IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, May 9-10, 1991

# Hardware Interface Configuration for Motion Control of the PUMA-560 and the Mitsubushi RM-501 Robots

Mile Erlic , Kevin Jones , and W.-S. Lu

Department of Electrical and Computer Engineering, University of Victoria
Victoria, British Columbia, Canada V8W 2Y2

*Abstract* – **Most results in manipulator dynamics and control are based solely on computer simulation. This technique enables the researcher to verify theoretical findings in a "quasi-real" fashion only. In fact, simulation techniques have difficulty accounting for practically important quantities such as uncertainty in the model parameter values, measurement noise, friction and backlash.**

**The experimental study in motion control of manipulators requires, on the other hand, substantial hardware modification since controllers that come with industrial robots often have structures which are too rigid to accommodate custom-designed motion control algorithms. In order to overcome limitations due to simulation studies only, our research team decided to accomplish some hardware modifications which would make manipulator motion control experimentation possible.**

**In this paper we report a recently developed hardware interface prototype configured for PUMA-560 robots as well as Mitsubishi RM-501 robots.**

## I. Introduction

### A. Mitsubishi RM-501 Control Hardware

The robot system being used by the Department of Electrical and Computer Engineering at the University of Victoria for courses in control systems and robotics is the Mitsubishi RM–501 Manipulator with the Move Master II Control Unit (MMCU) [1].

The manipulator has five degrees of freedom and an open and close end effector gripper. Each joint is powered by a permanent magnet electric motor and movements are actuated through gear, belt and chain transmissions. The manipulator has a system of optical encoders and limit switches placed at each joint which provide the control unit with joint position information.

The MMCU consists of logic and power electronics as well as robot control software which boots from ROM on power up. One Zilog Z–80 microprocessor is used for joint level control and robot language interpretation.

Although we had experienced much success with the implementation of the RM–501 robot system in undergraduate laboratories, we quickly discovered that the MMCU would be inadequate for manipulator dynamics and control research. The MMCU is an embedded system which was designed to interpret commands from a robot command language and execute simple point to point tasks. Since we were unable to find a simple way of reconfiguring the unit for low level manipulator motion control with a minimum probability of disabling the unit for operation in undergraduate laboratories it was decided to bypass the MMCU with a new much simpler interface.

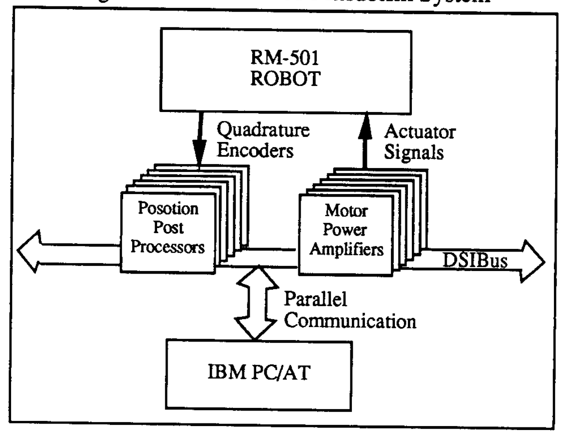
### B. The Control Hardware Modifications (Mitsubishi)

The primary modification the the RM-501 system was the replacement of the MMCU with hardware based on the Driver–Sensor Interface Bus (DSIBus) (Fig. 1) [3]. The DSIBus was designed so that it could connect to any parallel computer communication port. The robot hardware could then communicate to any standard computing system. The two plug-in boards designed for the DSIBus were the position post processor board and motor power amplifier board. The position post processor board converts quadrature joint encoder signals to parallel data. There is one such board for each axis of the robot. The motor power amplifier boards contain D/A electronics as well as power electronics. This board is used to give a precisely controlled control signal to the electric motor actuators of the RM-501 manipulator.

In our experiments with the Mitsubishi, the DSIBus was configured with five position post processor boards and six motor power amplifier. Communication to the DSIBus was facilitated with a custom parallel card plugged into an IBM PC/AT.

