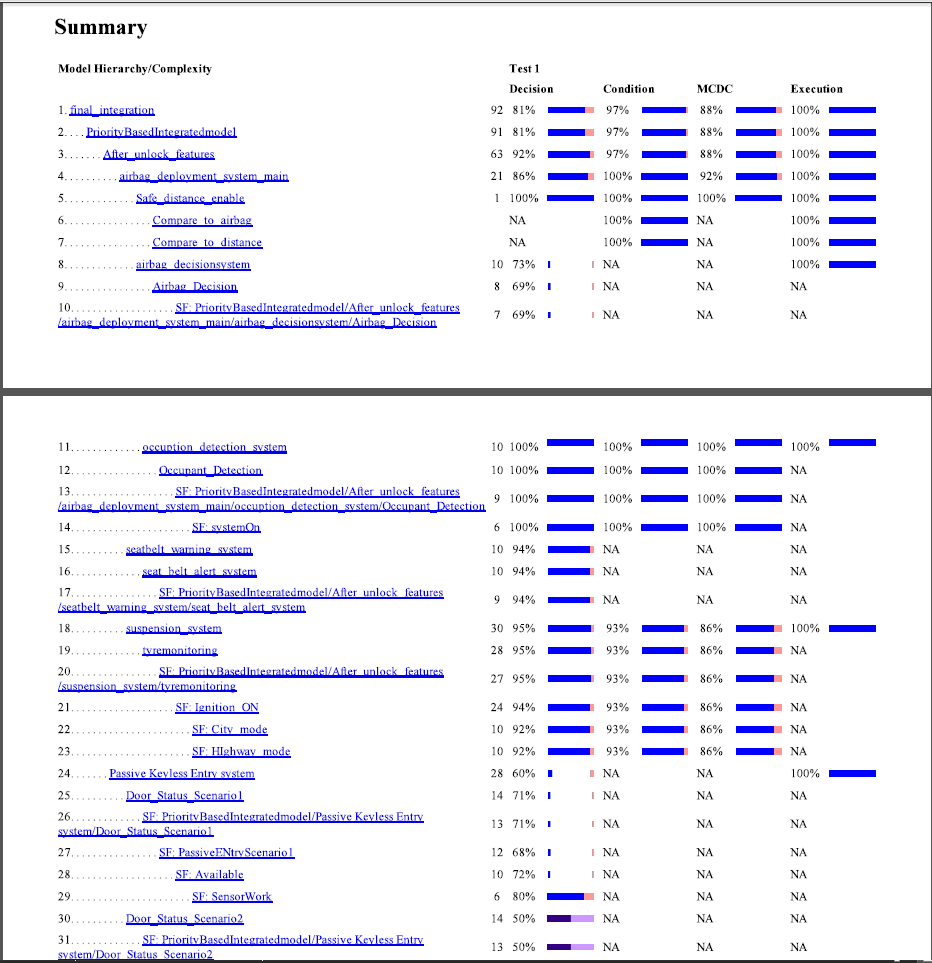


Figure 41: Final Iteration of Code Coverage

## METRIC ANALYSIS

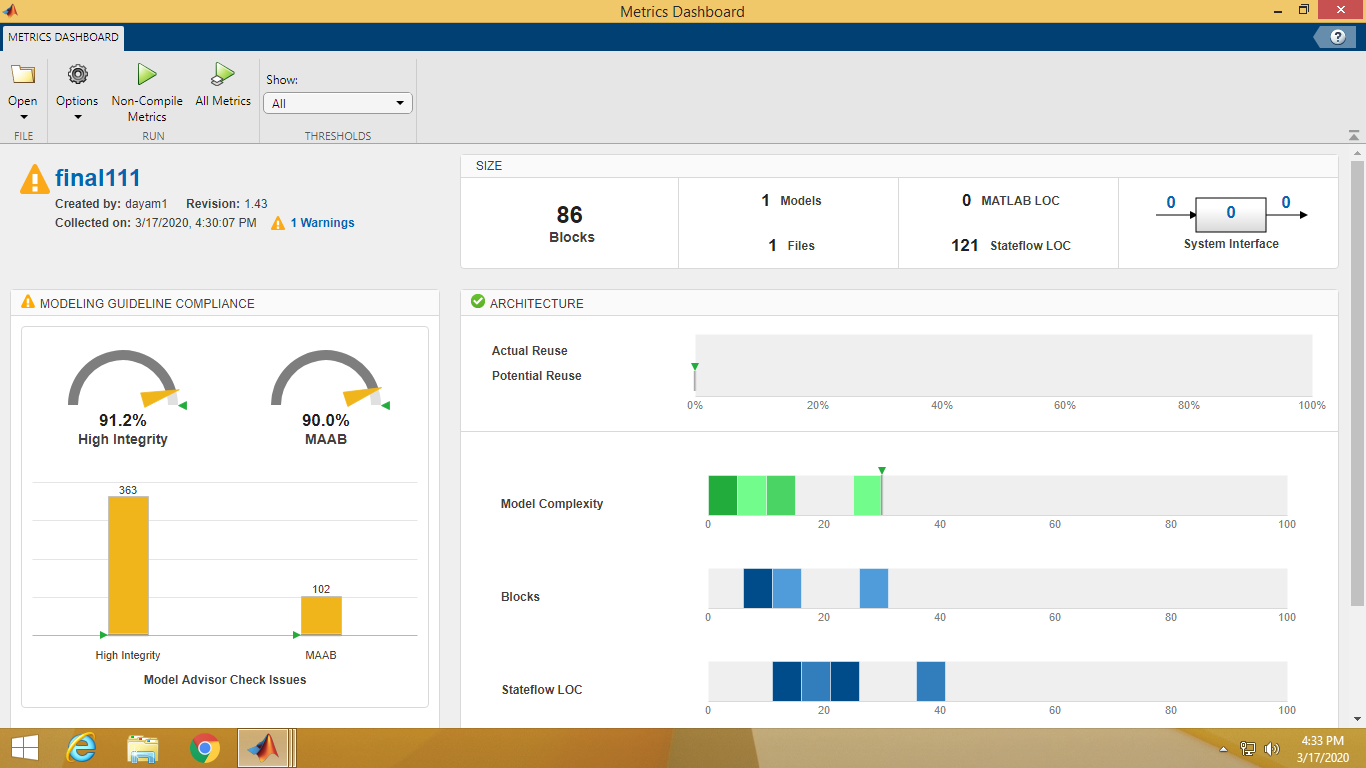


Figure 42: Metric Analysis

# LEARNINGS AND DIFFICULTIES OVERCOMED

## LEARNINGS

|  |  |  |  |
| --- | --- | --- | --- |
| SL NO. | TEAM | SOFTWARE | OTHERS |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |

Table 22: Learnings

## DIFFICULTIES OVERCOMED

### TECHNICAL

|  |  |  |
| --- | --- | --- |
| SL NO. | DIFFICULTIES | SOLUTIONS |
| 1 | Network issues during  team communications | Specific time of day chosen to get better signal strength |
| 2 | MATLAB not available. Some had previous versions while for others the license expired | Some were able to download the recent MATLAB 2020 version. However some had academic versions |
| 3 | Connectivity issues for a group interactive connect | Used Zoom meeting app to create a connect to explain work and part for each group members |
| 4 | Embedded C integration of system caused a great chaos within the individual outputs | Only some parts were implemented using the priority based feature to run the others |
| 5 | SimulIDE caused a great problem as the number of PINs were similar and already used | Had to decrease some components like the PWM for 2 systems. Some features were dropped |

Table 23: Technical Difficulties Overcomed

### NON-TECHNICAL

|  |  |  |
| --- | --- | --- |
| SL NO. | DIFFICULTIES | SOLUTIONS |
| 1 | Lockdown in Pune caused all the team members to stay away from each other | Frequent connects using the Zoom app was made to work on the integration |
| 2 | Misunderstanding in preparing for the presentation | Division of systems and parts of the presentation was divided equally and timed |
| 3 |  |  |

Table 24: Non-technical Difficulties Overcomed

# APPENDIX

## SIMULINK GENERATED EMBEDDED C CODE

ert\_main.c

/\*

\* File: ert\_main.c

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#include <stddef.h>

#include <stdio.h> /\* This ert\_main.c example uses printf/fflush \*/

#include "final\_integration.h" /\* Model's header file \*/

#include "rtwtypes.h"

/\*

\* Associating rt\_OneStep with a real-time clock or interrupt service routine

\* is what makes the generated code "real-time". The function rt\_OneStep is

\* always associated with the base rate of the model. Subrates are managed

\* by the base rate from inside the generated code. Enabling/disabling

\* interrupts and floating point context switches are target specific. This

\* example code indicates where these should take place relative to executing

\* the generated code step function. Overrun behavior should be tailored to

\* your application needs. This example simply sets an error status in the

\* real-time model and returns from rt\_OneStep.

