同济大学
TONGJI UNIVERSITY

硕士学位论文

（专业学位）

基于 AUTOSAR 架构和 Simulink 模型的汽车仪表系统研制

姓　　名：贺可军

学　　号：1231559

所在院系：汽车学院

职业类型：工程硕士

专业领域：车辆工程

指导教师：罗峰 教授

副指导教师：胡鹏

二〇一七年三月

同济大学
TONGJI UNIVERSITY

A dissertation submitted to

Tongji University in conformity with the requirements for

the degree of Master of Engineering

The development of instrument cluster based on AUTOSAR and Simulink models

Candidate: He Kejun

Student Number: 1231559

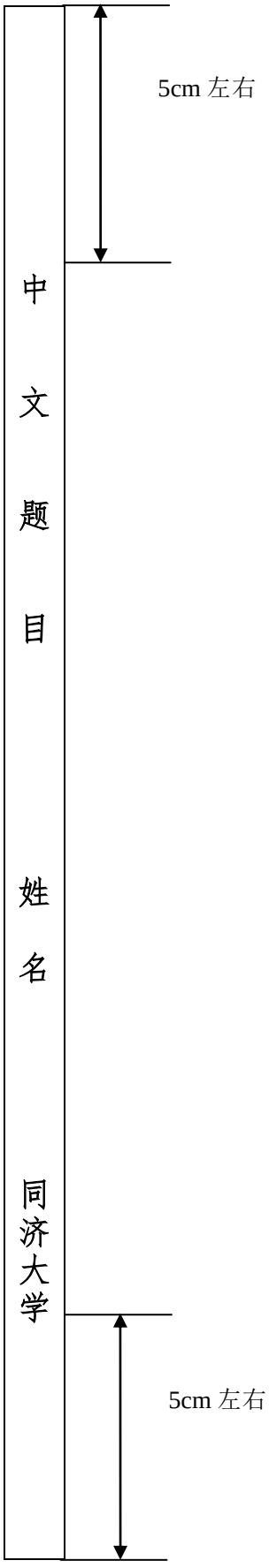
School/Department: School of Automotive Studies

Discipline: Engineering Master

Major: Automotive Engineering

Supervisor: Prof. Luo Feng

Mar, 2017


5cm 左右
中文题目
姓名
同济大学
5cm 左右

学位论文版权使用授权书

本人完全了解同济大学关于收集、保存、使用学位论文的规定，同意如下各项内容：按照学校要求提交学位论文的印刷本和电子版本；学校有权保存学位论文的印刷本和电子版，并采用影印、缩印、扫描、数字化或其它手段保存论文；学校有权提供目录检索以及提供本学位论文全文或者部分的阅览服务；学校有权按有关规定向国家有关部门或者机构送交论文的复印件和电子版；在不以赢利为目的的前提下，学校可以适当复制论文的部分或全部内容用于学术活动。

学位论文作者签名：贺可军

2017 年 3 月 5 日

同济大学学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：贺可军

2017 年 3 月 5 日

摘要

随着汽车上的电子零部件的增多，汽车组合仪表上集成的显示和报警功能也越来越多，越来越复杂。采用 AUTOSAR 的架构可以显著的提高汽车电子软件的可移植性，可以缩短产品的开发周期，受到各大主机厂和供应商的欢迎。基于模型的开发以其快速仿真验证及高可靠的代码生成，也日渐成为汽车电子零部件开发的一种趋势。

本文首先介绍了汽车仪表及汽车零部件软件架构的发展历史，对 AUTOSAR 软件架构的特点和实现机制给出了详细的分析和描述，对 AUTOSAR 架构中的操作系统，网络管理，诊断及虚拟功能总线等的原理及实现机制做了详细的分析和总结。

其次，概述了汽车组合仪表需要满足的功能要求，分析和设计了汽车组合仪表中主要的硬件电路：电源模块，数据采集模块，背光模块，通信模块，显示模块，步进电机模块等。在基于 AUTOSAR 的软件架构基础上，设计了汽车组合仪表的软件架构，对 AUTOSAR 操作系统的配置和应用层任务的设计，AUTOSAR 网络管理是配置和应用层的设计，AUTOSAR 诊断的开发等做了详细的描述，并对 AUTOSAR 中的虚拟功能总线相关的运行时环境层配置和代码生成工具做了详细设计。

汽车组合仪表的应用层采用基于模型开发的方法，搭建了应用层的模型开发和自动代码生成平台，在该平台上设计了汽车仪表的车速表模型，转速表模型，油量表模型及平均油耗模型，瞬时油耗模型，续驶里程模型等。设计了模型的自动在环测试平台，对在环测试的原理及实现方法做了详细描述，并开发了相关的自动化测试工具链，对设计的模型做了在环测试和验证。

最后以设计的某款数字组合仪表为例，导入设计的模型，生成全部仪表应用层代码，将生成的代码和运行时环境层的代码，基础软件层的代码进行集成编译链接，生成可执行的文件，烧录到仪表中，通过自动化的黑盒测试和手动功能测试，测试了仪表的基本功能和通信，通过这些测试，有效的验证了基于 AUTOSAR 和 Simulink 模型设计的汽车组合仪表的正确性和可靠性。

关键词：AUTOSAR，运行时环境，在环测试，基于模型的设计，汽车组合仪表

ABSTRACT

There are many features integrated to the instrument cluster, the software is more complexity than before. AUTOSAR aims to improve the reusability of automotive electronics software, it can short the product development cycle, and welcomed by the OEMs and suppliers. The model based design also becomes the trend of automotive electronics control unit.

Firstly, the paper introduced the history of instrument cluster and automotive electronics architecture, then focus on descript the principle of AUTOSAR OS, AUTOSAR network management, diagnostic and running time environment.

Secondly, generally descript the requirement of instrument cluster, design the instrument cluster hardware, such as the power module, the data collection module, the illumination module, the communication module, the display module, the stepper motor module. Then based on AUTOSAR architecture, detail descript the configuration of AUTOSAR OS and application tasks schedule, AUTOSAR network and diagnostic configuration and feature application development, at last detailed design RTE configuration and code generation tools.

The application layer follow the model based design process, I established the model development and code generation platform in Simulink. It include the speedometer models, tachometer models, fuel tank models and average fuel economic models, instance fuel consumption models, fuel range models. I also created the model in the loop test platform, detailed descript the principle of model in the loop test, software in the loop test and processor in the loop test. Then developed the tool testing chain to support the model in the loop test, also verified the cluster models in this testing platform.

At last, imported all the models to one type of instrument cluster and generate the application code, then compile the code with the RTE and BSW layer, and link all the source code to generate the execution files, download the file to target board, and run the function test cases through automatically black box testing and manually testing. All the modules pass the test, and it verify that the cluster which based on AUTOSAR and Simulink can meet the requirement reliability.

Key Words: AUTOSAR, RTE, loop test, model based design, Instrument Cluster

目录

第 1 章 绪论

1.1 课题研究的背景及意义

汽车工业从诞生至今，走过了 100 多年的发展历程，汽车仪表作为现代汽车关键的零部件，广泛的采用电子技术和计算机技术，成为现代汽车智能化的信息控制中心。在汽车仪表上，主要显示汽车行驶速度，行驶里程，发动机转速，机油压力，发送机冷却液温度，油箱燃油量，报警指示灯以及车辆行驶过程中的位置，路线,导航信息等，汽车仪表是一个集传感器采集，信号处理，显示和控制于一体的复杂机电设备，可以有效的辅助安全驾驶。

由于现代汽车的电子设备越来越复杂，需要在仪表上显示的内容越来越多，对仪表的显示精度要求也越来越高，传统的模拟电路仪表难以满足现代汽车工业的要求，因此，步进电机式数字显示仪表成为现代仪表的主流，而全液晶显示的仪表是未来汽车仪表发展的必然趋势。随着信息技术的高度发展，汽车仪表已从单个仪表电子化迈向集成化、系统化、网络化和智能化。尤其是在采用 CAN 总线传输整车实时信息的汽车上，大量的实车共享信息在总线上传输，可以很方便的在智能化的数字仪表上显示。

随着大量的整车信息在仪表上显示，再加上整车普遍采用 CAN 总线的通信方式，汽车仪表的软件越来越庞大，系统的复杂度也越来越高，不同的 OEM 厂商和供应商都在各自开发自己专有的电控单元，与此同时，各大主机厂和一级供应商都在思考如何才能快速的在不同的软硬件平台间切换，以达到降低产品成本和提高产品质量的目的。广大的嵌入式设计师也面对着如何才能更快的向市场交付产品，并以更加低廉的成本保证复杂系统的协调一致的挑战，他们既要保证产品代码的相对稳定，又要能适应快速的客户需求变化和修改。

为了使得产品具有广泛的可移植性，可扩展性和方便产品的更新换代，由汽车制造商，零部件供应商及其他半导体供应商和软件公司联合推出了汽车开放系统架构标准--AUTOSAR，AUTOSAR 为车辆系统提供了基于标准接口的通用软件架构。AUTOSAR 的架构自上而下的分为三层：应用层，运行时环境，基础软件层，应用层由各个独立的应用软件组件组成，运行时环境层通过提供一致的接口和服务，以完成不同的软件组件之间，软件组件与基础软件层之间的通信，基础软件层包含对操作系统的实现，非易失性存储器的管理，通信协议栈的管理，输入输出端口的抽象等，可以对运行时环境之上的应用层软件提供操作系统服务，存储

管理服务，总线通信服务等。

而汽车电子软件架构向 AUTOSAR 发展后，基于传统的手工编写代码的方式，越来越不能适应日新月异的项目变化，为此必须要考虑改变传统的项目开发方式。在传统手写代码的开发模式下，编程人员需要花费大量的时间对所编写的程序进行调试，查错和验证。这耗费了大量的工作量，并无形中延长了项目的开发周期，同时手工编写的代码依靠每个工程师个人的能力，每个工程师的编程水平参差不齐，编写出的代码的可读性很差，还要花费很多精力添加注释说明，这无形中降低了软件的可靠性，增加了代码出错的可能性。目前基于模型的开发设计，使用 Matlab 和 Simulink 等开发工具，因为其描述能力强，扩展性能好，成为汽车电子软件开发中比较流行的开发模式。基于模型的设计可以使工程师的精力聚焦于建模，算法策略和仿真，通过快速原型仿真，可以及早的发现产品中的缺陷。

本课题以 AUTOSAR 的软件架构和基于模型的设计开发方法，深入研究和探讨了仪表的软硬件架构和详细设计，并系统性的搭建了基于模型的设计的开发和仿真平台，自动化的实现了模型在环测试的要求。从而可以在不同项目上最大可能复用已开发的模型，降低开发的成本，加快产品开发周期。

1.2 研究现状

1.2.1 汽车仪表的研究现状

汽车仪表的发展史是伴随着电子技术和计算机技术的发展逐步进化的，从工作原理上来看，主要经历了五代产品，第一代汽车仪表是基于机械作用力而工作的机械式仪表；第二代汽车仪表的工作原理基于电测量技术，即通过各类传感器将被测的电量转换成电信号加以测量，称之为电气式仪表；第三代汽车仪表主要基于模拟电子技术，称为模拟电路式电子仪表；到第四代演化为步进电机式全数字汽车仪表；第五代产品是全液晶显示汽车仪表，目前汽车仪表正在经历第四代向第五代转型时期。

国际市场上汽车电子仪表应用主流有三种形式：第一种形式是通过对车速里程表和车速表，转速表进行电子化改造，目前这种形式的仪表一般出现在比较低端的乘用车或者商用车上，属于比较低端的汽车仪表类型。第二种形式是将车速表，转速表，油量表，水温表，各种报警指示信号灯等集成在一起，并且使用单片机控制步进电机的运转和液晶显示器的显示，这就是所谓的组合仪表，这种仪表是目前占市场份额最大的汽车仪表类型，该过程是全数字化的，不仅指示的精度比较高，而且仪表还具有自诊断的功能。第三种形式即是大屏全液晶显示仪表，

这也是国际汽车电子仪表的发展趋势，汽车上所有的指针式仪表和报警信息全部集成到一块大的液晶显示屏上，显示的内容更加丰富。

我国汽车仪表的发展伴随汽车工业一道，自 50 年代起步至今已有 60 多年，生产企业超过百家，经历了从简单仿制到自主开发的漫长历程，经过 80 年代与 90 年代的大规模技术改造和引进、消化、吸收国外先进技术，改变了过去产品品种单一、质量不高、工艺落后、生产与发展速度缓慢的局面，目前产品品种可基本满足国内生产品种中低档汽车车型的配套需求。产品已从原先只能简单指示汽车行驶过程中速度、燃油量、冷却液温度、机油或制动压力、发动机转速及燃油传感器、温度传感器、压力传感器的传统型仪表发展到机械式、电气式、电子式三大系列，并逐步向集成全液晶显示数字化仪表方向发展。

1.2.2 AUTOSAR 软件架构研究现状

AUTOSAR 是由全球汽车制造商、部件供应商及其他电子、半导体和软件系统公司联合建立，各成员保持开发合作伙伴关系。自 2003 年起，各伙伴公司携手合作，致力于为汽车工业开发一个开放的、标准化的软件架构。AUTOSAR 这个架构有利于车辆电子系统软件的交换与更新，并为高效管理愈来愈复杂的车辆电子、软件系统提供了一个基础。此外，AUTOSAR 在确保产品及服务质量的同时，提高了成本效率。国内的各大汽车厂商、科研院校也越来越关注 AUTOSAR 带来的标准化的设计、开发、验证，从而大幅提高汽车电子的研发效率和研发质量。

宝马集团自 2001 年即开始在称为 BMW Standard Core 的架构下，在电子控制单元 ECU 中运用标准化基础软件。该软件覆盖车辆管理系统各个层面的功能，包括执行（如车辆能量流管理系统、停车准备功能），系统管理（如系统的编码与诊断），到系统定制（如个性化定制功能，可设定特殊条件的服务定制功能）。现在，应用于全新 7 系的 BMW Standard Core 软件系统通过 AUTOSAR 架构实现对车载网络、系统内存管理以及大部分的系统诊断功能。此外，全新 BMW 7 系所采用的多个 ECU 的运行系统与 AUTOSAR 架构相匹配，允许各应用程序独立运行。

一汽,长安等整车厂技术研究院也于 2009 年开始利用 AUTOSAR 标准的工具进行 ECU 的设计、开发、验证。值得注意的是，普华基础软件股份有限公司作为 AUTOSAR 的成员，已经在中国联合上汽、一汽、长安、奇瑞等主要主机厂和部分院校成立了中国汽车电子基础软件自主研发与产业化联盟，旨在中国推广和发展 AUTOSAR 架构。普华已具备一些列基于 AUTOSAR 的工具链，如 OrientaisStudio，BSW 配置工具等。

浙江大学 ESE 实验中心从 2004 年开始关注 AUTOSAR，并率先加入了 AUTOSAR

组织。目前浙江大学ESE实验中心已经成功开发出一套符合AUTOSAR标准的集成的电控单元开发开发工具链，它可以用于ECU软件架构、网络系统配置、基础软件核配置、诊断、标定和仿真测试，支持从上到下、软件为中心的快速迭代开发模式。另外，ESE实验室中心已经开发出符合AUTOSAR标准的操作系统、通信等基础软件模块。

重庆邮电大学的程安宇，王刚，张居林等在深入研究AUTOSAR软件架构的基础上，改进了传统汽车仪表的软件架构，以控制器局域网络（CAN）通信模块为样例，介绍了基于AUTOSAR架构的汽车仪表的CAN驱动层，CAN接口层及CAN网络管理的设计和实现，该研究很好的验证了基于AUTOSAR的软件架构的汽车仪表具有良好的安全可靠性和可移植扩展性[1]。

山东重工集团公司的孙颖和王建俊等对基于 AUTOSAR 的汽车电控单元的自动代码生成技术做了深入研究和探讨，他们以汽车开放式系统架构标准AUTOSAR的方法，基于普华基础软件和Simulink工具探讨了汽车电控系统软件自动代码生成技术的开发流程。他们以汽车车灯控制系统为代表，对基础软件配置、软件组件封装、逻辑功能开发和运行时环境实现等关键环节进行了介绍。并对建模系统的在线仿真和汽车试验控制箱，也做了充分的实验和论证[2]。

同济大学的张允，钟再敏等以新型的机电符合传动装置动力总成控制器为基础，开发了一个基于AUTOSAR和多核处理器的机电复合传动控制软件。其中详细介绍了AUTAOSAR的理论基础：模板及方法论，对AUTOSAR标准中的虚拟功能总线的实现机理做了详细论述，对AUTOSAR中的基础软件服务：如通信，存储，诊断，操作系统等做了详细介绍，这些对本文的研究都有很好的借鉴意义[3]。

1.2.3 基于模型的开发研究现状

Simulink 是动态和嵌入式等系统的建模与仿真工具，也是基于模型设计的基础。Simulink自带了1000多个用户模块，可实现与有限状态机的无缝链接，扩展对复杂系统的建模能力，基本上它可以快速地创建基于嵌入式器件的应用模型，完成精确的系统模型描述，可以针对任何能够用数学来描述的系统进行建模，例如动力学系统、控制制导系统、通信系统、船舶及汽车等。Simulink 还提供了丰富的功能块以及不同的专业模块集合，利用 Simulink 几乎可以做到不书写一行代码完成整个动态系统的建模工作。State flow 状态机基于有限状态机的理论使用自然的，便于理解的形式，使复杂的逻辑关系清晰简单，一些传统方法很难实现的算法利用其建模非常容易，特别适用于对复杂的事件驱动系统进行建模和仿真。用户只通过简单直观的鼠标操作，就可以轻而易举地构造出复杂的系统，生成可靠的C代码。利用RTWEC等工具为用户算法自动生成嵌入式代码，这

是一种高效、实用的方法，目前国内外各大公司在进行新产品开发时已广泛采用。它的核心思想是让工程师把精力集中于算法的研究上，把枯燥、困难的代码编写工作留给计算机去自动完成，这样可以大大缩短产品的开发周期，同时避免了人为引入的错误，降低了风险。

Yingping Huang, Alexandros Mouzakitis 等实现了基于模型硬件在环测试和机器视觉识别技术的组合仪表自动测试平台，通过硬件在环模拟发送实车上的 CAN 信号，机器视觉识别系统内置图像识别算法，用来检测组合仪表在收到实车 CAN 信号后表头，报警，LCD 显示等的变化，该基于模型 HIL 的自动化测试平台对我们在该论文测试环节提到的黑盒自动测试平台的开发有很大的借鉴意义[4]。

同济大学汽车学院的冯江波和刘亚军教授在与 AUTOSAR 兼容的 Matlab/Simulink 自动代码生成技术方面做了很深入的研究，通过实例化的语言描述了如何使用 Matlab/Simulink 生成与 AUTOSAR 标准相兼容的方法，在研究中，还以增程式燃料电池电动汽车为例子，对基于 AUTOSAR 标准的建模过程做了实例化的分析和研究[5]。

清华大学汽车安全与节能国家重点实验室的华剑锋和何彬等人，以燃料电池整车控制器为例子，利用 Simulink 实时仿真环境，工业用数据板卡，CAN 通讯设备，系统的设计了燃料电池汽车整车控制器硬件在环实时仿真测试平台，该设计对本文在在环自动化测试测试平台的搭建有积极的帮助[6]。

1.3 论文研究的意义及内容结构

1.3.1 本文的研究意义

传统的汽车电子控制软件的开发采用的是以硬件为基础的开发方式，对软件开发者来说，需要了解从软件架构到硬件实现的具体细节，而且程序的可重用性比较差，可靠性也很难保证，很难在不同的硬件平台上移植已经经过验证的应用层算法。本课题探索了基于 AUTOSAR 架构的汽车组合仪表软件架构，该架构使得软件工程师不必过于关注底层硬件和通信机制，可以大大缩短项目的开发周期，并且促使汽车电子零部件的软硬件分离，提高了软件的复用率，有利于构建稳定的汽车软件开发平台。

国内在 AUTOSAR 的研究应用方面相对比较落后，本课题的研究有利于搭建一个基于 AUTOSAR 的软件架构的汽车组合仪表开发平台，促进国内汽车仪表电子在 AUTOSAR 应用上的进步。在 AUTOSAR 架构中，对应用层的开发模式研究的相对比较少，这一方面是因为应用层复杂多变，很难形成一个稳定的应用层开发平台；

另一方面也是由于应用层每一个主机厂或者零部件供应商都有自己的开发工具和方式，很难统一。同时，国内在基于Simulink的模型设计在汽车组合仪表上的应用研究的也比较少，本课题旨在融合两者的优点的基础上搭建一个基于Simulink的模型开发和仿真平台，着眼于架构的稳定性和应用层代码的可重用性和可移植性，推动基于模型的开发在汽车组合仪表领域的应用。

1.3.2 本文主要研究内容和结构

本课题是基于瑞萨公司的V850系列微控制器，参照AUTOSAR的软件架构，结合Vector公司提供的AUTOSAR软件开发工具Geny,DaVinci Configurator等，实现了汽车组合仪表平台的硬件设计，软件架构设计，操作系统配置，应用层模型开发及代码生成，最终实现了基于AUTOSAR架构的汽车组合仪表软硬件开发及测试平台。

本课题主要的研究内容及研究思路如下：

1. 从汽车组合仪表需要满足的基本功能入手，先后对汽车仪表在整车网络中的拓扑结构，汽车仪表的软件架构及AUTOSAR架构中的关键的技术做了详细的描述，对其原理及实现机制做了详细的分析和总结。

2. 给出了汽车组合仪表的硬件框图，设计了汽车组合仪表的相关硬件电路：电源模块，数据采集模块，背光模块，通信模块，显示模块，步进电机模块等。

3. 基于AUTOSAR的软件架构要求，设计了该组合仪表的软件架构，在DaVinci Configurator中配置实现了AUTOSAR操作系统，AUTOSAR网络通信，AUTOSAR诊断等，并自行设计了运行时环境层的配置和代码自动生成工具。

4. 设计了应用层的模型开发和自动代码生成平台，详细设计实现了汽车组合仪表中的车速表模型，转速表模型，油量表模型及平均油耗模型，瞬时油耗模型，续驶里程模型等。

5. 设计了模型的自动在环测试平台，开发了相关的自动化工具链，通过该在环设计平台，可以自动的对设计的应用层模型执行模型在环测试，软件在环测试和处理器在环测试。

6. 将应用层代码和BSW层的代码进行编译和链接，生成可执行文件，烧录到控制器中，通过自动化的黑盒测试和手动功能测试，全面测试了组合仪表集成后的功能和性能，验证了组合仪表的功能符合设计的要求。

第 2 章 汽车组合仪表系统设计

2.1 汽车组合仪表系统功能描述

汽车仪表是驾驶员与汽车进行信息交互的重要接口和界面，对车辆的安全行驶有着重要的作用，现代汽车组合仪表一般位于驾驶室的正前方，在仪表盘上集成了表征车辆当前工作状态的各种指示装置。汽车组合仪表上指示的信息可以分为以下几类：指针指示类信息，燃油经济性信息，报警指示灯信息，其他类信息等。

当前主流的汽车组合仪表一般采用四个机械步进电机表头加中间一块液晶显示屏，组合仪表的功能外观如下图 2.1 所示：

图 2.1 组合仪表功能外廓示意图

2.1.1 指针指示类信息

指针指示类信息主要包含车速表，转速表，油量表，水温表等四个机械式指针表。采用机械表头指示这四种信息，主要是从可靠性和安全性的角度出发，车速表用来指示当前车辆行驶的速度，车速表的指示单位一般是千米/小时，刻度盘的指示范围一般在 0 到 220 千米/小时之间，车速表指示的车速值需要比实际的车速值要偏大，并且按照法规的要求在不同的速度区间进行偏移，车速表在车辆行驶过程中的不同路况条件下都能平稳准确的指示当前的车速值，不能发生晃动或者抖动的情况。转速表用来指示发动机运转时的转数，一般刻度盘上的单位设置为千转/分钟，乘用车的最大转速值一般是 8000 转/分钟，有些车辆上会要

求转速表上明确标注发动机怠速运行时的位置。油量表用来指示油箱中剩余燃油量，单位为升，一般刻度盘上设置空油（E），1/4 油，1/2 油，满油（F）这几种指示位置，在车辆正常行驶时，油箱油量的变化相对比较慢，要求油量表可以稳定准确的指示当前的油箱的剩余油量，而当车辆在加油的时候，需要在点火钥匙打到运行状态时，燃油表可以快速的指示当前油箱的油量。水温表用来指示当前的发动机冷却液温度，单位为摄氏度，刻度盘指示的范围要求从 50 摄氏度到 140 摄氏度，一般发动机正常运转时，冷却液的温度维持在 90 摄氏度左右，因此要求水温表在 90 摄氏度即刻度盘的中间位置时，指示尤其要稳定，不能出现指针晃动的情况。

2.1.2 燃油经济性信息

燃油经济性信息主要是在组合仪表当中的 LCD 显示屏上显示，其中主要包括瞬时油耗，平均油耗，最佳油耗，续驶里程等。瞬时油耗指示当前一段时间内发动机的喷油量，瞬时油耗的单位是升/100 公里，计算公式为当前燃油量的累计除以当前里程的累计再乘以 100。平均油耗是以小计里程记录的这段距离内，消耗的燃油量，单位是升/100 公里，计算公式为累积的喷油量除以小计里程的距离再乘以 100。最佳油耗指示当前 50 公里，100 公里，500 公里内统计计算的最佳油耗值，通过分段累加计算 20 个平均油耗值，从里面挑选出耗油最小的一个值显示出来。续驶里程用来指示当前油箱剩余油量还能再行驶的最大距离，单位是公里，计算公式是油箱剩余油量除以平均油耗，续驶里程值可能会因为不同路况平均油耗的变化而时大时小，续驶里程值对长途行车来说具有比较大的参考意义。

2.1.3 报警指示灯信息

报警指示灯信息主要有 LED 指示灯报警和 LCD 弹出报警组成，其中 LED 指示灯报警是独立控制每一个 LED 指示灯，乘用车上大概有 20 到 40 个 LED 指示灯不等，这其中有些指示灯用来指示车上特定的模块发生故障时，比如发动机故障指示灯，会点亮这些仪表上的 LED 指示灯，还有一些指示灯用来指示车辆某一个模块的状态信息，比如，日间行车灯，左右转向灯等。仪表上一般常用的 LED 指示灯有以下这些：安全气囊模块故障报警指示灯，刹车防抱死模块故障报警指示灯，电池电压低报警指示灯，刹车模块故障指示灯，日间行车灯指示灯，后雾灯指示灯，前雾灯指示灯，燃油量低指示灯，远光灯指示灯，小灯指示灯，机油压力低指示灯，安全带指示灯，发送机故障指示灯，左右转向指示灯等。LCD 屏上弹出

的报警一般是安全性比较低的模块的报警，数量也比 LED 报警指示灯要多，可以多达几百个。当车辆在行驶的过程中，仪表检测到某些弹出报警的条件触发了，就会在当前的 LCD 屏上弹出这些报警，以提醒驾驶员及时停车检修。

