Reconfigurable Mechanisms for Application Control (RMAC): Architecture

Wei Li[1], Edward Red[2], Greg Jensen[3] and Mark Evans[4]

[1]General Motors, China, weili109@yahoo.com
[2]Brigham Young University, USA, ered@et.byu.edu
[3]Brigham Young University, USA, cjensen@byu.edu
[4]Direct Controls, Inc., USA, evans@directcontrols.net

ABSTRACT

It is most difficult to introduce a disruptive method of control that could radically improve manufacturing efficiency when it would change, even eliminate, conventional manufacturing practices. Nevertheless, this paper introduces such a method to the research community for direct control of a mechanism by an application like CAD/CAM. Called ***RMAC*** for *Reconfigurable Mechanism for Application Control (**RMAC**)* [1], RMAC has been developed and tested at Brigham Young University as a logical extension of the direct control of mechanisms. Its architectural precursor, ***direct control***, is now in commercial form, and being applied to modern manufacturing applications.

The dynamic reconfigurable control architecture described in this paper allows a single mechanism to be controlled by a variety of applications and control laws, changing its control and kinematics characteristics according to the application demands. RMAC uses a software, rather than hardware-based, paradigm that assigns a device driver to a mechanism class or model, depending on the intended application purpose. The design strategy uses dynamic-link libraries (DLL) to form a mechanism device driver.

Keywords: direct control, reconfigurable machine control, application control

1. INTRODUCTION

For decades the machine tool industry has been limited by traditional control process languages such as APT, CL, and M&G code. Under this planning and control paradigm, NC programming is an "off-line" activity; that is, the CAD geometry used to create a generic NC program like APT and CL must be post-processed into specific M&G control commands for a particular NC machine. Production can be hindered when part designs experience manufacturing perturbations, such as changing a fillet size or shifting the center of a hole slightly.

Also consider the increasing demand for sculptured end products. Modern machine tools are frequently required to handle motion entities represented by NURBS (Non-Uniform Rational B-Spline). A NURBS surface may be represented inside the CAD program by parametric equations, but to be machined, surface curves must be tessellated into thousands of line and arc segments to meet the controller machining accuracies. Some modern controllers will reconstitute these block tessellations into NURBS to improve feed rates, but with cumulative loss in shape accuracy.

Moreover, starting with the generation of the APT and CL files from the CAM software, each subsequent ASCII control file loses associativity with the parent process. Any modification to part geometry or process file is not reflected in the original design model stored in the CAD system. This unidirectional data flow multiplies the number of design to manufacturing process cycles when several machining iterations are required to generate an acceptable physical part from a CAD model. As a result, traditional CNC control methods severely limit the flexibility proposed for modern concurrent/agile manufacturing systems.

In response to these limitations, the last decade or so has witnessed a wave of Open Architecture Control (OAC) systems. The European ESPRIT III Project of OSACA (Open System Architecture for Controls within Automation Systems) [2-3], the U.S. open control project OMAC (Open Modular Architecture Controller) [4], and the Japanese OSEC Architecture (Open System Environment for Controller) [5] are examples of proposed open-architecture controller standards aimed at replacing closed proprietary CNC systems. In academia, as early as 1988, Wright et al. [6] proposed the MOSAIC (Machine Tool Open System Advanced Intelligent Controller) architecture. Koren et al. [7], representing the Engineering

Research Center for Reconfigurable Machining System at the University of Michigan, proposed an open CNC system named UMOAC.
More recently, the goal is to develop reconfigurable machine tool controllers that allow end users to have even more flexible control systems on their factory floors, by allowing machine tool vendors to specialize in particular control modules rather than complete systems. The approach is to divide a machine tool into a set of autonomous functional units that can "plug-and-play" to form complete systems for particular customer needs. The European MOSYN (Modular Synthesis of Advanced Machine Tools) project [8] and the reconfigurable control project [9] at the University of Michigan aim at developing modular and reconfigurable controllers.
These open control projects use the PC-platform and object-oriented program languages such as C++. Off-the-shelf OS's, such as Windows and UNIX, allow the developers to quickly implement any new advanced technology into their control platform, thus benefiting end users. However, their implementation still depends upon the antiquated M&G code (and post-processing), which is often customized to be machine-specific and vendor-specific. As a result, most open/reconfigurable control architectures, like those previously discussed, respect the business boundaries between machine control and process planning applications. Controllers are simply not integrated with CAD/CAM application software. Tool paths derived from complex part models are still post-processed into thousands of line or arc segments, sometimes recombining them into complex mathematical forms like NURBS, before the controller can process them on a physical part. A truly seamless integration of CAD/CAM with the control systems is still not feasible under these various architectures.