\*/

void rt\_OneStep(void);

void rt\_OneStep(void)

{

static boolean\_T OverrunFlag = false;

/\* Disable interrupts here \*/

/\* Check for overrun \*/

if (OverrunFlag) {

rtmSetErrorStatus(final\_integration\_M, "Overrun");

return;

}

OverrunFlag = true;

/\* Save FPU context here (if necessary) \*/

/\* Re-enable timer or interrupt here \*/

/\* Set model inputs here \*/

/\* Step the model for base rate \*/

final\_integration\_step();

/\* Get model outputs here \*/

/\* Indicate task complete \*/

OverrunFlag = false;

/\* Disable interrupts here \*/

/\* Restore FPU context here (if necessary) \*/

/\* Enable interrupts here \*/

}

/\*

\* The example "main" function illustrates what is required by your

\* application code to initialize, execute, and terminate the generated code.

\* Attaching rt\_OneStep to a real-time clock is target specific. This example

\* illustrates how you do this relative to initializing the model.

\*/

int\_T main(int\_T argc, const char \*argv[])

{

/\* Unused arguments \*/

(void)(argc);

(void)(argv);

/\* Initialize model \*/

final\_integration\_initialize();

/\* Attach rt\_OneStep to a timer or interrupt service routine with

\* period 0.1 seconds (the model's base sample time) here. The

\* call syntax for rt\_OneStep is

\*

\* rt\_OneStep();

\*/

printf("Warning: The simulation will run forever. "

"Generated ERT main won't simulate model step behavior. "

"To change this behavior select the 'MAT-file logging' option.\n");

fflush((NULL));

while (rtmGetErrorStatus(final\_integration\_M) == (NULL)) {

/\* Perform other application tasks here \*/

}

/\* Disable rt\_OneStep() here \*/

/\* Terminate model \*/

final\_integration\_terminate();

return 0;

}

final\_integration.c

/\*

\* File: final\_integration.c

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#include "final\_integration.h"

#include "final\_integration\_private.h"

/\* Named constants for Chart: '<S12>/Airbag\_Decision' \*/

#define final\_integr\_IN\_NO\_ACTIVE\_CHILD ((uint8\_T)0U)

#define final\_integrati\_IN\_standbyState ((uint8\_T)3U)

#define final\_integratio\_IN\_normalState ((uint8\_T)2U)

#define final\_integration\_IN\_fireState ((uint8\_T)1U)

/\* Named constants for Chart: '<S13>/Occupant\_Detection' \*/

#define final\_integration\_IN\_systemOff ((uint8\_T)1U)

#define final\_integration\_IN\_systemOn ((uint8\_T)2U)

/\* Named constants for Chart: '<S8>/seat\_belt\_alert\_system' \*/

#define final\_integrati\_IN\_Seatoccupied ((uint8\_T)4U)

#define final\_integration\_IN\_Buzzer ((uint8\_T)1U)

#define final\_integration\_IN\_Idle\_state ((uint8\_T)2U)

#define final\_integration\_IN\_SeatbeltOn ((uint8\_T)3U)

/\* Named constants for Chart: '<S9>/tyremonitoring' \*/

#define final\_integrati\_IN\_HIghway\_mode ((uint8\_T)2U)

#define final\_integrati\_IN\_Ignition\_OFF ((uint8\_T)1U)

#define final\_integratio\_IN\_Ignition\_ON ((uint8\_T)2U)

#define final\_integration\_IN\_City\_mode ((uint8\_T)1U)

/\* Named constants for Chart: '<S4>/Door\_Status\_Scenario1' \*/

#define final\_integrat\_IN\_VehicleSensor ((uint8\_T)3U)

#define final\_integration\_IN\_Available ((uint8\_T)1U)

#define final\_integration\_IN\_DoorSensor ((uint8\_T)1U)

#define final\_integration\_IN\_OutofRange ((uint8\_T)2U)

#define final\_integration\_IN\_ScanOff ((uint8\_T)2U)

#define final\_integration\_IN\_SensorWork ((uint8\_T)1U)

#define final\_integration\_IN\_Signal ((uint8\_T)2U)

/\* Block signals (default storage) \*/

B\_final\_integration\_T final\_integration\_B;

/\* Block states (default storage) \*/

DW\_final\_integration\_T final\_integration\_DW;

/\* Real-time model \*/

RT\_MODEL\_final\_integration\_T final\_integration\_M\_;

RT\_MODEL\_final\_integration\_T \*const final\_integration\_M = &final\_integration\_M\_;

/\* Forward declaration for local functions \*/

static void final\_integration\_Ignition\_ON(void);

/\* Function for Chart: '<S9>/tyremonitoring' \*/

static void final\_integration\_Ignition\_ON(void)

{

if (final\_integration\_B.seatoccupiedout == 0.0) {

final\_integration\_DW.is\_Ignition\_ON = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

final\_integration\_DW.is\_c6\_final\_integration =

final\_integrati\_IN\_Ignition\_OFF;

} else if (final\_integration\_DW.is\_Ignition\_ON ==

final\_integration\_IN\_City\_mode) {

if (final\_integration\_B.FromWs[6] == 1.0) {

final\_integration\_DW.is\_Ignition\_ON = final\_integrati\_IN\_HIghway\_mode;

}

} else {

/\* case IN\_HIghway\_mode: \*/

if (final\_integration\_B.FromWs[6] == 0.0) {

final\_integration\_DW.is\_Ignition\_ON = final\_integration\_IN\_City\_mode;

}

}

}

/\* Model step function \*/

void final\_integration\_step(void)

{

boolean\_T ig\_sb;

/\* FromWorkspace: '<S5>/FromWs' \*/

{

real\_T \*pDataValues = (real\_T \*) final\_integration\_DW.FromWs\_PWORK.DataPtr;

real\_T \*pTimeValues = (real\_T \*) final\_integration\_DW.FromWs\_PWORK.TimePtr;

int\_T currTimeIndex = final\_integration\_DW.FromWs\_IWORK.PrevIndex;

real\_T t = final\_integration\_M->Timing.t[0];

/\* Get index \*/

if (t <= pTimeValues[0]) {

currTimeIndex = 0;

} else if (t >= pTimeValues[221]) {

currTimeIndex = 220;

} else {

if (t < pTimeValues[currTimeIndex]) {

while (t < pTimeValues[currTimeIndex]) {

currTimeIndex--;

}

} else {

while (t >= pTimeValues[currTimeIndex + 1]) {

currTimeIndex++;

}

}

}

final\_integration\_DW.FromWs\_IWORK.PrevIndex = currTimeIndex;

/\* Post output \*/

{

real\_T t1 = pTimeValues[currTimeIndex];

real\_T t2 = pTimeValues[currTimeIndex + 1];

if (t1 == t2) {

if (t < t1) {

{

int\_T elIdx;

for (elIdx = 0; elIdx < 7; ++elIdx) {

(&final\_integration\_B.FromWs[0])[elIdx] =

pDataValues[currTimeIndex];

pDataValues += 222;

}

}

} else {

{

int\_T elIdx;

for (elIdx = 0; elIdx < 7; ++elIdx) {

(&final\_integration\_B.FromWs[0])[elIdx] =

pDataValues[currTimeIndex + 1];

pDataValues += 222;

}

}

}

} else {

real\_T f1 = (t2 - t) / (t2 - t1);

real\_T f2 = 1.0 - f1;

real\_T d1;

real\_T d2;

int\_T TimeIndex= currTimeIndex;

{

int\_T elIdx;

for (elIdx = 0; elIdx < 7; ++elIdx) {

d1 = pDataValues[TimeIndex];

d2 = pDataValues[TimeIndex + 1];

(&final\_integration\_B.FromWs[0])[elIdx] = (real\_T) rtInterpolate(d1,

d2, f1, f2);

pDataValues += 222;

}

}

}

}

}

/\* FromWorkspace: '<S2>/FromWs' \*/

{

real\_T \*pDataValues = (real\_T \*) final\_integration\_DW.FromWs\_PWORK\_n.DataPtr;

real\_T \*pTimeValues = (real\_T \*) final\_integration\_DW.FromWs\_PWORK\_n.TimePtr;

int\_T currTimeIndex = final\_integration\_DW.FromWs\_IWORK\_g.PrevIndex;

real\_T t = final\_integration\_M->Timing.t[0];

/\* Get index \*/

if (t <= pTimeValues[0]) {

currTimeIndex = 0;

} else if (t >= pTimeValues[9]) {

currTimeIndex = 8;

} else {

if (t < pTimeValues[currTimeIndex]) {

while (t < pTimeValues[currTimeIndex]) {

currTimeIndex--;

}

} else {

while (t >= pTimeValues[currTimeIndex + 1]) {

currTimeIndex++;

}

}

}

final\_integration\_DW.FromWs\_IWORK\_g.PrevIndex = currTimeIndex;