2.1.4 其他类信息

除了指针指示类信息，燃油经济性信息，报警指示灯信息外，还有一些其他的指示信息也需要在仪表上显示，主要包含总里程信息，小计里程信息，档位信息等。总里程信息记录了从车辆出厂上路到报废整个生命周期内总的行驶距离，并且总里程信息需要保存到非易失性存储器中，即使电池掉电也要能确保再次恢复历史数据，总里程的计算有两种方法，一种方法是直累积轮胎旋转的计数值，通过轮胎的周长关系，计算出当前行驶的距离；还有一种方法是间接从车上其他计算总里程的零部件获取已经计算完成的总里程数据，直接在仪表上存储和显示，在该课题中采用的是第二种方法，直接从车身控制器发送的报文中获取总里程数据。小计里程主要用来给驾驶员短期记录从某一个起始点到某一个终点间的距离，小计里程是可以通过按键清零的，而总里程数据是不能通过任何方式清零的。变速箱的档位信息也需要在仪表上显示，一般要显示的档位信息包括：驻车档（P），倒车档（R），空档（N），前进档（D），手动档（M）等。

2.2 某整车系统的网络结构

通信技术的发展促使汽车进入网络化，智能化时代，汽车网络已经成为汽车电子的重要组成部分，汽车仪表作为整车上信息显示的中枢，需要和车辆上绝大多数的零部件交互，车辆上的很多零部件的状态信息都需要在仪表上显示，传统的汽车仪表需要连接大量的线束，通信效率既低下，成本又高，自带控制器局域网络（CAN）总线通信模块的汽车仪表可以很方便的实现与汽车上的大部分零部件的交互并且还有自诊断功能。面对日益复杂的车载网络通信环境，如何通过嵌人式软件高效的、安全的、并同时保证传输实时性的实现汽车网络电控单元间通信成为了汽车网络通信中重要问题。

汽车车载网络一般采用的是总线式拓扑结构，CAN 总线是目前汽车工业中广泛采用的控制器通信局域网协议，CAN 总线具有结构简单，可靠性高，通信实时性强等优点。汽车组合仪表（IPC）在某整车网络结构中的位置如下图 2.2。

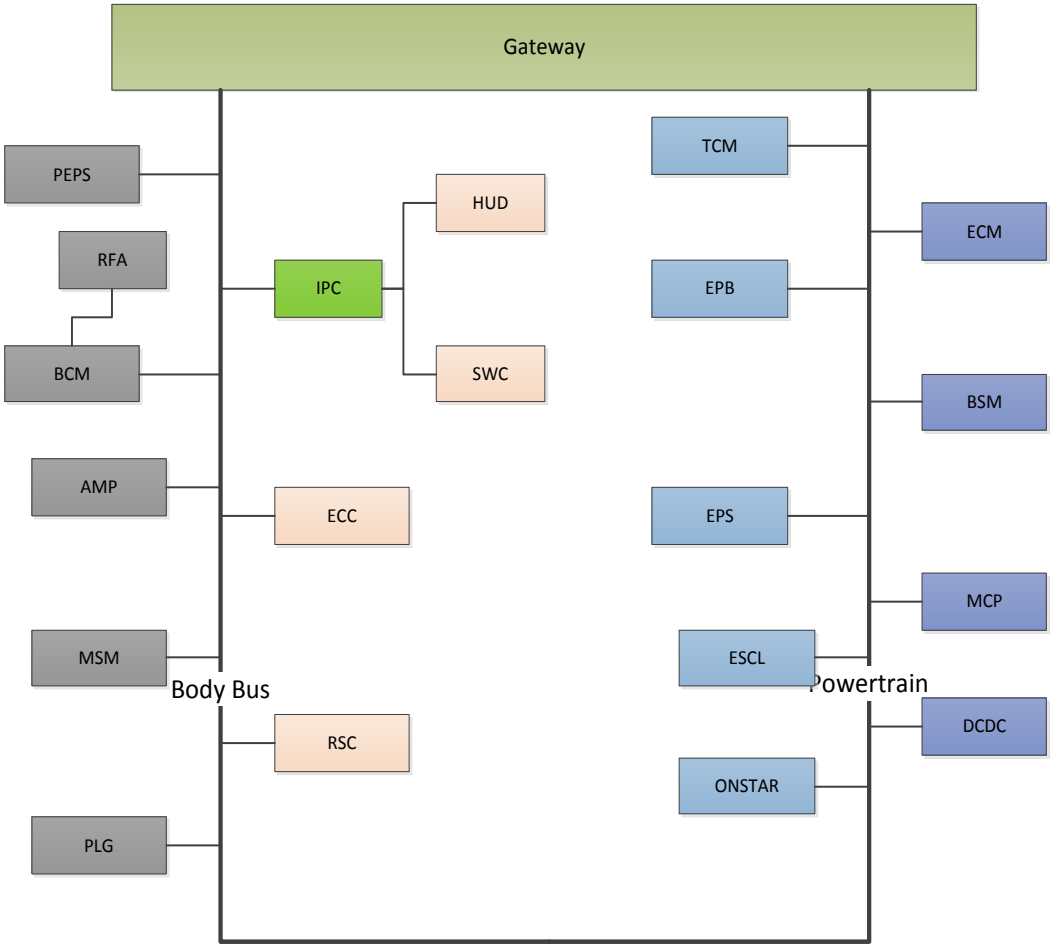


图 2.2 仪表在某整车网络系统结构图

该型车上有两条不同的 CAN 总线子网：车身总线子网和动力总线子网，两条子网通过网关连接在一起，通过网关交互不同网段上的信息。其中车身子网上挂接了无钥匙启动控制器，车身控制器，电子气候控制器和组合仪表等，车身 CAN 总线上采用的是 AUTOSAR 的部分网络管理策略，在整车休眠的时候，特定唤醒的节点只与特定的模块通信，以达到降低整车功耗的目的。动力子网上挂接了变速箱控制器，发动机控制器，电动助力转向控制器，电子驻车控制器等，这些模块对信号传输实时性要求比车身总线上模块要高，采用的是通用公司局域网的网络管理策略。

车身子网上的网络管理是采用的是 AUTOSAR 的网络管理策略，这也要求整车的软件架构也要尽可能的符合 AUTOSAR 的架构要求。AUTOSAR 架构遵循在标准上统一，在实现上竞争的原则，旨在为汽车电子软件开发领域提供开放的，标准化的软件架构以及一个标准的软件运行时环境，使得软件可以独立于硬件平台之外，可以显著的提高软件的复用率，缩短软件的开发周期，提升整个汽车电子行业的效率。接下来重点探讨一下 AUTOSAR 软件架构，AUTOSAR 操作系统和 AUTOSAR 网络管理和诊断及虚拟功能总线等核心关键技术。

2.3 AUTOSAR 软件架构

未采用 AUTOSAR 标准架构的软硬件是耦合在一起的，软硬件不能分离开，软件的更改和移植有很大的不便，AUTOSAR 采用分层分模块设计的思想，这样的设计可以最大可能的使软件和硬件分开，以达到方便切换硬件平台，最大可能提高软件的复用率。其架构自上而下可以分为三层：应用层，运行时环境，基础软件层。标准 AUTOSAR 的软件架构示意图如图 2.3 所示：

图 2.3 AUTOSAR 软件架构示意图

应用层包含了所有的应用软件组件，应用层的软件架构 AUTOSAR 没有统一的规范，因为不同的厂家不同的 ECU 产品根本无法抽象为一个统一的架构，因此这部分的架构是由用户自己设计的，应用层通过运行时环境层与基础软件层进行数据交互。

AUTOSAR 运行时环境位于应用层与基础软件层之间，是应用层软件和基础软件通信的桥梁，无论通信是在不同的 ECU 之间还是在同一个 ECU 内部，运行时环境均通过提供统一的接口和服务来实现不同的软件组件间的通信和调度。

运行时环境层之下是基础软件层，基础软件又分为服务层，微控制器抽象层，微处理器驱动层。其中服务层位于基础软件的最上层，主要对应用层软件提供以下三种服务：系统服务，存储服务，通信服务。系统服务主要提供操作系统的任务调度，电子控制单元管理等，存储服务主要是对非易失性存储器的读写服务，通信服务主要是处理来自总线上的网络报文收发等服务。微控制器抽象层与微处理器驱动层通过标准的 API 函数交互，将控制器的结构进行了抽象处理，如控制

器与外设的连接方式等，它提供的 API 可以访问外设，但是并不关心外设的位置以及连接方式。微处理器驱动层，包括微处理器相关的外设驱动，它为上层提供基于硬件或者独立于硬件的服务，如非易失性存储器驱动，通信驱动，输入输出端口驱动，模拟数字转换驱动等。

AUTOSAR 标准在制定时尽可能囊括汽车电子 ECU 单元的共性外设的驱动，但是有一些外设比如复杂的传感器和显示设备，这些设备的时序相对复杂，每一种型号的驱动都不尽相同，为了兼容这些特殊的设备驱动程序，AUTOSAR 标准中单独划出一个复杂设备驱动模块，用户可根据不同的外设按照统一的 AUTOSAR 接口设计属于本电子控制单元特殊的一些设备驱动函数。

2.4 AUTOSAR 虚拟功能总线

在 AUTOSAR 系统架构中，软件组件的交互是基于虚拟功能总线进行的，在虚拟功能总线上，软件组件间通过端口交互，端口控制了软件组件间的通讯的内容和语义，虚拟功能总线使得设计者在设计应用层的软件组件时，不必考虑他们会具体被分配到哪一个电子控制单元上，也不用考虑网络拓扑结构和 ECU 在网络中的通讯方式，这就意味着，通过虚拟功能总线，在车辆间的 ECU 架构确定前，就可以先期展开系统的功能设计，所以在 AUTOSAR 中，在虚拟功能总线的设计阶段，可以完全不用关心软件组件间的真实的通讯方式是什么，只需要在虚拟功能总线上实现通信即可，其他的工作都是通过系统配置实现的。在 AUTOSAR 中，软件组件间的接口与软件组件与基础软件层间的通讯接口通过运行时环境工具配置实现的。本文也是借鉴了 AUTOSAR 中虚拟功能总线的概念，设计实现了运行时环境层的配置工具链，这会在后续章节 4.5 中详细描述。不同的应用层软件组件通过虚拟功能总线交互的示意图如下：

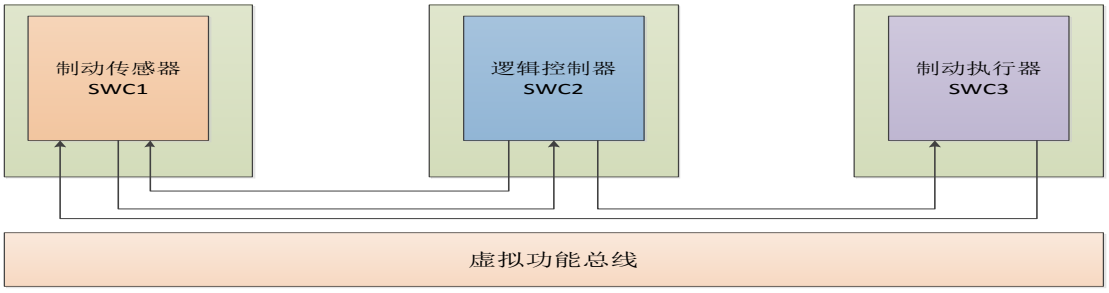


图 2.4 AUTOSAR 虚拟功能总线示意图

2.5 AUTOSAR 操作系统

AUTOSAR 操作系统是对汽车电子类开放系统和对应接口标准（OSEK）操作系

统的扩展，OSEK 是欧洲汽车工业中的一项标准，目的是为汽车电子设备提供统一的开放式的系统接口，其目标是使得软件可以在不同的项目中移植和重用，OSEK 操作系统中的基本概念有：任务，调度，中断，资源，计数器和警报等，AUTOSAR 操作系统继承了 OSEK 操作系统上述所有的内容。

本课题选用的操作系统配置工具为 Vector 公司的 Davinci Configurator，在 AUTOSAR 中对操作系统的接口及功能做了标准化要求，对其具体实现没有做强制规定，我们在此以 Vector AUTOSAR 操作系统为基础，具体描述一下 AUTOSAR 操作系统的功能和实现原理。

2.5.1 AUTOSAR 操作系统任务调度

如果系统需要在同一时间执行不同的活动，我们称之为并行的系统。在实时系统中，每一个并行的活动可以认为是一个独立的任务，几乎所有的应用层的代码都是放在任务中调度的。所有的实时操作系统都会有一个任务调度器，它可以根据固定的任务优先级来切换任务，这些优先级是在任务设计时在配置工具中分配的。优先级代表了这个任务的重要性和紧急程度，AUTOSAR 支持两种任务调度策略。

第一种，抢占式任务调度，如图 2.5 所示，抢占式的策略是在准备执行的任务中挑选优先级最高的任务来执行，如果一个任务在执行（图中的任务 1），有一个更高优先级的任务（图中的任务 2）准备执行了，那么高优先级的任务会抢占低优先级正在执行的任务，当高优先级的任务执行完毕后，之前被抢占的任务返回继续执行。

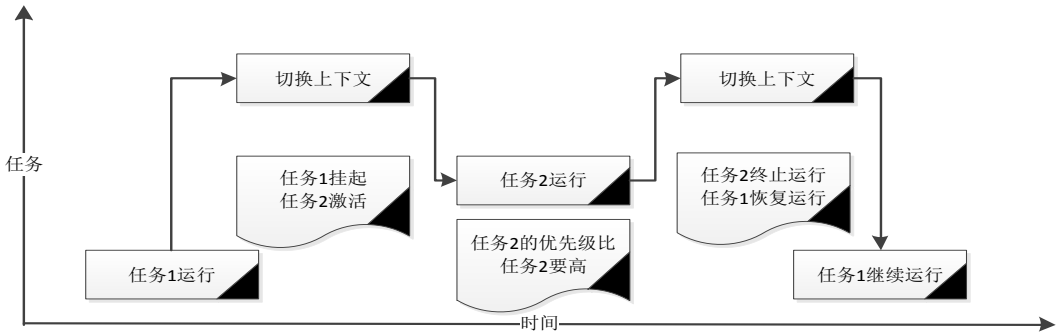


图 2.5 抢占式任务调度策略

第二种，非抢占式任务调度，如图 2.6 所示，非抢占式的策略是操作系统依然执行任务中优先级最高的任务，但是如果有一个更高优先级的任务准备执行了，那么仍然会运行当前的任务直到任务结束，一个非抢占式的任务会完整的运行直到结束。

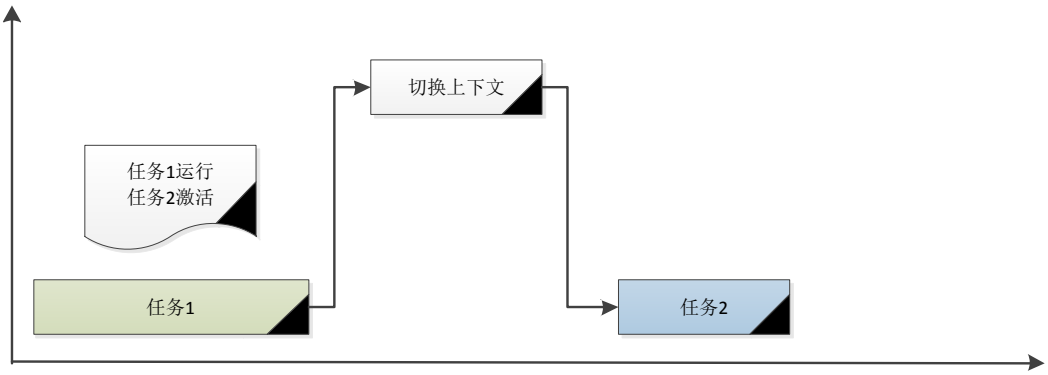


图 2.6 非抢占式任务调度策略

2.5.2 AUTOSAR 操作系统任务状态

操作系统基本的任务运行过程是：开始，执行，终止，其任务状态切换如下图所示：基本的任务有以下这些状态：挂起状态，准备运行状态，运行状态，等待状态。任务默认的状态是挂起，任务激活后进入准备状态，准备状态下，任务虽然是可以运行了，但是还没有真正的运行起来。一个任务终止后会进入挂起状态，之后可以被再次激活，并不断的重复上述过程。

任务只有在激活后，才能运行。激活指的是把任务从挂起状态转变为准备状态，或者将任务丢入任务队列中排队，没激活一次任务，只能运行一次，如果任务激活的数量超过限制就会报错，应用程序会报错。AUTOSAR 中任务的激活分成两种方式，一种是直接激活，直接调用 ActiveTask()API 激活，ActiveTask(TASK ID)可以把对应的任务变为准备状态。另外还有一种间接激活任务的方式，即通过警报激活，每次警报到期后，可以激活任务。在本课题的设计过程中，我们选择直接激活任务的方式激活每一个任务。

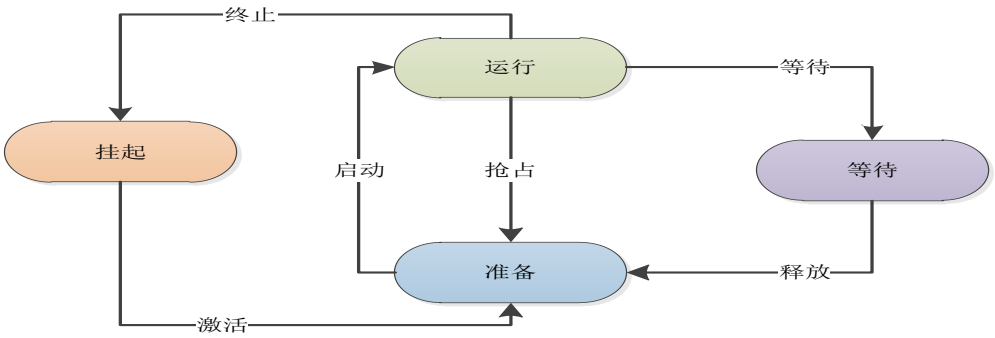


图 2.7 操作系统任务状态切换

2.6 AUTOSAR 网络通信

AUTOSAR 通信协议栈支持很多通信协议，包括控制器局域网（CAN），Flexray，局部互联协议（LIN），车载以太网等，本课题主要研究的通信协议栈是 CAN 总线，以 CAN 总线为例介绍 AUTOSAR 的通讯协议栈的设计过程，AUTOSAR 通信协议栈信号流转图如下图 2.8 所示：

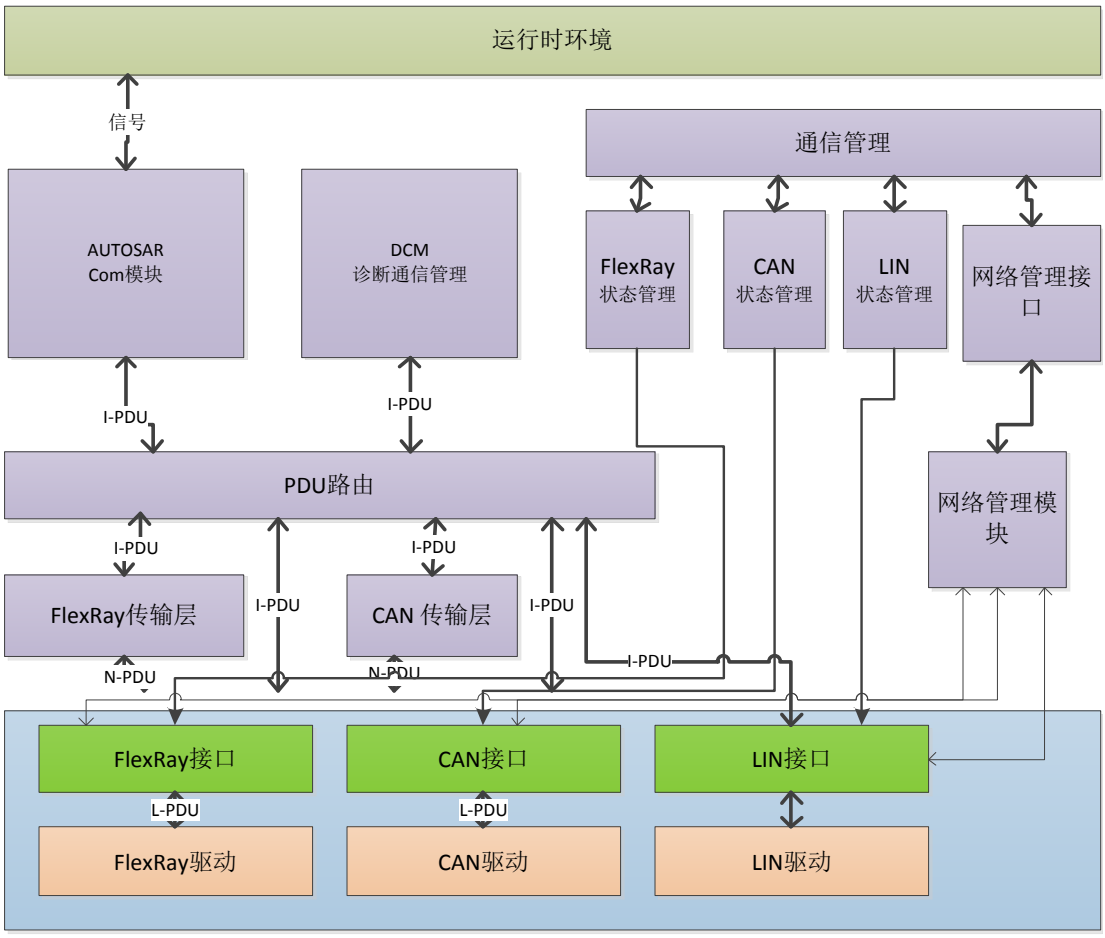


图 2.8 AUTOSAR 通信协议栈信号流转图

AUTOSAR 的网络通信协议栈的架构中按各模块提供通信服务的功能不同，将通信中间件主要划分为通信接口映射层、通信传输服务层、通信管理服务层及通信调度服务层四个层次：通信接口映射层用于保证应用程序接口、数据类型与通信中间件标准接口、数据类型的一致性；通信传输服务层包含的通信栈是信号传输的实际承载者，保证数据的期望实时性和可靠性传输；通信管理及通信 调度服务层是实时性保证的重要手段，维护信号属性、传输状态及总线状态，并且其调度点及实时相关参数可由用户灵活配置，用户可通过自由配置通信协议栈参数而使其满足通信系统的不同应用场合的需求。下面重点介绍 AUTOSAR 中的网络管理策略（带部分网络唤醒），AUTOSAR 中的网络管理策略包含以下三种模式：睡眠模式，预睡眠模式，网络模式。

AUTOSAR 网络管理的状态迁移图如下图 2.9 所示：

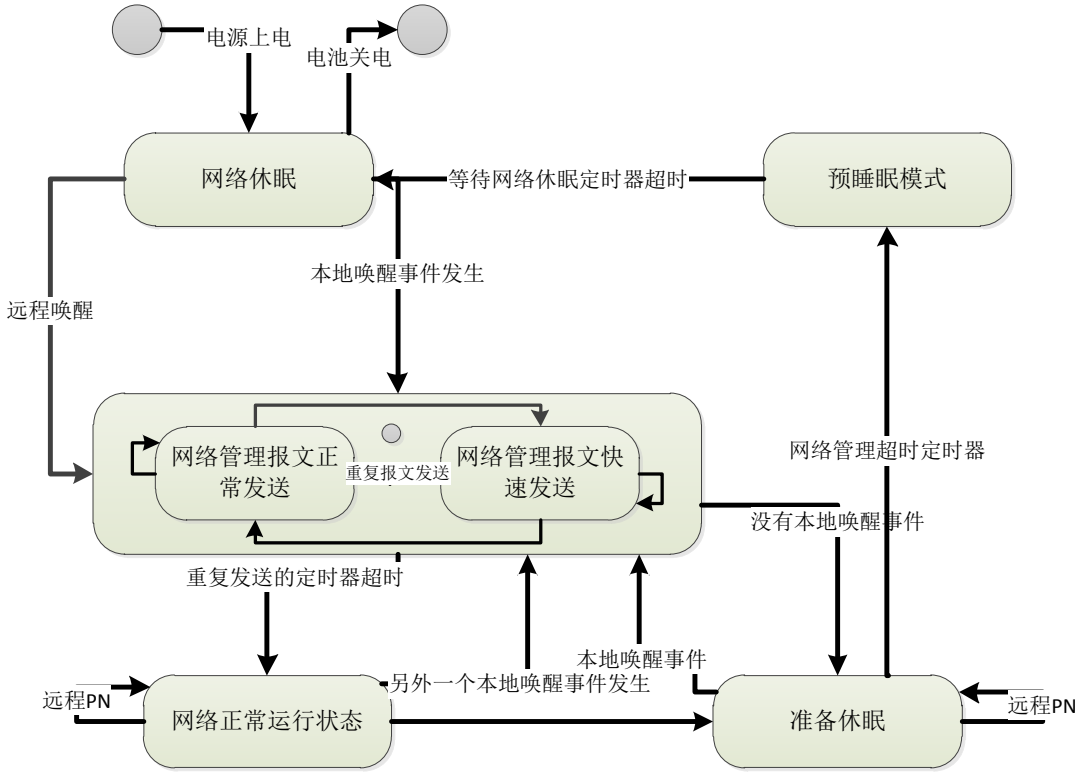


图 2.9 AUTOSAR 网络管理状态迁移图

在睡眠模式下，当节点没有本地网络唤醒及远程唤醒请求时，ECU 通信控制器切换至睡眠模式，ECU 功耗降低至适当水平。在睡眠模式下，节点的网络管理报文和应用报文禁止发送，并且不能对总线上的报文进行应答。同时节点在该模式下，如果检测到有效的唤醒源，节点必须被唤醒。

在预睡眠模式下，总线活动静止下来，最终达到总线上没有活动，ECU 通信控制器状态应处于工作模式。在该模式下，节点的网络管理报文和应用报文禁止发送，但应该对总线上的报文进行应答。节点的网络管理状态必须保持预睡眠模式一段时间，这段时间是可以配置的，一旦超时，网络管理状态应该离开预睡眠模式，进入睡眠模式。

网络模式又可以分为三种内部状态：重复报文状态，常规操作状态，准备睡眠状态。在重复报文状态下包含两个子状态：部分网络唤醒报文快发模式和部分网络唤醒报文正常发送模式。

节点在进入部分网络唤醒报文快速发送状态时，必须开启或重置网络超时定时器，为了快速唤醒网络，必须以快速周期发送网络管理报文，同时不得发送正常周期网络管理报文，与唤醒请求相关的所有应用报文必须在第一帧快速部分网络唤醒报文发送开始后延迟一段时间后才能发送。进入部分网络唤醒报文正常发送状态后，节点必须以正常周期发送网络管理报文。若节点因远程唤醒请求进入部分网络唤醒报文正常发送状态，必须开启网络管理超时定时器，与该远程请求