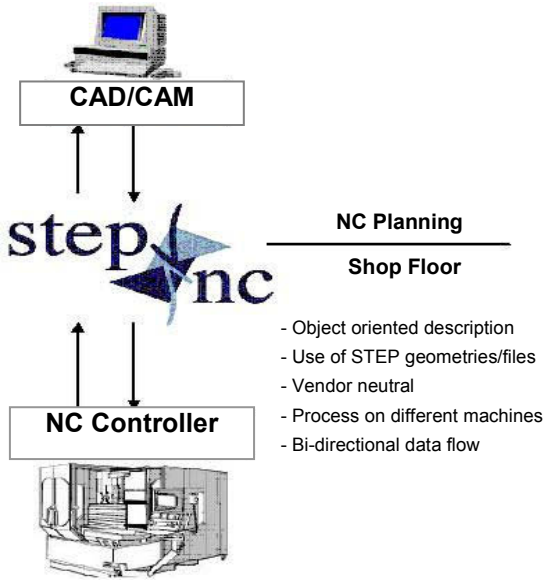

Fig. 1: STEP-NC machining process.

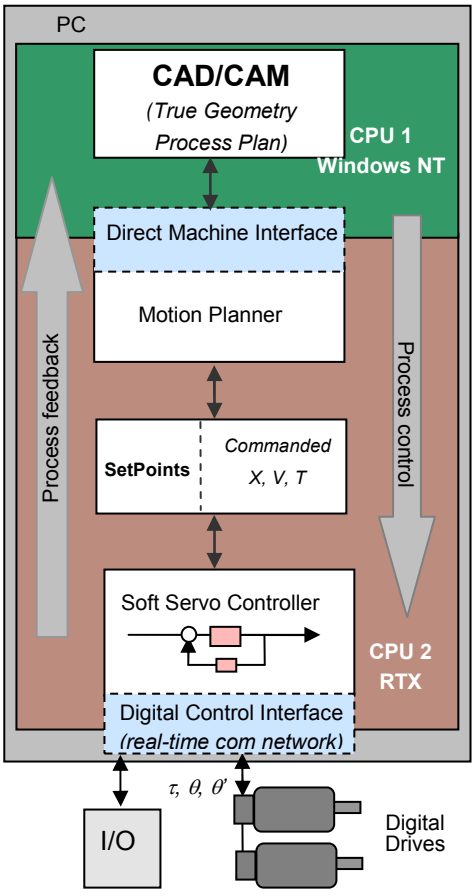

Fig. 2: DMAC system architecture.

1.1 STEP-NC

As a logical extension to the open control efforts, a more recent standards-based effort is represented by the STEP-NC program. STEP-NC is a neutral data description language designed to be CAM independent and NC machine-tool independent. Currently, under the IMS project [10-11], called STEP-NC in Europe and Asia, and Super Model in USA, industrialists and academics are collaborating to deliver a new data model as an ISO 14649 standard for CNC machines and to develop STEP-NC controllers. ISO 14649 is a new data interface developed to replace the old ISO 6983, known as M&G code, which functions as a bridge between CAM applications and the NC controller.
Even though STEP-NC provides a more robust link between CAM systems and CNC machine tools, it does not take the integration process far enough. The STEP-NC process shown in Fig. 1 still requires the translation of the CAM process plan into a STEP-NC file that loses associativity with the CAD model before being processed by the controller. Unfortunately, the STEP standards inherently respect, and thus maintain, the business boundaries between control/machine companies and process application companies, using files to bridge these boundaries. ISO 14649 perpetuates much of the overhead associated with file management and the business costs of conforming at all levels to the standard.