/\* Post output \*/

{

real\_T t1 = pTimeValues[currTimeIndex];

real\_T t2 = pTimeValues[currTimeIndex + 1];

if (t1 == t2) {

if (t < t1) {

{

int\_T elIdx;

for (elIdx = 0; elIdx < 2; ++elIdx) {

(&final\_integration\_B.FromWs\_h[0])[elIdx] =

pDataValues[currTimeIndex];

pDataValues += 10;

}

}

} else {

{

int\_T elIdx;

for (elIdx = 0; elIdx < 2; ++elIdx) {

(&final\_integration\_B.FromWs\_h[0])[elIdx] =

pDataValues[currTimeIndex + 1];

pDataValues += 10;

}

}

}

} else {

real\_T f1 = (t2 - t) / (t2 - t1);

real\_T f2 = 1.0 - f1;

real\_T d1;

real\_T d2;

int\_T TimeIndex= currTimeIndex;

{

int\_T elIdx;

for (elIdx = 0; elIdx < 2; ++elIdx) {

d1 = pDataValues[TimeIndex];

d2 = pDataValues[TimeIndex + 1];

(&final\_integration\_B.FromWs\_h[0])[elIdx] = (real\_T) rtInterpolate

(d1, d2, f1, f2);

pDataValues += 10;

}

}

}

}

}

/\* Chart: '<S4>/Door\_Status\_Scenario2' \*/

if (final\_integration\_DW.is\_active\_c5\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c5\_final\_integration = 1U;

final\_integration\_DW.is\_c5\_final\_integration = final\_integration\_IN\_ScanOff;

} else if (final\_integration\_DW.is\_c5\_final\_integration ==

final\_integration\_IN\_Available) {

if (final\_integration\_DW.is\_Available == final\_integration\_IN\_SensorWork) {

if (final\_integration\_B.FromWs\_h[1] == 0.0) {

switch (final\_integration\_DW.is\_SensorWork) {

case final\_integration\_IN\_DoorSensor:

final\_integration\_DW.count = 1.0;

final\_integration\_DW.is\_SensorWork = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

case final\_integration\_IN\_OutofRange:

final\_integration\_DW.count = 749.0;

final\_integration\_DW.is\_SensorWork = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

case final\_integrat\_IN\_VehicleSensor:

final\_integration\_DW.count = 99.0;

final\_integration\_DW.is\_SensorWork = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

}

final\_integration\_DW.is\_Available = final\_integration\_IN\_Signal;

} else {

switch (final\_integration\_DW.is\_SensorWork) {

case final\_integration\_IN\_DoorSensor:

final\_integration\_B.DoorEntryStatus\_S2 = 0.0;

final\_integration\_DW.count--;

break;

case final\_integration\_IN\_OutofRange:

final\_integration\_B.DoorEntryStatus\_S2 = 1.0;

if (final\_integration\_DW.count == 749.0) {

final\_integration\_DW.is\_SensorWork = final\_integrat\_IN\_VehicleSensor;

final\_integration\_B.DoorEntryStatus\_S2 = 1.0;

final\_integration\_DW.count = 749.0;

} else {

final\_integration\_DW.count--;

}

break;

default:

/\* case IN\_VehicleSensor: \*/

final\_integration\_B.DoorEntryStatus\_S2 = 1.0;

if (final\_integration\_DW.count == 99.0) {

final\_integration\_DW.is\_SensorWork = final\_integration\_IN\_DoorSensor;

final\_integration\_B.DoorEntryStatus\_S2 = 0.0;

final\_integration\_DW.count = 99.0;

} else {

final\_integration\_DW.count--;

}

break;

}

}

} else {

/\* case IN\_Signal: \*/

if (final\_integration\_B.FromWs\_h[1] < 1000.0) {

final\_integration\_DW.is\_Available = final\_integration\_IN\_SensorWork;

final\_integration\_DW.is\_SensorWork = final\_integration\_IN\_OutofRange;

final\_integration\_B.DoorEntryStatus\_S2 = 1.0;

final\_integration\_DW.count = 899.0;

}

}

} else {

/\* case IN\_ScanOff: \*/

final\_integration\_DW.is\_c5\_final\_integration =

final\_integration\_IN\_Available;

final\_integration\_DW.is\_Available = final\_integration\_IN\_Signal;

}

/\* End of Chart: '<S4>/Door\_Status\_Scenario2' \*/

/\* Outputs for Enabled SubSystem: '<S1>/After\_unlock\_features ' incorporates:

\* EnablePort: '<S3>/Enable'

\*/

if (final\_integration\_B.DoorEntryStatus\_S2 > 0.0) {

/\* Chart: '<S8>/seat\_belt\_alert\_system' \*/

if (final\_integration\_DW.is\_active\_c3\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c3\_final\_integration = 1U;

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_Idle\_state;

final\_integration\_B.seatoccupiedout = 0.0;

final\_integration\_B.buzzer = 0.0;

} else {

switch (final\_integration\_DW.is\_c3\_final\_integration) {

case final\_integration\_IN\_Buzzer:

final\_integration\_B.buzzer = 1.0;

if (final\_integration\_B.FromWs[0] == 0.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_Idle\_state;

final\_integration\_B.seatoccupiedout = 0.0;

final\_integration\_B.buzzer = 0.0;

} else {

if (final\_integration\_B.FromWs[1] == 1.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_SeatbeltOn;

final\_integration\_B.buzzer = 0.0;

}

}

break;

case final\_integration\_IN\_Idle\_state:

final\_integration\_B.seatoccupiedout = 0.0;

final\_integration\_B.buzzer = 0.0;

if (final\_integration\_B.FromWs[0] == 1.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integrati\_IN\_Seatoccupied;

final\_integration\_B.seatoccupiedout = 1.0;

}

break;

case final\_integration\_IN\_SeatbeltOn:

final\_integration\_B.buzzer = 0.0;

if (final\_integration\_B.FromWs[0] == 0.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_Idle\_state;

final\_integration\_B.seatoccupiedout = 0.0;

final\_integration\_B.buzzer = 0.0;

}

break;

default:

/\* case IN\_Seatoccupied: \*/

final\_integration\_B.seatoccupiedout = 1.0;

if (final\_integration\_B.FromWs[1] == 1.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_SeatbeltOn;

final\_integration\_B.buzzer = 0.0;

} else {

if (final\_integration\_B.FromWs[1] == 0.0) {

final\_integration\_DW.is\_c3\_final\_integration =

final\_integration\_IN\_Buzzer;

final\_integration\_B.buzzer = 1.0;

}

}

break;

}

}

/\* End of Chart: '<S8>/seat\_belt\_alert\_system' \*/

/\* Logic: '<S13>/Logical Operator' incorporates:

\* DataTypeConversion: '<S13>/Cast To Boolean'

\* Logic: '<S3>/NOT'

\*/

ig\_sb = ((final\_integration\_B.seatoccupiedout != 0.0) &&

(!(final\_integration\_B.buzzer != 0.0)));

/\* Chart: '<S13>/Occupant\_Detection' \*/

if (final\_integration\_DW.is\_active\_c2\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c2\_final\_integration = 1U;

final\_integration\_DW.is\_c2\_final\_integration =

final\_integration\_IN\_systemOn;

} else if (final\_integration\_DW.is\_c2\_final\_integration ==

final\_integration\_IN\_systemOff) {

if (!ig\_sb) {

final\_integration\_B.airbag\_en2 = 0.0;

final\_integration\_DW.is\_c2\_final\_integration =

final\_integration\_IN\_systemOn;

}

} else {

/\* case IN\_systemOn: \*/

if (ig\_sb) {

final\_integration\_B.airbag\_en2 = 1.0;

final\_integration\_DW.is\_c2\_final\_integration =

final\_integration\_IN\_systemOff;

}

}

/\* End of Chart: '<S13>/Occupant\_Detection' \*/

/\* Outputs for Enabled SubSystem: '<S7>/airbag\_decisionsystem' incorporates:

\* EnablePort: '<S12>/Enable'

\*/

/\* Logic: '<S7>/AND' incorporates:

\* Constant: '<S15>/Constant'

\* DataTypeConversion: '<S13>/Cast To Boolean2'

\* Logic: '<S10>/Logical Operator'

\* RelationalOperator: '<S15>/Compare'

\*/

if ((final\_integration\_B.airbag\_en2 != 0.0) && (final\_integration\_B.FromWs[3]

>= 20.0) && (final\_integration\_B.airbag\_en2 != 0.0)) {

/\* Chart: '<S12>/Airbag\_Decision' incorporates:

\* Constant: '<S12>/initial\_vol'

\* DiscreteIntegrator: '<S12>/acceleration\_integrator'

\* Sum: '<S12>/Add'

\*/

if (final\_integration\_DW.is\_active\_c1\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c1\_final\_integration = 1U;

final\_integration\_DW.is\_c1\_final\_integration =

final\_integratio\_IN\_normalState;