相关的所有应用报文必须从节点检测到相关请求后延迟一段时间才能发送。

当节点因发生本地唤醒事件需要与网络上其他节点进行通信时，必须保持在常规操作状态，在常规操作状态下，节点一旦接收或发送一条网络管理报文，或者网络管理超时定时器超时，网络管理超时定时器应该立即重置。节点在常规操作状态下若成功接收另一个相关的远程唤醒请求，则与该远程请求相关的所有应用报文必须从节点检测到相关请求后延迟一段时间后才能发送。在该状态下，节点的网络管理报文和应用报文必须正常发送。

节点进入准备睡眠状态后，必须停止发送网络管理报文，若节点在该状态下成功接收另一个相关的远程唤醒请求，则与该远程请求相关的所有应用报文必须从节点检测到相关请求后延迟一段时间后才能发送。在准备睡眠状态下，节点一旦接收到一条网络管理报文，网络管理超时定时器应该立刻重置，网络管理超时定时器超时，节点的网络管理状态应该进入预睡眠模式。

2.7 AUTOSAR 诊断协议栈

汽车诊断可以使得早期的故障得以发现，能把当前系统产生的故障监控下来，并以故障码的形式存储起来，有利于提高系统的安全性和稳定性。一般来说，ECU 中的电子诊断系统需要有下面这些功能：监控系统，发现故障并存储，降级或者关闭系统，系统从故障中恢复。诊断的应用场景有很多，在开发阶段主机厂或者供应商利于诊断进行软硬件开发，做版本测试等功能，在生产阶段，主机厂和供应商可以使用诊断对产品下线做最终的下线测试和版本管理，在主机厂售后服务阶段，又可以利用诊断系统进行车辆维修时的故障分析。

AUTOSAR 诊断服务栈支持汽车行业常用的诊断标准--统一诊断服务（UDS）协议，AUTOSAR 中诊断相关的模块关系图如下 2.10 所示：

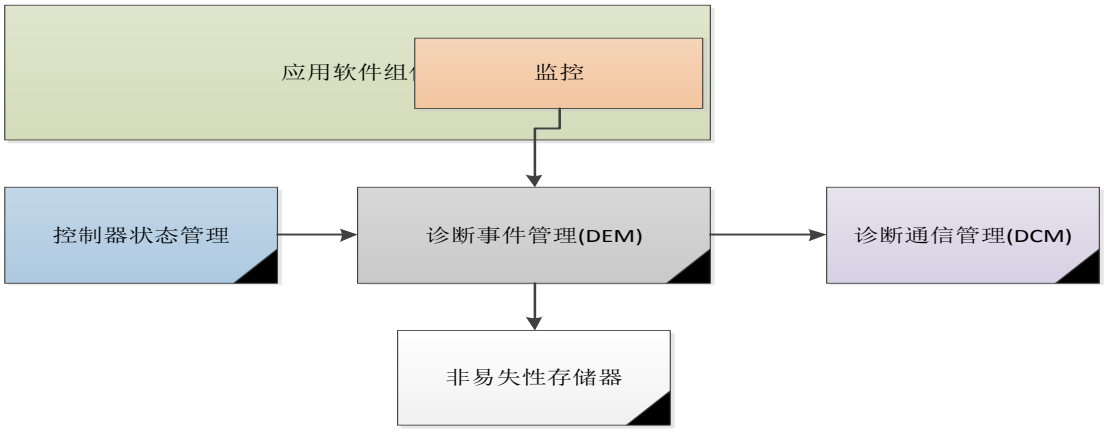


图 2.10 AUTOSAR 诊断框架图

通过该诊断框架图我们可以看出，诊断事件管理模块 DEM 和诊断通信管理模块 DCM 这两个模块是基础的核心模块，DEM 负责整个诊断事件的管理，而 DCM 负责诊断通信的管理,DCM 模块需要向诊断仪发送的诊断数据（故障码等）需要从 DEM 获取，DEM 向应用层负责监控的应用软件组件提供对诊断事件的服务接口，此外，DEM 还需要与非易失性存储器交互，以存储相关的诊断故障码。

2.8 本章小结

本章首先介绍了主流的汽车仪表的系统功能，对组合仪表上常用的指针类信息，燃油经济性信息，报警指示灯信息等提出了指标和要求。然后以某整车网络结构为例子，介绍了汽车组合仪表在整车网络架构中的位置及采用的网络管理策略等，从而引出了整车上采用 AUTOSAR 的网络管理策略，进而引伸到了对 AUTOSAR 中的几个关键技术的探讨，包括 AUTOSAR 的软件架构的分析，AUTOSAR 虚拟功能总线的描述，AUTOSAR 操作系统的任务调度机制，AUTOSAR 网络通信的信号传输流向，AUTOSAR 网络管理的策略，最后到 AUTOSAR 的网络诊断协议栈的概述等，这些关键性技术的分析和描述，为后续的开发奠定的坚实的理论基础。

第 3 章 汽车组合仪表硬件设计

3.1 汽车仪表硬件架构框图

一般来说，汽车组合仪表上需要集成车速表，转速表，油量表，水温表，报警指示灯，声音报警，LCD 显示等，所以在硬件设计上主要分成下面这些模块：微控制器模块，电源供电模块，指示灯控制模块，显示驱动模块，步进电机驱动模块，输入信号采集模块，背光控制模块，CAN 通信模块等，硬件系统框图如下 3.1 所示。

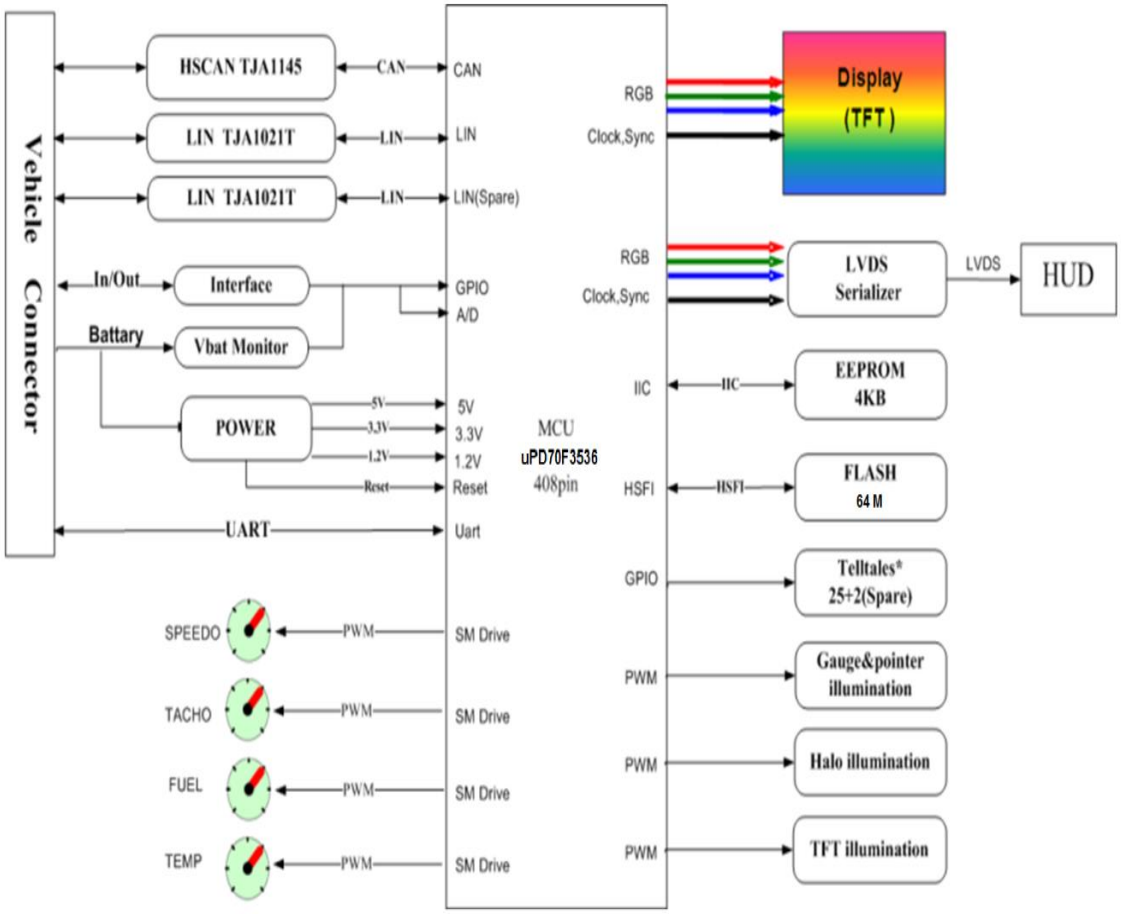


图 3.1 组合仪表平台系统框架图

3.1.1 微控制器模块

微控制器模块是整个仪表控制系统的核心，是各种信号处理和计算的中枢。该组合仪表平台的微控制器选用的是日本瑞萨公司的 V850 系列 Dx4 芯片，该芯

片拥有 3M 的内部 ROM 容量，256K 的 RAM 容量，并可以支持扩展 64M 的外部存储器，运行时钟频率最大可以达到 160MHZ，可以驱动 4.2 英寸全彩色 TFT 的显示，性能稳定可靠，在汽车仪表领域得到广泛的应用，微控制器模块的硬件特性概述如下所示：

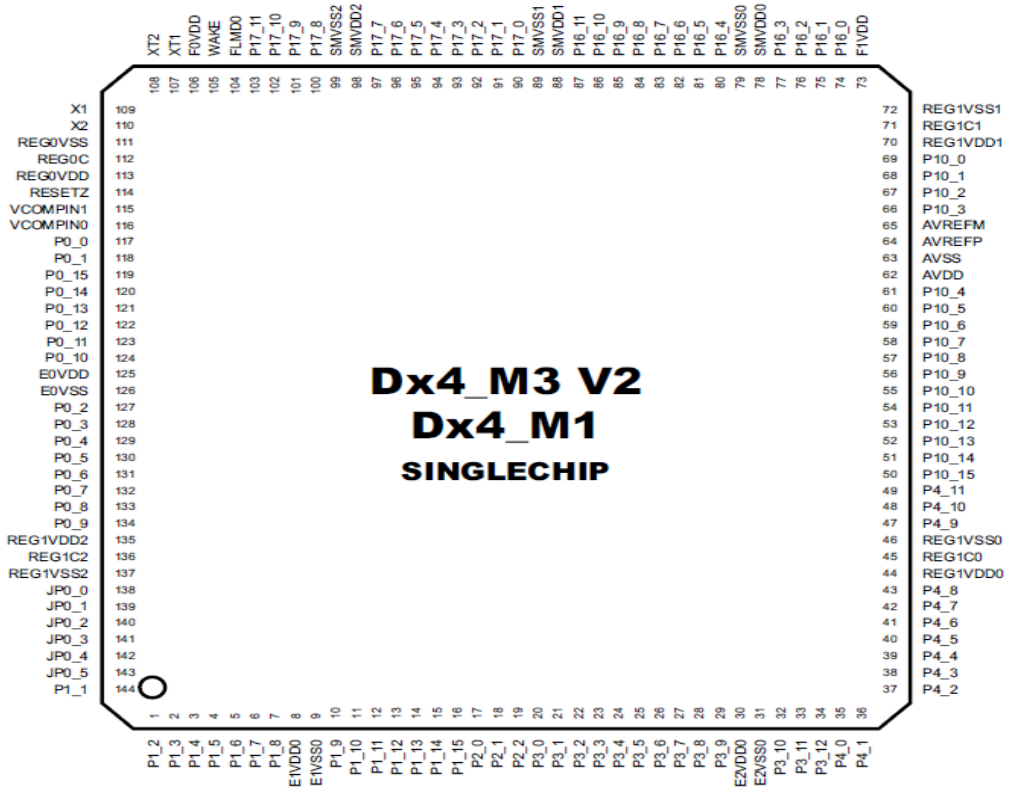


图 3.2 组合仪表平台微控制器概述

该 Dx4 系列芯片具有最大 144 个引脚，有 16 路 12 位的模拟数字转换器，有 5 个 16 通道 16 位的定时器，3 个 4 通道 32 位定时器，1 个给操作系统提供定时的专用定时器。另外有 3 路 CAN 通道（每路 CAN 通道自带 64 个硬件缓存），2 路 I2C 通道，可以支持 11 路外部硬线中断。芯片性能十分强大，完全能满足复杂组合仪表的设计要求，被各个主机厂和零部件供应商广泛使用。

3.1.2 电源供电模块

电源模块为整个仪表系统提供稳定的电力供应，在该仪表平台中，我们选用 12V 转 5V 的开关电源，可以实现对输入的车载电压信号滤波和稳压，并输出稳定的 5V 直流电压给微控制器芯片，再通过一个低压差线性稳压器转换为核心的 3.3V 电压提供给图像显示引擎，系统休眠后电源的功耗要求低于 120 微安，仪表需要工作在深睡眠的模式下，才能达到如此低的功耗要求，电源供电模块的电路设计原理图如下：

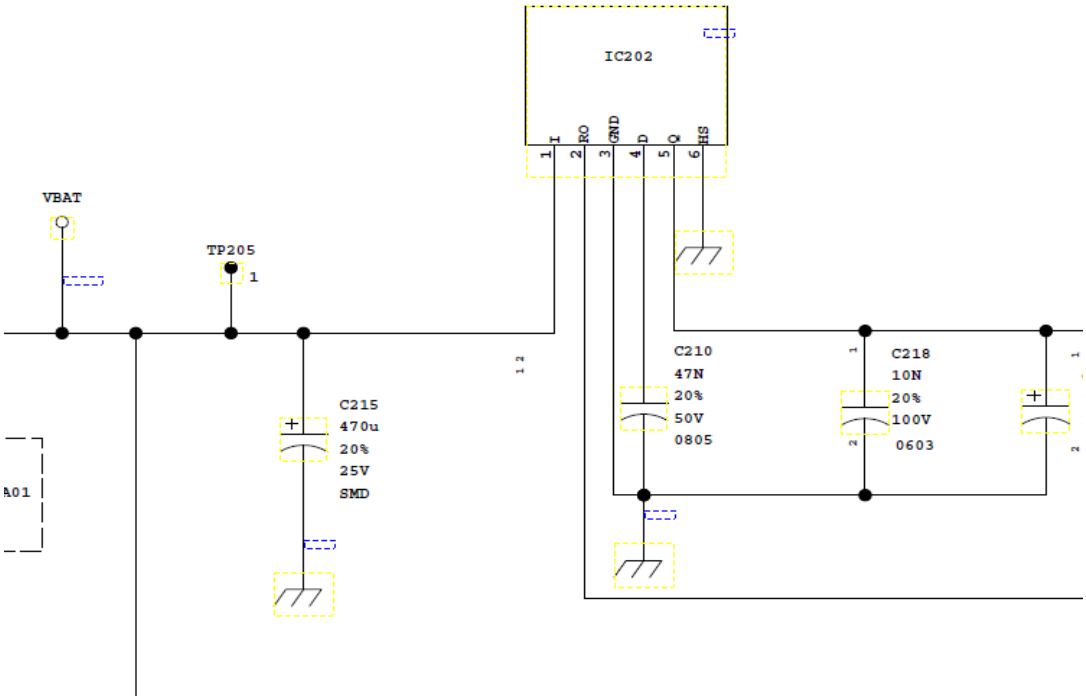


图 3.3 组合仪表平台电源供电模块

TPS7A6050 电源芯片可以支持 4V 到 40V 的宽电压输入，最大输出 300 毫安的驱动电流，在睡眠状态下，可以达到 2 微安的极低功耗，芯片内置集成了上电复位延时模块，可以灵活的设置芯片上电后供电输出稳定的时间，另外芯片内还有过流保护和短路保护电路，可以有效的保护芯片因为过流导致的损毁。

3.1.3 指示灯控制模块

指示灯控制模块用来指示当前的车辆报警信息，当车辆上某些关键部件发生故障时，通过指示灯可以及时有效的告知驾驶员，避免危险的发生。所有的报警指示灯电路都通过一个三极管实现与高压 5V 的驱动电路隔离，通过微控制器输出的高低电平信号控制该三极管的通断实现开关指示灯的报警功能，指示灯控制模块的设计原理如下图：

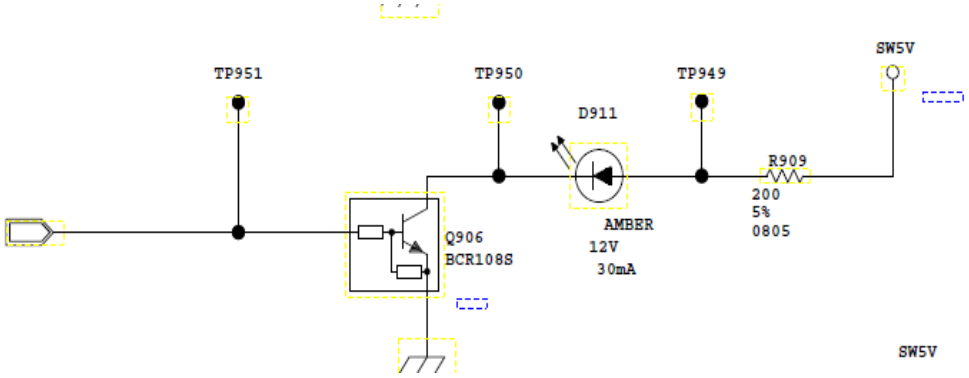


图 3.4 组合仪表平台指示灯控制模块

3.1.4 显示驱动模块

显示驱动模块用来驱动 4.2 寸彩色液晶显示屏，该显示屏是仪表显示系统的核心，红，绿，蓝三原色分别使用 8 根数据线传输，在布线时数据线需要考虑并行传输的信号的完整性问题，需要采用等长布线的方式以达到并行传输的信号的衰减阻抗一致的目的，显示驱动模块的原理图如下图所示：

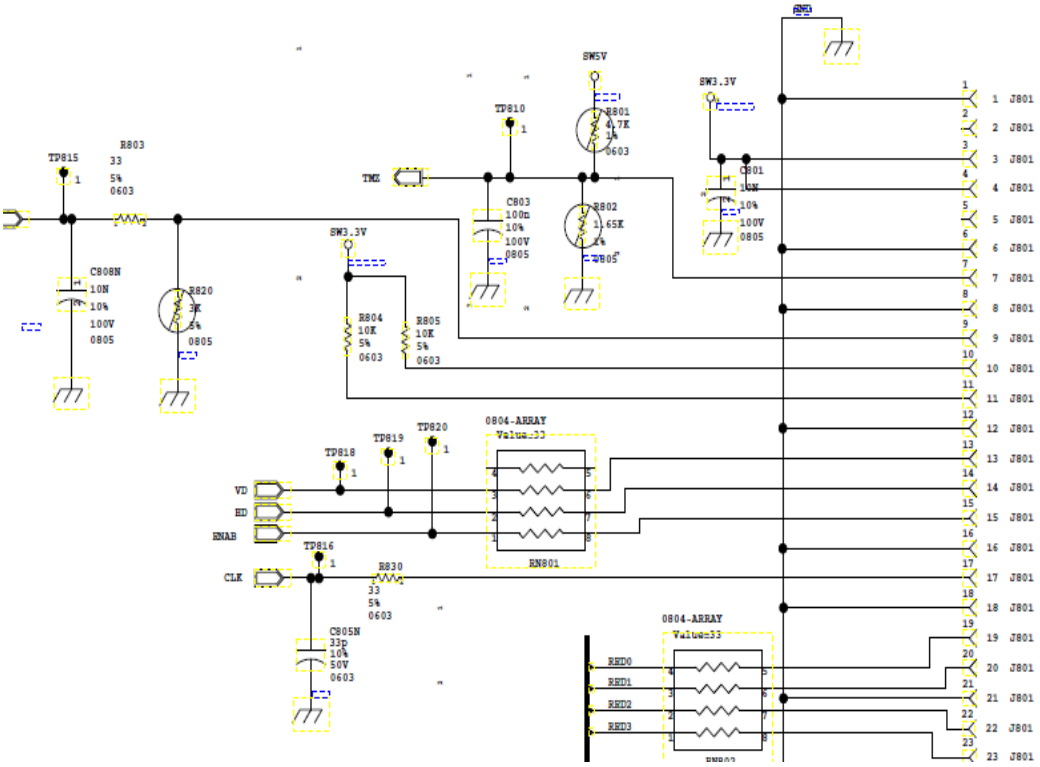


图 3.5 组合仪表平台显示驱动模块

3.1.5 步进电机驱动模块

组合仪表的车速表，转速表，油量表，水温表等四个表头采用机械式步进电机驱动，由于微控制器芯片内部自带步进电机的驱动电路，不需要在芯片外再增加额外的驱动芯片，只需要简单的将驱动电路连接到步进电机上就可以了，步进电机驱动电路如下图所示。

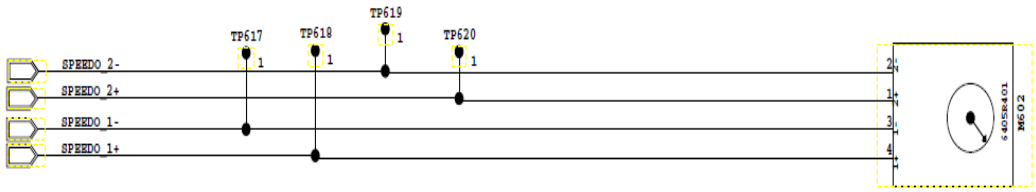


图 3.6 组合仪表平台步进电机驱动模块

3.1.6 输入信号采集模块

组合仪表的指示灯状态信号，是通过接插件的硬线连接输入，此处采集的车载信号主要是一些开关信号，信号从车载接插件上输入仪表后，通过分压电路后进入微控制器，使用控制器的通用输入输出功能采集到信号的变化情况，输入信号采集模块的原理图如下所示：

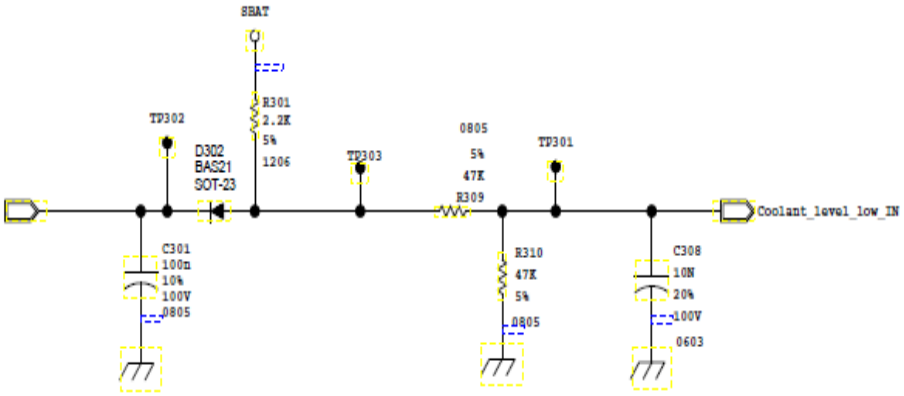


图 3.7 组合仪表平台输入信号采集模块

3.1.7 背光控制模块

汽车组合仪表要求在全天任何光照情况下都能清晰的指示车辆当前的状况，因此需要使用背光来应对一天中不同时间段的光线变化，目前的仪表平台中共设计三路背光，分别是仪表表头背光，指示灯背光，LCD 背光。背光输入的是方波信号，通过一个三极管外接到背光 LED 上，当输出不同的占空比的方波时，就可以调整背光指示灯的亮度，从而达到调整背光的目的，背光控制模块原理图如下所示：

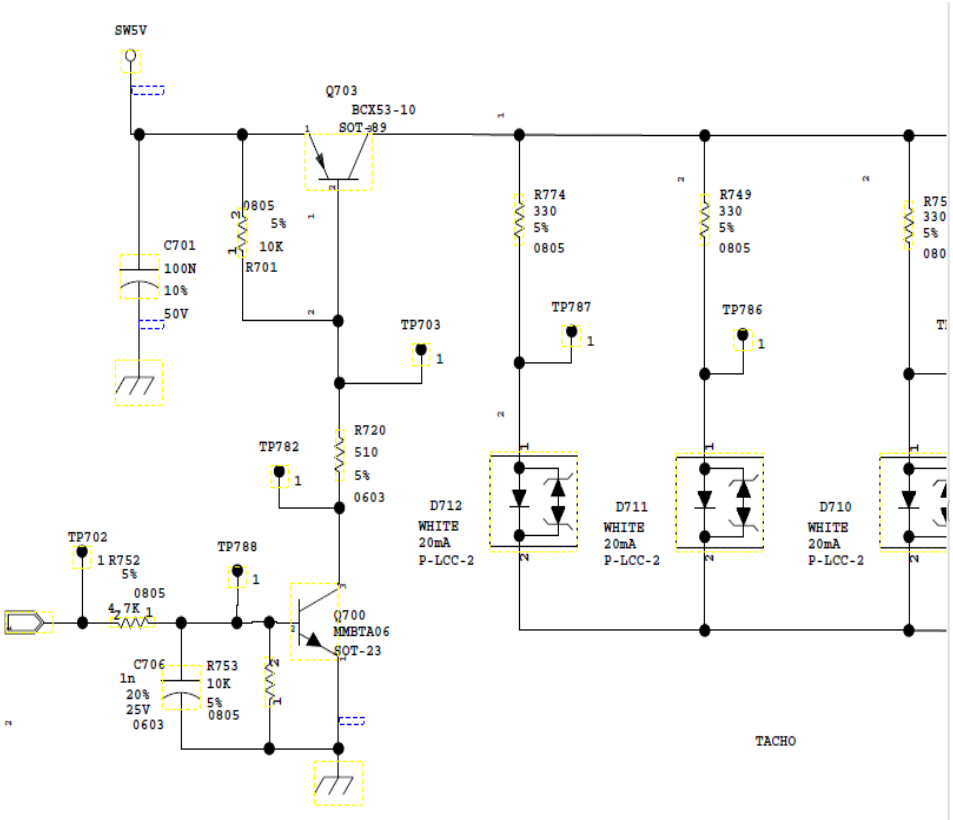


图 3.8 组合仪表平台背光控制模块

3.1.8 CAN 通信模块

网络管理采用 AUTOSAR 的部分网络管理策略，恩智浦公司的 TJA1145 收发器可以满足 ISO 11898-6 标准的要求。该收发器具有 SPI 通信接口，通过发送相关的配置命令，可以设置收发器中的接收滤波器，只允许特定的远程网络唤醒报文通过，从而达到部分唤醒网络的目的，在低功耗模式下只有微安级的电流消耗，可以满足车辆低功耗的严苛要求。为了保护芯片不受大电流的冲击，在 CAN_H 和 CAN_L 之间加了浪涌保护器件，可以有效防止意外大电流对收发器芯片的冲击，电路原理图如下：