It was then possible to develop state of the art motion control algorithms on the PC/AT utilizing popular software development tools, i.e. Borland's TURBOC.

Figure 1. The Modified Mitsubishi System




### C. PUMA-560 Control Hardware

The PUMA-560's MARK II control hardware is similar to the RM-501 MMCU control hardware in many ways. It is based on an embedded system controlled by the robot command language, VAL II. It also has a teach pendent, serial port and external I/O. The manipulator sensor data is processed by encoder circuitry and there are motor power amplifier boards to actuate the manipulator.

In order to make the PUMA-560 a true research tool its propriatory control hardware would also have to be bypassed in much the same way as the RM-501 hardware. Since this system was not being used in undergraduate labs, it was decided to take a chance and utilize as much of the MARK II controller as possible.

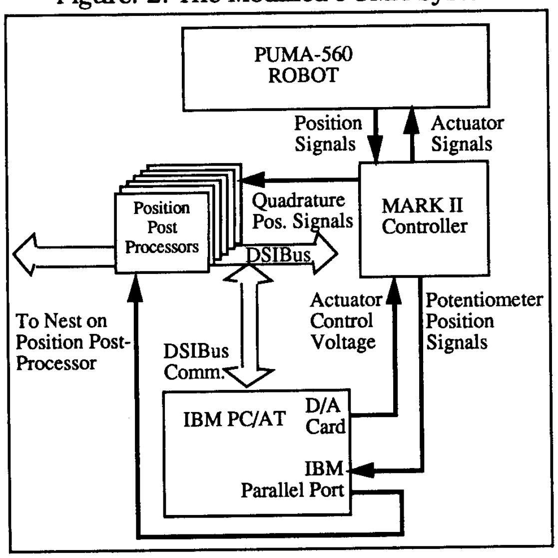
### D. The Control Hardware Modifications (PUMA)

With the PUMA-560, it was desired to utilize as much of the MARK II controller as possible. As a first attempt, manipulator position data was received by an IBM PC/AT through a serial connection from the MARK II controller. The data rate, however, was only 33Hz, an unacceptable rate for our research goals.

It was then decided to use the DSIBus with position post-processor boards to obtain much higher position update rates. The motor power amplifier boards in the MARK II controller could still be used by disabling appropriate connections to the MARK II CPU. The PUMA-560 uses a hybrid method to determine the absolute joint positions of its links. Potentiometers at each joint are used for absolute positioning and joint encoder are used for accurate relative position measurements. It was therefore necessary to determine the joint position from the potentiometers. For simplicity, voltage comparators were used to identify a calibrate position for each joint. Once these calibrate positions were determined, the joint encoders could update joint positions.

The entire system described is shown in Fig. 2. The modified system was successfully used by an International Submarine Engineering group. They programmed an IBM PC/AT to control the PUMA with Control System Probe (CSP), an object oriented, real time system.

Figure. 2: The Modified PUMA System




## II. THE MODIFIED MITSUBISHI SYSTEM

### A. The Driver Sensor Interface Bus

The DSIBus was designed so that the motor power amplifier boards and joint sensor post–processor boards could be addressed through a high speed parallel communication interface from a host computer.

It was also designed with computer independence as one of the goals. This was done so that the appropriate computation engine could be selected as an independent consideration at any point in the history of the manipulator. It was evident that for our purposes the IBM PC/AT would be used, due primarily to (i) its proven open architecture technology and (ii) the large selection of development software.

The DSIBus is a simple bus with a sixteen bit bi–directional data bus, an eight bit address bus, one +5V supply and one +30V supply. All backplane signal traces are 50 mil. at .156 in. spacing; power and ground traces are 100 mil. at the same spacing. Table 5.1 gives the pin–out description of the DSIBus. With eight bits of address space, the DSIBus can theoretically address 256 servo-driver and sensor post–processors.

For operating the RM–501, five addresses were required for the position post–processors An additional six were required for the motor power amplifiers.