} else {

switch (final\_integration\_DW.is\_c1\_final\_integration) {

case final\_integration\_IN\_fireState:

if (final\_integration\_DW.acceleration\_integrator\_DSTATE + 1.0 < 12.88)

{

final\_integration\_DW.is\_c1\_final\_integration =

final\_integrati\_IN\_standbyState;

}

break;

case final\_integratio\_IN\_normalState:

if (final\_integration\_DW.acceleration\_integrator\_DSTATE >=

final\_integration\_DW.Vth) {

final\_integration\_DW.is\_c1\_final\_integration =

final\_integrati\_IN\_standbyState;

}

break;

default:

/\* case IN\_standbyState: \*/

if ((final\_integration\_DW.acceleration\_integrator\_DSTATE + 1.0 >=

22.54) || (!(final\_integration\_DW.acceleration\_integrator\_DSTATE <

final\_integration\_DW.Vth))) {

final\_integration\_DW.is\_c1\_final\_integration =

final\_integration\_IN\_fireState;

} else {

final\_integration\_DW.is\_c1\_final\_integration =

final\_integratio\_IN\_normalState;

}

break;

}

}

/\* End of Chart: '<S12>/Airbag\_Decision' \*/

/\* Update for DiscreteIntegrator: '<S12>/acceleration\_integrator' \*/

final\_integration\_DW.acceleration\_integrator\_DSTATE += 0.1 \*

final\_integration\_B.FromWs[4];

}

/\* End of Logic: '<S7>/AND' \*/

/\* End of Outputs for SubSystem: '<S7>/airbag\_decisionsystem' \*/

/\* Chart: '<S9>/tyremonitoring' \*/

if (final\_integration\_DW.is\_active\_c6\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c6\_final\_integration = 1U;

final\_integration\_DW.is\_c6\_final\_integration =

final\_integrati\_IN\_Ignition\_OFF;

} else if (final\_integration\_DW.is\_c6\_final\_integration ==

final\_integrati\_IN\_Ignition\_OFF) {

if (final\_integration\_B.seatoccupiedout == 1.0) {

final\_integration\_DW.is\_c6\_final\_integration =

final\_integratio\_IN\_Ignition\_ON;

final\_integration\_DW.is\_Ignition\_ON = final\_integration\_IN\_City\_mode;

}

} else {

/\* case IN\_Ignition\_ON: \*/

final\_integration\_Ignition\_ON();

}

/\* End of Chart: '<S9>/tyremonitoring' \*/

}

/\* End of Outputs for SubSystem: '<S1>/After\_unlock\_features ' \*/

/\* Chart: '<S4>/Door\_Status\_Scenario1' \*/

if (final\_integration\_DW.is\_active\_c4\_final\_integration == 0U) {

final\_integration\_DW.is\_active\_c4\_final\_integration = 1U;

final\_integration\_DW.is\_c4\_final\_integration = final\_integration\_IN\_ScanOff;

} else if (final\_integration\_DW.is\_c4\_final\_integration ==

final\_integration\_IN\_Available) {

if (final\_integration\_DW.is\_Available\_o == final\_integration\_IN\_SensorWork)

{

if (final\_integration\_B.FromWs\_h[0] == 900.0) {

switch (final\_integration\_DW.is\_SensorWork\_a) {

case final\_integration\_IN\_DoorSensor:

final\_integration\_DW.count\_l = 101.0;

final\_integration\_DW.is\_SensorWork\_a = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

case final\_integration\_IN\_OutofRange:

final\_integration\_DW.count\_l = 901.0;

final\_integration\_DW.is\_SensorWork\_a = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

case final\_integrat\_IN\_VehicleSensor:

final\_integration\_DW.count\_l = 751.0;

final\_integration\_DW.is\_SensorWork\_a = final\_integr\_IN\_NO\_ACTIVE\_CHILD;

break;

}

final\_integration\_DW.is\_Available\_o = final\_integration\_IN\_Signal;

} else {

switch (final\_integration\_DW.is\_SensorWork\_a) {

case final\_integration\_IN\_DoorSensor:

if (final\_integration\_DW.count\_l > final\_integration\_DW.proximity) {

final\_integration\_DW.is\_SensorWork\_a =

final\_integrat\_IN\_VehicleSensor;

final\_integration\_DW.proximity = 750.0;

final\_integration\_DW.count\_l = 101.0;

} else {

final\_integration\_DW.count\_l++;

}

break;

case final\_integration\_IN\_OutofRange:

final\_integration\_DW.count\_l++;

break;

default:

/\* case IN\_VehicleSensor: \*/

if (final\_integration\_DW.count\_l > final\_integration\_DW.proximity) {

final\_integration\_DW.is\_SensorWork\_a =

final\_integration\_IN\_OutofRange;

final\_integration\_DW.proximity = 900.0;

final\_integration\_DW.count\_l = 751.0;

} else {

final\_integration\_DW.count\_l++;

}

break;

}

}

} else {

/\* case IN\_Signal: \*/

if (final\_integration\_B.FromWs\_h[0] > 0.0) {

final\_integration\_DW.is\_Available\_o = final\_integration\_IN\_SensorWork;

final\_integration\_DW.is\_SensorWork\_a = final\_integration\_IN\_DoorSensor;

final\_integration\_DW.proximity = 100.0;

final\_integration\_DW.count\_l = 0.0;

}

}

} else {

/\* case IN\_ScanOff: \*/

final\_integration\_DW.is\_c4\_final\_integration =

final\_integration\_IN\_Available;

final\_integration\_DW.is\_Available\_o = final\_integration\_IN\_Signal;

}

/\* End of Chart: '<S4>/Door\_Status\_Scenario1' \*/

/\* Update absolute time for base rate \*/

/\* The "clockTick0" counts the number of times the code of this task has

\* been executed. The absolute time is the multiplication of "clockTick0"

\* and "Timing.stepSize0". Size of "clockTick0" ensures timer will not

\* overflow during the application lifespan selected.

\*/

final\_integration\_M->Timing.t[0] =

(++final\_integration\_M->Timing.clockTick0) \*

final\_integration\_M->Timing.stepSize0;

{

/\* Update absolute timer for sample time: [0.1s, 0.0s] \*/

/\* The "clockTick1" counts the number of times the code of this task has

\* been executed. The resolution of this integer timer is 0.1, which is the step size

\* of the task. Size of "clockTick1" ensures timer will not overflow during the

\* application lifespan selected.

\*/

final\_integration\_M->Timing.clockTick1++;

}

}

/\* Model initialize function \*/

void final\_integration\_initialize(void)

{

/\* Registration code \*/

{

/\* Setup solver object \*/

rtsiSetSimTimeStepPtr(&final\_integration\_M->solverInfo,

&final\_integration\_M->Timing.simTimeStep);

rtsiSetTPtr(&final\_integration\_M->solverInfo, &rtmGetTPtr

(final\_integration\_M));

rtsiSetStepSizePtr(&final\_integration\_M->solverInfo,

&final\_integration\_M->Timing.stepSize0);

rtsiSetErrorStatusPtr(&final\_integration\_M->solverInfo, ((const char\_T \*\*)

(&rtmGetErrorStatus(final\_integration\_M))));

rtsiSetRTModelPtr(&final\_integration\_M->solverInfo, final\_integration\_M);

}

rtsiSetSimTimeStep(&final\_integration\_M->solverInfo, MAJOR\_TIME\_STEP);

rtsiSetSolverName(&final\_integration\_M->solverInfo,"FixedStepDiscrete");

rtmSetTPtr(final\_integration\_M, &final\_integration\_M->Timing.tArray[0]);

final\_integration\_M->Timing.stepSize0 = 0.1;