1.2 DMAC

To overcome these limitations, the ***Direct Machining And Control (DMAC)*** research group at Brigham Young University began in the 90's to develop an open-architecture controller [12-17] that directly interfaces to application software like CAD/CAM. **Error! Reference source not found.** demonstrates the DMAC system architecture. As shown, DMAC by-passes the need to post-process a CAM process into the file and language protocol required by autonomous controllers. CAD/CAM applications and the controller software concurrently communicate on the same PC. The CAD master model and CAM process directly drive the physical machine through a *Direct Machine Interface*, which practically eliminates the post-processing and multiple file handling processes of other open and reconfigurable control architectures.

2. RMAC AND ITS ADVANTAGES

Advances in high speed networks, digital electronics, and PC operating systems stimulated the development of direct control machining, and further commercialization. The simplicity of the direct control architecture further stimulated a new method for mechanism control, called *Reconfigurable Mechanism for Application Control (**RMAC**)* [1], as depicted in Fig. 3. Built on the direct machining architecture, RMAC seamlessly integrates CAD/CAM applications with the mechanism control system on the same computer. RMAC's architectural goal is to make mechanisms look like ***part printers*** connected to application software, thus hiding the mechanism configuration complexity within a *Device Driver* from the application user. Under RMAC the application and machine controller can reside concurrently on the same PC, and maintain mathematical tool paths without need for conversion.
RMAC and direct control demonstrate the following advantages and unique design features:

- ***Application and control on one compute***r, breaking down the current business and product barriers, resulting in reduced implementation costs.
- ***Software-based control architecture***, with all the version updating and modification capabilities that software brings to the production environment. RMAC reduces the need for dedicated hardware and firmware to embed process control algorithms used in feedback control, and to scan/control sensors and other I/O.
- ***Elimination of servo-control hardware***, such as 1) the general purpose servo-card, 2) PLC's, and 3) redundant computer boards for separate control and operator interfaces. To control motor torque, digital format amplifier boards are still required, although it is anticipated that much of the hardware form will be replaced by software as computers and networks become even faster.
- ***Replacement of a dedicated process machine*** with a *general purpose, reconfigurable mechanism*, limited only by motor size, kinematics and structural features.
- ***Direct, high fidelity, data and process communicatio***n between the application (e.g., CAD/CAM) and the software control core, using shared memory and API (Application Programming Interface) function calls, without need for data conversion into forms required by some communication standard or protocol. The process planning application becomes the *virtual operator* and the mechanism becomes the slave to the application.
- ***Simpler digital PC-based implementation*** that reduces the need for proprietary, more expensive, hardware solutions. RMAC borrows many of its architectural concepts from the PC's de-facto standards environment.

- ***Mechanism representation as a generic set of motors***, transmission elements, sensors and I/O, as limited by each mechanism's kinematics and structural classification (robot, mill, lathe, 3-axis, 5-axis, etc.), and by motor(s) size. Under the RMAC architecture a device driver configures the mechanism into the operational form required for a unique application. Loading a new device driver reconfigures the mechanism into a new machine, without need to structurally modify the mechanism or change controller hardware.
- ***Near optimal machine selection*** because the RMAC Device Driver Manager allows application users to select or reconfigure a machine tool for a particular application. A built-in database search engine allows users to easily and quickly narrow down their machine selections and select the relevant mechanism device driver. Machine-specific configurations and capabilities, such as machine spatial and power limits, maximum feed rate, etc., are integrated into a mechanism device database and accessible through the device driver.
- ***Mechanism-specific control algorithm***s can be designed and built in software as separate *dynamic-link libraries* (DLL's). A configuration system allows run-time mapping of mechanism-specific controls for the selected machine tool using relevant DLL's.

