**Body Control Module I4 BMW**

**Software Requirements**

1. Software shall be organized in the AUTOSAR structure per modules as such: application layer, run time environment and basic software.
2. Software shall implement the following applications to satisfy the project requirements: Bluetooth Communication (Btc), Central Lock (CenLoc), Diagnostic Control (DiagCtrl), Exterior Lights (ExtLights), Interior Lights (IntLights), Parking Distance Controller (Pdc), Security Alarm (SecAlm), Sensor Control (SenCtrl).
3. Software shall implement the run time environment as an interface between application layer and basic software to respect the standard requirement of interactions between software components.
4. Software shall implement the following basic software modules: Universal Asynchronous Receive Transmit Protocol Handler (UartH), Diagnostic Event Manager (Dem), Analog to Digital Converter Handler (AdcH), Port Handler (PortH), Timer Handler (TimH), Free Real Time Operating System (FreeRTOS), Project Includes (Std\_Types.h), Cyclic Redundancy Check (Crc), Electronic Control Unit Manager (EcuM), System Manager (SystemManager), WatchdogManager (WatchdogManager).
5. Software shall implement own application main definition.
6. Software shall perform initialization of the flash memory, low level drivers, system clock, hardware abstraction layer, operating system, operating system start and peripheral drivers initialization in this order before software can be in run state.
7. Basic software modules shall implement error status mechanisms, if possible, to determine faults in the system.
8. Software shall use the hardware timer as a timebase for the operating system.
9. Software shall configure a watchdog for timing, system integrity requirements.
10. Software shall organize the operating system in task based on the safety levels of the applications.
11. Software shall send deactivate each task for 5 miliseconds.
12. Software shall implement semaphore as means of task synchronization.
13. Software shall allocate enough RAM for the operating system.
14. Software shall implement reset reasons.
15. Software shall implement reset capabilities in case of faults.
16. Software shall be able to detect faults in the software.
17. Software shall be able to count how many times each fault occured.
18. Software shall define the following faults:
    1. Power on reset
    2. Brown out reset
    3. Software reset
    4. Watchdog reset
    5. Low power reset
    6. Hardware reset
    7. Non-maskable interrupt reset
    8. Memory fault reset
    9. Usage fault reset
    10. Bus fault reset
    11. Malloc failed reset
    12. Adc error internal fault
    13. Adc error dma fault
    14. Adc error overrun fault
    15. Uart parity error fault
    16. Uart noise error fault
    17. Uart frame error fault
    18. Uart overrun fault
    19. Uart DMA fault
    20. Timer two fault
    21. Timer three fault
    22. Timer four fault
    23. Timer five fault
19. Software shall be able to set hardware reset DTC when one of the following occurs:
    1. Brown out reset
    2. Low power reset
    3. Hardware reset
    4. Non-maskable interrupt reset
    5. Memory fault reset
    6. Usage fault reset
    7. Bus fault reset
20. Software shall be able to set software reset DTC when one of the following occurs:
    1. Software reset
    2. Watchdog reset
    3. Stack overflow reset
    4. Malloc failed reset
21. Software shall be able to set peripheral errro DTC when one of the following occurs:
    1. Adc error internal
    2. Adc error overrun
    3. Adc error DMA
    4. Uart parity error
    5. Uart noise error
    6. Uart frame error
    7. Uart overrun error
    8. Uart DMA error
    9. Timer two error
    10. Timer three error
    11. Timer four error
    12. Timer five error
22. Software shall be able to initialize the nested vectored interrupt controller with interrupt priorities defined in the software.
23. Software shall be able to set diagnostic trouble codes for the faults in the software.
24. Software shall define clear priorities for each task based on the safety level of the task.
25. Software shall implement an operating system counter to detect aliveness of the software.
26. Software shall implement idle hook in the operating system.
27. Software shall implement tickless idle concept in the operating system for the improved efficiency of the software.
28. Peripheral drivers shall be initialized in an exact order to prevent wrong initialization.
29. Software shall configure the bluetooth communication application software component to process commands received from the UART bus.