/\* Start for FromWorkspace: '<S5>/FromWs' \*/

{

static real\_T pTimeValues0[] = { 0.0, 1.0, 1.0, 2.0, 2.0, 3.0, 3.0, 4.0, 4.0,

5.0, 5.0, 6.0, 6.0, 7.0, 7.0, 8.0, 8.0, 9.0, 9.0, 10.0, 10.0, 11.0, 11.0,

12.0, 12.0, 13.0, 13.0, 14.0, 14.0, 15.0, 15.0, 16.0, 16.0, 17.0, 17.0,

18.0, 18.0, 19.0, 19.0, 20.0, 20.0, 21.0, 21.0, 22.0, 22.0, 23.0, 23.0,

24.0, 24.0, 25.0, 25.0, 26.0, 26.0, 27.0, 27.0, 28.0, 28.0, 29.0, 29.0,

30.0, 30.0, 31.0, 31.0, 31.5, 31.5, 31.9, 31.9, 32.3, 32.3, 32.7, 32.7,

33.1, 33.1, 33.5, 33.5, 33.9, 33.9, 34.3, 34.3, 34.7, 34.7, 35.1, 35.1,

35.5, 35.5, 35.9, 35.9, 36.3, 36.3, 36.7, 36.7, 37.1, 37.1, 37.5, 37.5,

37.9, 37.9, 38.3, 38.3, 39.0, 39.0, 40.0, 40.0, 41.0, 41.0, 42.0, 42.0,

43.0, 43.0, 44.0, 44.0, 45.0, 45.0, 46.0, 46.0, 47.0, 47.0, 48.0, 48.0,

49.0, 49.0, 50.0, 50.0, 51.0, 51.0, 52.0, 52.0, 53.0, 53.0, 54.0, 54.0,

55.0, 55.0, 56.0, 56.0, 57.0, 57.0, 58.0, 58.0, 59.0, 59.0, 60.0, 60.0,

61.0, 61.0, 62.0, 62.0, 63.0, 63.0, 64.0, 64.0, 65.0, 65.0, 66.0, 66.0,

67.0, 67.0, 68.0, 68.0, 69.0, 69.0, 70.0, 70.0, 71.0, 71.0, 72.0, 72.0,

73.0, 73.0, 74.0, 74.0, 75.0, 75.0, 76.0, 76.0, 77.0, 77.0, 78.0, 78.0,

79.0, 79.0, 80.0, 80.0, 81.0, 81.0, 82.0, 82.0, 83.0, 83.0, 84.0, 84.0,

85.0, 85.0, 86.0, 86.0, 87.0, 87.0, 88.0, 88.0, 89.0, 89.0, 90.0, 90.0,

91.0, 91.0, 92.0, 92.0, 93.0, 93.0, 94.0, 94.0, 95.0, 95.0, 96.0, 96.0,

97.0, 97.0, 98.0, 98.0, 99.0, 99.0, 100.0 } ;

static real\_T pDataValues0[] = { 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0,

0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0,

0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0,

0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 1.0,

1.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0,

0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0,

1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

78.0, 78.0, 68.0, 66.0, 55.0, 55.0, 87.0, 87.0, 43.0, 43.0, 30.0, 30.0,

20.0, 20.0, 10.0, 10.0, 70.0, 70.0, 88.0, 88.0, 99.0, 99.0, 66.0, 66.0,

54.0, 54.0, 34.0, 34.0, 44.0, 44.0, 89.0, 89.0, 86.0, 86.0, 10.0, 10.0,

54.0, 54.0, 44.0, 44.0, 99.0, 99.0, 75.0, 106.0, 106.0, 106.0, 106.0, 50.0,

25.0, 25.0, 33.0, 33.0, 26.0, 26.0, 57.0, 57.0, 78.0, 78.0, 57.0, 57.0,

78.0, 78.0, 86.0, 10.0, 10.0, 10.0, 10.0, 54.0, 54.0, 54.0, 54.0, 44.0,

44.0, 44.0, 44.0, 99.0, 99.0, 99.0, 99.0, 75.0, 75.0, 106.0, 106.0, 106.0,

106.0, 106.0, 106.0, 106.0, 106.0, 50.0, 50.0, 25.0, 25.0, 25.0, 25.0,

33.0, 33.0, 33.0, 33.0, 66.0, 66.0, 55.0, 55.0, 87.0, 87.0, 43.0, 43.0,

30.0, 30.0, 20.0, 20.0, 10.0, 10.0, 70.0, 70.0, 88.0, 88.0, 99.0, 99.0,

66.0, 66.0, 54.0, 54.0, 34.0, 34.0, 44.0, 78.0, 78.0, 86.0, 10.0, 10.0,

54.0, 54.0, 44.0, 44.0, 99.0, 99.0, 75.0, 106.0, 106.0, 106.0, 106.0, 50.0,

25.0, 25.0, 33.0, 33.0, 66.0, 78.0, 86.0, 10.0, 10.0, 54.0, 54.0, 44.0,

44.0, 99.0, 99.0, 75.0, 106.0, 106.0, 78.0, 78.0, 44.0, 99.0, 55.0, 55.0,

55.0, 87.0, 87.0, 43.0, 43.0, 30.0, 30.0, 20.0, 20.0, 10.0, 10.0, 70.0,

55.0, 55.0, 87.0, 87.0, 43.0, 43.0, 30.0, 30.0, 20.0, 20.0, 10.0, 10.0,

70.0, 70.0, 88.0, 88.0, 99.0, 99.0, 66.0, 66.0, 54.0, 54.0, 34.0, 34.0,

44.0, 78.0, 78.0, 86.0, 10.0, 10.0, 106.0, 106.0, 50.0, 86.0, 10.0, 10.0,

54.0, 54.0, 44.0, 44.0, 99.0, 99.0, 75.0, 21.0, 22.0, 22.0, 24.0, 24.0,

10.0, 10.0, 18.0, 18.0, 23.0, 23.0, 29.0, 29.0, 50.0, 50.0, 23.0, 23.0,

16.0, 16.0, 13.0, 13.0, 21.0, 21.0, 22.0, 22.0, 19.0, 19.0, 14.0, 14.0,

24.0, 24.0, 15.0, 15.0, 19.0, 19.0, 11.0, 11.0, 22.0, 22.0, 17.0, 17.0,

22.0, 22.0, 24.0, 24.0, 10.0, 10.0, 18.0, 18.0, 23.0, 23.0, 29.0, 29.0,

50.0, 50.0, 23.0, 23.0, 50.0, 50.0, 23.0, 23.0, 23.0, 19.0, 19.0, 19.0,

11.0, 11.0, 11.0, 11.0, 22.0, 22.0, 22.0, 22.0, 17.0, 17.0, 17.0, 17.0,

22.0, 22.0, 22.0, 22.0, 24.0, 24.0, 24.0, 24.0, 10.0, 10.0, 10.0, 10.0,

18.0, 18.0, 18.0, 18.0, 23.0, 23.0, 23.0, 23.0, 29.0, 29.0, 17.0, 17.0,

17.0, 22.0, 22.0, 24.0, 24.0, 10.0, 10.0, 18.0, 18.0, 23.0, 23.0, 29.0,

29.0, 50.0, 50.0, 23.0, 23.0, 50.0, 50.0, 23.0, 23.0, 23.0, 19.0, 19.0,

11.0, 11.0, 22.0, 22.0, 17.0, 17.0, 22.0, 22.0, 24.0, 24.0, 10.0, 10.0,

18.0, 18.0, 23.0, 23.0, 29.0, 17.0, 17.0, 22.0, 22.0, 24.0, 24.0, 10.0,

10.0, 18.0, 18.0, 23.0, 23.0, 29.0, 29.0, 50.0, 50.0, 23.0, 23.0, 50.0,

21.0, 22.0, 29.0, 50.0, 24.0, 24.0, 10.0, 10.0, 18.0, 18.0, 23.0, 23.0,

29.0, 29.0, 50.0, 50.0, 23.0, 23.0, 17.0, 22.0, 22.0, 24.0, 24.0, 10.0,

10.0, 18.0, 18.0, 23.0, 23.0, 29.0, 29.0, 50.0, 50.0, 23.0, 23.0, 50.0,

50.0, 23.0, 23.0, 23.0, 19.0, 19.0, 11.0, 11.0, 22.0, 22.0, 17.0, 17.0,

23.0, 29.0, 17.0, 10.0, 18.0, 18.0, 23.0, 23.0, 29.0, 29.0, 50.0, 50.0,

23.0, 9.0, 12.0, 12.0, 45.0, 45.0, 54.0, 54.0, 67.0, 67.0, 12.0, 12.0,

47.0, 47.0, 34.0, 34.0, 67.0, 67.0, 23.0, 23.0, 89.0, 89.0, 23.0, 23.0,

52.0, 52.0, 9.0, 9.0, 76.0, 76.0, 13.0, 13.0, 67.0, 67.0, 34.0, 34.0, 90.0,

90.0, 28.0, 28.0, 49.0, 49.0, 12.0, 12.0, 45.0, 45.0, 54.0, 54.0, 67.0,

67.0, 12.0, 12.0, 47.0, 47.0, 34.0, 34.0, 67.0, 67.0, 34.0, 34.0, 67.0,

67.0, 67.0, 34.0, 34.0, 34.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0,

