# A NEW APPROACH TO LABORATORY MOTOR CONTROL
# MMCS
# THE MODULAR MOTOR CONTROL SYSTEM

*Peter I. Corke*

**MS-CIS-89-17**
**GRASP LAB 175**

**Department of Computer and Information Science**
**School of Engineering and Applied Science**
**University of Pennsylvania**
**Philadelphia, PA 19104**

**February 1989**

---

**Acknowledgements:** This work was supported in part by NSF IRI84-10413-AO2, MCS-8219196-CER, CCR8716975, NSF/DCR grants 8501482, NSF/DMC 8512838, U.S. Army grants DAA29-84-K-0061, DAA29-84-9-0027.

# A New Approach to Laboratory Motor Control

# MMCS

# The Modular Motor Control System

Peter I. Corke[1]
pic@grasp.cis.upenn.edu

Computer and Information Science Department
University of Pennsylvania
Philadelphia, PA 19104

February 23, 1989

[1]Research Scientist, CSIRO Division of Manufacturing Technology, Melbourne, Australia.

**Abstract**

Many projects within the GRASP laboratory involve motion control via electric servo motors, for example robots, hands, camera mounts and tables. To date each project has been based on a unique hardware/software approach.

This document discusses the development of a new modular, and host independent, motor control system, MMCS, for laboratory use. The background to the project and the development of the concept is traced.

An important hardware component devloped is a 2 axis control motor control board that can be plugged into an IBM PC bus or connected via an adaptor to a high performance workstation computer.

To eliminate the need for detailed understanding of the hardware components, an abstract controller model is proposed. Software implementing this model has been developed in a device driver for the Unix operating system. However for those who need or wish to program at the hardware level, the manual describes in detail the various custom hardware components of the system.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Overview

This first chapter discusses the motivation for a new modular, and host independent, motor control system for laboratory use. The background to the project and the development of the concept is traced. A number of possible solutions are proposed and discussed, leading to a general description of the system that has been implemented.

Chapter 2 describes in detail an abstract programmer's model of the axis controller. Details of the servo interface hardware are hidden, allowing the applications programmer to concentrate on higher level control. The software implements position, velocity or torque control, selectable per axis, and the closed loop dynamics may be modified by a digital compensation network.

Chapter 3 describes an interactive graphical tool that allows a user to configure the axis controller, perform diagnostics and perform joint level motions.

The last two chapters are not essential reading for casual programmers, but are essential for those programming at the hardware level.

Chapter 4 describes the function performed by the host bus adaptor and also the axis controller bus, which is the same as IBM/PC bus. Details such as redefinition of some signal lines[1], and the addressing conventions used are covered. It describes in detail the hardware implementation of the VME host to PC bus adaptor that was built.

Chapter 5 describes in detail the hardware and programming details for the Mark I servo interface board.

[1] It's not as bad as it sounds

## 1.2 Background and Motivation

Many projects within the GRASP lab. involve motion control via electric servo motors, for example robots, hands, camera mounts and tables. Each project has been based on a unique hardware/software approach. In the last few years the approaches have included

- VAL-II control language receiving commands over a serial line from a host computer
- RCCL (Hayward and Paul)
- RFMS multiprocessor (Zhang and Paul)

The first approach is limited by communications speed, and is not suitable for real-time sensor based control. The RFMS controller has proved in practice to be very difficult to program, and does not seem to have realized the full potential of its parallel hardware architecture.

RCCL is a very general robot programming environment and is capable of real-time sensor based control, as has been demonstrated by various projects within the lab. It does have the drawback that it is tightly coupled to the VAX architecture and Unimate robots and their controllers

RCCL provides the programmer with a particular model of the robot and its environment. This model, based on kinematic position equations and cartesian representation using homogeneous transforms, is very powerful, however there are many applications to which it is not well suited. It is at this point that the inherent inflexibility of RCCL becomes a problem, and the application programmer's effort goes increasingly into outwitting and thwarting RCCL "features".

Based on discussions with robot users in the laboratory the following points were made

1. Robot control hardware. It was considered that the best platform for robot control would be a powerful single processor system like a workstation. A single thread machine is inherently easier to program, and a workstation provides an integrated environment for program development and high speed execution. To allow a workstation to perform robot control an interface is required to the robot's electronic subsystems.

2. Robot interface. The RCCL controllers use a relatively high level interface to the Puma robot. The Unimate controller boxes provide position servo capability, A/D[2] and D/A[3] converters etc. A functionally more general interface was designed for the RFMS project, but the interface was physically limited to use within the RFMS(board size, connectors etc).

[2] Analog to digital
[3] Digital to analog

Professor Paul commisioned a final year project to build a general purpose 6 axis interface for the MicroVAX Qbus, but this was never finished and there is some doubt as to whether MicroVAXs and Qbus are the hardware platform to use in the future.

The author suggested a more general solution, based on the technology developed for the RFMS. The axis controller would be modular, thus allowing it to be expanded easily to cope with changing requirements, for example 7 axis robot, robot + hand, or two cooperating robots. Most importantly the axis controllers would be independent of the host processor bus, whether it be Multibus, VMEbus or Qbus. A simple electronic adaptor would connect the axis controller bus to the host bus, and would represent a relatively small fraction of the total system complexity, thus allowing easy migration to new host computing platforms. It was decided that the axis controller bus should be the IBM-PC bus, due to the variety of compatible products in the marketplace.

3. Robot control software. Based on experience with RCCL and CSIRO's ARCL robot controller[5] it has been decided to redesign the robot control software so as to be very modular, as opposed to the "monolithic" structure of RCCL. The structure looks like comprising a number of simple interfaces and functional blocks, implemented as libraries, and on which the applications programmer can build. The detailed work would be tackled by Gaylord Holder as a Master's project. A number of considerations in the design are:

    - at the lowest level it must be able to interface with the existing RCI interface to Unimate controllers, as well as the new MMCS hardware.
    - at the highest level it must provide a similar level of functionality to the RCCL programming environment, since this is one (despite its limitations) with which many workers are familiar. Within this new programming environments different programming tools will hopefully spring up and eventually replace RCCL.

To restate this, a new motion controller should

- be based on a fast single thread processor
- contain a host independent and modular motor interface
- be accessible via a small and modular software library

The remainder of this document is concerned with the first two points only.