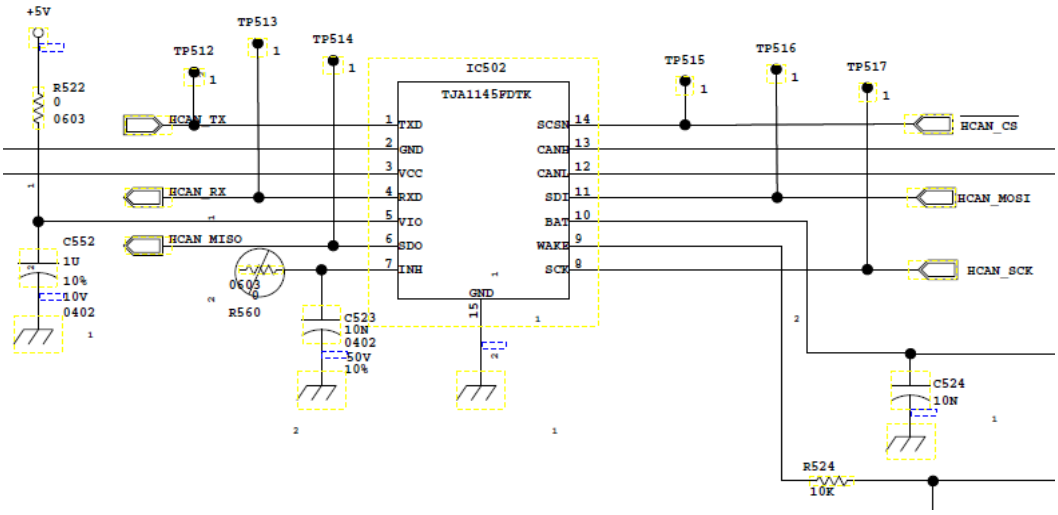


图 3.9 组合仪表平台 CAN 通信模块

3.2 本章小结

本章首先概述了汽车组合仪表的硬件架构框图，从总体上对组合仪表的硬件功能做了描述，然后分小节对组合仪表上的主要模块的硬件电路展开设计，分析了微控制器模块，电源模块，指示灯控制模块，显示驱动模块，步进电机驱动模块，输入信号采集模块，背光控制模块，CAN 总线通信模块等的设计原理，尤其对用到的关键的元器件及技术做了重点的说明，重点介绍了主控制器芯片，电源转换芯片，CAN 通信收发器芯片等，通过这些分析和描述，表明硬件电路的设计完全可以满足组合仪表的系统要求。

第 4 章 汽车组合仪表软件设计

4.1 组合仪表平台的软件架构

在汽车电子零部件发展的早期，有相当多的汽车电子零部件的规模都比较小，受工程师认识，客户规模和项目进度的影响，经常不做任何架构的设计，直接以实现功能为目标进行编码。这种设计方法从短期看是满足了进度，成本，功能各方面的需求，但从长远来看，在可扩展性和可维护性上付出的成本比最初节约的成本要高得多，尤其是当电子零部件的功能越来越复杂时，产品竞争越来越激烈的时候，这种差距尤其明显。如何设计一个好的软件框架结构，并且可以快速的实现在不同的项目和硬件平台间切换，就显得尤为重要。

在汽车电子领域，受限于成本的压力和技术的进步，会经常性的切换不同的硬件平台，一般传统的 ECU 在架构设计的时候不够充分，层次也不够清晰，应用层软件、中间层算法和底层的驱动全部契合在一起，没有很严格的接口标准的约束，这样的结构不但在 ECU 软件开发的过程中会导致接口调用的混乱，而且会导致在切换不同的项目或硬件平台时，很难复用以前的软件，在开发新的项目时，几乎所有的软件从应用层到底层驱动都需要重新再写一遍，浪费很多的人力和财力。

传统ECU的软件架构已经不能适应未来ECU的功能变化的要求，基于AUTOSAR的汽车电子软件架构，可以实现最大程度上的软件复用，可以减少硬件平台切换时人力和财力的投入，AUTOSAR 的软件架构是未来汽车电子零部件开发的趋势。基于 AUTOSAR 的软件架构开发的组合仪表平台架构如下图 4.1 所示。

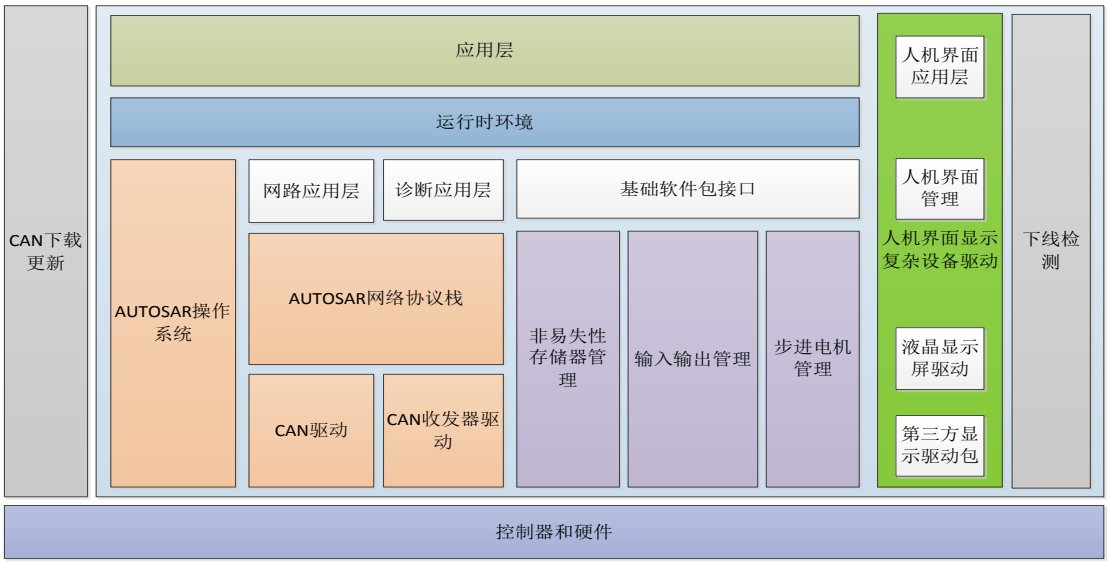


图 4.1 组合仪表平台软件架构

该仪表开发平台采用类似 AUTOSAR 的软件架构，从底层到顶层也是分为三层：基础软件层，运行时环境层，应用层。基础软件层又分为操作系统模块，网络通信协议栈和诊断模块，非易失性存储器管理模块，输入输出管理模块，步进电机管理模块和人机界面显示驱动模块。这其中操作系统模块，网络协议栈和诊断复用了从 Vector 公司购买的 AUTOSAR 的软件组件，这几个模块是在 Vector 公司的 Davinci configurator 工具中配置生成的。对非易失性存储器的存储算法和步进电机的驱动算法以及显示模块是以复杂设备驱动的形式移植到整体的软件架构中，输入输出模块封装了对底层输入输出端口的操作。运行时环境层，该层是由运行时环境配置工具生成代码，在运行时环境工具中可以配置应用层软件组件通信的端口，实现与基础软件层中各个软件模块的互联。运行时环境层之上的应用层软件采用 Simulink 建模实现，搭建了基于 Simulink 模型的设计和仿真平台。CAN 下载更新模块和下线检测模式作为相对独立的功能模块，主要用来满足客户在线刷新软件和工厂批量生产检测的要求。

下面将分别介绍该平台中的 AUTOSAR OS 操作系统配置和开发，AUTOSAR 网络通信配置和开发，AUTOSAR 诊断配置和开发，运行时环境工具开发，应用层模型开发和代码自动生成等。

4.2 AUTOSAR OS 配置和开发

AUTOSAR OS 的开发主要包含两部分的内容：操作系统的配置开发和操作系统应用层任务的调度开发。

4.2.1 AUTOSAR OS 的配置

平台采用的 AUTOSAR 操作系统是直接从 Vector 公司购买的，操作系统包含静态代码和动态配置代码，动态配置代码需要在 Davinci configurator 工具中配置，主要在配置工具中配置任务的种类，任务调度方式，操作系统运行的时钟心跳定时器，任务的堆栈大小，中断任务的设置等，操作系统在 Davinci configurator 中的配置如下图：

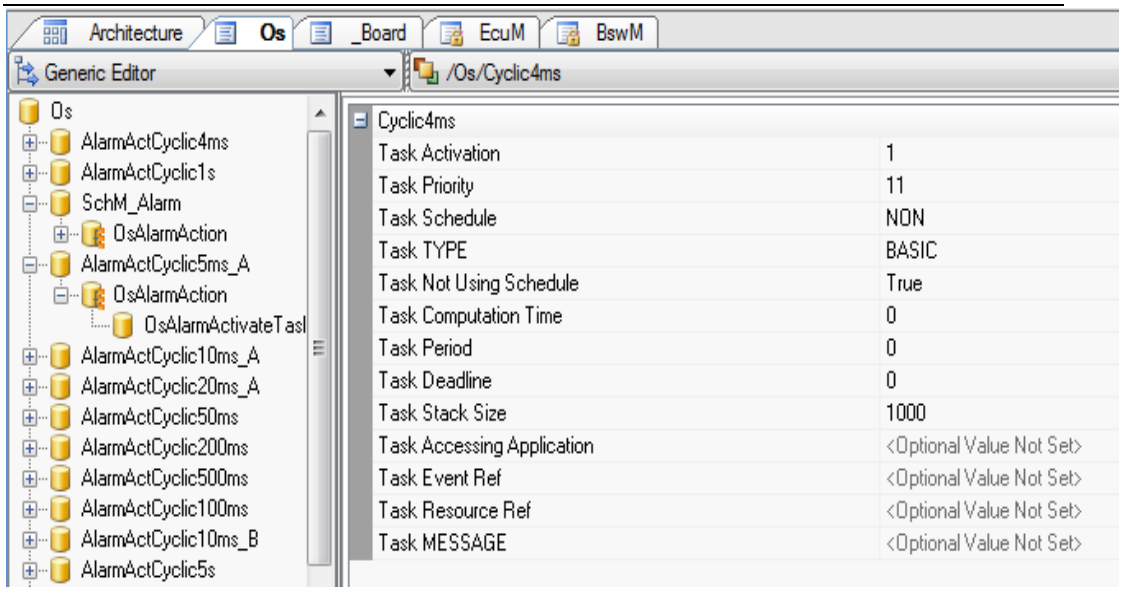


图 4.2 操作系统在 Davinci configurator 中的配置

根据应用层任务的需要，配置生成如下的应用层调度任务列表。

表 4.1 操作系统调度任务列表

任务名称	启动时间	任务类型	任务如何终止	堆栈大小（字节）
4 毫秒	3 微妙	周期调度任务	自行终止	1K
5 毫秒	1 微妙	周期调度任务	自行终止	1K
10 毫秒	2 微妙	周期调度任务	自行终止	1K
20 毫秒	5 微妙	周期调度任务	自行终止	1K
50 毫秒	9 微妙	周期调度任务	自行终止	1K
100 毫秒	10 微妙	周期调度任务	自行终止	1K
200 毫秒	8 微妙	周期调度任务	自行终止	1K
500 毫秒	18 微妙	周期调度任务	自行终止	1K
1 秒	19 微妙	周期调度任务	自行终止	1K
循环任务	20 微妙	循环调度任务	无终止	2K

在本组合仪表平台中，我们采用的是非抢占式任务调度策略，之所以选择非抢占式任务调度，是由于我们在任务调度的时候不用担心不同的任务访问同一数据会造成的并行问题，因为这种调度策略不允许同一数据被并行访问。

其中周期调度的任务除了会被中断打断外，正在运行的周期调度任务不可以被其他的任务抢占，每一个周期任务运行结束后会自行终止退出，这种类型的任务实时性比较好，主要用来处理应用中的需要周期性定时调度的任务，而循环调度的任务则是一直在运行的，可以被周期性调度的任务打断，实时性没有周期性调度的任务好，主要用来处理液晶显示屏上的图形描画等。

4.2.2 操作系统的应用层任务设计

应用层中每一个具体任务需要加入操作系统的调度任务表中，这些任务主要包含信号采集任务，报警指示灯的控制任务，背光控制，步进电机的控制，液晶显示屏上的图形描画显示任务等，各个应用层的任务在操作系统调度表中的排布如下表 4.2 所示。

表 4.2 OS 调度任务列表

任务名称	任务内容
4 毫秒	4 毫秒基准的定时器
5 毫秒	应用层网络信号更新 指示灯报警控制
10 毫秒	网络管理及网络状态切换 数字和模拟信号采集 电源模式控制及电压监控 背光控制 车速表，转速表，油量表，水温表信号采集及控制
20 毫秒	ComM 状态管理 LCD 上显示的弹出报警管理
50 毫秒	LCD 上显示的报警灯控制
100 毫秒	步进电机的驱动 油耗相关的计算
200 毫秒	休眠状态的管理
500 毫秒	500 毫秒的定时器
1 秒	1 秒的定时器
循环任务	显示描画图片管理

4.3 AUTOSAR 通信的配置和开发

平台采用的 AUTOSAR 的通信协议栈也是直接从 Vector 公司购买的，网络通信及网络管理代码是在 Vector 公司提供的 Geny 配置工具中生成的。下面分别介绍 Com 模块，ComM 模块，Dcm 模块，Dem 模块，PduR 模块等的配置开发及网络应用层的开发。

4.3.1 Com 模块

AUTOSAR 中 Com 模块介于 RTE 和 PDU Router 模块之间，它的基本功能是将 RTE 发送的信号以一定的规则打包发送，另外是将底层传输来的 I-PDU 拆分成各个信号发送给 RTE。在 Com 模块还定义发送 I-PDU 的方式，以及对接收到的信号进行作滤波处理。除此之外还能在每次成功发送或者接收到数据后通知 RTE。

ComSignals 和 ComIPdus 分别是 COM 中信号和 I-PDU 配置信息的集合。ComIPdu 中包括 I-PDU 的方向，类型，和信号组。而 ComSignals 中包含信号在 I-PDU 中的起始位置，长度，字节大小端顺序，信号的数据类型等。

Com 模块在接收到信号时，可以对其进行过滤操作，如果通过过滤操作，则数据会被发到 RTE，否则数据被丢弃。Com 模块的配置如下：

Configurable Options	Com
Common	
Configuration Variant	Variant 1 (Pre-compile Configuration)
Version Info Api	☑ *
Dev Error Detect	☐ *
Prod Error Detect	☑ *
Miscellaneous	
Use Rte	☐ *
DeInit Api	☑ *
Reception DM Api	☑ *
Get Status Api	☑ *
Trigger IPDU Send Api	☑ *
Trigger Transmit Api	☑ *
Signal Invalidation Api	☑ *
Configuration Id	
Get Configuration Id Api	☑ *
Configuration Id	0*
Configuration String	
Configuration String	*
ComConfigurationTimeBase	
Configuration Time Base [s]	0.001*
Configuration Rx Time Base [s]	0.001
Configuration Tx Time Base [s]	0.001
Addons	
Enable Timeout Flags	☐ *
Signal Access Macro Api	☑
Tx Timeout For Tx Mode None	☐ *
Dynamic DLC Support	☐ *
Optimizations	
Signal Reception Filtering	☑ *

图 4.3 AUTOSAR Com 模块配置

4.3.2 ComM 模块

ComM 模块是通信管理的总模块，它直接封装了 CAN 等网络状态管理和网络管理服务，它负责向通信管理者提供总的服务。ComM 收集来自不同 SWC 的通信请求并协调通信。当 ComM 收到 CAN 通信请求后，首先进行仲裁，以判断是否进入通信模式，ComM 同时控制 CanSM 模块进行 CAN 总线状态的管理，目的是使得总线一直处于唤醒的状态。ComM 模块还会控制 NM 模块，使得总线上其他的 ECU 模块同时处于通信状态。当一个模块请求全通信模式时，ComM 模块会检查通信是否被允许，并阻止 ECU 在通信时关闭。ComM 模块的配置如下图所示：

Configurable Options	ComM
Common	
Configuration Variant	Variant 1 (Pre-compile Configuration)
Version Info Api	☑
Dev Error Detect	☐
Prod Error Detect	☑ *
User Configuration File Path	* ...
Service Component Description	
Use Rte	☐
Software Component Name	ComM*
Req Mode Port Prefix	UR_*
Current Mode Port Prefix	UM_*
Miscellaneous	
ComM Synchronous Wake Up	☐ *
Mode Limitation Settings	
Mode Limitation Enabled	☐ *
Partial Network Configuration	
Pnc Support	☑
Pnc Prepare Sleep Timer [s]	1.5*
Pnc Gateway Enabled	☐ *

图 4.4 AUTOSAR ComM 模块配置

4.3.3 PduR 模块

PduR 模块作用是为 I-PDU 提供路由，路由的使用模块主要有接口通信模块：Com,CanIf,CanNm 等，传输协议模块：CanTp，Dcm，I-PDU 的路由全部是通过 I-PDU 的 ID 静态配置得到的。PduR 模块主要执行三种类型的操作：从下层模块接收 I-PDU 转发到上层模块，按照上层模块的请求，将 I-PDU 发送到下层模块，作为 PDU 网关。PduR 的配置如下图：

Configurable Options	PduR
Common	
Configuration Variant	Variant 1 (Pre-compile Configuration)
Version Info API	☑
Dev Error Detect	☐ *
Prod Error Detect	☑ *
Pre-compile Configuration	
Dynamic DLC Support	☐ *
PduR 2 Provide Rx Buffer Calls	
PduRCanTp2ProvideRxBufferCalls	☑
PduRFrTp2ProvideRxBufferCalls	☐ *
Miscellaneous	
Balance Routing Time	☐ *
PduR Use RAM Manager	☐ *

图 4.5 AUTOSAR PduR 模块配置

4.3.4 网络与应用层的接口设计

在 Com 模块之上，通过调用 Com 层提供的 Com_ReceiveSignal 函数，把接收到的信号重新封装到应用层的数据结构中，处理过程如下流程图所示：

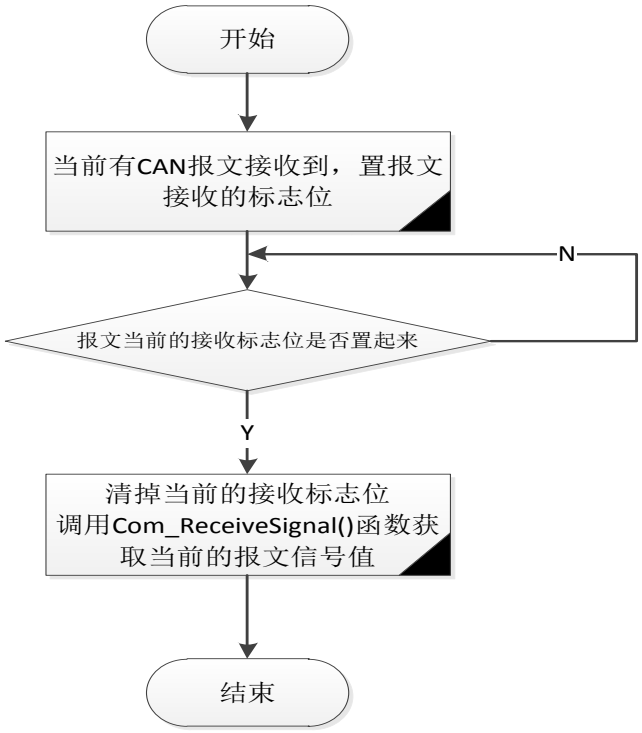


图 4.6 网络应用层报文接收处理过程

仪表需要接收和发送的应用报文，在网络应用层定义如下类型的数据结构，将接收到的每一帧报文中的每一个信号全部解析出来，放到该数据结构中，应用层模块通过 RTE 层直接调用相应的网络接口。

Structure Name	VNIM_RX_PPEI_ENGINE_GENERAL_STATUS_4_t				
Description	network application message structure				
Structure Scope	Public				
Defined in	vnim_app_signals.h				
Member List	Variable	Type	Min	Max	Description(If any)
	msg_flag	vnim_unsigned8	0	255	message signal
	engclttmpv	vnim_unsigned8	0	255	message signal
	barprsabsv	vnim_unsigned8	0	255	message signal
	barprsabs	vnim_unsigned8	0	255	message signal
	engclttmp	vnim_unsigned8	0	255	message signal
	engoilhotio	vnim_unsigned8	0	255	message signal

图 4.7 网络应用层报文数据结构

通过在 OS 任务中调用如下的函数，把报文帧中的信号更新到网络应用层的

数据结构中。该函数的入口参数和返回值全部为空，在函数内部对网络应用层的全局数据结构进行操作，更新信号到该数据结构中。

Name	vnim_update_msg_PPEI_Engine_General_Status_4					
Description	update com signal to network application					
Return type	void					
Return Range	**Min**		**Max**		**Pointer type**	No
Defined in	vnim_app_signals.h					
Input Parameters Details						
S.No	Variable Type	Variable name	Input/Output	Type remarks	Description	
1	void	void				
	Min		Max			
2						
	Min		Max			

图 4.8 网络应用层接口函数

4.4 AUTOSAR 诊断的配置和开发

AUTOSAR 的诊断主要包含 Dem 模块，Dcm 模块的配置及诊断应用层的开发。

4.4.1 Dem 模块

Dem 模块是负责诊断事件管理及其相关数据处理和存储的模块。Dem 还向 Dcm 模块提供故障的信息，Dem 对报故障的诊断时间进行判断，在故障确认后会对事件相关的故障码和诊断信息进行锁存，Dem 模块主要有两大功能：事件存储管理和事件状态管理。事件存储管理包括对在 Dem 模块内外的事件记录，更新和移除等，事件状态管理是处理监控器的监控结果，事件的状态与目前的环境及监控结果有关联，所以事件状态管理子模块需要合理的管理这些情况。Dem 的配置如下：

Configurable Options	CanTp
— Common	
Configuration Variant	Variant 1 (Pre-compile Configuration)
Version Info Api	☑
Dev Error Detect	☐
Prod Error Detect	☑ *
Oper Not Supported Report Dem	☐ *
E Comm Report Dem	☐ *
User Config File	- ...
— Miscellaneous	
Main Function Period [ms]	10
— Precompile configuration features	
Padding Active	☑ *
Support RxIndication	☑ *
Support TxConfirmation	☑ *
— CanTp SDUs	
Add TP channel	...
+ Advanced Options	

图 4.9 Dem 模块配置

4.4.2 Dcm 模块

Dcm 模块对于诊断通信服务提供常用的功能及接口，诊断模块在 ECU 开发，制造，和服务阶段都可能会被外部诊断设备访问。Dcm 模块确保诊断数据流的正确性和诊断状态的管理。另外 Dcm 模块还检测当前的服务请求在当前的会话模式下是否支持。Dcm 模块的配置如下：

Configurable Options	Dcm
Common	
Configuration Variant	Variant 1 (Pre-compile Configuration)
Version Info Api	☑
Dev Error Detect	☐ *
User Config File	*
Software Component Template	
Use Rte	☐
Software Component Name	Dcm*
Miscellaneous	
Task Time [ms]	10*
Respond All Request	☑ *
Paged Buffer Enabled	☐ *
Dsl Diag Resp Force Resp Pend En	☐ *
Periodic Tx Type	Type II
Roe Tx Type	Type II
Pseudo Parallel Request Handling	Exclusive Silent
Runtime Limiter	10*
Unspecified Service Support	☑
Variant Mode Selection	SIMPLE_ECU
Diagnostic Request Notifications	
Supplier Notification Support	☐
Manufacturer Notification Support	☐

图 4.10 Dcm 模块配置

4.4.3 UDS 诊断应用层的开发

在统一诊断协议中，通过调用不同的诊断服务满足不同的应用需求，常用的统一诊断协议中的诊断服务如下表所示，在该平台中是通过在 CAN dela studio 软件中编辑诊断的配置文件，在诊断配置文件中写明诊断需要支持哪些服务号和数据服务号，然后将诊断配置文件导入 Davinci Configurator 工具中，生成符合统一诊断协议框架的代码，最后在预留的诊断应用层接口添加诊断应用层函数。

UDS Diagnostic Services ID	sub- function	Operation Software	Bootloader Software
0x10	01	M	M
	02	C1	M
	03	M	M
0x27	01	C2	M
	02	C2	M
	03	C3	N/A
	04	C3	N/A
0x28	00	M	M
	01	U	N/A
	02	U	N/A
	03	M	M
0x3E	00	M	M
0x22	--	M	M
0x23	--	U	N/A
0x2A	--	M	N/A
0x2C	01	M	N/A
	02	U	N/A
	03	M	N/A
0x2E	--	M	M
0x85	01	M	M
	02	M	M
0x14	--	M	N/A
0x19	01	M	N/A
	02	M	N/A
	03	C4	N/A
	04	C4	N/A
	0A	M	N/A
0x2F	--	C5	N/A
0x31	01	C6	M
	02	C6	M
	03	U	N/A
0x34	--	N/A	M
0x36	--	N/A	M
0x37	--	N/A	M

图 4.11 AUTOSAR 诊断服务

4.5 AUTOSAR 运行时环境层工具开发

在标准的 AUTOSAR 架构中，所有的软件组件是基于虚拟功能总线设计，虚拟功能总线是不同的软件组件间信息沟通的桥梁，而运行时环境是虚拟功能总线的具体实现，运行时环境即 RTE 是介于应用层和基础软件层间的接口层，它提供应用层不同的组件间通信的路由，应用层组件可以通过运行时环境层访问同一个 ECU 中不同的应用组件，也可以通过 ECU 的通信模块访问不同 ECU 上的组件，这些信息的路由全部由运行时环境层和运行时环境配置工具完成。

在该组合仪表平台中设计实现了运行时环境层的功能，通过运行时环境配置

管理工具，可以很方便的实现不同的应用层软件组件间信息的交互。该工具是使用 C++语言编写，通过读取应用层的不同软件组件间的接口配置表，获取不同的软件组件间数据的通道信息，并按照标准的 AUTOSAR 读写函数实现访问基础软件层的软件。

运行时环境层代码生成工具界面如下图 4.12 所示：

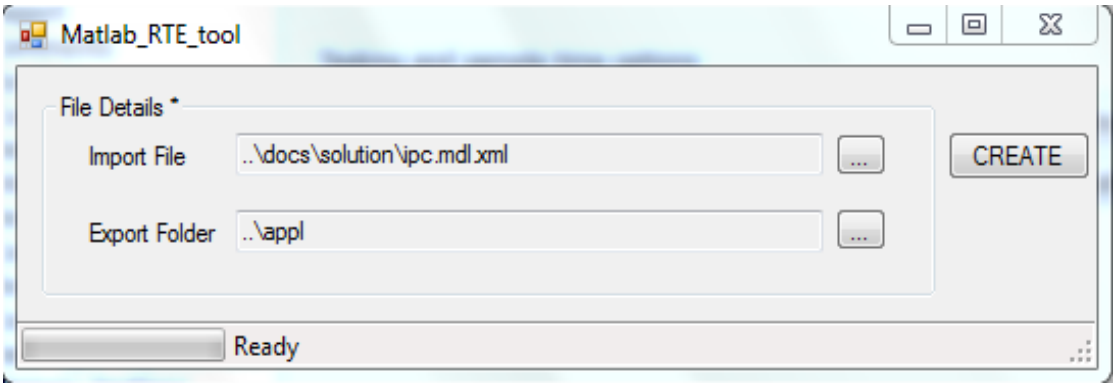


图 4.12 RTE 层代码生成工具

运行时环境层读取数据接口函数和运行时环境写入数据接口函数如下表 4.3 和表 4.4 所示，两者参考使用标准的 AUTOSAR 读写函数接口。

表 4.3 运行时环境读取数据接口函数

Name	PnRte_Read_CanSysPwrMd					
Description	从CAN总线上读取当前的发动钥匙模式					
Return type	TeIPC_e_PwrMod					
Return Range	**Min**	0	**Max**	3	**Pointer type**	No
Input Parameters Details						
S.No	Variable Type	Variable name	Input/Output	Type remarks	Description	
1						
	Min		Max			
2						
	Min		Max			

