# 'New lamps for old!' A method to rejuvenate old robots through the use of a simple set of microcontroller-based control boards to replace outdated control systems

P. Brunn[a] and A. W. Labib[b] (corresponding author)
[a] *Total Technology,* [b] *Department of Strategy and Business Systems (SBS), Portsmouth Business School, University of Portsmouth, P01 3DE*
*E-mail: ashraf.labib@port.ac.uk*

**Abstract** The paper describes the design, development, testing and use of a microcontroller- and PC-based control system which was used to repair and enhance an ASEA IRB6 welding robot in the authors' laboratory. The principles described could be applied to any robot of similar age and to provide a low-cost route to revitalise any working robot hardware that is limited by an outdated control system. The proposed approach addresses a problem within many manufacturing systems operating in industry.

**Keywords** robots; microcontroller-based control boards

## Introduction

As part of an MPhil project [1] to investigate the integration of machine vision with robotic devices for use in manufacturing systems, it was decided to repair a non-functioning ASEA robot that was in the possession of the Manufacturing Division, Department of Mechanical, Aerospace Manufacturing Engineering at UMIST. The robot had been previously investigated by an undergraduate student, who found that it was beyond repair, but the earlier history and the cause of the current malfunctions were unknown. There was, however, no obvious damage and the major elements of the device seemed still to be in place.

At the start of the project herein described, there was technician support available within the divisional laboratory facilities and consequently it was considered sensible to invest the time and effort, of both student and supervisory staff, to investigate, repair and reuse this robot. The addition of a PC as the main high-level control device meant that the robot could be integrated with other equipment using standard interfaces and controlled using generally well understood, standard, third-generation programming languages.

This paper, based on material first presented at the 33rd International MATADOR conference, Manchester, in July 2000, outlines the development and use of standard microcontroller techniques to retro-fit control boards and hence make the complete robot system suitable for a wide range of teaching and research activities.

## The ASEA IRB6 welding robot

The robot arm used for this work, the ASEA IRB6, is a five-axis robot (see Fig. 1), with three major axes and two wrist axes, all of which are electrically driven by d.c. servo motors. The waist is directly driven through a harmonic gearbox, the shoulder and elbow joints are driven by recirculating ball nuts on lead screws, and the wrist axes are driven remotely through linkages from drives in the base of the robot. Each axis has a velocity control loop (tacho generator on the motion shaft) inserted within a position control loop (resolvers), where both measurement devices are powered from and respond to the main control boards. The original design of this robot dates from 1973 and it is believed that this particular model was first used in 1975.

Initial attempts at fault finding established that the robot could not be started into any form of controlled motion, since the start-up and initial calibration procedures would not complete successfully. At this time the robot was moved from one laboratory area to another, due to a reorganization in the building occupation, and its work cell could be redesigned to allow access to the rear of the control cabinet without entry into the restricted workspace of the robot itself. Special connectors were developed to allow access to components and test points on the circuit boards of the original controller to investigate the cause of the problem further.

It was possible to establish that each axis could be powered up and moved at a defined speed, but only under open-loop control. The safety circuits were all working correctly and would allow the robot to start up if the main controller confirmed that all axes were properly positioned (which unfortunately it would not under normal operation). Within each axis servo system, both the tachometer and the resolver indicating position were shown to be working correctly. All the faults could thus be traced to the circuit boards containing the main control computer system.