90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 90.0, 45.0,

45.0, 54.0, 54.0, 54.0, 54.0, 67.0, 67.0, 67.0, 67.0, 12.0, 12.0, 12.0,

12.0, 47.0, 47.0, 12.0, 12.0, 45.0, 45.0, 54.0, 54.0, 67.0, 67.0, 20.0,

20.0, 50.0, 50.0, 34.0, 34.0, 67.0, 67.0, 23.0, 23.0, 89.0, 89.0, 23.0,

23.0, 52.0, 52.0, 76.0, 76.0, 50.0, 50.0, 13.0, 13.0, 67.0, 67.0, 45.0,

45.0, 14.0, 14.0, 88.0, 88.0, 49.0, 49.0, 42.0, 42.0, 45.0, 45.0, 24.0,

24.0, 67.0, 67.0, 12.0, 12.0, 99.0, 99.0, 34.0, 34.0, 67.0, 67.0, 50.0,

50.0, 99.0, 99.0, 9.0, 12.0, 9.0, 12.0, 50.0, 50.0, 45.0, 45.0, 54.0, 54.0,

67.0, 67.0, 12.0, 12.0, 47.0, 47.0, 34.0, 34.0, 67.0, 67.0, 45.0, 45.0,

54.0, 54.0, 67.0, 67.0, 20.0, 20.0, 50.0, 50.0, 34.0, 34.0, 67.0, 67.0,

23.0, 23.0, 89.0, 89.0, 23.0, 23.0, 52.0, 52.0, 76.0, 76.0, 50.0, 50.0,

13.0, 13.0, 67.0, 67.0, 42.0, 45.0, 45.0, 99.0, 99.0, 34.0, 34.0, 67.0,

67.0, 50.0, 50.0, 99.0, 99.0, -100.0, -80.0, -80.0, 60.0, 60.0, 45.0, 45.0,

65.0, 65.0, 100.0, 100.0, 76.0, 76.0, 23.0, 23.0, 98.0, 98.0, 45.0, 45.0,

119.0, 119.0, 102.0, 102.0, 99.0, 99.0, 66.0, 66.0, 34.0, 34.0, 56.0, 56.0,

22.0, 22.0, 97.0, 97.0, 67.0, 67.0, 9.0, 9.0, 78.0, 78.0, -80.0, -80.0,

60.0, 60.0, 45.0, 45.0, 65.0, 65.0, 100.0, 100.0, 76.0, 76.0, 23.0, 23.0,

98.0, 98.0, 23.0, 23.0, 0.0, 0.0, 0.0, 97.0, 97.0, 97.0, 67.0, 67.0, 67.0,

67.0, 9.0, 9.0, 9.0, 9.0, 78.0, 78.0, 78.0, 78.0, -80.0, -80.0, -80.0,

-80.0, 60.0, 60.0, 60.0, 60.0, 45.0, 45.0, 45.0, 45.0, 65.0, 65.0, 65.0,

65.0, 100.0, 100.0, 100.0, 100.0, 76.0, 76.0, -50.0, -50.0, -10.0, -10.0,

10.0, 10.0, 30.0, 30.0, 50.0, 50.0, 50.0, 50.0, 100.0, 100.0, 110.0, 110.0,

100.0, 100.0, 40.0, 40.0, 80.0, 80.0, 140.0, 140.0, 50.0, 50.0, 50.0, 50.0,

15.0, 15.0, 17.0, 17.0, 100.0, 100.0, 100.0, 100.0, 100.0, 100.0, 88.0,

88.0, 100.0, 100.0, 50.0, 50.0, 50.0, 50.0, 44.0, 44.0, 66.0, 66.0, 50.0,

50.0, 50.0, 50.0, 100.0, 100.0, 100.0, 100.0, 50.0, 50.0, -100.0, -50.0,

-100.0, -80.0, 100.0, 100.0, 60.0, 60.0, 45.0, 45.0, 65.0, 65.0, 100.0,

100.0, 76.0, 76.0, 23.0, 23.0, 98.0, 98.0, -10.0, -10.0, 10.0, 10.0, 30.0,

30.0, 50.0, 50.0, 50.0, 50.0, 100.0, 100.0, 110.0, 110.0, 100.0, 100.0,

40.0, 40.0, 80.0, 80.0, 140.0, 140.0, 50.0, 50.0, 50.0, 50.0, 15.0, 15.0,

17.0, 17.0, 100.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 100.0, 100.0, 100.0,

100.0, 50.0, 50.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0,

0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0,

1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0,

1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0,

0.0 } ;

final\_integration\_DW.FromWs\_PWORK.TimePtr = (void \*) pTimeValues0;

final\_integration\_DW.FromWs\_PWORK.DataPtr = (void \*) pDataValues0;

final\_integration\_DW.FromWs\_IWORK.PrevIndex = 0;

}

/\* Start for FromWorkspace: '<S2>/FromWs' \*/

{

static real\_T pTimeValues0[] = { 0.0, 0.96, 0.96, 4.0, 6.16, 7.5, 7.5, 10.0,

10.0, 100.0 } ;

static real\_T pDataValues0[] = { 0.0, 100.0, 100.0, 402.14067278287462,

616.81957186544344, 750.0, 750.0, 1000.0, 1000.0, 1000.0, 1000.0, 904.0,

904.0, 600.0, 384.0, 250.0, 250.0, 0.0, 0.0, 0.0 } ;

final\_integration\_DW.FromWs\_PWORK\_n.TimePtr = (void \*) pTimeValues0;

final\_integration\_DW.FromWs\_PWORK\_n.DataPtr = (void \*) pDataValues0;

final\_integration\_DW.FromWs\_IWORK\_g.PrevIndex = 0;

}

/\* SystemInitialize for Enabled SubSystem: '<S1>/After\_unlock\_features ' \*/

/\* SystemInitialize for Enabled SubSystem: '<S7>/airbag\_decisionsystem' \*/

/\* SystemInitialize for Chart: '<S12>/Airbag\_Decision' \*/

final\_integration\_DW.Vth = 1.0;

/\* End of SystemInitialize for SubSystem: '<S7>/airbag\_decisionsystem' \*/

/\* End of SystemInitialize for SubSystem: '<S1>/After\_unlock\_features ' \*/

}

/\* Model terminate function \*/

void final\_integration\_terminate(void)

{

/\* (no terminate code required) \*/

}

/\*

\* File trailer for generated code.

\*

\* [EOF]

\*/

final\_integration.h

/\*

\*

\* File: final\_integration.h

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#ifndef RTW\_HEADER\_final\_integration\_h\_

#define RTW\_HEADER\_final\_integration\_h\_

#include <math.h>

#ifndef final\_integration\_COMMON\_INCLUDES\_

# define final\_integration\_COMMON\_INCLUDES\_

#include "rtwtypes.h"

#include "rtw\_continuous.h"

#include "rtw\_solver.h"

#endif /\* final\_integration\_COMMON\_INCLUDES\_ \*/

#include "final\_integration\_types.h"

/\* Macros for accessing real-time model data structure \*/

#ifndef rtmGetErrorStatus

# define rtmGetErrorStatus(rtm) ((rtm)->errorStatus)

#endif

#ifndef rtmSetErrorStatus

# define rtmSetErrorStatus(rtm, val) ((rtm)->errorStatus = (val))

#endif

#ifndef rtmGetT

# define rtmGetT(rtm) (rtmGetTPtr((rtm))[0])

#endif

#ifndef rtmGetTPtr

# define rtmGetTPtr(rtm) ((rtm)->Timing.t)

#endif

/\* Block signals (default storage) \*/

typedef struct {

real\_T FromWs[7]; /\* '<S5>/FromWs' \*/

real\_T FromWs\_h[2]; /\* '<S2>/FromWs' \*/

real\_T DoorEntryStatus\_S2; /\* '<S4>/Door\_Status\_Scenario2' \*/

real\_T seatoccupiedout; /\* '<S8>/seat\_belt\_alert\_system' \*/

real\_T buzzer; /\* '<S8>/seat\_belt\_alert\_system' \*/

real\_T airbag\_en2; /\* '<S13>/Occupant\_Detection' \*/

} B\_final\_integration\_T;

/\* Block states (default storage) for system '<Root>' \*/

typedef struct {

real\_T acceleration\_integrator\_DSTATE;/\* '<S12>/acceleration\_integrator' \*/

real\_T count; /\* '<S4>/Door\_Status\_Scenario2' \*/

real\_T proximity; /\* '<S4>/Door\_Status\_Scenario1' \*/

real\_T count\_l; /\* '<S4>/Door\_Status\_Scenario1' \*/

real\_T Vth; /\* '<S12>/Airbag\_Decision' \*/

struct {

void \*TimePtr;

void \*DataPtr;

void \*RSimInfoPtr;

} FromWs\_PWORK; /\* '<S5>/FromWs' \*/

struct {

void \*TimePtr;

void \*DataPtr;

void \*RSimInfoPtr;

} FromWs\_PWORK\_n; /\* '<S2>/FromWs' \*/

struct {

int\_T PrevIndex;

} FromWs\_IWORK; /\* '<S5>/FromWs' \*/

struct {

int\_T PrevIndex;

} FromWs\_IWORK\_g; /\* '<S2>/FromWs' \*/

uint8\_T is\_active\_c5\_final\_integration;/\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_c5\_final\_integration; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_Available; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_SensorWork; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_active\_c4\_final\_integration;/\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_c4\_final\_integration; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_Available\_o; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_SensorWork\_a; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_active\_c6\_final\_integration;/\* '<S9>/tyremonitoring' \*/

uint8\_T is\_c6\_final\_integration; /\* '<S9>/tyremonitoring' \*/

uint8\_T is\_Ignition\_ON; /\* '<S9>/tyremonitoring' \*/

uint8\_T is\_active\_c3\_final\_integration;/\* '<S8>/seat\_belt\_alert\_system' \*/

uint8\_T is\_c3\_final\_integration; /\* '<S8>/seat\_belt\_alert\_system' \*/

uint8\_T is\_active\_c2\_final\_integration;/\* '<S13>/Occupant\_Detection' \*/

uint8\_T is\_c2\_final\_integration; /\* '<S13>/Occupant\_Detection' \*/

uint8\_T is\_active\_c1\_final\_integration;/\* '<S12>/Airbag\_Decision' \*/

uint8\_T is\_c1\_final\_integration; /\* '<S12>/Airbag\_Decision' \*/

} DW\_final\_integration\_T;

/\* Real-time Model Data Structure \*/

struct tag\_RTM\_final\_integration\_T {

const char\_T \* volatile errorStatus;

RTWSolverInfo solverInfo;

/\*

\* Timing:

\* The following substructure contains information regarding

\* the timing information for the model.