表 4.4 运行时环境写入数据接口函数

Name	PnRte_Write_IpcPwrMod				
Description	把当前的电源模式从SWC写入CAN网络				
Return type	void				
Return Range	**Min**		**Max**		**Pointer type**	
Input Parameters Details					
S.No	Variable Type	Variable name	Input/Output	Type remarks	Description
1	eIPC_e_PwrMo	value			
	Min ->	0	Max ->	3	
2					
	Min->		Max->		

4.6 汽车组合仪表应用层功能模型开发

仪表的应用层功能主要采用 Mathworks 公司的 Matlab Simulink 工具设计，基于模型的开发的设计思路，设计实现了仪表所有应用层的模型，应用层的代码使用 Matlab 中的嵌入式代码生成器自动生成。

汽车组合仪表上需要实现下面这些模型：车速表输出当前车辆行驶的车速值，转速表需要输出当前车辆行驶的转速值，油量表需要指示正确的当前油箱中的剩余油量，另外在组合仪表的液晶显示屏上还需要显示多个与车辆行驶状况相关的信息页面，这其中包括：发动机当前的瞬时油耗值，车辆的平均油耗值，车辆的最佳油耗值，车辆的可续驶里程值等，下面分别介绍这些模型的详细设计。

4.6.1 车速表控制模型

仪表上电后，车速表首先执行复位动作，然后指针回零到机械零位，此时为了校验仪表指针能否正确的指示，仪表指针会先从零点开始在整个表盘上旋转一周到最大值点，然后再从最大值点回到机械零位。在正确的指示车速值之前，首先要对输入的车速信号值做滤波处理，以获得干净稳定的车速输出，经过滤波的车速值还需要按照法规的要求，做一定比例的偏移，目的是使得指示的车速值比实际的车速值要大，然后将车速值对应转换为步进电机的步数（表盘的 1 度等于步进电机 12 步），通过如下的分段查找表获取当前车速值对应步进电机的步数。通过调用底层的步进电机驱动指示当前的车速值。

表 4.5 车速值与步进电机步数对应表

实际车速值	0	20	60	100	140	200	220
偏移车速值	0	21	63	105	147	210	220
步进电机值	48	338	917	1497	2077	2656	2084

车速表的控制流程图如图 4.13 所示：

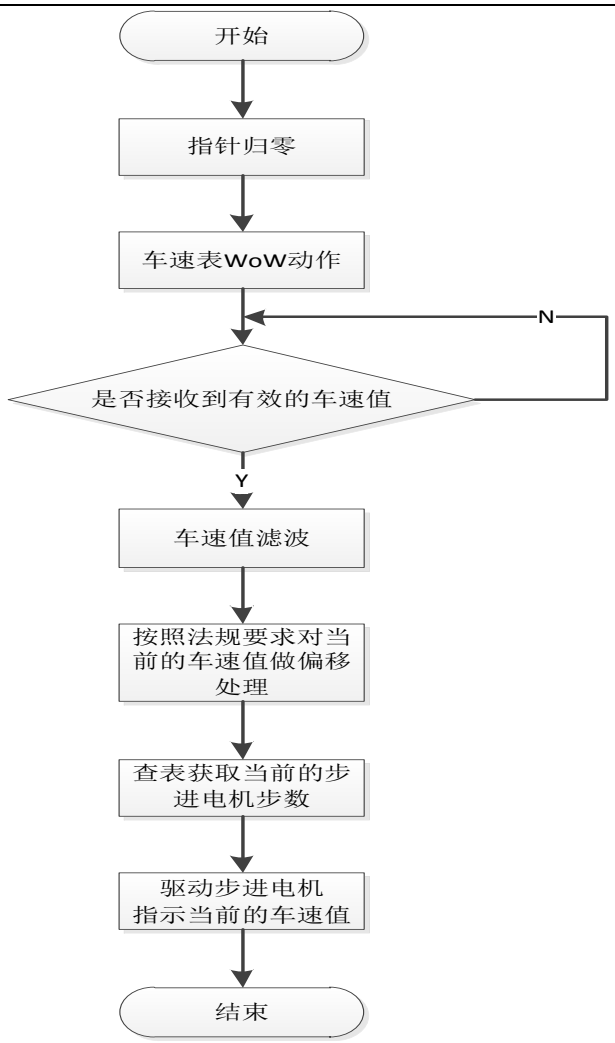


图 4.13 车速表控制流程图

车速表控制模型如下图 4.14 所示。

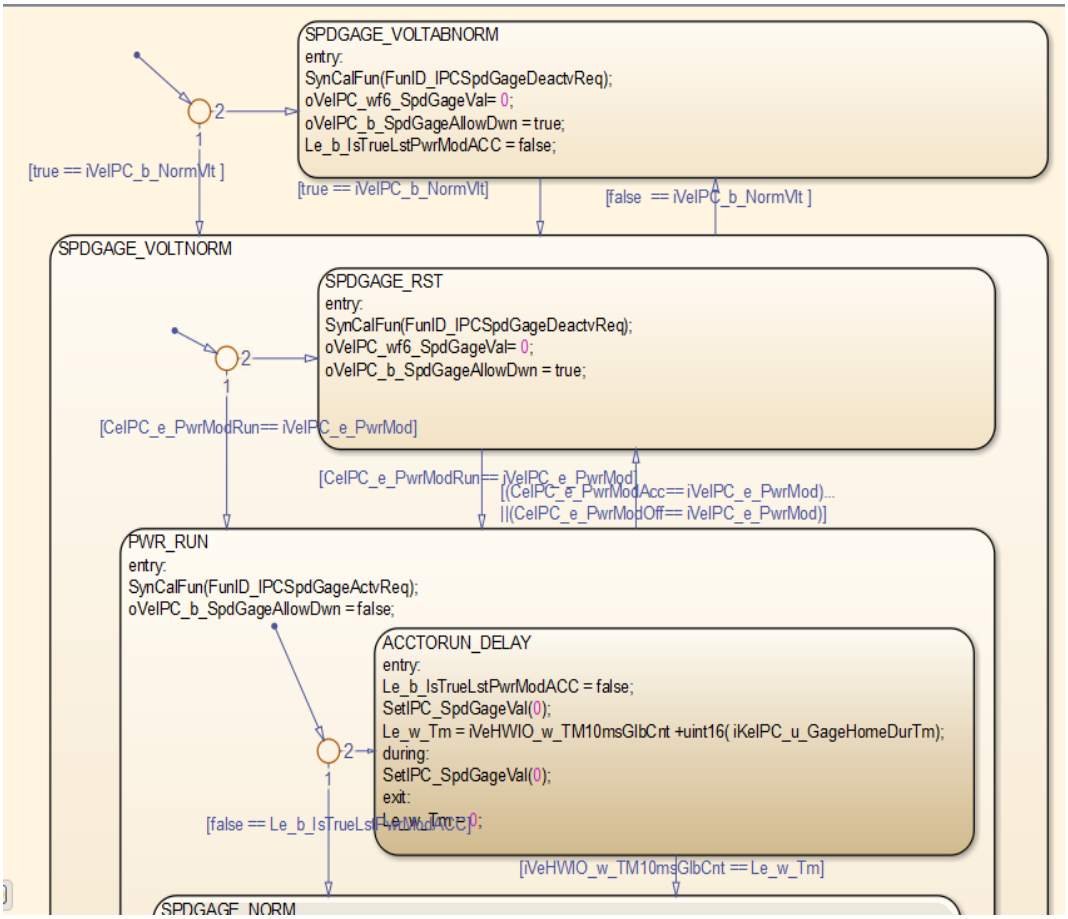


图 4.14 车速控制模型

在该车速表控制的状态迁移图中，分为电压正常状态，电压异常状态，点火开关运行状态，点火开关关闭状态，车速表正常运行状态，车速表复位状态等。当仪表上电时，首先检测当前系统的电源电压值，如果处于 9-16V 状态，则是电压正常状态，否则迁移到电压异常状态，在电压异常状态时，需要发出当前车速表不激活的状态，并将该状态存储到非易失性存储器中。当在电压正常状态时，首先判断点火开关是否处于运行状态，如果是在运行状态，则发出当前车速表激活的信号，否则进入车速表复位状态，在复位状态需要发出并存储车速表不激活的状态。在点火开关运行状态，需要判断当前的启动钥匙状态是 ACC 状态还是 RUN 状态，如果是 ACC 的状态，则车速表指向 0 的位置，否则车速表需要根据当前输入的车速值信号，正确的指示当前的车速值。

4.6.2 转速表控制模型

仪表上电后，转速表首先执行复位动作，然后指针回零到机械零位，此时为了校验仪表指针能否正确的指示，仪表指针会先从零点开始在整个表盘上旋转一

周到最大值点，然后再从最大值点回到机械零位。输入的发动机转速值通过如下的分段查找表获取当前发动机转速对应步进电机的步数，通过调用底层的步进电机驱动在转速表上指示当前的转速值。

表 4.6 转速值与步进电机步数对应表

实际转速值	0	400	500	1000	4000	8000
步进电机值	48	258	363	468	1098	1938

转速表的控制流程图如下图 4.15

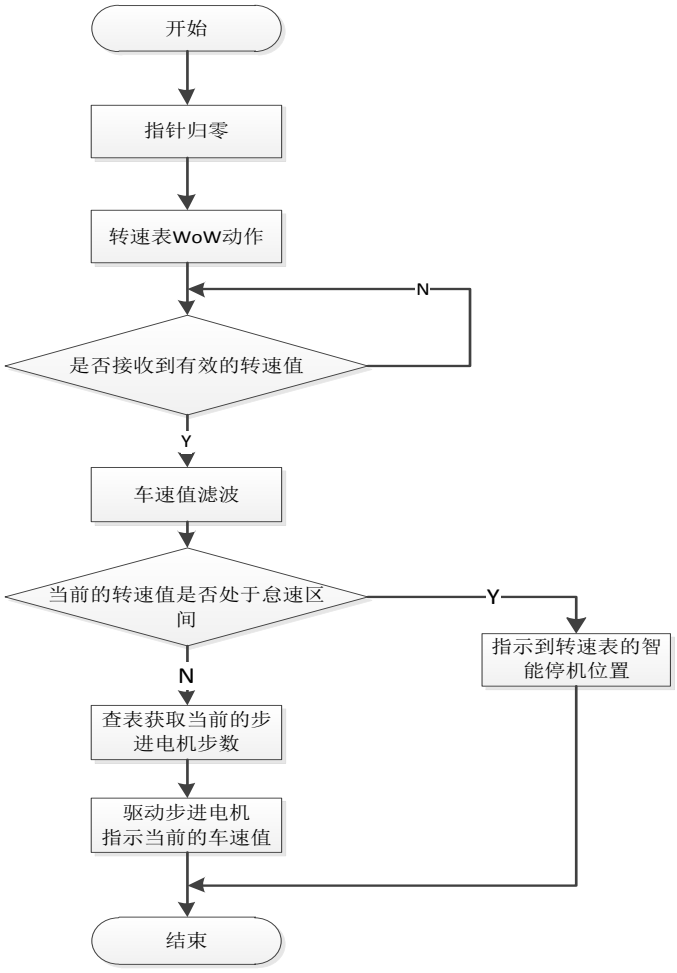


图 4.15 转速表控制流程图

转速表控制模型如下图 4.16 所示：

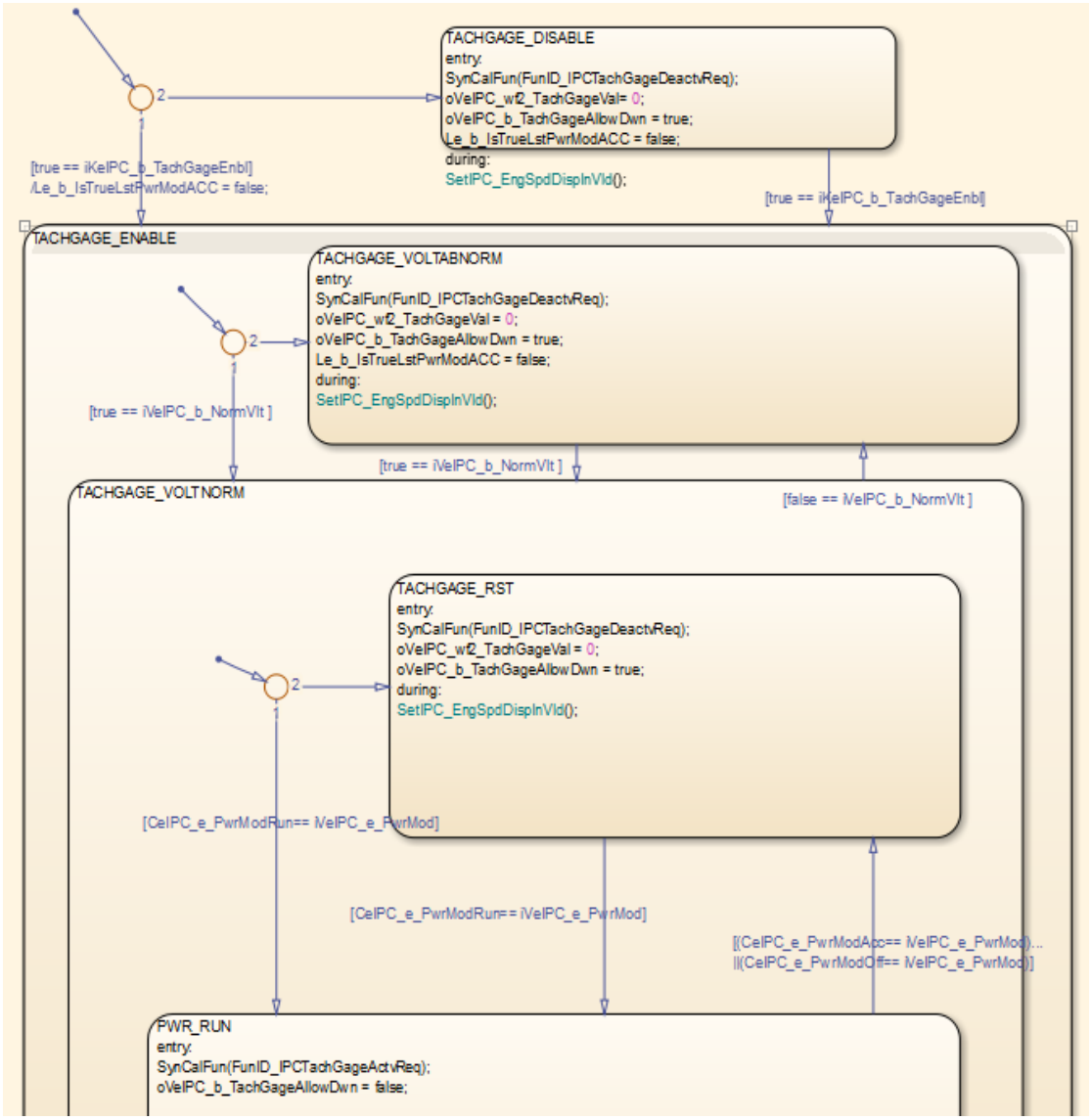


图 4.16 转速控制模型

在该转速表控制的状态迁移图中，分为转速表使能状态，转速表非使能状态，电压正常状态，电压异常状态，转速表正常运行状态，转速表复位状态等。如果转速表使能标定量为真，则使能转速表，否则转速表处于非激活状态，当仪表上电时，检测当前系统的电源电压值，如果处于 9-16V 状态，则是电压正常状态，否则迁移到电压异常状态，在电压异常状态时，需要发出当前转速表不激活的状态，并将该状态存储到非易失性存储器中。当在电压正常状态时，首先判断点火开关是否处于运行状态，如果是在运行状态，则发出当前转速表激活的信号，否则进入转速表复位状态，在复位状态需要发出并存储转速表不激活的状态。在正常运行状态，转速表需要根据当前输入的发动机转速值，正确的指示当前的转速值。

4.6.3 油量表控制模型

油量表用来指示当前车辆油箱中的剩余油量，组合仪表上的油量表一般如下图 4.17 所示：

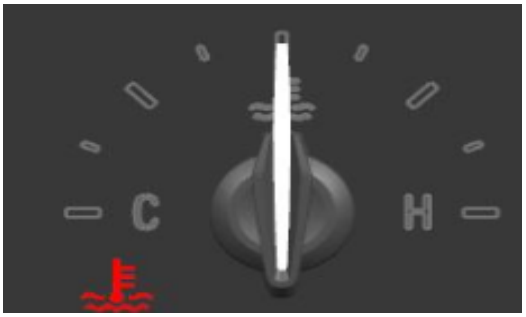


图 4.17 油量表示意图

仪表上电后，油量表首先执行复位动作，然后指针回零到机械零位，此时为了校验仪表指针能否正确的指示，仪表指针会先从零点开始在整个表盘上旋转一周到最大值点，然后再从最大值点回到机械零位。从油箱采集当前的油量信息，并将该滤波后的油量信号输出给油量表的步进电机驱动模块，并控制油量表在不同的工况下切换，可以稳定的指示油箱当前的油量，油量表的控制模型如下图 4.18 所示。

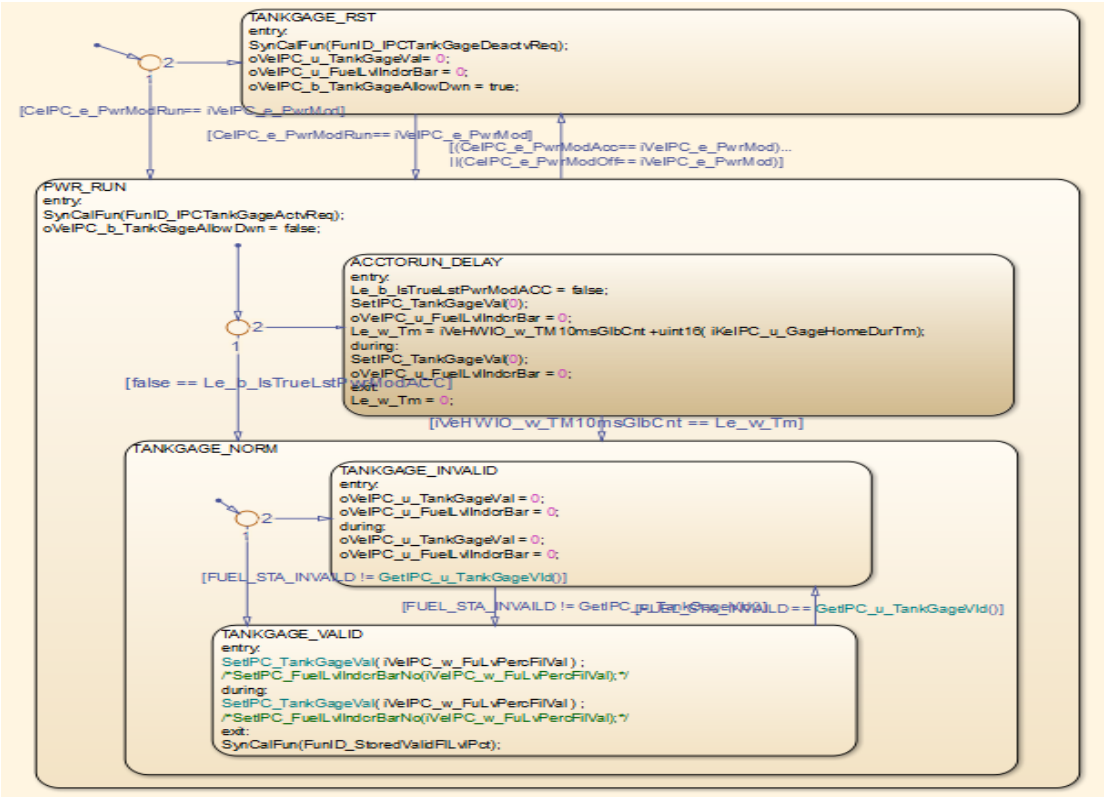


图 4.18 油量表控制模型

在该油量表控制的状态迁移图中，分为油量表复位状态，油量表正常状态，油量表有效状态，油量表无效状态等。如果钥匙点火是运行状态，则进入油量表正常指示状态，在正常指示状态中，通过获取当前的油量表信号值是否有效，从而决定是否进入油量表有效状态，在油量表有效状态，依据输入的油量信号值，正确的指示当前油箱的剩余油量，否则油量表指示到 0 的状态。在复位状态，同样需要让油量表指示到 0 的状态。

4.6.4 瞬时油耗模型

瞬时油耗模型累计当前发动机喷油嘴的喷油量，将累计的喷油量值输出给液晶显示屏上，在瞬时油耗界面显示发动机当前的瞬时喷油值，瞬时油耗显示的情况如下图 4.19 所示：

图 4.19 瞬时油耗在组合仪表上的显示

瞬时油耗的模型如下图 4.20 所示。

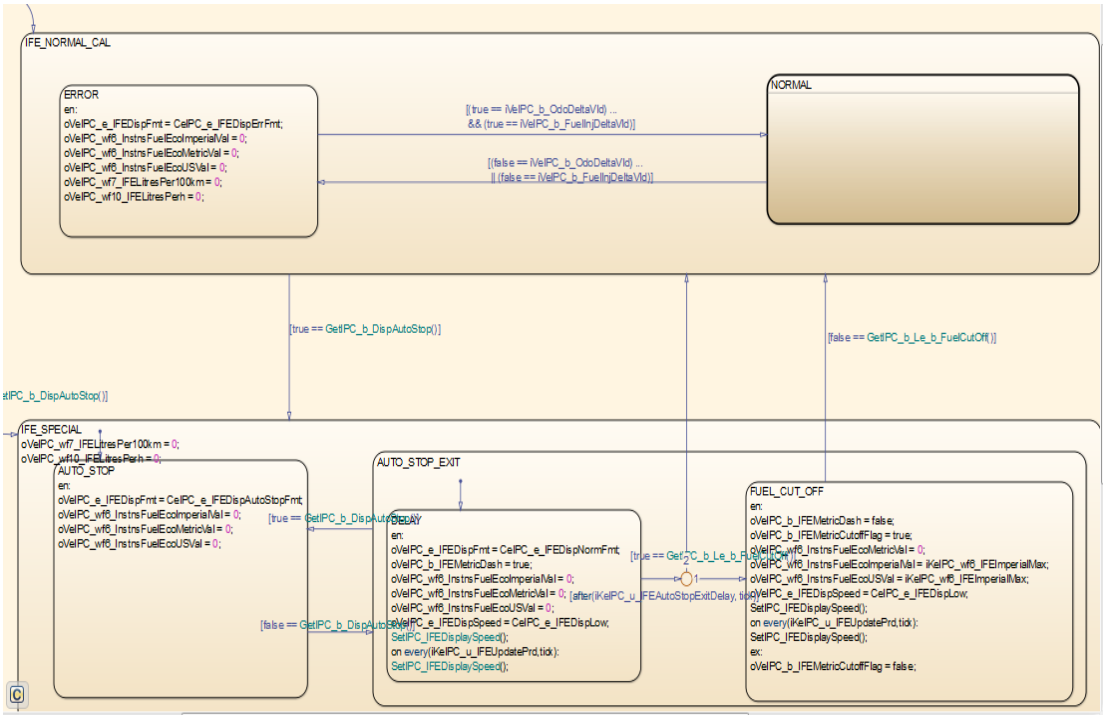


图 4.20 瞬时油耗模型

在该瞬时油耗模型的状态迁移图中，可以分为瞬时油耗计算状态，瞬时油耗特殊状态，在瞬时油耗计算状态中又可以分成错误状态和正常计算状态，在瞬时油耗特殊状态中，分为自动停机状态，自动停机退出状态，在自动停机退出状态又分为延时状态和燃油中断状态。首先通过判断当前的自动停机标志位是否被置上，以确定是否进入自动停机状态，如果进入自动停机状态，需要在 LCD 屏上显示自动停机的文字，否则进入瞬时油耗计算状态，在计算的过程中如果发生报文丢失或者信号变成无效时，需要切换成错误状态，显示瞬时油耗值为 0。在智能停机状态时，延迟 1 秒钟后自动退出自动停机状态，判断当前的燃油是否中断供应，如果燃油中断供应，则显示瞬时油耗值为 0。

4.6.5 平均油耗模型

平均油耗模型不停的统计发动机在一定的距离后的喷油量，并将两者做除得到平均的油耗值输出到液晶屏上显示，平均油耗在组合仪表上的显示情况如图 4.21 所示：

图 4.21 平均油耗

平均油耗的控制计算模型如下图 4.22 所示。

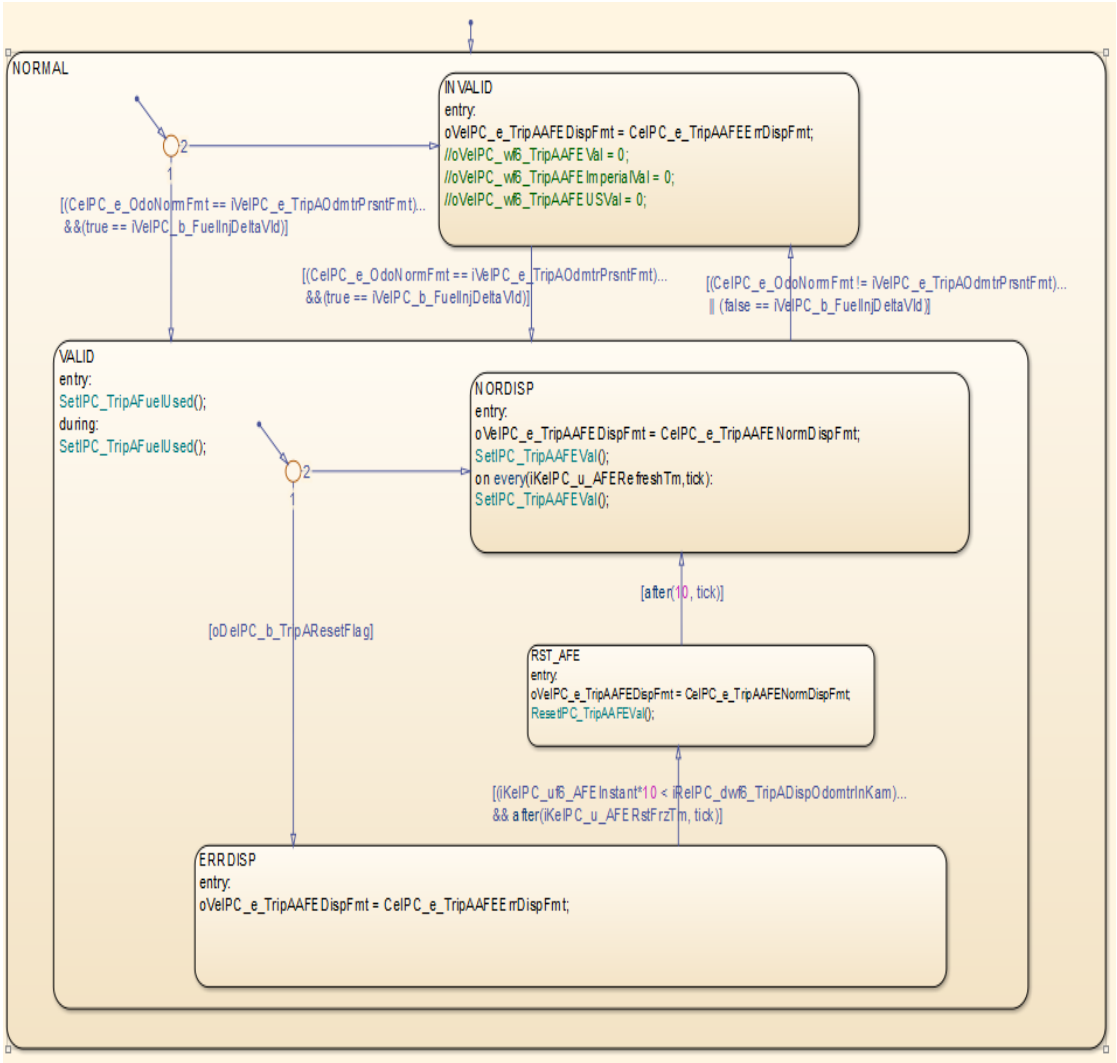


图 4.22 平均油耗模型

在该平均油耗模型的状态迁移图中，可以分为信号无效状态，信号有效状态，正常计算状态，错误显示状态，复位状态等。首先通过判断当前的喷油信号值是否有效，以确定是进入信号有效状态还是信号无效状态，在信号有效状态时，开始累加当前的喷油量和小计里程，如果没有接收到按键清零的事件发生，则正常计算和显示当前一段里程的平均油耗值，如果有按键事件发生则显示异常- -的状态。同时如果信号为无效状态，也会显示- -的状态。

