Automation of Initial Signal Value Verification in BCM Messages Using CAPL

Project Description:

This project automates the verification of initial signal values in Body Control Module (BCM) messages using CAPL automation scripting in CANoe. The primary objective is to validate whether specific signals (sig_213 and sig_214) within a BCM message are initialized to zero when the message is received. The system waits for the BCM message, checks the signal values, and logs the results as either a pass or fail based on the expected initial value.

Precondition: The system is powered on, and **BCM_msg** transmission is active.

Test Cases:

Test case Id	Action	Expected Result
TC_213_01	1.Initialize initial value to 0. 2. and wait time to 1000ms 3. Verify if is received correctly.	The system should receive the BCM message, and **sig_213** should have a valid initial value (**0**).
TC_214_01	1.Initialize initial value to 0. 2. and wait time to 1000ms 3. Verify if **sig_214** is received correctly.	The system should receive the BCM message, and **sig_214** should have a valid initial value (**0**).

CAPL code for Radar ECU:

```
includes
{
}
variables
{
  message BCM_msg a;
  msTimer t1, t2;
}

On start
{
  setTimer(t1, 500);
  setTimer(t2, 200);
}

on timer t1                    // Timer t1 triggers every 500ms and sends BCM_msg
{
  output(a);
  setTimer(t1, 500);
}

on timer t2                    // Timer t2 triggers every 200ms and sends BCM_msg
{
  output(a);
  setTimer(t2, 200);
}
```

CAPL Automation for testing Initial signal value:

```
includes
{

}

variables
{
  int msg_waittime = 1000;
}

void MainTest()
{
  testReportFileName("Automation of Finding Initial Value");
  testModuleTitle("Test Case for Initial Value");
  testModuleDescription("Finding if the initial value of signals is 0 or not");

  testTC1();
}

testcase testTC1()
{
  long initial_val = 0;
  long ret_val, ret_val2, ret_val3;

  testCaseTitle("TC1", "Initial Value Result");

  ret_val = testWaitForMessage(BCM_msg, msg_waittime);    // Wait for the BCM message
  testWaitForTimeout(5000);

  if (ret_val == 1)
  {
    // Check if signal values match the expected initial value (0)
    ret_val2 = checkSignalMatch(sig_213, initial_val);
    ret_val3 = checkSignalMatch(sig_214, initial_val);

    if (ret_val2 == 1)
    {
      testStepPass("1.1", "BCM signal 213 received correct signal data (0)");
    }
    else
    {
      testStepFail("1.1", "BCM signal 213 received incorrect signal data");
    }

    if (ret_val3 == 1)
    {
      testStepPass("1.2", "BCM signal 214 received correct signal data (0)");
    }
    else
    {
      testStepFail("1.2", "BCM signal 214 received incorrect signal data");
    }
  }
  else
  {
    testStepFail("1.0", "BCM message was not received within the wait time");
  }
}
```