\*/

struct {

uint32\_T clockTick0;

time\_T stepSize0;

uint32\_T clockTick1;

SimTimeStep simTimeStep;

time\_T \*t;

time\_T tArray[2];

} Timing;

};

/\* Block signals (default storage) \*/

extern B\_final\_integration\_T final\_integration\_B;

/\* Block states (default storage) \*/

extern DW\_final\_integration\_T final\_integration\_DW;

/\* Model entry point functions \*/

extern void final\_integration\_initialize(void);

extern void final\_integration\_step(void);

extern void final\_integration\_terminate(void);

/\* Real-time Model object \*/

extern RT\_MODEL\_final\_integration\_T \*const final\_integration\_M;

/\*-

\* These blocks were eliminated from the model due to optimizations:

\*

\* Block '<Root>/Scope' : Unused code path elimination

\* Block '<Root>/Scope1' : Unused code path elimination

\* Block '<S13>/Cast To Boolean1' : Eliminate redundant data type conversion

\*/

/\*-

\* The generated code includes comments that allow you to trace directly

\* back to the appropriate location in the model. The basic format

\* is <system>/block\_name, where system is the system number (uniquely

\* assigned by Simulink) and block\_name is the name of the block.

\*

\* Use the MATLAB hilite\_system command to trace the generated code back

\* to the model. For example,

\*

\* hilite\_system('<S3>') - opens system 3

\* hilite\_system('<S3>/Kp') - opens and selects block Kp which resides in S3

\*

\* Here is the system hierarchy for this model

\*

\* '<Root>' : 'final\_integration'

\* '<S1>' : 'final\_integration/PriorityBasedIntegratedmodel'

\* '<S2>' : 'final\_integration/Signal Builder'

\* '<S3>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features '

\* '<S4>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system'

\* '<S5>' : 'final\_integration/PriorityBasedIntegratedmodel/Signal Builder'

\* '<S6>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /System Requirements'

\* '<S7>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main'

\* '<S8>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system'

\* '<S9>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system'

\* '<S10>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable'

\* '<S11>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/System Requirements'

\* '<S12>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem'

\* '<S13>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system'

\* '<S14>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/Compare\_to\_airbag'

\* '<S15>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/Compare\_to\_distance'

\* '<S16>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/System Requirements'

\* '<S17>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem/Airbag\_Decision'

\* '<S18>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem/System Requirements'

\* '<S19>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system/Occupant\_Detection'

\* '<S20>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system/System Requirements'

\* '<S21>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system/System Requirements'

\* '<S22>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system/seat\_belt\_alert\_system'

\* '<S23>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system/System Requirements'

\* '<S24>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system/tyremonitoring'

\* '<S25>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/Door\_Status\_Scenario1'

\* '<S26>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/Door\_Status\_Scenario2'

\* '<S27>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/System Requirements'

\*/

#endif /\* RTW\_HEADER\_final\_integration\_h\_ \*/

/\*

\* File trailer for generated code.

\*

\* [EOF]

\*/ /\*

\*

\* File: final\_integration.h

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#ifndef RTW\_HEADER\_final\_integration\_h\_

#define RTW\_HEADER\_final\_integration\_h\_

#include <math.h>

#ifndef final\_integration\_COMMON\_INCLUDES\_

# define final\_integration\_COMMON\_INCLUDES\_

#include "rtwtypes.h"

#include "rtw\_continuous.h"

#include "rtw\_solver.h"

#endif /\* final\_integration\_COMMON\_INCLUDES\_ \*/

#include "final\_integration\_types.h"

/\* Macros for accessing real-time model data structure \*/

#ifndef rtmGetErrorStatus

# define rtmGetErrorStatus(rtm) ((rtm)->errorStatus)

#endif

#ifndef rtmSetErrorStatus

# define rtmSetErrorStatus(rtm, val) ((rtm)->errorStatus = (val))

#endif

#ifndef rtmGetT

# define rtmGetT(rtm) (rtmGetTPtr((rtm))[0])

#endif

#ifndef rtmGetTPtr

# define rtmGetTPtr(rtm) ((rtm)->Timing.t)

#endif

/\* Block signals (default storage) \*/

typedef struct {

real\_T FromWs[7]; /\* '<S5>/FromWs' \*/

real\_T FromWs\_h[2]; /\* '<S2>/FromWs' \*/

real\_T DoorEntryStatus\_S2; /\* '<S4>/Door\_Status\_Scenario2' \*/

real\_T seatoccupiedout; /\* '<S8>/seat\_belt\_alert\_system' \*/

real\_T buzzer; /\* '<S8>/seat\_belt\_alert\_system' \*/

real\_T airbag\_en2; /\* '<S13>/Occupant\_Detection' \*/

} B\_final\_integration\_T;

/\* Block states (default storage) for system '<Root>' \*/

typedef struct {

real\_T acceleration\_integrator\_DSTATE;/\* '<S12>/acceleration\_integrator' \*/

real\_T count; /\* '<S4>/Door\_Status\_Scenario2' \*/

real\_T proximity; /\* '<S4>/Door\_Status\_Scenario1' \*/

real\_T count\_l; /\* '<S4>/Door\_Status\_Scenario1' \*/

real\_T Vth; /\* '<S12>/Airbag\_Decision' \*/

struct {

void \*TimePtr;

void \*DataPtr;

void \*RSimInfoPtr;

} FromWs\_PWORK; /\* '<S5>/FromWs' \*/

struct {

void \*TimePtr;

void \*DataPtr;

void \*RSimInfoPtr;

} FromWs\_PWORK\_n; /\* '<S2>/FromWs' \*/

struct {

int\_T PrevIndex;

} FromWs\_IWORK; /\* '<S5>/FromWs' \*/

struct {

int\_T PrevIndex;

} FromWs\_IWORK\_g; /\* '<S2>/FromWs' \*/

uint8\_T is\_active\_c5\_final\_integration;/\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_c5\_final\_integration; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_Available; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_SensorWork; /\* '<S4>/Door\_Status\_Scenario2' \*/

uint8\_T is\_active\_c4\_final\_integration;/\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_c4\_final\_integration; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_Available\_o; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_SensorWork\_a; /\* '<S4>/Door\_Status\_Scenario1' \*/

uint8\_T is\_active\_c6\_final\_integration;/\* '<S9>/tyremonitoring' \*/

uint8\_T is\_c6\_final\_integration; /\* '<S9>/tyremonitoring' \*/

uint8\_T is\_Ignition\_ON; /\* '<S9>/tyremonitoring' \*/

uint8\_T is\_active\_c3\_final\_integration;/\* '<S8>/seat\_belt\_alert\_system' \*/

uint8\_T is\_c3\_final\_integration; /\* '<S8>/seat\_belt\_alert\_system' \*/

uint8\_T is\_active\_c2\_final\_integration;/\* '<S13>/Occupant\_Detection' \*/

uint8\_T is\_c2\_final\_integration; /\* '<S13>/Occupant\_Detection' \*/

uint8\_T is\_active\_c1\_final\_integration;/\* '<S12>/Airbag\_Decision' \*/

uint8\_T is\_c1\_final\_integration; /\* '<S12>/Airbag\_Decision' \*/

} DW\_final\_integration\_T;

/\* Real-time Model Data Structure \*/

struct tag\_RTM\_final\_integration\_T {

const char\_T \* volatile errorStatus;

RTWSolverInfo solverInfo;

/\*

\* Timing:

\* The following substructure contains information regarding

\* the timing information for the model.