4.6.6 最佳油耗模型

最佳油耗模型统计在不同的距离，比如 50 公里，100 公里，500 公里路程中所计算得到的最佳油耗值，并将该值在液晶屏上显示，最佳油耗在组合仪表上的显示如下图 4.23 所示，

图 4.23 最佳油耗

最佳油耗的控制和计算模型如下图 4.24 所示。

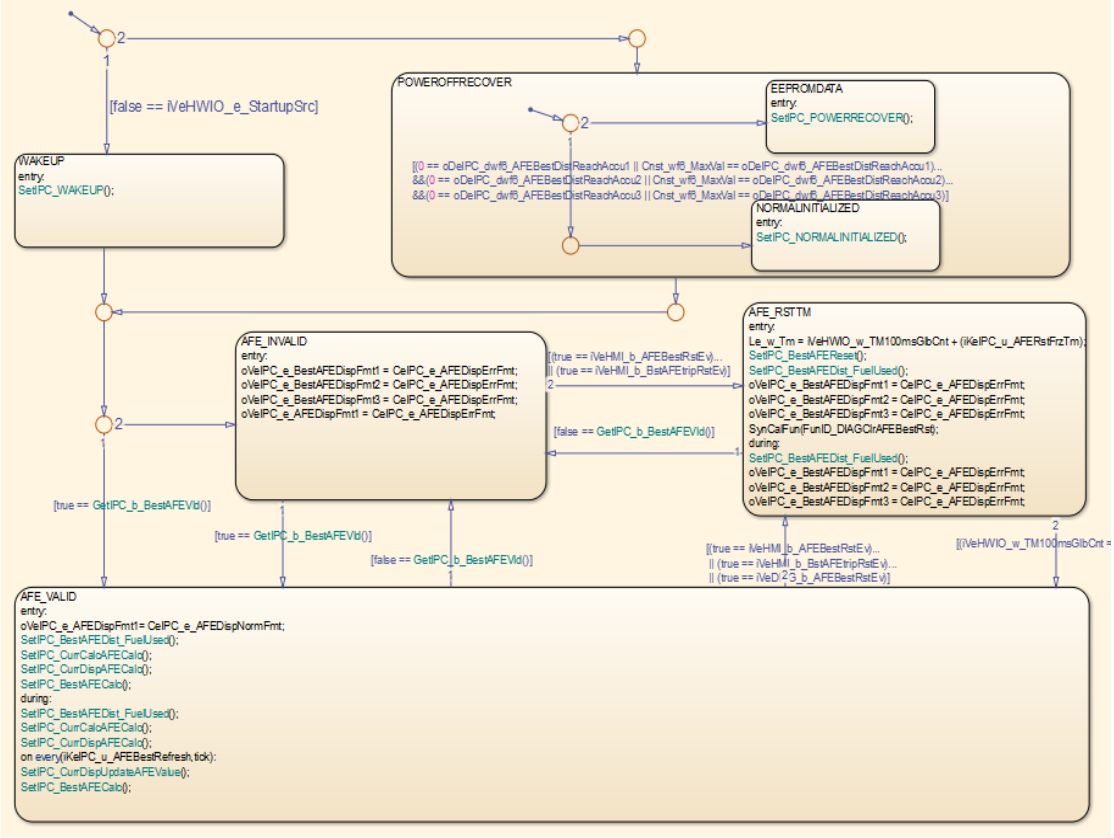


图 4.24 最佳油耗模型

在该最佳油耗模型的状态迁移图中，可以分为最佳油耗无效状态，最佳油耗有效计算状态，最佳油耗恢复状态，最佳油耗复位状态等。通过判断当前最佳油耗计算相关的应用报文是否有效接收到来确定是进入最佳油耗有效计算状态还是最佳油耗无效状态，在无效状态需要显示- -，在有效计算状态，根据当前一段距离内累计算的平均油耗值与历史计算的平均油耗值比较，选取历史计算的一个最低的油耗数值来显示。当在有效计算状态发生复位事件时，需要把当前的最佳油耗值的累积清零，最佳油耗值显示- -。另外最佳油耗值要求在仪表休眠唤醒后可以从睡眠前累计的油耗和距离继续计算，这就需要在最佳油耗复位状态把

这些中间计算的变量值从备份 RAM 中恢复，继续进行计算和显示。

4.6.7 续始里程模型

续始里程模型是通过当前油箱中剩余的燃油量除以平均的油耗值，得出当前车辆还能行驶的距离，值得说明的是该值可能会随着车辆在不同路况下的平均油耗值的不同而浮动，续始里程在组合仪表上的显示如下图 4.25 所示：

图 4.25 续航里程

续航里程的模型如下图 4.26 所示。

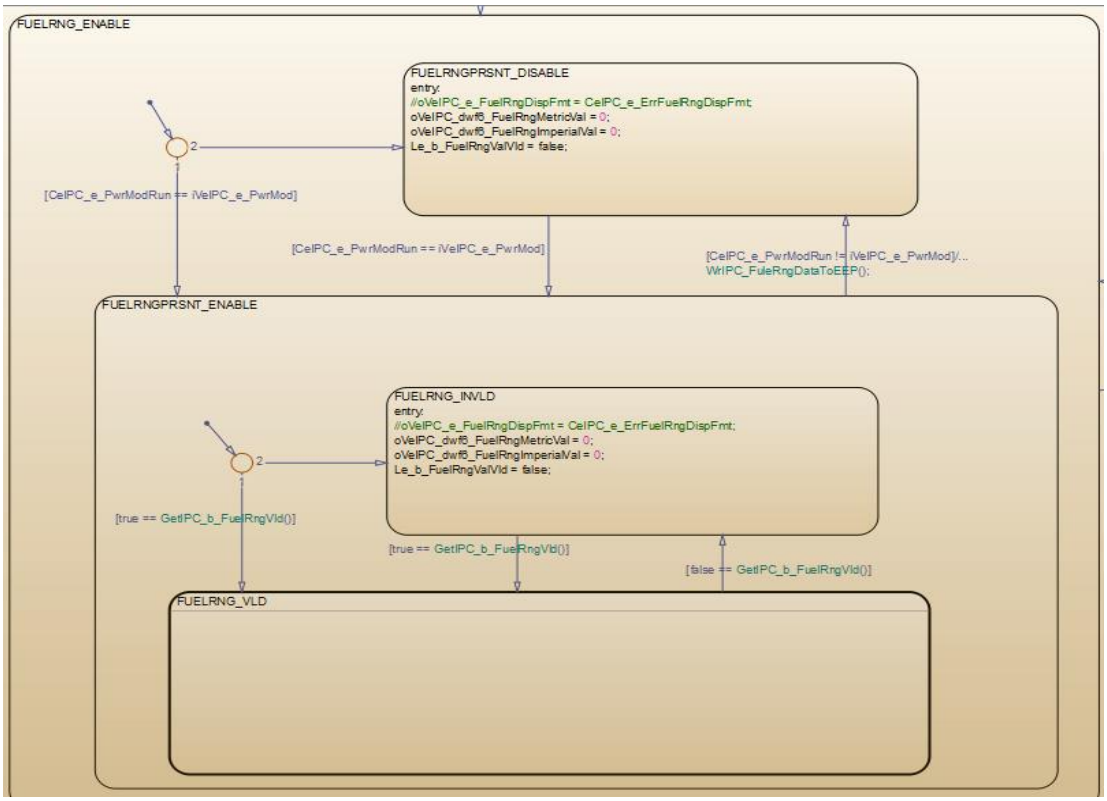


图 4.26 续驶里程模型

在该续驶里程模型的状态迁移图中，可以分为续驶里程使能状态，续驶里程

非使能状态，信号有效计算状态和信号无效错误显示状态。首先通过判断当前的点火钥匙状态是运行状态，来确定是否进入续驶里程使能状态，在续驶里程使能状态会对当前的计算续驶里程的信号的有效性进行判定，如果信号有效，则进入信号有效计算状态，根据输入的油箱剩余油量和平均油耗模块计算的平均油耗值，来估算出当前的可续驶里程。如果发生信号无效的情况，则进入信号无效错误显示状态，在液晶显示屏上显示- -。

4.7 应用层自动代码生成

4.7.1 模型数据管理

Simulink 模型在代码生成过程中，数据的管理是很重要的工作，数据管理主要是对 Simulink 模型中的两类数据进行管理，一类是信号，一类是参数，可以简单的把信号对应到函数上，把参数对应到全局变量上。信号可以通过 GetSet 函数获取，参数则是不通过程序运行而发生变化的，参数的变化，一般是通过人工调节完成的，也就是参数调节，参数调节的目的是为了选择合适的参数以得到最佳的性能。数据管理的方式有多种，在这里使用数据对象进行数据管理，这里的“对象”和“面向对象编程”里面的“对象”具有相同的意义。

在 Simulink 中事先为用户定义的 Simulink Package 包里有两种类，分别为 Simulink.Signal 和 Simulink.Parameter 这两个类。用户可以通过这两个类定义相应的对象，然后通过类提供的属性定义数据的属性。其实这两个类里面除了属性之外，还定义了方法，一般情况下，我们管理数据，使用属性就够了。在很多时候，如果 Simulink Package 不能完全满足用户的所有要求，需要用户定义自己的包，按照面向对象里面的一些概念，我们可以从 Simulink Package 里继承并创建自己的包。在该代码自动生成平台中，我们也是用数据对象的方法定义所有模型的输入输出数据。

4.7.2 模型接口管理表

为了方便的管理众多模型的输入输出参数，我们使用了如下的模型接口管理表格，其中将参数分为输入参数和输出参数，输入参数又分为变量和参数，变量对应外部的接口函数，参数对应全局的标定数据。在每一个变量的属性列中包含名称，数据类型，精度，初始值，范围，存储类型，变量来源，RTE 函数接口，BSP 驱动函数接口等，该模型管理表同时也是 RTE 层的接口配置管理工具。

表 4.7 模型接口管理表

		Name	Event	Data Type	Initial Value(E)	Resolution	Initial Value(N)
Input	Variables	VeCAN_b_SysPwrModPrsnt		boolean			
		VeCAN_b_PNReqMonitor		boolean			
		VeCAN_e_SysPwrMd		TeIPC_e_PwrMod			
		VeHWIO_e_RunCrank		TeHWIO_e_DisSwtSt			
		VeHWIO_w_TM10msGlbCnt		uint16		E=N*1	
	Calibration	Name		Data Type	Nominal Value(E)	Resolution	Nominal Value(N)
		KeIPC_b_Bpmm		boolean	TRUE		TRUE
	Event	Name	Sender				
		Evt_Initialize	-				
Output	Variables	Name	Event	Data Type	Initial Value(E)	Resolution	Initial Value(N)
		VeIPC_e_PwrMod		TeIPC_e_PwrMod	CeIPC_e_PwrModOff		CeIPC_e_PwrModOff
		VeIPC_e_SysBkUpPwrMod		TeIPC_e_PwrMod	CeIPC_e_PwrModOff		CeIPC_e_PwrModOff

对每一个输入和输出变量都创建一个 Simulink.Signal 的对象，并把表格中相应的属性赋给该对象，以 VeCAN_b_SysPwrModPrsnt 该信号为例，创建属于该信号的 Simulink.Signal 的对象，如下图 4.27 所示。

```
VeCAN_b_SysPwrModPrsnt = Simulink.Signal;
VeCAN_b_SysPwrModPrsnt.CoderInfo.StorageClass = 'Custom';
VeCAN_b_SysPwrModPrsnt.CoderInfo.CustomStorageClass = 'GetSet';
VeCAN_b_SysPwrModPrsnt.CoderInfo.CustomAttributes.GetFunction = 'PnRte_Read_CanSysPwrModPrsnt';
VeCAN_b_SysPwrModPrsnt.CoderInfo.CustomAttributes.Setfunction = 'PnRte_Write_CanSysPwrModPrsnt';
```

图 4.27　Simulink.Signal 类型

对每一个输入标定参数都创建一个 Simulink.Parameter 的对象，并把表格中相应的属性赋给该对象，以标定参数 KeIPC_b_Bpmm 为例，创建属于该标定参数的 Simulink.Parameter 的对象，如下图 4.28 所示。

```
KeIPC_b_Bpmm = Simulink.Parameter;
KeIPC_b_Bpmm.Value = 1;
KeIPC_b_Bpmm.CoderInfo.StorageClass = 'ImportedExtern';
KeIPC_b_Bpmm.DataType = 'boolean';
```

图 4.28　Simulink.Parameter 类型

为了批量化的处理这些模型的输入输出参数，使用 python 语言编写了自动化的数据转换脚本，通过运行该自动化脚本，可以从模型数据管理表中读取相应的配置，为每一个输入和输出变量及参数生成 Simulink.Signal 和 Simulink.Parameter 的对象，所有的对象全部生成到一个 Matlab 脚本中，在代码生成的时候会把该脚本在当前的代码生成控件打开，从而能够读取到模型输入输出型号的每一个对外的接口，生成相应的代码。

以下是使用 python 语言编写的数据参数转换工具。

```
#This file using for generate matlab shell file(.m file) to set up interface betwe

import os
import sys
import xlrd
import re

def  main():
    #generate matlab shell file(.m file) to set up interface between matlab model
    model_signal_interface_file = ur'..\workspace\IPC_app_model_signal_interface.m
    model_calibration_interface_file = ur'..\workspace\IPC_app_model_calibration_i
    fw = open(model_signal_interface_file,'w+')
    fp = open(model_calibration_interface_file,'w+')

    #Read matlab model configure information from excel file
    filename = '../docs/IPC_Model_Interface(IPC).xls'
    excel = xlrd.open_workbook(filename)

    #Get all worksheet names
    worksheet_names = excel.sheet_names()
    #print len(worksheet_names)
    #print worksheet_names
    for sheet_name in worksheet_names:
        #Ignore no model data sheet
        pattern = re.compile(r'IPC_')

        if pattern.match(sheet_name):
            #print sheet_name
            sheet = excel.sheet_by_name(sheet_name)
            fw.write('%%%s Component Signal Interfaces between matlab model and RT
            fp.write('%%%s Component Calibration Data Interfaces between matlab mo
            #Get row length of the selected sheet
            num_rows = sheet.nrows
            #print num_rows
            for row in range(0,num_rows):
                    #Get storage class of matlab model varibles
                storage class = sheet.cell(row,17).value
```

图 4.29 数据参数转换工具

4.7.3 Simulink 自动代码生成环境参数配置

在 Simulink 模型构建好了之后，接下来就是代码的自动生成。为了生成代码需要对代码生成参数做配置，首先需要配置的就是系统目标文件，此平台我们选择 ert.tlc，ert.tlc 是实时嵌入式代码生成器的系统目标文件，大量事实表明，使用实时嵌入式代码生成器生成的代码在效率和可读性等方面足以与优秀的手写代码媲美，这也是我们选择该系统目标文件的主要原因。

在 Set Objective 选项卡中选择要生成的目标代码的优化顺序，根据系统的具体需求可以配置为执行效率优先，ROM 空间优先，RAM 效率优先，可追溯性，可调试性。设置不同的优化选项，最终生成的代码的大小和执行效率会有一些差异，在此我们优先选择配置为执行效率优先，代码生成优化配置如下图 4.30。

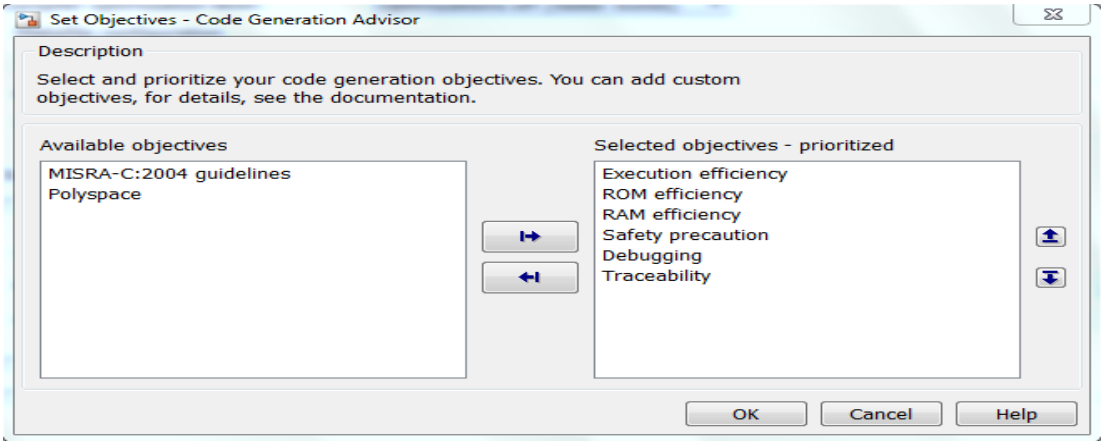


图 4.30 代码生成优化配置

在 Hardware Implementation 选项卡中需要配置当前生成代码运行的硬件，主要是配置处理器的类型，处理器的厂家，及数据类型的长度，多字节大小端存储模式。我们选用的处理器是瑞萨生产的 V850 系列芯片，char 型数据长度为 8 比特，long 型数据长度为 32 比特，多字节数据存储模式为小端，其硬件处理器的配置如下图 4.31。

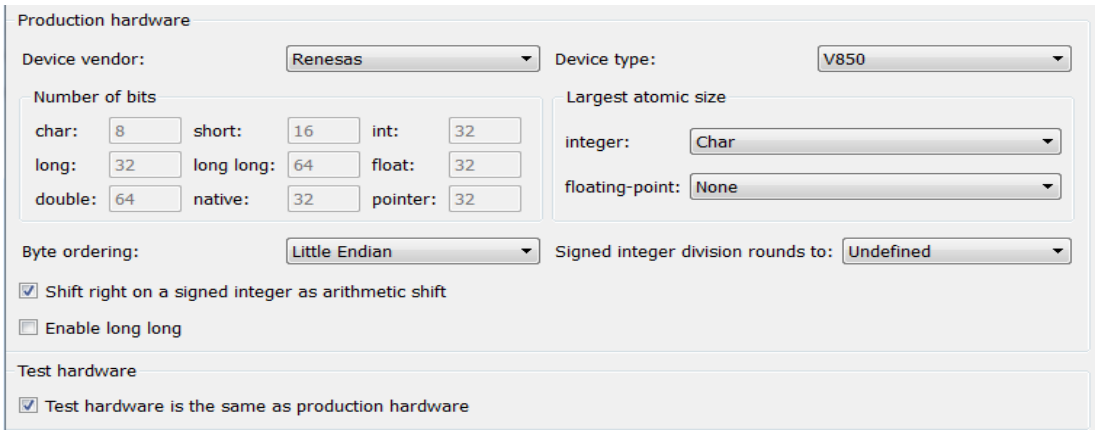


图 4.31 硬件处理器配置

这些配置完成后，直接点击 Code generate 选项卡中的 Build 选项就可以直接生成代码，但是为了方便的管理所有模型的代码生成，我们将这些配置保存为一个 ConfigParam.mat 格式的配置文件，该配置可以在生成代码时，在当前的工作空间打开，并把这些配置信息读取到当前的工作空间，这样就可以方便的通过 Matlab 脚本命令调用这些代码生成的配置，从而能方便的实现批量化的对所有模型的代码生成。

第 5 章 汽车组合仪表应用层模型在环测试开发

仪表在开发过程中，需要设计用例验证设计的正确性，在开发完成后还需要设计功能测试用例，以确认仪表的设计是否符合客户的需求，在验证的时候，我们既用到了白盒测试，也用到了黑盒测试。该平台的应用层主要使用 Simulink 模型开发完成，因此对应用层的模型首先有在环仿真测试的要求，本文设计了自动化在环仿真测试平台。

5.1 模型在环仿真验证系统架构

所谓的在环验证，指的是将控制器模型和被控制对象二者在模型层面，代码层面和处理器层面分别形成闭环，将相同的外部信号激励输入控制器模型，代码及处理器中，比较输出的结果。Matlab 中的在环测试，通常包含以下几种：模型在环测试（MIL）、软件在环测试（SIL）、处理器在环测试（PIL）、硬件在环测试（HIL），下面会分别介绍模型在环测试、软件在环测试、处理器在环测试，暂不包含硬件在环，由于硬件在环主要是针对易损坏的被控对象，本实验平台主要用于测试汽车仪表，一般不会导致被控对象的损坏，没有做 HIL 的必要，因此 HIL 不在本文的讨论之列。

模型在环测试（MIL），软件在环测试（SIL）及处理器在环测试（PIL）的系统架构如下图所示，测试用例统一输入模型及模型生成的代码 S-function 和处理器中，软件在环的结果和处理器在环的结果分别与模型输出的结果比较，如果两者一致，则通过测试。其中模型在环和软件在环是在 Matlab/Simulink 中完成的，处理器在环需要有硬件处理器的支持。

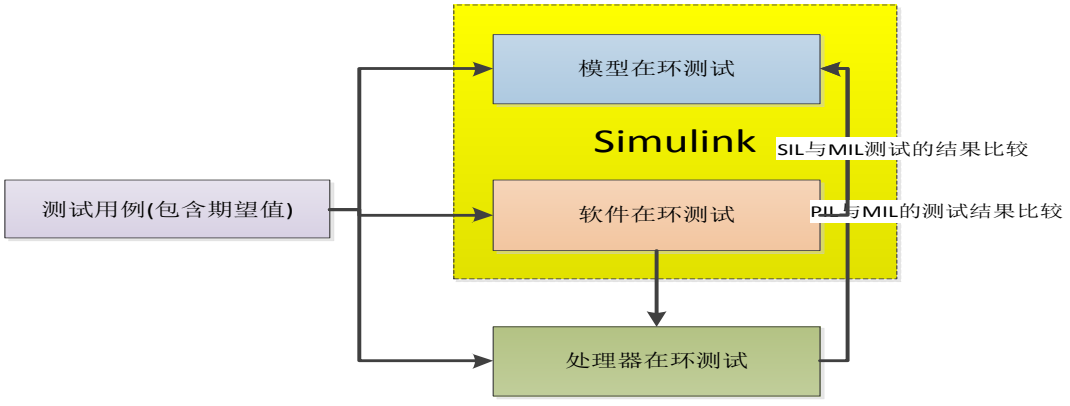


图 5.1 模型在环仿真系统架构

5.2 模型在环验证

模型在环验证，指的是在 Simulink 模型里将控制器模型和被控制对象模型二者连起来形成闭环，就是所谓的 MIL 测试。

5.2.1 模型在环验证的目的

模型在环测试需要有被控对象模型，但在建立模型在环测试的时候，不一定非要建立控制器和被控对象的闭环，在不少的应用中，控制器模型的输出是一些开关量，类似这种输出，我们可以很方便的给出期望的结果，在此种情况下，没有必要将被控对象与控制器连接成闭环，可以直接对输出的结果与期望值比较，从而很方便的实现模型的仿真。本在环仿真平台中的模型在环即是利用模型的输出与期望值进行比较搭建的，没有把被控对象与模型连接形成闭环。

将模型的外部输入信号和调整的参数输入模型，模型输出的结果和期望值做比较，二者一致，则设计的模型符合需求，二者不一致，则可以在模型开发的早期即可以通过仿真发现模型中隐藏的问题。如果设计的用例足够充分，达到足够高的覆盖率，那么经过模型在环测试后的控制算法，在 Simulink 仿真后就可以认为是正确的。

5.2.2 模型在环验证的设计

本论文中以汽车仪表作为模型在环测试的例子，汽车仪表主要是一个被动显示的器件，模型输出的都是一些开关量，比如控制某个指示灯的开关，此时我们没有必要做一个灯泡的模型放在 Simulink 里，我们可以根据需求很方便的求得期望输出值，因此模型在环的平台在搭建的时候，采用的是期望值与模型输出的直接比较，两者比较如果一致，则证明模型设计时合理的，如果不一致，就需要对模型做出修正，模型在环验证的框架结构如下图 5.2 所示。