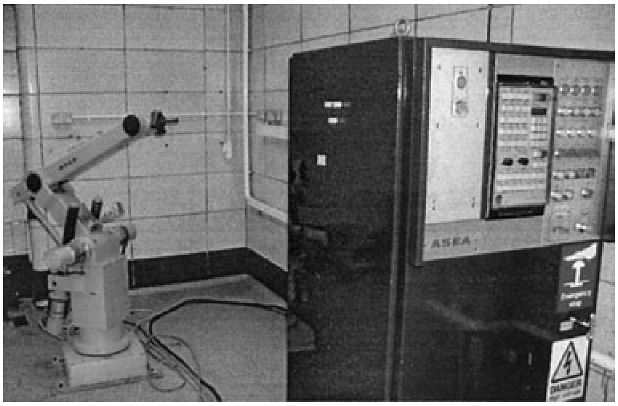

Fig. 1 *The ASEA robot.*

It quickly became apparent that it would not be practical or possible simply to repair the main circuit boards, for several reasons. Many of the components used in the original design were now obsolete and direct replacements were no longer available. It was doubtful whether pin-compatible replacements would be available in more modern logic parameters of whether these replacements would be compatible with the timing requirements of the remaining original parts. The processor system (Intel 8008) used in the original controller was very slow and of very limited capability (8-bit data and address busses, 14-bit program counter).

To complicate matters further, it seemed likely that the control code for the processor, stored in 7 kb of PROM (non-volatile read-only memory), was corrupt and following a general enquiry the manufacturer's agents were unable (or unwilling) to provide replacement PROM. They were, however, most helpful in providing documentation on the system. In particular, they were able to supply circuit diagrams of both the electro-mechanical and the electronic elements of the controller and the arm itself, without which the project would have been much more difficult or even impossible within the time available.

## The chosen repair philosophy

The investigation into the probable cause of malfunction of the ASEA robot and its possible repair had clearly shown that the majority of the parts were both in good condition and of fairly traditional design. They operated correctly when isolated from the main controller and were likely to continue to operate properly for the foreseeable future. The main control computer, however, was obsolete in both design and performance and was very definitely non-functional. The associated program and data storage facilities were similarly obsolete. The chosen approach was therefore to salvage as much hardware as possible but to replace the entire control system, with the exception of the main power supplies, motor control circuits and safety interlocks. This decision required a further choice as to the structure of the new controller, in particular regarding the relationship between the individual axis controllers and the overall control system. In the original design, the 8008 microcontroller had been responsible for the management of all functions, from inverse kinematics calculations down to axis speed setting and position comparison. It was thus both slow and limited in its capacity to interface with the outside world. It had required a specialist operating system and this led to difficulty in getting it to communicate with other devices.

One alternative considered was to employ a ruggedised PC to carry out all the control functions, since this would have considerably more raw processor power and standard devices to store programs and data and a range of conventional interfaces to communicate more widely. However, a single PC with a conventional, low-cost, operating system would not have been able to provide real-time control of each axis and simultaneously the necessary number-crunching power for geometric planning.

Other industrial control options were investigated but none had the price and power advantages of the use of a standard PC based on an Intel processor, when viewed in terms of overall robot and cell control functionality. Notwithstanding the academic constraints on finance, there were clear advantages to be gained from a low-cost yet flexible solution, since even at this stage it appeared that this repair exercise could provide a route by which other robots (and even other machines used in the department's research activities) could be upgraded within a limited budget.

The strengths of a PC-based system are its ability to provide overall control and system optimisation but the weaknesses are in the real-time control of individual axes. It was therefore decided to use single-chip microcontrollers to provide low-level control and simple I/O for each axis, and a conventional PC, with a standard type of parallel interface, to issue commands to and monitor the progress of these axis controllers. Additional circuitry would be required to provide the necessary power supplies and other 'housekeeping' functions, such as the sinusoidal excitation signals for the resolvers and the logic signals for overload monitoring and detection. This solution meant that a modular axis controller board could be designed, built and tested and then duplicated for each axis and for spares.

The wide variety and low cost of the many simple chip microcontrollers currently available would fit within the limited budget of the project. It was anticipated (correctly) that the circuit boards needed for these axis controllers would be relatively simple and thus could be designed, made, populated and tested in house. The following section discusses the particular family of microcontrollers selected for this project and then describes the operation of the axis controllers in more detail.