The sixteen bit bi–directional data bus reads data from the the position post–processor boards. It also writes data to the motor power amplifiers. In either case, the sixteen bit dynamic range provided was sufficient for our experiments. The two power supplies selected for the DSIBus are +5V and +30V. The +5V supply was selected for compatibility with low power Shottky transistor transistor logic (LSTTL) and the +30V supply was selected for power electronics.

The results of an actual implementation has proven the DSIBus to be a useful component in a robot manipulator control research.

Table 1: DSIBus Pin-Out Description

| Pin | Function | Pin | Function |
|---|---|---|---|
| DSI 1 | Gnd | DSI 15 | D13 |
| DSI 2 | D0 | DSI 16 | D14 |
| DSI 3 | D1 | DSI 17 | D15 |
| DSI 4 | D2 | DSI 18 | A0 |
| DSI 5 | D3 | DSI 19 | A1 |
| DSI 6 | D4 | DSI 20 | A2 |
| DSI 7 | D5 | DSI 21 | A3 |
| DSI 8 | D6 | DSI 22 | A4 |
| DSI 9 | D7 | DSI 23 | A5 |
| DSI 10 | D8 | DSI 24 | A6 |
| DSI 11 | D9D | DSI 25 | A7 |
| DSI 12 | 10D | DSI 26 | +5V |
| DSI 13 | 11D | DSI 27 | Gnd+2 |
| DSI 14 | 12 | DSI 28 | 4V |

### B. The IBM PC-Bus

The IBM PC Bus has sixty two connections which are used for direct memory access (DMA), interrupt handling, data lines and control lines. In the design of a PC Bus/DSIBus parallel interface, only address lines (A0 – A9), data lines (D0 – D7) and control lines AEN, RESET, IOR and IOW were used. The IBM PC has only ten address lines decoded for input–output (I/O) operations. Of the 1024 bytes of I/O space which results, IBM has assigned I/O locations to many devices and only hexadecimal addresses 0x300 to 0x31F can be used for prototype cards. Since there is a possibility that third party peripheral boards will be used by the PC and may occupy some of the prototype I/O space, there is a provision in the PC Bus/DSIBus interface to change the PC address of our computer communication interface card. This provision is implemented with switches so that the parallel communication board can be moved around in the I/O space of the PC Bus by different configurations of these switches.

### C. Communication Prototype Card

Now that a brief discussion of both the PC Bus and DSIBus has been made, the design of the prototype PC communication card connecting the DSIBus should be easy to follow. Before beginning with the description, however, it should be clarified that the design has been modified from a design diagrammed in a Byte Magazine article [4]. Detailed physical and functional descriptions of all integrated circuits (IC's) can be found in references [5] and [6].

The design of the Communication prototype card is simple and consists of three dual in line package integrated circuits. These are the Intel8255 programmable parallel peripheral interface (PPI), the 74LS682 address comparator and the 74LS245 bus buffer.

Consider first bus buffering and address decoding. The 74LS245 was selected as the bi–directional bus transceiver. PC Bus data is transmitted to the Intel8255 data lines or Intel8255 data is transmitted to the PC Bus depending on the logical level at pin 1 (DIR) of the bus buffer 74LS245. A low on pin 1 invokes a PC Bus read whereas a high invokes a PC Bus write. Data isolation occurs when pin 19 (G) is high.

Board selection is done with the address comparator 74LS682 combined with an eight switch DIP. Depending on the comparator output, the Intel8255 is selected for PC Bus/DSIBus data transfer.

The Intel8255 has three eight bit bi–directional ports with high impedance three–state capability whose I/O configuration is programmable. Once the Intel8255 has been selected, the state of the A0, A1, RD and WR pins determine port configurations. The register map of the Intel8255 is given in table 2.

The Intel8255 operates in three modes depending on the control word written to it. These are mode 0 basic input/output; mode 1 strobed input/output and mode 2 bi–directional bus.