\*/

struct {

uint32\_T clockTick0;

time\_T stepSize0;

uint32\_T clockTick1;

SimTimeStep simTimeStep;

time\_T \*t;

time\_T tArray[2];

} Timing;

};

/\* Block signals (default storage) \*/

extern B\_final\_integration\_T final\_integration\_B;

/\* Block states (default storage) \*/

extern DW\_final\_integration\_T final\_integration\_DW;

/\* Model entry point functions \*/

extern void final\_integration\_initialize(void);

extern void final\_integration\_step(void);

extern void final\_integration\_terminate(void);

/\* Real-time Model object \*/

extern RT\_MODEL\_final\_integration\_T \*const final\_integration\_M;

/\*-

\* These blocks were eliminated from the model due to optimizations:

\*

\* Block '<Root>/Scope' : Unused code path elimination

\* Block '<Root>/Scope1' : Unused code path elimination

\* Block '<S13>/Cast To Boolean1' : Eliminate redundant data type conversion

\*/

/\*-

\* The generated code includes comments that allow you to trace directly

\* back to the appropriate location in the model. The basic format

\* is <system>/block\_name, where system is the system number (uniquely

\* assigned by Simulink) and block\_name is the name of the block.

\*

\* Use the MATLAB hilite\_system command to trace the generated code back

\* to the model. For example,

\*

\* hilite\_system('<S3>') - opens system 3

\* hilite\_system('<S3>/Kp') - opens and selects block Kp which resides in S3

\*

\* Here is the system hierarchy for this model

\*

\* '<Root>' : 'final\_integration'

\* '<S1>' : 'final\_integration/PriorityBasedIntegratedmodel'

\* '<S2>' : 'final\_integration/Signal Builder'

\* '<S3>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features '

\* '<S4>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system'

\* '<S5>' : 'final\_integration/PriorityBasedIntegratedmodel/Signal Builder'

\* '<S6>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /System Requirements'

\* '<S7>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main'

\* '<S8>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system'

\* '<S9>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system'

\* '<S10>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable'

\* '<S11>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/System Requirements'

\* '<S12>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem'

\* '<S13>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system'

\* '<S14>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/Compare\_to\_airbag'

\* '<S15>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/Compare\_to\_distance'

\* '<S16>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/Safe\_distance\_enable/System Requirements'

\* '<S17>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem/Airbag\_Decision'

\* '<S18>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/airbag\_decisionsystem/System Requirements'

\* '<S19>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system/Occupant\_Detection'

\* '<S20>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /airbag\_deployment\_system\_main/occuption\_detection\_system/System Requirements'

\* '<S21>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system/System Requirements'

\* '<S22>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /seatbelt\_warning\_system/seat\_belt\_alert\_system'

\* '<S23>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system/System Requirements'

\* '<S24>' : 'final\_integration/PriorityBasedIntegratedmodel/After\_unlock\_features /suspension\_system/tyremonitoring'

\* '<S25>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/Door\_Status\_Scenario1'

\* '<S26>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/Door\_Status\_Scenario2'

\* '<S27>' : 'final\_integration/PriorityBasedIntegratedmodel/Passive Keyless Entry system/System Requirements'

\*/

#endif /\* RTW\_HEADER\_final\_integration\_h\_ \*/

final\_integration\_private.h

/\*

\*

\* File: final\_integration\_private.h

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#ifndef RTW\_HEADER\_final\_integration\_private\_h\_

#define RTW\_HEADER\_final\_integration\_private\_h\_

#include "rtwtypes.h"

/\* Private macros used by the generated code to access rtModel \*/

#ifndef rtmIsMajorTimeStep

# define rtmIsMajorTimeStep(rtm) (((rtm)->Timing.simTimeStep) == MAJOR\_TIME\_STEP)

#endif

#ifndef rtmIsMinorTimeStep

# define rtmIsMinorTimeStep(rtm) (((rtm)->Timing.simTimeStep) == MINOR\_TIME\_STEP)

#endif

#ifndef rtmSetTPtr

# define rtmSetTPtr(rtm, val) ((rtm)->Timing.t = (val))

#endif

/\* Used by FromWorkspace Block: '<S5>/FromWs' \*/

#ifndef rtInterpolate

# define rtInterpolate(v1,v2,f1,f2) (((v1)==(v2))?((double)(v1)): (((f1)\*((double)(v1)))+((f2)\*((double)(v2)))))

#endif

#ifndef rtRound

# define rtRound(v) ( ((v) >= 0) ? floor((v) + 0.5) : ceil((v) - 0.5) )

#endif

#endif /\* RTW\_HEADER\_final\_integration\_private\_h\_ \*/

final\_integration\_types.h

/\*

\* File: final\_integration\_types.h

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#ifndef RTW\_HEADER\_final\_integration\_types\_h\_

#define RTW\_HEADER\_final\_integration\_types\_h\_

/\* Forward declaration for rtModel \*/

typedef struct tag\_RTM\_final\_integration\_T RT\_MODEL\_final\_integration\_T;

#endif /\* RTW\_HEADER\_final\_integration\_types\_h\_ \*/

rtwtypes.h

/\*

\* File: rtwtypes.h

\*

\* Code generated for Simulink model 'final\_integration'.

\*

\* Model version : 1.67

\* Simulink Coder version : 9.2 (R2019b) 18-Jul-2019

\* C/C++ source code generated on : Mon Apr 6 04:54:41 2020

\*

\* Target selection: ert.tlc

\* Embedded hardware selection: Intel->x86-64 (Windows64)

\* Code generation objectives: Unspecified

\* Validation result: Not run

\*/

#ifndef RTWTYPES\_H

#define RTWTYPES\_H

/\* Logical type definitions \*/

#if (!defined(\_\_cplusplus))

# ifndef false

# define false (0U)

# endif

# ifndef true

# define true (1U)

# endif

#endif

/\*=======================================================================\*

\* Target hardware information

\* Device type: Intel->x86-64 (Windows64)

\* Number of bits: char: 8 short: 16 int: 32

\* long: 32

\* native word size: 64

\* Byte ordering: LittleEndian

\* Signed integer division rounds to: Zero

\* Shift right on a signed integer as arithmetic shift: on

\*=======================================================================\*/

/\*=======================================================================\*

\* Fixed width word size data types: \*

\* int8\_T, int16\_T, int32\_T - signed 8, 16, or 32 bit integers \*

\* uint8\_T, uint16\_T, uint32\_T - unsigned 8, 16, or 32 bit integers \*

\* real32\_T, real64\_T - 32 and 64 bit floating point numbers \*

\*=======================================================================\*/

typedef signed char int8\_T;

typedef unsigned char uint8\_T;

typedef short int16\_T;

typedef unsigned short uint16\_T;

typedef int int32\_T;

typedef unsigned int uint32\_T;

typedef float real32\_T;

typedef double real64\_T;

/\*===========================================================================\*

\* Generic type definitions: boolean\_T, char\_T, byte\_T, int\_T, uint\_T, \*

\* real\_T, time\_T, ulong\_T. \*

\*===========================================================================\*/

typedef double real\_T;

typedef double time\_T;

typedef unsigned char boolean\_T;

typedef int int\_T;

typedef unsigned int uint\_T;

typedef unsigned long ulong\_T;

typedef char char\_T;

typedef unsigned char uchar\_T;

typedef char\_T byte\_T;

/\*===========================================================================\*

\* Complex number type definitions \*

\*===========================================================================\*/

#define CREAL\_T

typedef struct {

real32\_T re;

real32\_T im;

} creal32\_T;

typedef struct {

real64\_T re;

real64\_T im;

} creal64\_T;

typedef struct {

real\_T re;

real\_T im;

} creal\_T;

#define CINT8\_T

typedef struct {

int8\_T re;

int8\_T im;

} cint8\_T;

#define CUINT8\_T

typedef struct {

uint8\_T re;

uint8\_T im;

} cuint8\_T;

#define CINT16\_T

typedef struct {

int16\_T re;

int16\_T im;

} cint16\_T;

#define CUINT16\_T

typedef struct {

uint16\_T re;

uint16\_T im;

} cuint16\_T;

#define CINT32\_T

typedef struct {

int32\_T re;

int32\_T im;

} cint32\_T;

#define CUINT32\_T

typedef struct {

uint32\_T re;

uint32\_T im;

} cuint32\_T;

/\*=======================================================================\*

\* Min and Max: \*

\* int8\_T, int16\_T, int32\_T - signed 8, 16, or 32 bit integers \*

\* uint8\_T, uint16\_T, uint32\_T - unsigned 8, 16, or 32 bit integers \*

\*=======================================================================\*/

#define MAX\_int8\_T ((int8\_T)(127))

#define MIN\_int8\_T ((int8\_T)(-128))

#define MAX\_uint8\_T ((uint8\_T)(255U))

#define MAX\_int16\_T ((int16\_T)(32767))

#define MIN\_int16\_T ((int16\_T)(-32768))

#define MAX\_uint16\_T ((uint16\_T)(65535U))

#define MAX\_int32\_T ((int32\_T)(2147483647))

#define MIN\_int32\_T ((int32\_T)(-2147483647-1))

#define MAX\_uint32\_T ((uint32\_T)(0xFFFFFFFFU))

/\* Block D-Work pointer type \*/

typedef void \* pointer\_T;

#endif /\* RTWTYPES\_H \*/