30. Software shall configure UART at 9600 Baud rate to fit the requirement of the HC-05 bluetooth module.
31. Software shall configure the system clock at maximum 100 MHz speed.
32. Software shall configure the Btc to check the CRC of the received data with the initial values calculated at start-up in the Crc module.
33. Software shall be able to discard bluetooth commands if they the data validity is not correct.
34. Software shall define the following bluetooth messages for each type of command:
    1. Central lock on – 1
    2. Central lock off – 2
    3. High beam on – 3
    4. High beam off – 4
    5. Turn signal left on – 7
    6. Turn signal left off – 8
    7. Turn signal right on – 9
    8. Turn signal right off – 10
    9. Hazard lights on – 11
    10. Hazard lights off – 12
    11. Front fog lights on – 13
    12. Front fog lights off – 14
    13. Rear fog lights on – 15
    14. Rear fog lights off – 16
    15. Brake lights on – 17
    16. Brake lights off – 18
    17. Interior light on – 19
    18. Interior light off – 20
    19. Reverse lights on – 21
    20. Reverse lights off – 22
    21. Rotary light switch position zero – 23
    22. Rotary light switch position automatic lights – 24
    23. Rotary light switch position position lights – 25
    24. Rotary light switch position night time lights – 26
35. Software shall store the individual value of each command in a different variable and send it to the respective software module where it is needed.
36. Software shall process the data received on the UART bus on the receiving complete callback and process the data from ASCII to uint8 value and send it through RTE to the Btc.
37. Software shall implement command with value 99 for reset from the HC-05 module.
38. Software shall implement command with value 90 for diagnostic request.
39. Software shall send message „Present DTC in the system:” when the request is made.
40. Software shall send message „Light sensor fault” when the light sensor is faulty.
41. Software shall send message „Vibration sensor fault” when the vibration sensor is faulty.
42. Software shall send message „Rear parking sensor fault” when the rear parking sensor is faulty.
43. Software shall send message „Front parking sensor fault” when the front parking sensor is faulty.
44. Software shall send message „Hardware fault” when there is a peripheral fault, a hardware reset or a software reset in the software.
45. Software shall send message „HC-05 fault” when the UartH detects an error in the UART peripheral.
46. Software shall call for the refreshing of the watchdog during each transmission since the transmission takes time.
47. Software shall not allow for any type of command except for central lock on when the central lock is off.
48. Software shall implement the central lock mechanism, by using operating system timers.
49. Software shall, when the central locking is on, turn on the turn signals on for 250 miliseconds and off for 250 miliseconds using said operating system timer. This sequence applies also for the central lock buzzer. This sequence is executed twice.
50. Software shall, when the central lock is set to off, turn on the turn signals on for 250 miliseconds and off for 250 miliseconds using said operating system timer. This sequence applies also for the central lock buzzer. This sequence is executed once.
51. Software shall initialize the security alarm state each time the state of the central lock changes.
52. Software shall control an LED that turns on each time an operating system timer expires at 3,75 seconds, and should be on for 250 miliseconds before an operating system timer expires and turns it off.
53. Software shall control the follow me home concept by activating an operating system timer. After 20 seconds, the operating system timer expires and updates the variable in regard to the concept and turns off the exterior and interior lights controlled by the concept.
54. Software shall interrupt the follow me home concept if any light command is sent from the bluetooth.
55. Software shall be able to reset the security alarm when the state of the central lock module is changed.
56. Software shall be able to reset the counter variable that keeps track of how many inputs have been detected from the vibration sensor.
57. Software shall be able to trigger the security alarm as many times as the input is calculated as enough to trigger the alarm.
58. Software shall be able to control a buzzer and the exterior lightning when the security alarm activates.
59. Software shall turn on for 500 miliseconds and off for 500 miliseconds at full PWM signal the exterior lightning and a buzzer.
60. Software shall repeat the alarm sequence during 10 seconds, after it stops.
61. Software shall turn off the buzzer when the alarm is set to off.
62. Software shall be able to turn on and off the interior lights based on the activation or deactivation of the follow me home concept.