Table 2: 8255 Address Map

| A2 | A1 | A0 | RD | WR | Operation |
|---|---|---|---|---|---|
| X | 0 | 0 | 0 | 1 | port A -> PC Bus |
| X | 0 | 1 | 0 | 1 | port B -> PC Bus |
| X | 1 | 0 | 0 | 1 | port C -> PC Bus |
| X | 0 | 0 | 1 | 0 | PC Bus -> port A |
| X | 0 | 1 | 1 | 0 | PC Bus -> port B |
| X | 1 | 0 | 1 | 0 | PC Bus -> port C |
| X | 1 | 1 | 1 | 0 | PC Bus -> control |

In our Board only mode 0 was used in which the three ports A, B and C are separated into four regions; namely, port A, port B, port C (upper) and port C (lower). These can be programmed as input or output depending on the control word provided. In total, there are sixteen possible I/O configurations in mode 0. Only two configurations were required for our communication needs and these correspond to (i) all ports output, and (ii) ports A and B input and port C output. Port C is used as the address bus of the DSIBus; whereas, ports A and B are used for bi-directional data transfer to and from the DSIBus.

### D. Motor Driver Boards

The purpose of the motor power amplifier board is to provide a precisely controlled actuator driving force to each of the electric motors belonging to the Mitsubishi RM-501.

The design of the motor power amplifier board consists of four sub-circuits. These are (i) the address decoder and DSIbus buffer, (ii) the digital to analog signal amplifier circuit, (iii) power transistor and switching circuit and (iv) overload shutdown circuit.

The address decoder chosen for the digital to analog power amplifier boards as well as the position and velocity post-processor boards is the 74LS138. Sixteen addresses on the DSIBus have been defined using this chip.This addressing scheme satisfies the objectives of the driver-sensor interface in that all required RM-501 driver and position post-processor boards have a reserved address space.

The bus buffer on each board of the driver-sensor interface is based on the 74LS373 high impedance three-state latch buffer. This is an ideal chip where high output impedance three-stating is required during bus operations.

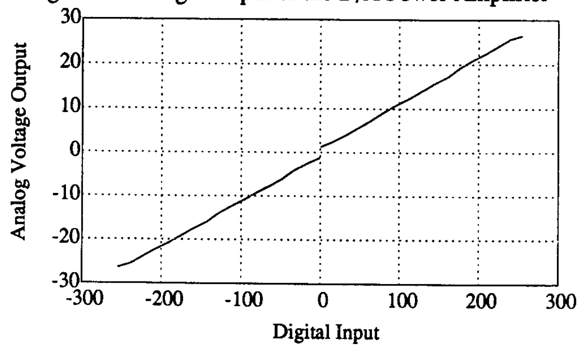
Every analog actuator when controlled by a microprocessor must have a digital to analog converter (DAC). In our system, eight bit data sent by the host CPU to the DSIBus is written to the buffered input of one of the digital to analog motor power amplifier boards. An eight bit digital to analog linear circuit based on the DAC-08C converts the digital signal to one of 256 levels between 0 volts and 2.5 volts. Amplification and offset calibration of this signal is made with two thirty volt single sided operational amplifiers. Variable amplifier gains are set with a 100K potentiometer so that an analog signal of up to 25V and 256 levels is provided to the power transistor stage.

Since the operational amplifier is only able to provide a maximum of 40 mA, power transistor stages are necessary for current amplification. In a design based on a maximum armature current of 3 A, a current amplification of 75 is required. Figure 1. shows the output linearity of the D/A motor power amplifier for no load conditions. Notice the good linearity between -25 V and 25 V.

Apart from providing linear digital to analog conversion, quick switching response and sufficient power to drive each robot motor, it is necessary to protect both the power amplifier circuit and robot motors from destructive overload conditions. The overload shutdown criterion is based on the level of motor armature current. A current sensor was implemented with a 0.1 Ohm, 2 Watt, 1% wire wound shunt resistor. The rest of the overload shutdown circuit is based on a single pole double throw relay , light emitting diode and silicon controlled rectifier.