## Axis velocity and position control modules based on PIC microcontrollers

The overall control philosophy chosen was to add considerable intelligence to each axis controller (replacing the much simpler passive devices controlled directly by the single processor of the original system) to allow the PC providing system control time for other functions (such as cell control, vision system analysis, etc.). Consequently it was necessary to select a suitable microcontroller which had the required functionality and the most suitable life-cycle cost. The PIC family of microcontrollers from Arizona Microchip was chosen, partly because of their extensive technical specification but also because the Manufacturing Division, Department of Mechanical, Aerospace and Manufacturing Engineering and the majority of the Electrical and Electronic Engineering Department at UMIST had considerable experience of their use. As new products appear on the internet (almost daily) it may have been possible to find an alternative with a better unit cost and performance specification, but the experience already gained using the PIC family's Harvard architecture and Reduced Instruction Set Computing (RISC) approach meant that development time would be kept to a minimum. This was coupled with high operating speed, a wide range of facilities (such as internal resets) and simple power requirements, which set the PIC family apart from most rivals (Fig. 2).

Six identical axis control modules were produced (one to act as a spare), based on the PIC 16C74A, which uses a 14-bit instruction bus and a separate 8-bit data bus. The device has 33 I/O pins, which can be configured as one 6-bit, three 8-bit

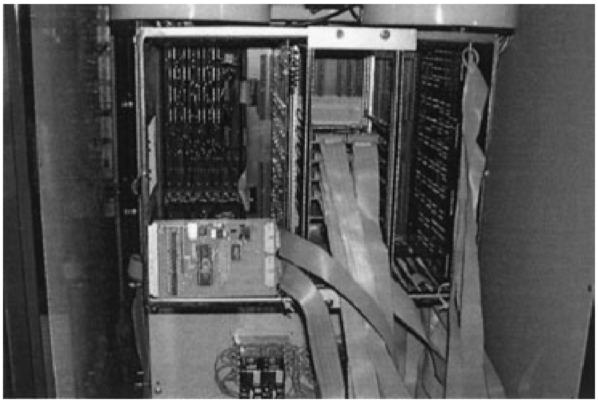

Fig. 2 *The robot controller.*

and one 3-bit digital I/O ports, some of which can also be used with other internal modes of the microcontroller, such as inputs to the on-board ADC or as parts of a serial port. A separate digital-to-analogue converter was used, together with a precision op amp and a voltage regulator to provide an output analogue voltage for the velocity setting sent to the axis motor drive unit associated with each board. A voltage comparator was used to convert each resolver's position feedback to a series of digital pulses for input to the microcontroller.

Each axis controller is designed to operate in one of three modes. The first, particularly suited to joint interpolated motion, requires that the supervisory computer sends to the axis controller a velocity and position value and, if these are accepted and the safety circuits are set correctly, a command to initialise the movement. The axis controller then starts to move the axis, using the drive's in-built circuits to ramp up to the set speed, continue the motion at constant speed until close to the final position and then decelerate the axis by reducing the speed setting until the final position had been reached with no overshoot. Once initiated, an axis motion of this type continues until completed (and then the microcontroller issues a message confirming success) unless the safety circuit is interrupted, in which case the axis stops immediately. The second mode of operation is very similar to the first except that the reduction in speed towards the end of the motion does not occur. A flag is set to indicate that the axis can receive a second command before the first is finished and on completion of the first motion the axis controller changes to the second speed setting without stopping the axis movement. This mode is ideally suited to controlled motion in a straight line and in the use of 'via' points, typical of several second-generation robot programming languages. The third mode allows the supervisory computer to set the speed voltage directly and control the axis open loop. It was initially used only for system testing but was retained in the final design in case this flexibility is required by subsequent uses.

All five axis controllers are normally synchronised to move together by a common 'go' command, and return digital values of their current velocity and position to the supervisory computer, while continuously checking the overall safety circuits. These are linked with the existing systems that monitor hardware performance, such as the motor drive thermal cut-outs, axis travel limit switches, etc.