63. Software shall perform diagnostic tests inside the DiagCtrl application and report the faults, if present, to the Dem module through Rte.
64. Software shall define faults as per components that are feasible to be tested, such as the parking sensors, light sensor, vibration sensor, bluetooth module, peripherals or software resets or faults.
65. Software shall deifne the following codes for components malfunction:
    1. Light sensor malfunction – 0x03
    2. Vibration sensor malfunction – 0x13
    3. Rear parking sensor malfunction – 0x02
    4. Front parking sensor malfunction – 0x12
    5. Bluetooth module malfunction – 0x22
66. Software shall be able to switch from one turn signal to another, when the commands for on are send from one to another.
67. Software shall be able to turn on the fog lights at full PWM gradually over time.
68. Software shall be able to turn on and off the interior light at full PWM gradually over time.
69. Software shall be able to control the interior light indenpendetly.
70. Software shall be able to turn on the central lock buzzer at full PWM signal.
71. Software shall be able to turn on the security alarm buzzer at full PWM signal.
72. Software shall be able to turn on the brake lights, high beam and reverse lights at full PWM signal.
73. Software shall reset the duty cycle register for each PWM channel each time the state of the said component changes.
74. Software shall implement the automatic mode for exterior lights and shall take the input from the sensor every 39 miliseconds based on the operating system counter.
75. Software shall turn on and off at full PWM signal the exterior lights based on the input from the light sensor.
76. Software shall turn on the low beam at 37,5% duty cycle and the rear position lights at 75% duty cycle when the rotary light switch position is select as position lights.
77. Software shall turn off the exterior lights when the position zero of the rotary light switch is selected.
78. Software shall turn on at full duty cycle the exterior lights when the position night time lights is selected on the rotary light switch.
79. Software shall define the parking distance controller safe distances as follows:
    1. Maximum distance – 20
    2. First distance – 17
    3. Second distance – 14
    4. Third distance – 11
80. Software shall define the parking distance controller buzzer frequency as follows:
    1. Period one – 500 miliseconds
    2. Period two – 250 miliseconds
    3. Period three – 125 miliseconds
    4. Period four – 75 milisecond
81. Software shall configure the port configuration as follows: two pins, one pin for each sensor, for triggering the sensor to emmit sound, one pin per sensor as input capture mode from a hardware timer to detect sound reception and calculate the time the sound has travelled from the sensor to the object and back and calculate the distance between the object and the sensor, one buzzer per snesor connected to a pin set as trigger.
82. Software shall control the parking sensors only when the reverse light is set to on.
83. Software shall use 75% of the value calculated from the timer input capture mode for precision.
84. Software shall use operating system timers to trigger the buzzers of the sensors.
85. Software shall get the data from the AdcH through Rte in SenCtrl and send it to the ExtLights and SecAlm applications.
86. Software shall get the input data from vibration and light sensors using AdcH and transmitting to the ExtLights and SecAlm applications through Rte from SenCtrl.
87. Software shall define the following diagnostic trouble codes:
    1. Light sensor malfunction – 0x03
    2. Vibration sensor malfunction – 0x13
    3. Rear parking sensor malfunction – 0x02
    4. Front parking sensor – 0x12
    5. Bluetooth module malfunction – 0x22
    6. Software reset – 0x35
    7. Hardware reset – 0x56
    8. Peripheral error – 0x99
88. Software shall update the values in SenCtrl each 200 miliseconds based on operating system counter and each time an analog to digital conversion has ben completed.
89. Software shall use the ADC in DMA mode.
90. Software shall not implement any blocking code that might affect the timing of the software.
91. Software shall define the ports configuration in the PortH module.
92. Software shall configure the hardware timers as such to respect the previous requirements in regards with PWM channels singals usage.
93. Software shall configure the ADC as such to measure the values of the two sensors and the value of controller temperature, and configure the analog watchdog to generate reset when the high threshold of 3000 is passed.
94. Software shall make use of the controller resource as optimized as possible.
95. Software shall use any peripheral in interrupt mode.