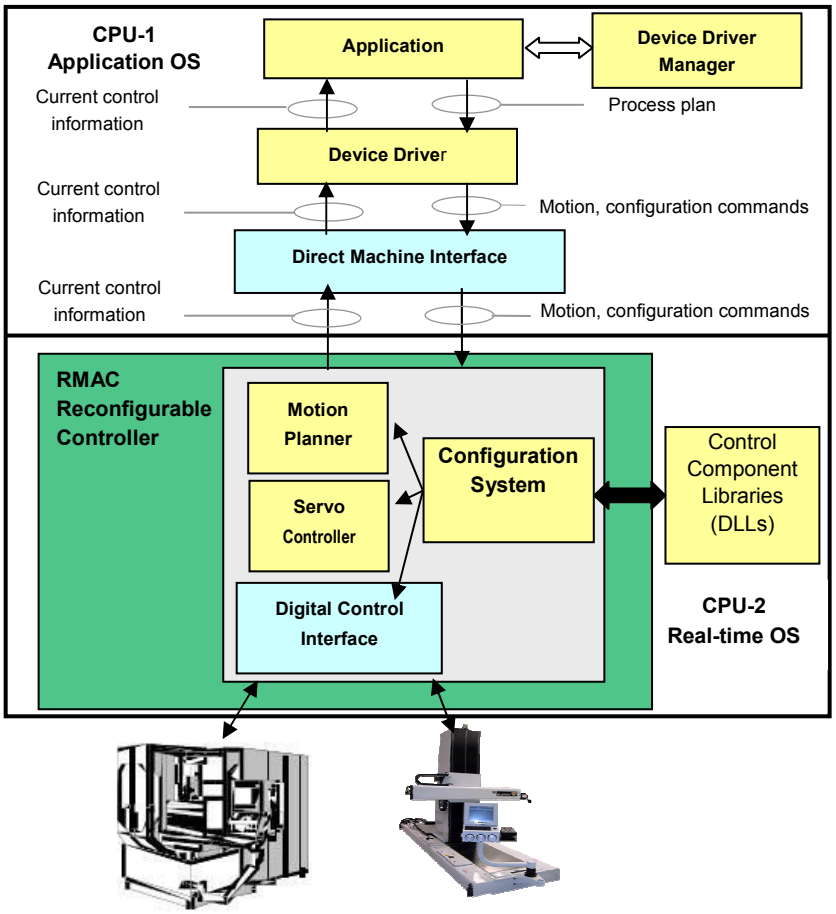

Fig. 3: RMAC architecture.

- ***RMAC changes how Product Data Management (PDM) systems would work***, by eliminating many stored files and modifying personnel access rules. Since a CAD/CAM process interface could be used to directly configure and drive a machine, the boundaries between design, process planning, and production become less defined, with less process conversion and data management required. In smaller shops, for

example, one person could do the part design, process planning, and production at a single machine. In larger facilities, the design-to-manufacturing processes could be greatly streamlined by eliminating unnecessary intermediate file conversions and handling. These advances cannot be easily realized with current practices or with any of the other open/reconfigurable control systems described earlier.

3. RMAC SOFTWARE ARCHITECTURE

As shown in Fig. 3 the overall software architecture is configured into distinct, hierarchically organized, software modules on one computer. The *Application* interacts with the *Device Driver Manager* to assign a proper device driver for the selected mechanism. The *Application* performs under a general purpose operating system (OS) like Windows XP on one CPU/Core of a dual CPU/Core computer. On the second CPU/Core, as instructed by the *Device Driver*, the *Configuration System* reconfigures the mechanism as a set of appropriate software control models.

Motion control and configuration commands flow from the *Application* to the RMAC reconfigurable controller through the *Device Driver*. The application software, such as CAD/CAM must have sufficient flexibility to directly control the mechanism through its API library. An interesting application, discussed in the companion paper *"Reconfigurable Mechanisms for Application Control (RMAC): Applications"* [18], uses PC-DMIS, a well-known CMM process planning package, to directly control the mill through a dimensional inspection process.

Process feedback flows from the reconfigurable controller back to the *Application* through the *Device Driver*, using COM interfaces or some other shared memory method. To allow for control and information flow, three interfaces and their interface APIs have been developed. Fig. 4 shows the necessary software modules and the interfaces defined within the RMAC architecture. The software system is composed of six different components described as follows, using CAD/CAM as the application example.

