浅论臂车控制器的软硬件开发模式

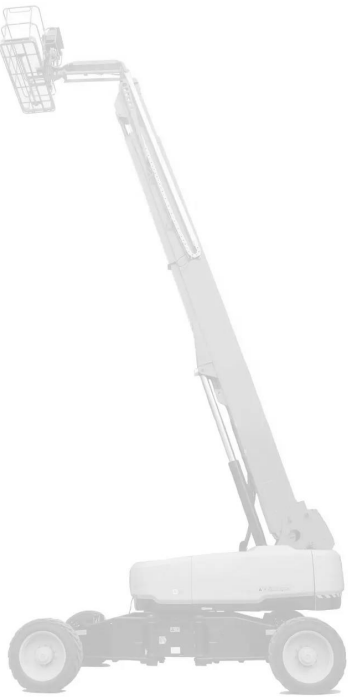
通常先有需求，再有开发模式。基于对臂车的认知与理解，我们选择适配的控制器开发模式来满足臂车的应用场景的主机厂需求。每个开发模式都有优劣，关键在于合理取舍和利用现有的技术资源。

首先，一台臂车的整机性能取决于机械、液压、控制器（硬件+软件）三方合作的结果，机电液是一个整体，在整体的前提下，控制器的功能需要最大限度地发挥机械和液压的特性并嵌入整机。由此，我们把臂车控制器分成三个层面的需求：

1）动作之间的控制逻辑；

2）执行机构的控制策略（算法）；

3）整机的性能提升（如安全空间的持续优化）。



市场上主流的软件开发模式是采用通用的控制器硬件+带操作系统的软件开发平台（如CODESYS和PLUS1）。这种开发模式有其历史的优势，在工程机械界沿用十几年，有广泛的客户基础，可以利用现有的成熟硬件并采用PLC的编程方式，现场对控制逻辑层面的需求实现非常方便，所见即所得，让工程师能专注于臂车应用的研究。同时基于通用控制器的已有的很多控制策略的模块可以方便地搭建框架，从而实现快速编程和调试。缺点是现有的控制策略的模块（第二层面）很有限，通用的控制模块不足以覆盖臂车的专项需求，所以需要进行控制策略的额外编程，这个对于工程师就提出了很高的要求，如和发动机的功率匹配就要求工程师对高等数学有很好的基础来实现编程（但如果供应商能根据用户需求和具体的液压件开发相应的专用模块，则另当别论）。不过整机性能的层面是通过现场的不断尝试来匹配完成，比较花时间，这是最大的不足。另外不得不提的是这种通用控制器的成本偏高，因为有太多的输入和输出的端口的资源闲置，还有就是增加了操作系统，对硬件的资源要求也偏高从而增加了成本。同时控制系统的增加也可能会增加系统的BUG点，所谓越简单的系统越可靠。但这种模式比较适合工程机械小批量多品种的特点。

市场上另一主流是采用通用（或专用）控制器硬件+C语言的开发平台。这种开发方式需要专业的懂C语言开

发的软件工程人员，同时还要懂应用，这就对编程的工程师提出了双重的要求。好处是这种编程模式将C语言的代码直接通过编译写入控制器硬件，少了一层操作系统的层面，相对上一种模式会直截了当，从而相对地减少BUG点，不过软件需要DEBUG的过程，对软件人员却是一个考验。优点是C代码的编写对于第二层面的控制策略的编写非常高效。另外这种控制器对硬件资源要求会略低，成本相应较低，因为不需要操作系统的加载。

还有一种方式是最近出现的，从汽车行业借鉴而来，沿用Bosch开发汽车整车控制器的流程。这种开发模式最大的优势是利用Matlab/Simulink来进行建模和仿真，在电脑中搭建臂车模型，然后模拟测试控制逻辑和控制策略，一旦测试完成，再自动生成C代码写入硬件，大大缩短现场调试的时间（相对于前两种的开发平台而言），现场只需要验证即可。另一优势是后期如果能和生产臂车的主机厂联合建立臂车的虚拟样机模型，在Matlab/Simulink里就直接和模型对接仿真，这样就能对臂车的整机性能的提升（第三层面）有真正的帮助，从而能改善整体上的匹配问题进而提升机电液的整机性能。另外需要提及的是Matlab里面有大量的数学工具，在臂车的控制策略即第二层面的编写上能直接调用数学工具，对开发控制策略（第二层面）有明显的开发优势。不过这种Matlab/Simulink的开发模式对于软件编程提出较高的要求，虽然Simulink也是可视化的编程模式。还有Simulink的编程模式不需要程序Debug，比起第二种方式相应减少了工作量。不过这种开发模式需要相应的工具链的配备并熟悉之，对使用者需要一个相对长的熟悉的过程。另外这种方式的开发通常会自行设计硬件来配合软件工程，这样会大大降低硬件成本，对于批量较大的臂车有明显的价格优势。

各花入各眼，如今的臂车控制器的开发平台选择多样，主机厂可以选择合适的资源进行开发并赢得市场上的竞争力。