The axis control circuit boards were mounted in the existing control cabinet, on a common back-plane, which also carried a new card that provided the excitation signals for the resolvers and other standard signals that were previously included in the processor boards of the original control computer. The use of a sheet of vero-board to form the back-plane (within a second-hand card cage) with sockets for each axis control board kept electrical noise to a minimum and allowed the use of nearly all the cables and connectors that made up the original system. There was considerable useful life left in virtually all the hardware elements of the robot and the controller, even though several of them had been superseded as technology evolved. It was therefore sensible to use as much as possible of this good-quality hardware, except for equipment such as the magnetic tape drive for program storage and the teach pendant, since the functions could be more easily duplicated (and enhanced) in the graphical user interface (GUI) developed on the new supervisory computer.

It is interesting to note that while the design, development and testing of the new circuit boards were not particularly straightforward (particularly when developing the resolver-reading conversion circuits) they were carried out by a research student whose first degree was in computation, with a supervisor whose main subject area was mechanical engineering. Thus, while some training in electronics was essential, it was not necessary to employ a specialist to design and test the circuits or to produce and populate the circuit boards that were used. It should therefore be equally possible for others who are primarily robot users rather than robot designers to undertake such an exercise to replace elements of a similar controller.

## The supervisory computer for geometric calculations and system control and optimisation

The system design includes considerable intelligence in each axis controller and therefore requires less continuous effort from the supervisory control computer. For playback control of simple taught programs, the supervisory system simply needs to send sets of position and speed information for each axis and commands to initiate motion. If sensor input and tool and/or gripper control are required, the supervisory system has simply to interface with the I/O channels of the robot and respond accordingly. Overlaid on these requirements is the need to monitor for alarm signals or incorrect position information. However, this does not need to be done in real time, since each axis microcontroller is also monitoring its own performance and the overall safety circuits.

For the system described herein, an IBM PC with a Pentium processor (running at 166 MHz) and 32 Mbyte of RAM was used for overall control. The system had ISA bus slots and thus could contain a relatively low-cost 48-bit programmable digital interface card based on two 8255 PIOs.

This computer had more than sufficient speed and processing power for this application and was available, so it served as an ideal platform not only to control the

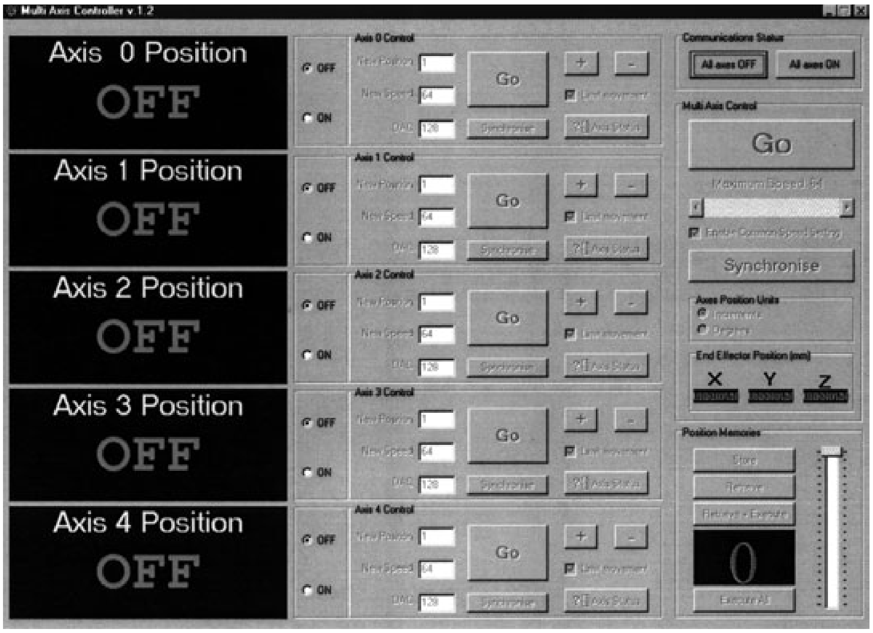

Fig. 3 *The developed user interface for the multi-axis controller.*

robot but also upon which to develop the software for both robot control and the vision-related aspects of the overall research project.