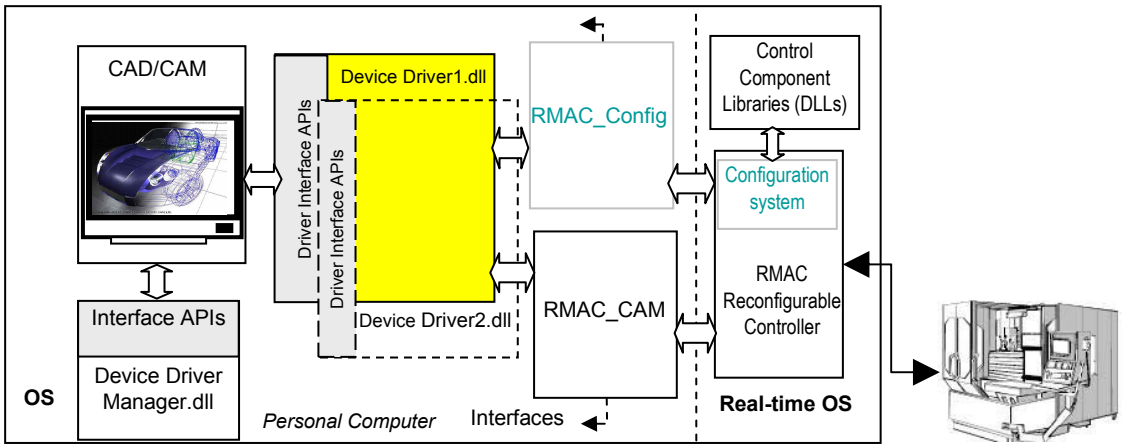

Fig. 4: RMAC software system overview.

- **CAD/CAM** creates a 3-D representation of physical model and, with input from the user, generates manufacturing process plans.
- **Device Driver Manager** maintains a device driver database relevant to a collection of different mechanism devices and their driver DLLs – see Fig. 5. It provides interface API's for CAD/CAM users to query for a proper machine and then locate a driver DLL for that machine. The *Device Driver Manager* is a DLL running independently from the CAD/CAM application. The device driver database contains the relevant device and related driver information accessible to the *Device Driver Manager*. Upon a user's selecting of a mechanism, the CAD/CAM application can search and locate a driver DLL through the interface APIs, provided by the *Device Driver Manager*. Then the application software can run-time load the device driver DLL for executing the intended process plans.
- ***Device Driver*** maintains a device database relevant to the details of a mechanism – see Fig. 6. These details include its structure, the number of axes, machine resolution, kinematics, and force/torque capabilities, etc., which can solely determine this mechanism's behavior. By accessing the database, the *Device Driver* is able to

properly configure the RMAC controller for the intended application. The driver then interfaces the application process commands directly to the physical mechanism through the reconfigured control software. The solid box shown in Fig. 4 represents the loaded device driver. The dashed box represents a different device driver DLL that can be loaded to replace the current driver software once the CAD/CAM user selects a different mechanism.

- ***RMAC_Config*** interface directs the configuration commands from the *Device Driver* to the *RMAC Controller* that will reconfigure the motion planner, servo controller, and underlying digital control interface. *RMAC_Config*, designed as a COM interface (can be other forms such as shared memory queue), will only interface with the configuration system inside the *RMAC Reconfigurable Controller*. Whenever the CAD/CAM user chooses a new mechanism, the *Device Driver* will call these interface API's to pass configuration commands to the *RMAC Controller* so that the controller software can be configured for the intended application.

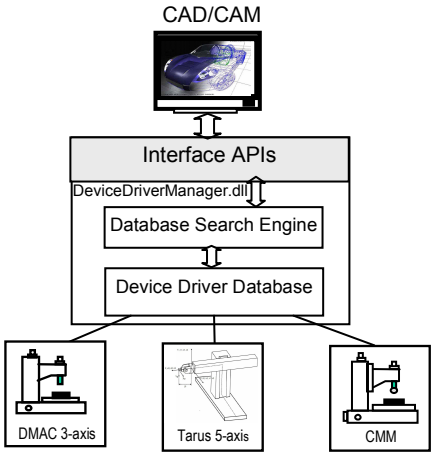

Fig. 5: Device driver manager.