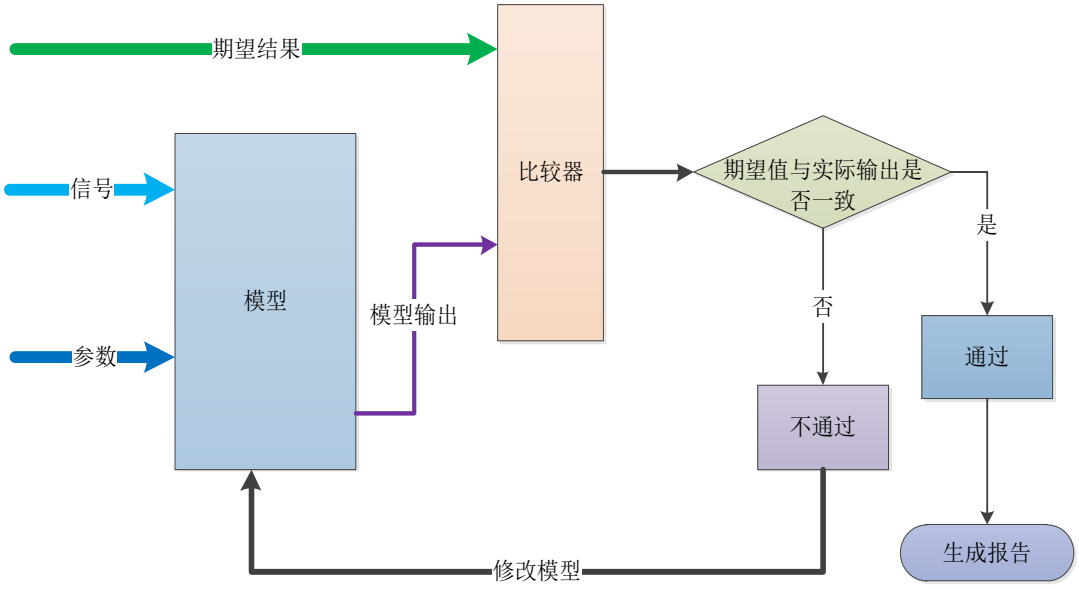


图 5.2 模型在环（MIL）框架结构

5.2.3 模型在环的实现

模型在环测试是完全在 Simulink 中实现的，首先在 Simulink 中打开需要测试的模型，给模型输入一组激励，然后运行该模型，给定的一组用例后，模型会产生一个特定的输出，把该输出与用例设计时的期望值比较，如果两者一致，则通过，否则不通过。在模型在环测试的时候，Simulink 会统计当前的用例对模型各个分支的覆盖情况，因此也会同步产生一个模型的覆盖率报告，该覆盖率报告中默认包含三种类型的覆盖：判定覆盖，条件覆盖，修改的判定和条件覆盖，三种类型的覆盖率会统计到报告中。通过设计大量的测试用例，提升模型测试的覆盖率，使之达到比较高的覆盖率值，完全可以在建模时期就能发现一些算法和逻辑上的错误，从这个角度来看，模型的在环测试等同于手工编写代码的单元测试。

5.3 软件在环验证

软件在环测试就是将控制算法模型生成的代码编译成 S-function，用它来替换原来模型中的控制器部分，再把相同的测试用例分别输入模型和 S-function，如果测试之后，S-function 输出的结果和模型输出的结果完全一致，那么我们可以间接推理得到生成的代码和用于代码生成的模型是一致的。

5.3.1 软件在环验证的目的

算法在经过模型在环验证之后，为了避免代码生成器可能出现的因自身 bug 而导致生成的代码不正确，此时我们使用软件在环验证，以尽可能早的发现代码生成器的问题。

软件在环测试是一种等效性测试，目的是为了验证代码和用于生成代码的模型之间相同输入激励的情况下行为上的一致性，这一个环节存在的必要性是因为我们对代码生成工具的不信任。

5.3.2 软件在环验证的设计

既然 SIL 是为了测试等效性，那么其实可以简化 SIL 测试，即不必通过闭环去做验证，而只要将控制器模型和控制器模型生成的代码编译成的 S-function 放到同一个模型里，给二者加载相同的输入，观测二者输出是否一致即可达到等效性验证的目的。软件在环验证的框架结构如下图 5.3 所示。

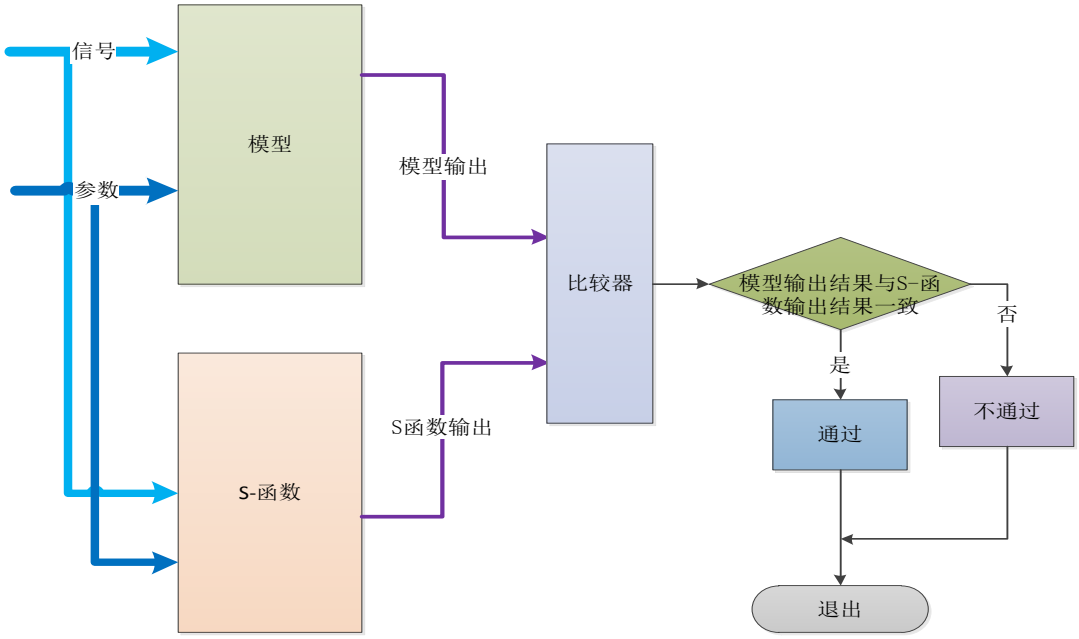


图 5.3 软件在环（SIL）框架结构

5.3.3 软件在环验证的实现

软件在环测试也是在 Simulink 中实现的，首先在 Simulink 中打开需要测试的模型，在如下的模型的代码生成配置中 Verification 项选择创建一个 SIL 测试，给模型输入一组激励，然后运行该模型，模型在运行的时候，会自动为该模型创建一个 S-function 即该模型生成的代码，该组激励也会同步输入该 S-function，需要手动的在 Simulink 中加入一个比较单元，比较 MIL 和 SIL 的运行结果，如果两者是一致的，则通过，否则不通过。

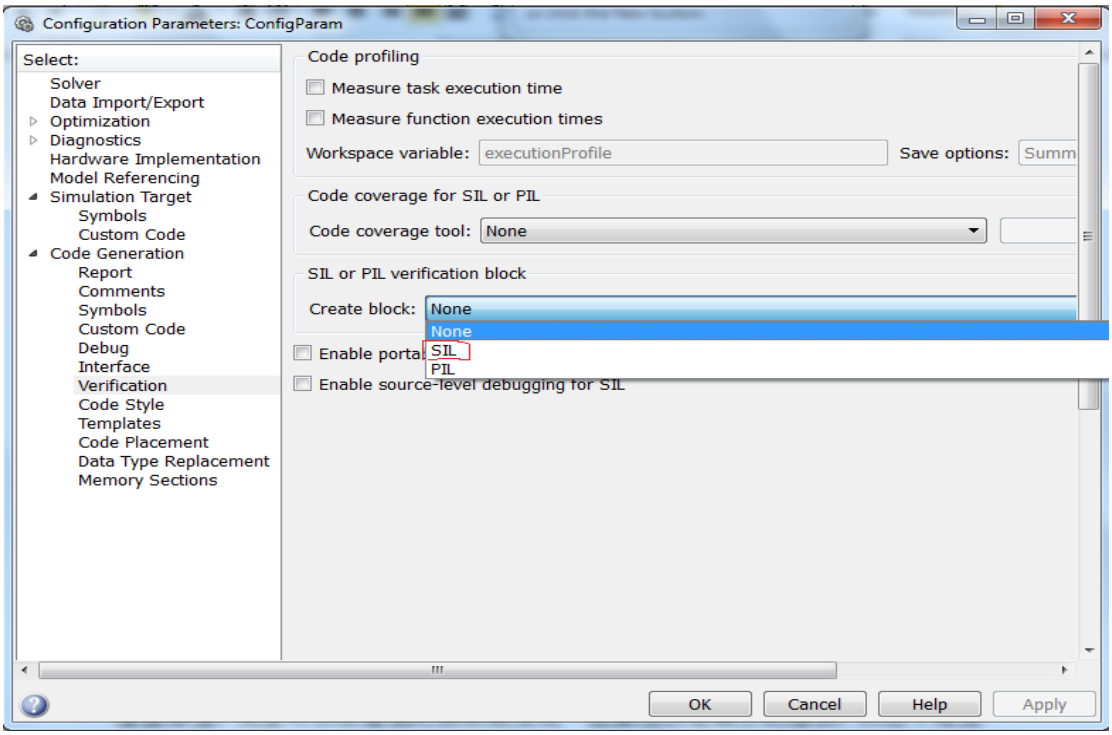


图 5.4 软件在环（SIL）的配置

5.4 处理器在环验证

处理器在环 PIL 指的是使用目标编译器将目标代码编译下载到目标处理器芯片中，将相同的输入激励分别输入模型和目标处理器，比较两者输出的结果。

5.4.1 处理器在环验证的目的

SIL 测试是验证代码和模型的一致性，代码运行在 Windows 平台上，并不能保证代码到目标处理器上的运行结果也能够和模型保持一致，所以我们还需要处理器在环测试，PIL 测试也是等效性测试，PIL 测试的目的有两个，一是可以帮我们及早发现编译器可能引入的错误，二是可以测量生成的算法在目标处理器上的最长运行时间。算法的最长运行时间对做嵌入式实时软件开发来说，比较重要，如果控制器的某些运行时间参数没有那么苛刻，那么处理器在环测试的意义就要大打折扣了。

5.4.2 处理器在环验证的设计

处理器在环测试时，首先要将模型生成的代码通过编译器编译成可执行文件，再通过调试工具或者其他下载手段将可执行文件下载到目标处理器中，通过串口将目标处理器和上位机 PC 相连，目标处理器中需要集成串口驱动程序，给

模型和目标处理器相同的输入激励，目标处理器的输出会通过串口发送给Matlab中的 Simulink 模型，通过比较模型的输出结果和目标处理器的输出结果，来验证模型和目标代码的有效性，处理器在环验证的框架结构如下图 5.5。

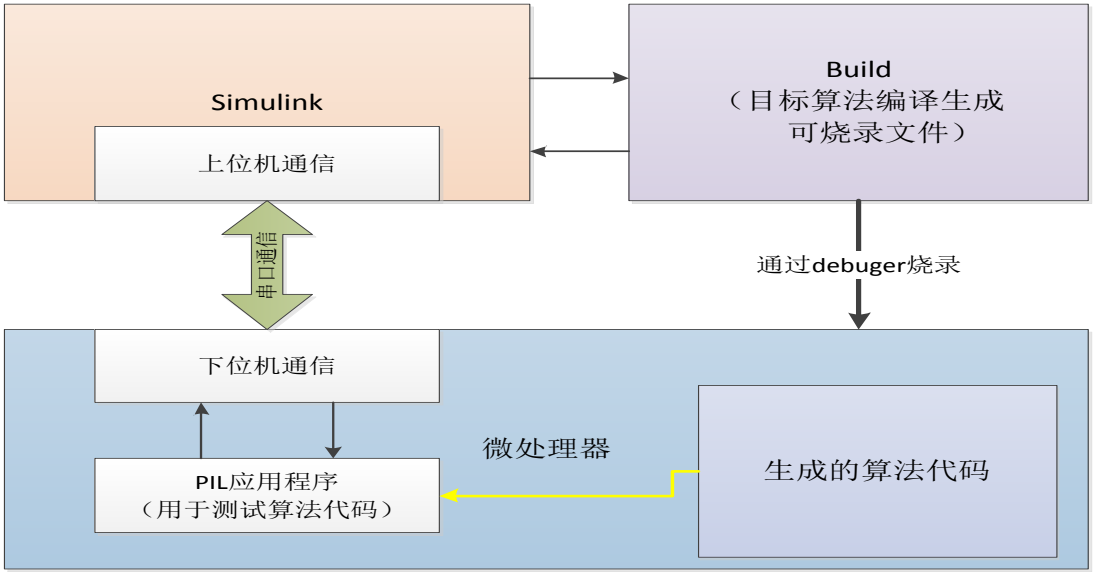


图 5.5 处理器在环（PIL）框架结构

5.4.3 处理器在环验证的实现

处理器在环测试主要是在硬件仿真板子上完成的，硬件的仿真板需要和上位机的 Matlab 通过 PC 机的串口通信，首先在 PC 机上将生成的模型代码使用目标板的编译器编译生成可执行文件，将该可执行文件通过调试工具或者其他下载工具下载到目标板上，目标板上需要有与上位机 PC 进行串口通信的驱动程序，目标板和上位机 PC 通过串口连接，设置统一的通信波特率，在目标板上运行下载的程序算法，同步在 Simulink 中也运行相应的模型，给定模型一组特定的激励，在如下的模型的代码生成配置中 Verification 项选择创建一个 PIL 测试，Simulink 会将该用例通过串口下发到目标板上运行，目标板上运行后的结果会通过串口发送给 Simulink，Simulink 会比较模型输出的结果与目标板上运行的结果，如果两者是一致的，并满足特定的任务运行时间要求，则通过，否则不通过。

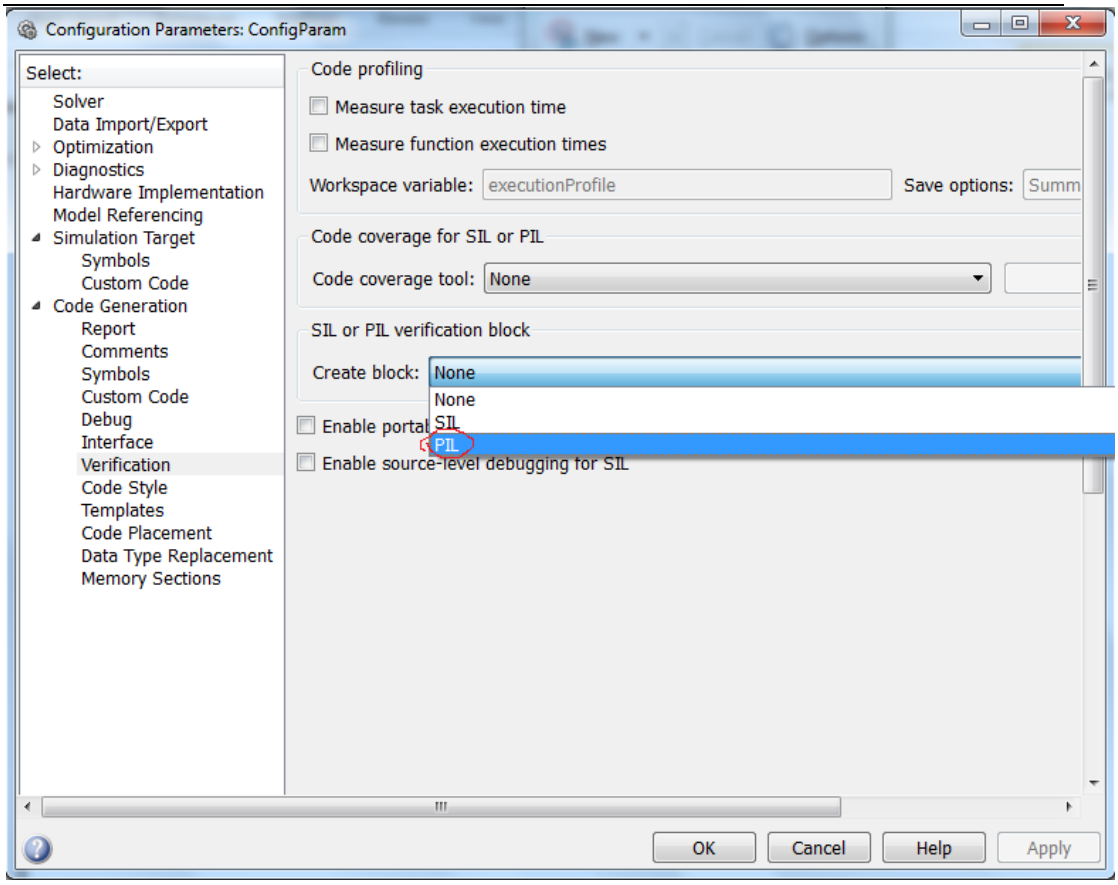


图 5.6 处理器在环（PIL）的配置

5.5 在环测试的工具链

在环测试过程中工作量最大的是测试用例设计以及测试向量的生成，测试用例一般会根据需求去设计测试用例，有时也会结合模型的结构设计测试用例，模型的测试既包含了黑盒测试又包含了白盒测试，如何把测试用例转换为测试向量，也十分关键，在基于模型的开发过程中，代码是可以自动生成的，在仿真测试的环节也同样追求测试的自动化，这其中需要一整套的自动化测试工具链的支持，为此我们使用 Matlab 脚本和 Python 脚本开发了很多测试辅助工具，通过这些测试辅助工具，可以很方便的生成用例模板，将测试用例转换为测试向量并自动的完成整个测试过程，最后自动生成测试报告。

5.5.1 用例模板编写工具

数据的管理是基于模型的开发的重点内容之一，同样如何管理海量的单元测试用例，并在项目的后期可以很方便的维护这些用例，也是需要面对的主要挑战之一。为此使用了如下的表格形式的用例管理工具来维护所有的测试用例，所有的测试用例全部在这个表格中设计完成，在设计用例数据时不直接使用数字而是

使用有一定意义的枚举型的变量来表示，可以很方便的阅读和维护。在表格的纵向的属性列中包含了输入变量，期望值及 MIL，SIL，PIL 测试的结果。

表 5.1 用例数据管理表

				Expected Results			Result			Tester	Notes/Test Data
VeHWIO_b_CheckSumError	KeIPC_wf3_OdoJumpLimt	VeCAN_u_VehicleSpeedInfTick	Step_Out	VeIPC_wf3_OdoDelta	VeIPC_b_OdoDeltaVld	VeIPC_u_OdoDeltaCount	Mil_Result	Sil_Result	Pil_Result		
false(0)	Default(0)	Default(0)	0	Default(0)	false(0)	Default(0)	Pass	Pass	Pass	xma6	2015/12/24
			1				Pass	Pass	Pass	xma6	2015/12/24
			2				Pass	Pass	Pass	xma6	2015/12/24
			3				Pass	Pass	Pass	xma6	2015/12/24
			4				Pass	Pass	Pass	xma6	2015/12/24
		1	5		true(1)		Pass	Pass	Pass	xma6	2015/12/24
			6				Pass	Pass	Pass	xma6	2015/12/24
		2	7			1	Pass	Pass	Pass	xma6	2015/12/24
			8			0	Pass	Pass	Pass	xma6	2015/12/24
			9				Pass	Pass	Pass	xma6	2015/12/24
			10				Pass	Pass	Pass	xma6	2015/12/24
			11				Pass	Pass	Pass	xma6	2015/12/24
			12				Pass	Pass	Pass	xma6	2015/12/24
			13				Pass	Pass	Pass	xma6	2015/12/24
true(1)			14				Pass	Pass	Pass	xma6	2015/12/24
			15				Pass	Pass	Pass	xma6	2015/12/24
			16		false(0)		Pass	Pass	Pass	xma6	2015/12/24
			17				Pass	Pass	Pass	xma6	2015/12/24
			18		true(1)		Pass	Pass	Pass	xma6	2015/12/24
		3	19			1	Pass	Pass	Pass	xma6	2015/12/24
		2	20			255	Pass	Pass	Pass	xma6	2015/12/24
			21		false(0)		Pass	Pass	Pass	xma6	2015/12/24

下面是 ABS 指示灯模型的测试用例编写的例子。

在 ABS 指示灯模型中，根据需求，我们设计了以下几种状态：诊断激活状态，诊断非激活状态，功能使能状态，功能关闭状态，信号有效状态，信号无效状态，常亮状态，熄灭状态，闪烁状态，自检状态。这些状态间通过不同的条件迁移，在设计用例时，每一个状态的迁入和迁出，都是一组单独的用例。模型在仿真的时候需要设定步进的长度，假设设定为 0.001 秒，即每过 1 毫秒采集一次外部的输入激励，则给定一组外部输入激励，使得当前的模型到达常亮状态，模型的期望输出是常亮，把期望的输出值填写到输出一栏，在 Matlab 中运行模型在环测试脚本时，会把模型实际的输出值与期望的输出值做比较，如果一致，则该用例通过，否则就不通过，这样通过设计很多组的输入激励和输出期望值，就可以覆盖到模型中的所有分支和路径，并且 Matlab 可以自动统计分支的覆盖情况，输出完整的覆盖率报告供测试最终结束的参考。

5.5.2 用例模板生成工具

为了能快速的生成测试用例模板，使用 Matlab 脚本语言编写了可以自动生成测试用例模板的自动化脚本程序 init.m。在 Matlab 中运行该 init.m 文件，就可自动生成初始的用例模板，用例模板中的属性列中的输入输出参数可以直接从模型的输入输出接口读取，节省了时间也防止了手动拷贝可能出现的错误，这样模型工程师只需要将精力投入到最核心的测试用例设计就可以了。

init.m 脚本文件如下图 5.7 所示：

```
%Output_array = zeros(1,Outports_num);
Output_name_array = {''};

Input_cnt = 1;
for m = 1:inports_num
    for j = 1:Num_temp
        if ( strcmp(sd(j).Scope,'Input') && (m == sd(j).Port))
            if (sd(j).Props.Array.Size > 1 )
                inports_num = inports_num + str2double(sd(j).Props.Array.Size) - 1;
                %Input_array(m) = sd(j).Port;
                for k = 1:str2double(sd(j).Props.Array.Size)
                    Input_name_array{Input_cnt} = strcat(sd(j).Name(2:length(sd(j).Name)),'[',num2st
                    Input_cnt = Input_cnt + 1;
                end
            else
                Input_name_array{Input_cnt} = sd(j).Name(2:length(sd(j).Name));
                Input_cnt = Input_cnt + 1;
            end
        end
    end
end

Output_cnt = 1;
for m = 1:Outports_num
    for j = 1:Num_temp
        if ( strcmp(sd(j).Scope,'Output') && (m == sd(j).Port))
            if (sd(j).Props.Array.Size > 1 )
```

图 5.7 init.m 脚本文件

5.5.3 测试向量转换工具

该测试用例的编写和维护是在用例管理表格中进行的，但是这些数据不可能直接被模型读取，Matlab/Simulink 模型可以读取数字形式的测试向量，这需要将测试用例管理表格中的数据转换为模型可以识别的测试向量，为了实现数据格式间的快速转换，使用 python 语言编写测试向量自动转换工具，通过运行该自动脚本，可以将测试用例模板中的用例转换为 excel 格式的测试向量。转换后的测试向量如下表格 5.2 所示。

表 5.2 测试向量转换表

VeCAN_w	VeCAN_b	VeCAN_b	VeCAN_w	VeCAN_b	VeCAN_b	VeCAN_b	VeCAN_b	VeCAN_b	VeCAN_b	VeHWIO_	KeIPC_wf	VeCAN_u
0	1	0	0	1	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	1	0	0	0	0	0
0	1	0	0	1	0	1	1	0	0	0	0	0
0	1	0	0	1	0	1	1	1	0	0	0	0
0	1	0	0	1	0	1	1	1	1	0	0	0
64	0	0	0	1	0	1	1	1	0	0	0	1
64	0	0	0	1	0	1	1	1	0	0	0	1
64	1	0	0	0	0	1	1	1	1	0	0	2
64	1	0	0	0	0	1	1	0	1	0	0	2
64	0	0	0	0	0	1	1	1	1	0	0	2
64	0	1	0	0	0	1	1	1	1	0	0	2
64	0	1	0	0	0	1	1	0	1	0	0	2
64	1	1	0	0	0	1	1	1	1	0	0	2
64	1	1	0	0	1	1	1	0	1	0	0	2
64	1	1	0	0	0	1	1	1	1	1	0	2
64	1	1	0	0	1	1	1	1	1	1	0	2
64	1	1	0	0	1	1	1	0	0	1	0	2
64	1	1	0	1	1	1	1	1	1	1	0	2
64	0	1	64	0	1	1	1	0	1	1	0	2
64	0	1	4	0	1	1	1	0	1	1	0	3
64	0	1	4	0	1	1	1	0	1	1	0	2
64	0	1	4	0	1	1	0	0	1	1	0	2

5.5.4 在环自动测试工具

模型用例编写完成后，需要把用例数据导入到模型中，并启动 MIL，SIL，PIL 测试，为了自动方便的完成这些测试准备工作，使用 Matlab 编写自动测试脚本工具 test_script.m，通过运行该脚本，可以自动的完成在环测试的环境配置，测试完成后，可以把 MIL，SIL，PIL 的测试结果自动回写到测试用例模板中。该自动测试脚本如下图 5.8 所示：

```
85 –    my_config_set = attachConfigSetCopy(model_name,config_set,true);
86 –    setActiveConfigSet(model_name, my_config_set.Name);
87 –    Const_char = 'ts';
88 –    Input_array = {''};
89 –    Input_array_sum = '';
90 –  for i = 1:inports_num
91 –        char_temp = num2str(i);
92 –        Input_array{i} = strcat(Const_char,char_temp);
93 –        if (i == inports_num)
94 –            Input_array_sum = strcat(Input_array_sum,Input_array{i});
95 –        else
96 –            Input_array_sum = strcat(Input_array_sum,Input_array{i},',');
97 –        end
98 –    end
99
00 –    set_param(model_name, 'LoadExternalInput', 'on',...
01                       'ExternalInput',Input_array_sum,...
02                       'SolverType','Fixed-step',...
03                       'Solver','FixedStepDiscrete',...
04                       'FixedStep','sample_time',...
05                       'StopTime','stop_time',...
06                       'SaveOutput','on',...
07                       'SaveFormat','StructureWithTime',...
08                       'SimCustomHeaderCode','#include "pn_rte_types.h"',...
09                       'RTWUseSimCustomCode','on',...
10                       'SystemTargetFile','ert.tlc',...
```

图 5.8 自动测试脚本文件

5.5.5 报告自动生成工具

模型的 MIL，SIL，PIL 测试自动完成后，需要将测试结果汇总并生成符合要求的报告，为了直观的展现覆盖率和 MIL，SIL，PIL 的仿真结果，使用图形化的曲线来描述最终的仿真结果，为方便快速的完成报告的生成，使用 Matlab 脚本编写了报告自动生成工具，通过该工具可以自动的生成如下格式的测试报告。

ABS 指示灯模型测试用例覆盖率报告如下图 5.9 所示：

Summary

Model Hierarchy:	Total		
	D1	C1	MCDC
1. IPC_OdoDeltaCalcn	100%	97%	81%
2. . . . IPC_OdoDeltaCalcn	100%	97%	81%
3. . . . . IPC_OdoDeltaCalcn	100%	97%	81%
4. . . . . SF: IPC_OdoDeltaCalcn	100%	97%	81%
5. . . . . SF: GetIPC_b_VehOdoVld	100%	100%	100%
6. . . . . SF: GetIPC_wf3_DistRollCntVal	100%	100%	100%
7. . . . . SF: GetIPC_wf3_OdoDelta	100%	NA	NA