In effect, any 32-bit PC could have been used. A PC was also the most cost-effective way to provide programme and data storage and standard communications with other devices and systems outside the robot workspace. The absence of a need for real-time control of each axis from within the PC meant also that a standard operating system could be used (in this case Windows 95), which had advantages in terms of both cost and operator training. In this particular project the development language and programming environment chosen were Borland's Delphi, since both research student and supervisor had experience in its use. The object-oriented nature of Delphi allowed quite complex and imaginative GUIs to be developed. Delphi also has the advantage that assembler code can be integrated with the high-level language where necessary for direct access to the hardware [2]. (The software could equally well have been developed in C++ had the authors' experience been in that area.)

A GUI that duplicated the original teach pendant and some of the front-panel switches was developed to allow the robot to be started, moved to its calibration position and then controlled to move through a series of taught points while accepting and sending signals from a range of I/O devices (Fig. 3). The geometry necessary to perform forward and inverse kinematic calculations was encoded in a linked software unit. It was thus possible to write simple programs that moved the robot between predefined points in Cartesian space and subsequently to specify the path along which the robot should move. Additional, application-specific features could easily have been added to the system controller at this stage.

The main objective of the particular project for which the ASEA robot was first repaired was to integrate a vision system with robot control. Consequently some considerable time was spent selecting and configuring a vision system and writing software that carried out image analysis and then selected control strategies prior to converting these to actual robot motion commands. The PC was quite capable of using the majority of its processing capabilities to control the image-grabbing and analysis system while also generating all the necessary control codes to move the robot in response to visual events in the workspace. Given the computer-intensive nature of vision analysis, it is fairly safe to assume that the supervisory PC would therefore be capable of handling almost any task in a real manufacturing cell or similar situation.

## System testing and use

System testing was split into two main areas, namely hardware testing and software validation after the detection and removal of all code 'bugs'. Obviously, there was only a very limited amount of hardware testing that could be accomplished without the supervisory computer and none that related to the system without the individual axis controllers. (Standard test instruments and a variable-voltage source had been used during the early stages of the project to establish that the motor drives, resolvers and tachometers were all working correctly.) A simple GUI that provided control of individual axes (in axis coordinate terms) and the learning and motion for taught points was written to text repeatability. Using a pair of LVDTs and a simple reference frame, as in [3], the repeatability of the new system at a range of points in the workspace was shown to be as good as that expected from the controller it replaced (based on the manufacturer's specification). However, within this project it was no possible to establish the accuracy of the new controller when moving to points specified in Cartesian space. This was because the exact link parameters were unknown, since they were not provided in the extensive literature supplied with the robot and subsequently by the manufacturer's agent [4]. It was possible to estimate these (based on measurements taken from the assembled robot) but not with sufficient accuracy to make it worthwhile then to use the data in inverse kinematics calculations that would be used as part of a programme of position accuracy testing. (The estimates could, however, be used to demonstrate that Cartesian motion could be controlled by a series of inverse kinematics calculations.)

Since the open-systems philosophy of the design of the operating computer and axis modules allows any user to control each axis directly with commands that have the same resolution as the axis resolvers, it would be relatively straightforward to write additional software to move the robot to any position in joint space that linked with other software controlling a range of measuring devices. It is anticipated that another project could use this system both to compare robot accuracy measurement approaches and at the same time to improve the estimates of link dimensions used in the inverse kinematics calculations.

In this project it was considered more important to demonstrate that the control program in the supervisory computer could keep up with or outpace the robot and that communication with the axis controllers was not an excessive overhead. A simple demonstration was written to use the video system, integrated with the robot control software in the PC, to recognise when a new object (a small disc) appeared in the work area (a white tabletop) and then to move the robot to point directly at that object wherever it landed. Recordings of processor usage (from the Windows system Monitor utility) showed that even a relatively slow Pentium processor had spare capacity when tracking moving objects, establishing when they had stopped moving and then controlling the robot to point to each new disc as it arrived in the workspace. Prior to this demonstration, more mundane testing of each unit of the software had enabled the authors to validate the key elements of the control software.