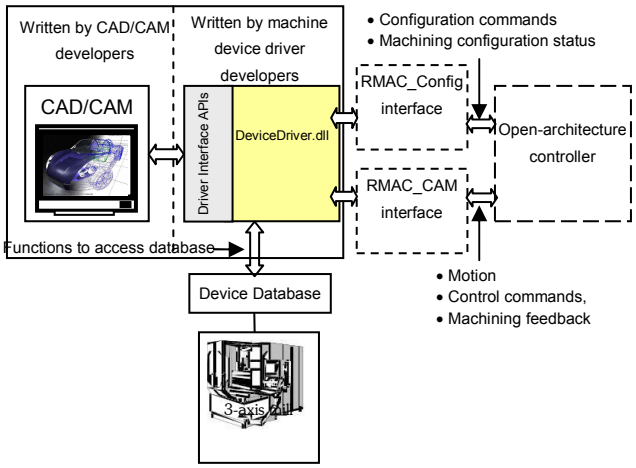

Fig. 6: Device driver.

- ***RMAC_CAM*** interface directs the motion and control commands to the *RMAC Controller*, receives the machining process feedback information, and sends it to the *Device Driver*. Presently, it is designed as a COM interface.
- ***RMAC Controller*** in Fig. 7 receives the configuration commands from the *Device Driver*. It uses the configuration system functions to map any mechanism-specific or application-specific control codes from the relevant DLL libraries. It then invokes kinematics representations, motion planning, storage buffers, and servo-algorithms to interpolate the motion and generate the necessary torque/current values to drive each individual actuator. The torque is delivered digitally to the motor amps over a real-time network like 1394. Other I/O commands are delivered similarly.

Control and feedback information both flow among these software modules through the following interfaces.

- ***Device Driver Manager*** interfaces to CAD/CAM **-** The *Device Driver Manager* exposes interface API's to an application like CAD/CAM for obtaining the necessary machine information. These function calls invoke the database search engine to search and locate a device and its driver DLL from the device driver database.
- ***Device Driver*** interfaces to CAD/CAM - The *Device Driver* exposes interface APIs to CAD/CAM for obtaining the detailed machine information. This machine information is then used to reconfigure the *RMAC Controller* necessary for executing manufacturing process plans on the selected machine tool.
- **Device Driver** interfaces to the *RMAC Controller* **-** The device driver software communicates with the *RMAC (reconfigurable) Controller* through two COM interfaces: *RMAC_Config* and *RMAC_CAM*. The *Device Driver* will only invoke *RMAC_Config* interface functions when CAD/CAM user selects a new mechanism. The configuration commands are sent to the *RMAC Controller* through *RMAC_Config* interface. *RMAC_CAM*

interface contains function calls that allow for CAD/CAM application to send motion and process commands to the *RMAC Controller*.