图 5.9 覆盖率报告

ABS 指示灯模型 MIL 仿真结果及 SIL 和 PIL 等效性比较结果报告如下图 5.10 所示：

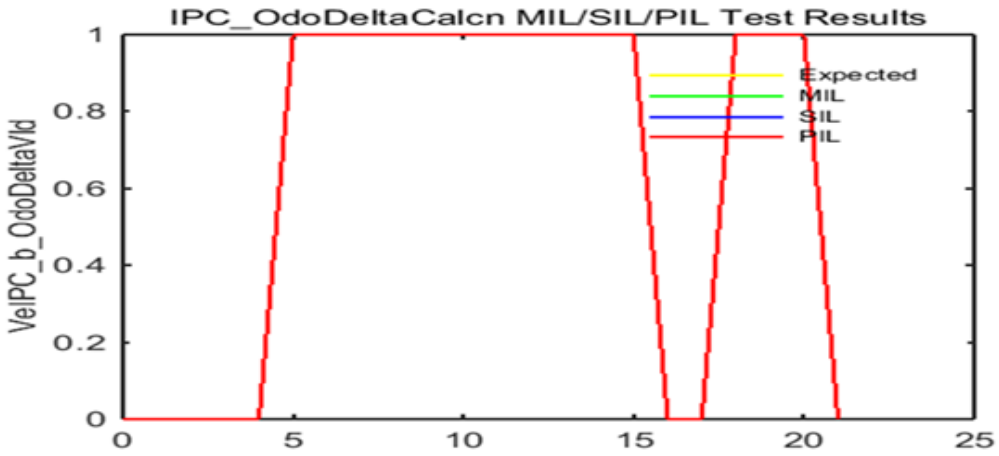


图 5.10 MIL，SIL，PIL 结果曲线

从模型的覆盖率报告上可以看出，判定覆盖（D1）和条件覆盖（C1）都达到了 95%以上，最难达到的修改的判定和条件覆盖（MCDC）也达到了 80%以上，可以得出我们的测试用例设计的十分充分，几乎覆盖了模型的每一个分支。从 MIL，SIL 和 PIL 的曲线图上可以看出，三者的结果完全一致，这表明生成的代码和模型完全一致，从这两个报告可以得出，我们设计的模型在功能逻辑上可以 100%的满足客户的要求。

第 6 章 汽车组合仪表功能测试与验证

仪表的应用层模型经过在环测试验证后，应用层和驱动算法集成后还需要进行完整的功能性测试，功能性测试主要是黑盒测试，是为了确认设计的产品与客户的需求是一致的。因此，功能测试用例是根据客户输入的具体需求编写的，其期望状态必须是客户所真实期望看到的结果。在此我们使用两种类型的黑盒测试方法，一种是自动化的黑盒测试，可以执行一些常规的，重复性的，高强度的极端测试，缺点是有一些复杂情况的测试用例无法执行，无法覆盖到；还有一种是手动测试，通过搭建手动测试台架，可以人工输入测试用例，执行一些自动化测试台架无法测试的复杂测试用例，该方法的缺点是费事费力。在本组合仪表的测试过程中，我们综合运用了这两种测试方法，两者互为补充，互相配合，达到全面深入的测试仪表功能的要求。下面分别介绍下这两种测试方法的实现及组合仪表的测试结果。

6.1 仪表黑盒自动化测试

为了加快测试的进度和提高测试用例的复用率，我们搭建了组合仪表黑盒自动化测试平台。该黑盒自动化测试平台如下图 6.1 所示，由输入部分，输出部分，测试脚本解析器，摄像头，硬件板卡等组成，其中测试脚本解析器是使用图形化的编程语言在 Labview 中实现，摄像头用来捕捉当前仪表上的显示情况，需求设计为表格形式的测试用例输入 Labview 解析器，会逐条解释输入的测试用例，通过摄像头捕捉当前仪表的显示结果与输入的期望值进行比较，如果两只一致则通过，否则该条用例测试不通过，测试结束后，自动输出测试报告。

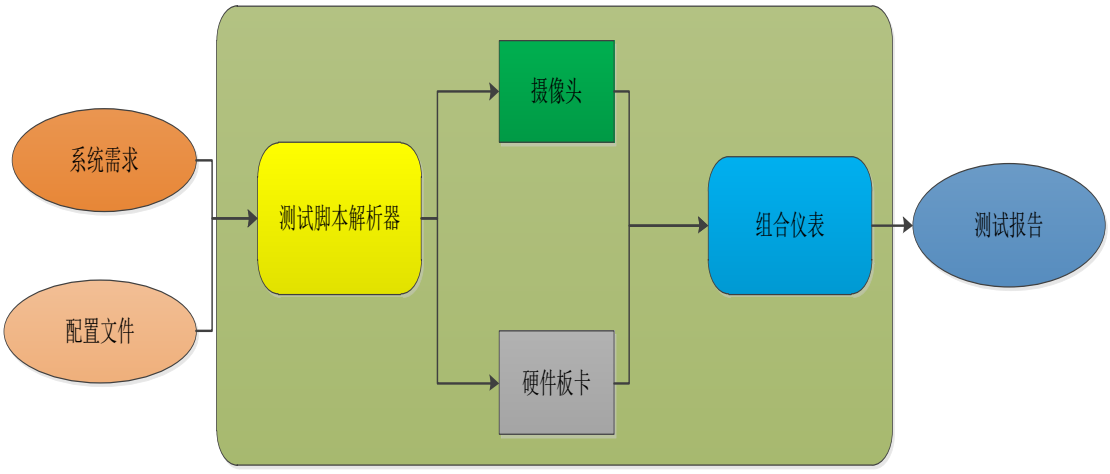


图 6.1 仪表黑盒测试自动化平台

6.2 仪表黑盒手动测试

组合仪表黑盒手动测试，主要是用来弥补自动化测试无法覆盖到的一些比较复杂的测试用例，虽然费事费力，但测试的意义极大，可以发现自动化测试无法发现的一些隐藏比较深层次的问题。测试过程一般包括测试环境（台架）的搭建，功能测试用例的编写，执行功能测试用例，测试结果的汇总整理等。

6.2.1 集成测试环境搭建

在做仪表的集成测试前，先要搭建集成测试的台架，对带 CAN 总线的仪表来说，仪表是总线上的一个节点，需要和车辆上的大部分 ECU 节点通信，通常使用 CAN 通信工具编写一些自动化的测试脚本来模拟实车上的不同 ECU 节点，有时候对比较复杂的 ECU 节点，通过编写 CAN 测试脚本的方式难以模拟出来，就需要将该节点直接连接到测试台架上。台架搭建好之后，仪表通过测试盒与测试台架连接到一起，测试盒可以模拟实车环境提供仪表正常工作需要的电源和通信端口，整个测试台架系统连接好后，给仪表上电，通过 CAN 总线收发工具发送实车相关的激励信号给仪表，按照事先写好的用例和期望值，观察仪表实际的输出反馈情况。集成测试环境如下图 6.2 所示。

图 6.2 仪表集成测试台架

6.2.2 功能测试用例编写

在功能测试前，需要先根据客户输入的需求，编辑功能测试用例，我们使用如下的表格编写和维护相关用例数据，在该表格中需要描述清楚测试的步骤及该条用例的期望值，最终的测试结果等。我们分别对仪表的车速表模块，转速表模块，油量表模块，水温表模块，指示灯模块，液晶显示模块，网络通信模块编写相应的功能测试用例。

Precondition	Test Steps	Expected Result	*He Kejun* *0*
	1. To make IGN to run state SysPwrMd = 2(Run) 2. FOAI_VehAhdIndRq = $1(Alert Level 1) 3. Adaptive Cruise Control VDA = Available 4. P_FCA_TYPE = ACC	1. The Tracking Lead Vehicle Indicator shall be illuminated. (Green)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. FOAI_VehAhdIndRq11E = $1(Alert Level 1) 3. P_FCA_TYPE = camera	1. The Tracking Lead Vehicle Indicator shall be illuminated.(Green)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. FOAI_VehAhdIndRq = $2(Alert Level 2) 3. Adaptive Cruise Control VDA = Available 4. P_FCA_TYPE = ACC	1. The Tracking Lead Vehicle Indicator shall be flashed at Flash Rate #3 (250 msec).(Green)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. FOAI_VehAhdIndRq11E = $2(Alert Level 2) 3. P_FCA_TYPE = camera	1. The Tracking Lead Vehicle Indicator shall be flashed at Flash Rate #3 (250 msec).(Green)	PASS

图 6.3 功能测试用例表

6.2.3 功能测试结果

在仪表测试台架上，按照编写的测试用例，逐条运行测试仪表的各个功能模块，在测试用例表中详细记录实际的测试结果，下面分别对仪表的车速表模块，转速表模块，油量表模块，指示灯模块，网络通信模块做了详细的测试。

Precondition	Test Steps	Expected Result	V2.3.3 *He Kejun* *63* *0* *0* Full Variant1
1. SysPwrMd=OFF	1. Send the message $121 signal SysPwrMd = 2(Run)	1.The Speedometer pointer goes from minimum position to maximum position for 2.5 sec	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. VehSpdAvgDrvn = Present (0x500) 3. VehSpdAvgDrvnV = $0(Valid) 4. Send message $c9	1. The presentation for speedometer guage shall be at 20 Km/h (based on the signal VehSpdAvgDrvn)	p
	1. To make IGN to run state SysPwrMd = 2(Run) 2. VehSpdAvgDrvn = Present (0x500) 3. VehSpdAvgDrvnV = $1(Invalid) 4. *VehSpdAvgNDrvn* = Present (0x500) 5. *VehSpdAvgNDrvn*V = $0(Valid) 6. Send message $c9	1. The presentation for speedometer guage shall be at 20 Km/h (based on the signal VehSpdAvgNDrvn)	p
1.BAT on; 2.IGN on; 3.Voltage=normal 4. SysPwrMd= 1(ACC)	1. To make IGN to run state SysPwrMd = 2(Run)	1.The Speedometer pointer shall remain in their minimum scale position for P_GAGE_HOME_DURATION. 2. The Speedometer pointer shall begin to move to its correct position within 150 msec.	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. *P_SPEEDOMETER_GAGE_ENABLED* = True 3. *Engine Virtual Device Availability* = *Not Present*	1.The IPC shall set speedometer pointer to its minimum scale position 2.The IPC shall set **Vehicle_Speed_Filtered** = 0	

图 6.4 车速表功能测试用例及结果

从该测试结果可以看出设计的测试用例可以覆盖车速表相关的各个功能点，并能全部通过测试，这充分表明了我们设计的组合仪表的正确性和可靠性。

			V2.3.3
Precondition	Test Steps	Expected Result	He Kejun
			58
			0
			0
			Full
			Variant1
1. SysPwrMd=OFF	1. To make IGN to run state SysPwrMd = 2(Run)	1.The Tachometer pointer goes from minimum position to maximum position for 2.5 sec	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. EngSpd = Present (0x640) 3. EngSpdStat = $0(Normal operation) 4. P_TACHOMETER_GAGE_ENABLED = True 5. P_HYBRID_TACH = False	1. The Tachometer gauge shall present the engine speed(400 rpm).	p
	1. To make IGN to run state SysPwrMd = 2(Run) 2. EngSpd = Present (0x960) 3. EngSpdStat = $0(Normal operation) 4. P_TACHOMETER_GAGE_ENABLED = True 5. P_HYBRID_TACH = True 6. EngSpd(0x960) > P_AUTO_STOP_ENGINE_OFF_SPEED	1. The Tachometer gauge shall present the engine speed(600 rpm).	p
	1. To make IGN to run state SysPwrMd = 2(Run) 2. EngSpd = Present (0x640) 3. P_TACHOMETER_GAGE_ENABLED = True 4. P_HYBRID_TACH = True 5. EngSpd(0x640) <= P_AUTO_STOP_ENGINE_OFF_SPEED 6. Engine Run Active = True	Position the tachometer gage pointer at the Auto Stop position (based on calibration P_HYBRID_TACH_AUTO_STOP)	p

图 6.5 转速表功能测试用例及结果

从该测试结果可以看出设计的测试用例可以覆盖转速表相关的各个功能点，并能全部通过测试，这充分表明了我们设计的组合仪表的正确性和可靠性。

Precondition	Test Steps	Expected Result	He Kejun 50 0 0 Full Variant1
1. SysPwrMd=OFF	1. To make IGN to run state SysPwrMd = 2(Run)	1.The Fuel guage pointer goes from minimum position to maximum position for 2.5 sec	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. FILvlPct = present(0x80) 3. FILvlPctV = $0(Valid) 4. P_FUEL_GAGE_ENABLED = True	1. The Fuel guage pointer shall present the fuel level as 50%.	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True 3. FILvlPct = not present	The IPC 1. shall set Fuel Level pointer to its minimum position	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True 3. FILvlPctV = $1(Invalid)	The IPC 1. shall set Fuel Level pointer to its minimum position	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True 3. FuelMdStat = $0(Gasoline Mode) 4. FILvlPct = not present	The IPC 1. shall set Fuel Level pointer to its minimum position	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True 3. FuelMdStat = $0(Gasoline Mode) 4. FILvlPctV = $1(Invalid)	The IPC 1. shall set Fuel Level pointer to its minimum position	p
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True 3. FuelMdStat = $1 (Natural Gas Mode) 4. FILvlTank2Pct = not present	The IPC 1. shall set Fuel Level pointer to its minimum position	p
1.BAT on; 2.IGN on;	1. To make IGN to run state SysPwrMd = 2(Run) 2. P_FUEL_GAGE_ENABLED = True	The IPC 1. shall set Fuel Level pointer to its minimum position	

图 6.6 油量表功能测试用例及结果

从该测试结果可以看出设计的测试用例可以覆盖油量表相关的各个功能点，并能全部通过测试，这充分表明了我们设计的组合仪表的正确性和可靠性。

Precondition	Test Steps	Expected Result	He Kejun 0
1.BAT on; 2.IGN on; 3.Voltage=normal 4.SysPwrMd=OFF	1. To make IGN to run state SysPwrMd = 2(Run)	1. The Airbag indicator shall be illuminated continuously for ??? Sec.	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. To turn on the Airbag telltale set to AirbgICIO= 1 3. Set the AirbgICDutCyc= 50% 4. Set the AirbgICIndPer= 0.5 Hz	1. The Airbag indicator shall be flashed at Flash Rate #1 (1sec)	PASS
1.BAT on; 2.IGN on; 3.Voltage=normal	1. To make IGN to run state SysPwrMd = 2(Run) 2. To turn on the Airbag telltale set to AirbgICIO= 1 3. Set the AirbgICDutCyc= 50% 4. Set the AirbgICIndPer= 0.5 Hz	1. The Airbag indicator shall be flashed at Flash Rate #1 (1sec)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. To turn on the Airbag telltale set to AirbgICIO= 1 3. Set the AirbgICDutCyc= 50% 4. Set the AirbgICIndPer= 1 Hz	1. The Airbag indicator shall be flashed at Flash Rate #2 (500msec)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. To turn on the Airbag telltale set to AirbgICIO= 1 3. Set the AirbgICDutCyc= 50% 4. Set the AirbgICIndPer= 2 Hz	1. The Airbag indicator shall be flashed at Flash Rate #3 (250msec)	PASS
	1. To make IGN to run state SysPwrMd = 2(Run) 2. To turn on the Airbag telltale set to AirbgICIO= **0** 3. Set the AirbgICDutCyc= 50% 4. Set the AirbgICIndPer= 2 Hz	1. The Airbag indicator shall be OFF	PASS

图 6.7 仪表指示灯模块功能测试用例及结果

从该测试结果可以看出设计的测试用例可以覆盖指示灯相关的各个功能点，并能全部通过测试，这充分表明了我们设计的组合仪表的正确性和可靠性。

ID	Norminal Period of the Msg (ms)		MIN (ms)		MAX (ms)	PASS or FAIL
$0C9	15		6.313	≤ Tperiodic ≤	23.810	OK
108	100		91.567	≤ Tperiodic ≤	105.455	OK
121	100		92.576	≤ Tperiodic ≤	105.565	OK
124	500		490.776	≤ Tperiodic ≤	503.972	OK
128	10		4.337	≤ Tperiodic ≤	14.864	OK
160	100		91.054	≤ Tperiodic ≤	105.310	OK
$17D	100		91.349	≤ Tperiodic ≤	107.031	OK
182	100		89.798	≤ Tperiodic ≤	108.970	OK
185	20		16.798	≤ Tperiodic ≤	23.970	OK

图 6.8 网络通信模块功能测试用例及结果

从该测试结果可以看出设计的测试用例可以覆盖网络相关的各个功能点，并能全部通过测试，这充分表明了我们设计的组合仪表的正确性和可靠性。

6.3 组合仪表测试结果分析

通过对组合仪表的应用层模型执行在环测试，仪表应用层的 100 多个模型全部通过在环测试，并且判定覆盖和条件覆盖的覆盖率可以达到 95%以上，修改的判定覆盖和条件覆盖率可以达到 90%以上，有效保证了应用层的模型的逻辑的正确性。

通过对组合仪表集成的软件执行自动化的黑盒测试和手动的黑盒测试用例，编写了几万条的黑盒测试用例，仪表全部能运行通过，全面的验证了该组合仪表的设计的正确性和可靠性。

第 7 章 结论与展望

7.1 结论

本课题设计实现了一个基于 AUTOSAR 架构和 Simulink 模型的汽车组合仪表通用开发平台，主要的工作内容如下：

1. 概述了汽车组合仪表需要满足的主要功能，对汽车仪表在网络中的拓扑结构给出详细的描述，对 AUTOSAR 架构中涉及到的关键的技术，比如 AUTOSAR 的操作系统，AUTOSAR 的网络管理，AUTOSAR 的诊断及 AUTOSAR 虚拟功能总线等，对其原理及实现机制做了详细的分析和总结。

2. 针对组合仪表需要实现的功能要求，设计了汽车组合仪表的硬件电路，对汽车组合仪表的主要关键电路，例如电源模块，数据采集模块，背光模块，通信模块，显示模块，步进电机模块等做了详细的设计描述。

3. 重点在基于 AUTOSAR 的软件架构基础上，设计了该组合仪表的软件架构，对 AUTOSAR 操作系统的配置和应用层任务的设计，AUTOSAR 网络配置和应用层的设计，AUTOSAR 诊断开发等做了详细的描述，设计实现了 AUTOSAR 中的虚拟功能总线相关的运行时环境工具，可以配置生成运行时环境层相关的代码。

4. 设计了应用层的模型开发和自动代码生成平台，在该平台上详细设计了汽车仪表主要功能相关的车速表模型，转速表模型，油量表模型及平均油耗值，瞬时油耗值，续驶里程值的计算过程。

5. 设计了模型的自动在环测试平台，开发了相关的自动化工具链，通过该在环设计平台，可以自动的对设计的应用层模型执行模型在环测试，软件在环测试和处理器在环测试。

6. 通过自动化的黑盒测试和手动功能测试，全面测试了组合仪表集成后的功能和性能，从而确保了设计符合系统的功能要求。

最后希望本文对国内汽车电子领域基于 AUTOSAR 架构的模型开发及仿真验证有积极的借鉴和帮助。

7.2 展望

目前在汽车电子领域采用 AUTOSAR 的架构成为一种趋势，采用 AUTOSAR 架构的汽车仪表也是仪表未来的发展趋势。在基于 AUTOSAR 的架构中，一般重点探讨的都是 BSW 等基础组件的设计，对应用层采用的框架涉及的很少，采用基于模型

的设计方法，可以方便的以图形化的方式在 Simulink 中展现应用层的框架，并使用成熟的嵌入式代码生成器自动生成高质量的软件代码，生成的软件不但有很好的一致性，而且软件硬件整合简单可靠，可大大降低解决问题的成本，这代表了汽车电子领域嵌入式产品软件开发的未来发展方向。

目前已进行的研究工作还仅仅停留在很初级的阶段，还有大量的研究工作可进一步向前推进，尤其在模型的优化方面，目前仪表上设计的这些模型仅仅是满足了仪表基本的功能需求，在性能方面是不是最优的，是否还有更高效的节约存储空间的设计，还需要进一步研究。

致谢

在职硕士研究生的求学过程是我人生的一段宝贵经历，因为有了这段经历，我的人生也有了不一样的精彩，在工作中学习，在学习中工作，理论结合实践经验，受益匪浅。

本课题于 2014 年底开始酝酿、构思，历时将近三年时间，在导师罗峰教授指导下逐渐明晰主题和框架；初稿形成后，导师又数次提出修改意见直至论文定稿。在此，特别感谢老师百忙之中的指导与关怀，使我对学术、工作和生活都有了更高的认识。

感谢延锋伟世通公司给予了我成长的平台，使我可以在工作过程中兼顾到学习，感谢在整个学习期间，公司的领导和同事给予的无私帮助和支持，在汽车仪表设计领域与诸多公司的专家交流颇多，了解了很多国内外的最新趋势和动态，在此致以诚挚谢意。

汽车工程是一个广而大的学科，既需要我们有广博的知识面，又需要我们能够在一个领域中钻研精通，感谢同济大学给了我这个优秀的学习平台，使我能够认清自己的不足，补齐短板，同时坚定了为中国汽车产业做大做强贡献自己的一份力量的决心。

贺可军

2017 年 3 月

参考文献

[1] 程安宇,王刚,张居林等.基于 AUTOSAR 架构的汽车仪表通信模块的设计.自动化与仪器，2013，Vol.6.

[2] 孙颖，王建俊，张承瑞.基于 AUTOSAR 的汽车电控系统代码自动生成技术.重庆理工大学学报，2014,Vol.28(3):34～36.

[3] 张允，钟再敏.基于 AUTOSAR 和多核处理器的机电复合传动控制算法实现.同济大学学报，2016.

[4] Yingping Huang, Alexandros Mouzakitis.Design Validation Testing of Vehicle Instrument Cluster Using Machine Vision and Hardware-in-the-loop.IEEE，2008.

[5] 冯江波,刘亚军.与 AUTOSAR 兼容的 Matlab_Simulink 自动代码生成技术[M]. 佳木斯大学学报，2011，Vol.29（6）:834～837.

[6] 何彬，华剑锋，章健勇等. 燃料电池汽车整车控制器仿真测试平台研究. 系统仿真学报，2005，Vol.17(7)：1694—1698.

[7] 徐超坤，朱婷，李威宣.基于模型的嵌入式 C 代码的实现与验证[M].单片机与嵌入式应用，2011，Vol.12（6）:15～17.

[8] Dionisio de Niz1, Gaurav Bhatia, and Raj Rajkumar. Model-Based Development of Embedded Systems: The SysWeaver Approach. Real-Time and Multimedia Systems Lab, Carnegie Mellon University,2006.

[9] 曹更彦，李银国,基于 MATLAB_Simulink 和 LabVIEW 的发动机仿真.重庆：重庆邮电大学，电子测试，2008,Vol.8:57～58.

[10] 杜剑维，王银燕，基于 dSPACE 的相继增压柴油机硬件在环仿真[M].哈尔滨：哈尔滨工程大学，2007.

[11] 朱虎，林立，刘正奇.基于 Matlab 的直流电机控制系统硬件在环设计.湖南：邵阳学院，2015.

[12] LAPUSAN CIPRIAN, MATIES VISTRIAN, BALAN RADU, HANCU OLIMPIU. MODELING AND SIMULATION METHODS FOR DESIGNING MECHATRONIC SYSTEMS. Journal of Engineering Studies and Research.

[13] 李彤，屈金标，岳杰等.车载 CAN 总线在环测试技术研究.通信技术，2012，Vol.1（45）:84～86.

[14] Laszlo Kis，Gergely Regula，Bela Lantos. Design and Hardware-in-the-Loop Test of the Embedded Control System of an Indoor Quadrotor Helicopter.IEEE，2007.

[15] W. Chaaban, M. Schwarz, B. Batchuluun, H. Sheng. A Partially Automated HiL Test Environment for Model-Based Development Using Simulink and OPC Technology. Department of Computer Architecture and System Programming University of Kassel.

[16] Hyun Chul Jo, Shiquan Piao, Sung Rae Cho and Woo Young Jung. RTE Template Structure for AUTOSAR based Embedded Software Platform. Department of Computer Architecture and System Programming University of Kassel. IEEE.

[17] http://www.autosar.org, AUTOSAR标准文档.

[18] 冯辉宗，刘先东，蒋建春等.基于 AutoSAR 规范的驱动代码生成工具箱设计与实现.电

子技术应用，2013 Vol.38（7）:33～35.
[19] 郭徐，陈华钧，王卫红等.汽车电子软件 AUTOSAR Service 的配置与实现.浙江：浙江大学，2014.
[20] 阴晓峰，刘武东.汽车电子系统软件开发新标准 AUTOSAR.西华大学学报，2010.
[21] 魏学哲，戴海峰，孙泽昌.汽车嵌入式系统开发方法、体系架构和流程.同济大学学报（自然科学版），2012,Vol.40（7）:1065～1067.
[22] 王林，佘庆，张激.AUTOSAR OS 调度表同步优化设计.计算机工程，2011, Vol.37（9）:36～37.
[23] 邓俊，李红，方正等，.AUTOSAR OS 存储保护方案的改进与实现.仪器仪表学报，2011，Vol.32（9）：2147～2149.
[24] 冯川，胡杰，颜伏伍等.符合 AUTOSAR 标准的 CAN 底层通信研究.武汉理工大学学报（信息与管理工程版），2013, Vol.35（6）：843～844.
[25] 陈海兰，罗晓敏，涂时亮.基于 AUTOSAR 的实时操作系统设计与实现.计算机工程，2012, Vol.38（20）：10～16.
[26] 王萍，赵玲，郑安豫.基于 Labview 的汽车仪表检测系统的设计.佳木斯大学学报（自然科学版），2015，Vol.33（2）：269～270.
[27] 蓝天，廖承林.基于虚拟仪器技术的步进电机式汽车仪表的设计.仪表技术与传感器网，2008，Vol.8：18～21.
[27] 岳晓峰，张娇.基于 SIFT 算法和改进最小二乘法的汽车仪表指针的识别.制造业信息化，2014，Vol.12：161～162.
[28] 谢超.敏捷开发方法在汽车仪表软件研发中的应用.研究与开发，2014，Vol.3：52～55.
[29] 陈鹏，汪至中.重型卡车嵌入式数字仪表的研究和开发.北京交通大学，2006.
[30] Hung-Yih Tsai, Jeng-Shyong Chen, Yuan-Yong Hsu,Development of a modular master－slave instrument cluster.IEEE，2011.
[31] 陈新，张桂香，肖奇云.电动汽车液晶数字仪表的设计.汽车工程，2013，Vol.35（3）：273～276.

个人简历、在读期间发表的学术论文与研究成果

个人简历：

贺可军，男，1985 年 7 月生。

2008 年 7 月毕业于中南民族大学计算机科学与技术专业获学士学位。

2008 年 8 月至 2011 年 3 月任职于武汉光庭汽车电子有限公司。

2011 年 4 月至今任职于延锋伟世通电子科技（上海）有限公司。

2012 年 3 月入同济大学在职攻读车辆工程硕士研究生。

已发表论文：

暂无