The MPhil project, of which the repair and use of the ASEA robot was only a part, had a planned time scale of only one year. This extended to nearly 18 months in total. Seven man-months were spent on the design, programming, manufacture and testing of the individual axis controllers. A further four man-months were spent in programming the supervisory computer and subsequent testing of robot motion control and the various systems. The remainder of the time was required for the initial project feasibility study, selection of vision hardware and software and final thesis preparation.

Development of the axis control hardware was delayed at one stage because of a lack of facilities to manufacture printed circuit boards to the required quality within the department and earlier by problems with interpreting the resolvers' output to the correct resolution.

With the experience gained so far and, equally importantly, knowing that it has already been done successfully, it should be possible to repeat the exercise of upgrading the controller of a similar robot within a much shorter time-scale. It would therefore be an attractive proposition for any organisation with a non-functional robot and could be well worth consideration when planning to upgrade a working production facility. Similarly, an exercise of this type would be an ideal training opportunity for new technician staff and provide experience in all aspects of robot programming and use in a production environment, with otherwise obsolete (and thus low-cost) equipment.

## Conclusions

The approach to the upgrade of an existing (non-functional) robot controller, used in this project, has been shown to be successful and relatively inexpensive. It employs a distributed processing system, whereby a supervisory computer manages overall control by coordinating the collective operation of a set of intelligent axis controllers. The supervisory PC is responsible for performing all the computationally intensive but not time-critical tasks, such as inverse kinematic calculations and trajectory generation. The axis controllers have the form of microcontroller-based single-board computers whose main function is to carry out the time-critical tasks of regulating the position and motion of their respective axes. The position control loop is based on a client–server architecture, where the system's supervisor, the client, requests the service of the axis control units, the servers, via a series of low-frequency messages, in order to execute the assignments required by the user. This offers a range of benefits over the more traditional architecture. Technically the new system makes more effective use of the supervisory computer's resources, communication is easier, the system is easily expandable and upgradeable, and can be safer in operation, since additional features can be added, such as collision avoidance through either software or hardware. Commercially the new controller has cost advantages, insofar as it makes use of low-cost microcontroller and PC hardware, it uses standard interface and communication protocols outside the robot cell, it uses standard data storage and retrieval devices, and operates with an industry-standard computer operating system. Code to control the system can be written and maintained using any suitable high-level language (C++ or in this case Delphi) by a programmer who does not have to be an expert in machine-level code.

Most significantly, this approach allows the owner of an existing robot to update and enhance the control system of that robot, at relatively low cost, while retaining and reusing the expensive hardware, which, even after many years of productive use, is probably still in good working order.

The project described here has demonstrated that it is both possible and practical with relatively limited resources to upgrade an old-style robot control system. A further consequence of this work was that all involved obtained a much better understanding of the operation and control of that robot. It was possible to build relatively simple diagnostics into the software of each axis microcontroller and it would have been even easier to build extensive diagnostics into the supervisory programs had that been considered a significant element of the project. Thus there are considerable long-term benefits that could be gained for an organisation employing a number of robots in terms of staff training, system maintenance and ongoing product and process improvement. Future research will involve the investigation of the development of a system that can demonstrate integration of vision and motion within the ASEA robot vision. The perceived application is to attach a CCD camera to the robot and built a transfer line of parts (say L shape) moving along it in different orientations. The camera will capture the image and position of the moving part on the line and the robot will react by moving towards it in real time.

## References

[1] C. Trigonis, *The Selection of Hardware and Software Control Element for Robot Dynamics*, MPhil thesis, UMIST (1998).

[2] P. Thurrot, *Delphi 3 Superbible*, (Waite Group Press, 1997).

[3] P. Brunn and F. Hidalgo, 'Low cost robot performance assessment', in *Int. Workshop on Advanced Robotics and Intelligent Machines*' (University of Salford, March 1997), pp. 163–170.

[4] *Programming and Operating ASEA's Industrial Robots*, ASE (now ABB) documentation (1975).