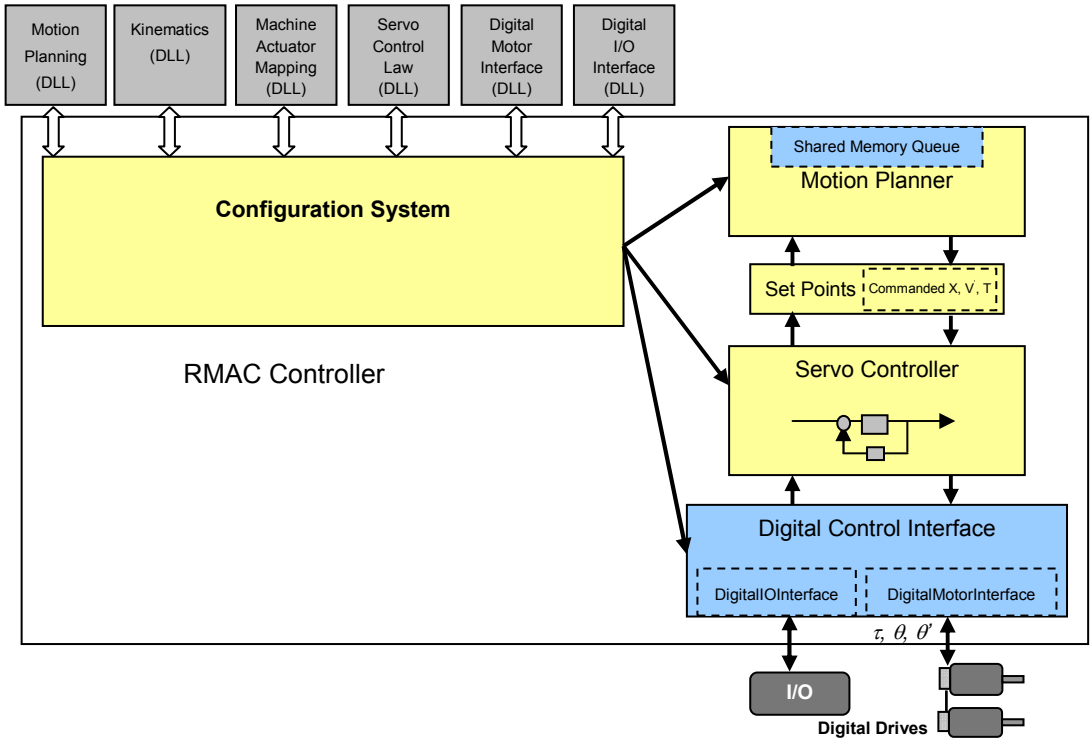

Fig. 7: RMAC reconfigurable controller.

4. RMAC APPLICATIONS

Direct control has been demonstrated on both machine tools and robots, using applications in Unigraphics, Catia, Alias Wavefront, GibbsCAM, and PC-DMIS to directly control mechanisms. Mechanism reconfiguration has been demonstrated by converting a Sugino 3-axis machine tool into a CMM, as described in greater detail in [18] and shown in Fig. 8. Direct Controls, Inc., has released a new line of sculpting robots using direct control technologies that provide both an economical and space advantage over larger prototyping machines – see [18].

Researchers and graduate students at BYU conducted a study of prototyping methods used by a large automobile manufacturer to design new automobiles. The study showed that times from shape concept to finished clay or foam prototype model could be reduced from 50% - 90% using direct control methods. Designers could conceive and then make the prototypes without the data conversion required today. Unfortunately, organizations that would have been displaced by these new technologies - some employing union personnel - reacted negatively.

Reference [18] - the companion paper - shows how RMAC reconfigures the mill in Fig. 9 according to a 3-axis application and a 2-axis application. Two device drivers were independently developed so the mill could be commanded as a three-axis sculpting mill or as a two-axis face mill. Fig. 9 shows the mill in the 3-axis configuration milling a car headlight directly from Catia CAD/CAM.

5. SUMMARY

This paper describes the RMAC architecture for mapping a mechanism into a device driver and using this device driver to reconfigure a machine for a particular application. RMAC provides applications, like CAD/CAM, direct control access to a new set of reconfigurable machines, releasing application process companies from having to adhere to standards

that perpetuate artificial business barriers. Unfortunately, direct control and RMAC prove highly disruptive to many companies and organizations because it has the potential to displace personnel or eliminate entire departments.
RMAC brings a number of advantages, namely control simplicity, reduction in hardware, and greater machine flexibility to the manufacturing floor, including in-process inspection of parts as they are being made. Much of the business overhead associated with maintaining bridge files such as APT, CL, M&G, or even STEP, can practically be eliminated. Instead, one master model drives the entire design-to-production processes. This overcomes many manufacturing hurdles evident in current practices. True bi-directional associativity is realized between applications like CAD/CAM and the machines they are directly controlling. As a result, last-minute changes or corrections to part geometries and process programs can instantly be fed back to design personnel for model updating.
Most important, RMAC architectures can drastically improve manufacturing efficiency by simplifying, even eliminating, conventional manufacturing practices, while the other open/reconfigurable research that we have reviewed still respects the current business boundaries.

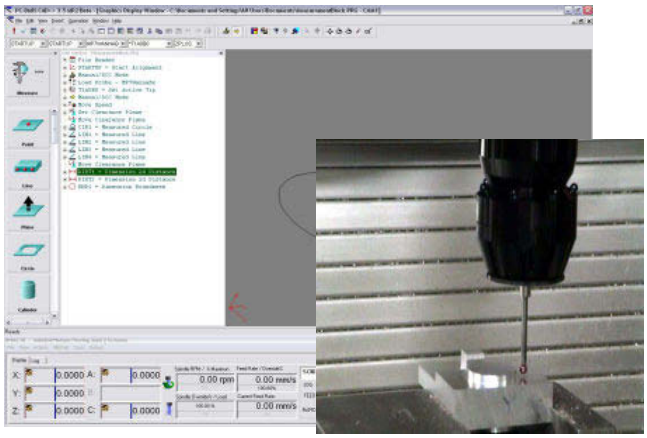

Fig. 8: PC-DMIS directly controlling a Sugino 3-axis mill.

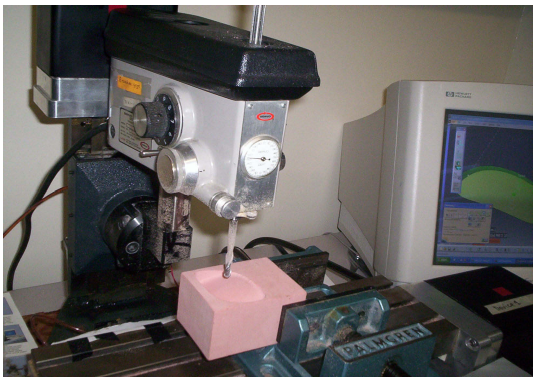

Fig. 9: Direct machining a car headlight.

6. REFERENCES

[1] Wei Li: Dynamic Reconfigurable Machine Tool Controller, Ph.D. Dissertation, Brigham Young University, Provo, Utah, 2005.
[2] OSACA Organization, http://www.osaca.org.
[3] Lutz, P.; Sperling, W.: OSACA – The Vendor Neutral Control Architecture, Proceedings of the European Conference on Integration in Manufacturing IiM'97, Dresden, Germany, 1997.
[4] OMAC Users Group, http://www.omac.org.
[5] Fujita, S.; Yoshida, T: OSE: Open System Environment for Controller, 7th International Machine Tool Engineers Conference, Nov. 1996, 234-243.
[6] Wright, P.; Schofield, S: Open-architecture Controllers for Machine Tools, Part 1: Design Principles, Journal of Manufacturing Science and Engineering, 120, May 1998, 417-424
[7] Koren, Y.: Open-Architecture Controllers for Manufacturing Systems, Open Architecture Control Systems, ITIA Series, 1998.
[8] Zatarain, M.; Lejardi, E.; Egana, F: Modular Synthesis of Machine Tools, Annals of the CIRP, 37/1, 1998, 333-336.
[9] Landers, R.; Min B.; Koren, Y.: Reconfigurable Machine Tools, Annals of the CIRP, 50(1), 2002, 269-275.
[10] Suh, S.; Chung, D.; Lee, B.; Cho, J.; Cheon, S.; Hong, H.; Lee, H.: Developing an Integrated STEP-Compliant CNC Prototype, Journal of Manufacturing Systems, 2002, 350-362.
[11] STEP-NC consortium, ESPRIT Project EP 29708, STEP-NC Final Report, version 1, Nov. 2001.J.
[12] Bosley, J.: A Modular, Open-Architecture Controller for Direct Machining, M.S. Thesis, Brigham Young University, Provo, Utah, 2000.

[13] McBride, C.: An Open Architecture Controller for Direct Machining and Control, M.S. Thesis, Brigham Young University, Provo, Utah, 2002.
[14] Miza, H.: Developing a Direct Machining Interface for Commercial CAM Systems, M.S. Thesis, Brigham Young University, Provo, Utah, 2003.
[15] Wei Li; Davis, T.; Jensen, G.; Red, E.: Rapid and Flexible Prototyping through Direct Machining, Computer-Aided Design and Applications, 1(1-4), 2004, 91-100.
[16] Cheng, Z.; Wei Li; Cheatham, R.; Wang, J.; Jensen, G.; Red, E.: A Direct Machining System for Commercial CAD/CAM Packages, ASME International Mechanical Engineering Congress Conference, IMECE-59992, November 13-19, 2004, Anaheim, California.
[17] Davis, T.; Carlson, S.; Red, E.; Jensen, G.; Sipfle, K.: Flexible in-process inspection through direct control, Measurement, 39, 2006, 57-72.
[18] Wei Li; Red, E.; Jensen, G.; Evans, M.: Reconfigurable Mechanisms for Application Control (RMAC): Applications, Computer-Aided Design and Applications, 4(1-4), 2007, 549-556.