The voltage drop across the 0.1 Ohm shunt resistor passes through a low pass filter with time constant 0.2 sec so that high frequency transients and short duration overload do not trigger the silicon controlled rectifier. The resulting signal is amplified by an operational amplifier. The overload shutdown condition can be

Figure 3: Voltage Output of the D/A Power Amplifier




changed by adjusting a potentiometer so that different armature current levels are chosen to trigger the SCR. Once the overload shutdown condition is triggered, the SCR switches on, the voltage drop across the pole of the relay increases, the open contact condition is actuated and the circuit which powers the RM-501 servo-motor opens.

## III. THE MODIFIED PUMA SYSTEM

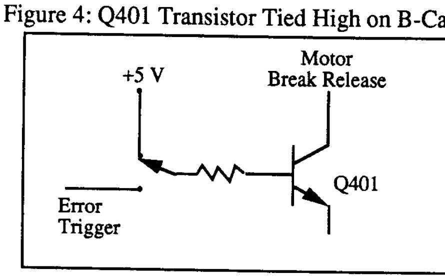
### A. The Modified Mark II Controller

The approach used to modify the Puma Mark II controller was different than that of the Mitsubishi system. Researchers wanted the flexibility of access to the stock VAL II system, as well as a high speed controller that was operated by a foreign host. For this reason, it was decided to use as much of the stock unit as possible.

It was also desired to use as much of the DSI hardware as possible. The motor power amplifier cards which were designed for the Mitsubishi robot would require substantial modification to operate the Puma. The DSI input cards, however, were ideal for the application, and are discussed following the output section. The Puma 560 Mark II controller contains two power amplifier boards, each of which operate three joints. To use the power amplifier boards in the Mark II, all that was required was to disconnect the ribbon cables between the Mark II digital to analog converter board and the power amplifier boards. These two cables were then connected to a six channel digital to analog converter in the new IBM PC/AT host computer. To select the original or modified controllers, the researcher would merely have to swap a couple of cables at the back of the Mark II controller before turning it on.

In order to gain control over the arm from a remote host, the Mark II shutdown mechanisms had to be disabled. One must remove the Puma controller's "B" card and make the modification shown below. After this, the user can control the arm power with the on/off buttons at the front of the Mark II controller.

Figure 4: Q401 Transistor Tied High on B-Card




The Mark II input card provided both digital (relative) position data from the shaft encoders, and analog (absolute) position data from potentiometers on each axis of the robot arm. A ribbon cable was used to obtain all of the signals of the input card. The ribbon cable was connected to a prototype card in an empty Q-Bus slot, which utilized a system of buffers and comparators to sample the digital and analog information. This information was then fed through another ribbon cable to a socket at the back of the Puma controller. The result was a generic data socket that provided all the position information from the Mark II controller, without any affect whatsoever to the standard Puma Mark II system operation. The pinout of the data socket is shown below.

Table 3: Position Information Socket

| Pin | Function | Pin | Function |
|---|---|---|---|
| 1 | J1 A | 27 | J4 B |
| 3 | J1 B | 29 | J4 INT |
| 5 | J1 INT | 31 | J4 NST |
| 7 | J1 NST | 33 | J5 A |
| 9 | J2 A | 35 | J5 B |
| 11 | J2 B | 37 | J5 INT |
| 13 | J2 INT | 39 | J5 NST |
| 15 | J2 NST | 41 | J6 A |
| 17 | J3 A | 43 | J6 B |
| 19 | J3 B | 45 | J6 INT |
| 21 | J3 INT | 47 | J6 NST |
| 23 | J3 NST | 49 | +5V, .5 Amp |
| 25 | J4 A | | |

• All even numbered pins are ground.
• J1 A - Joint one, encoder A

### *B. The Role of the DSIBus*

The DSI bus, developed for the Mitsubishi controller, was a natural choice for interfacing between the Position Information Socket and the host computer. The Position Information Socket provides digital counting pulses from the optical encoders for each axis of the Puma 560 robot arm. These pulse signals were then run to the DSI Bus Input Cards, and the position data could be read by the host computer, an IBM PC/AT, using the previously discussed Communication Prototype Card.

The chief difference between the Mark II system and the Mitsubishi system was in the way physical limits for each joint were detected. The Mitsubishi robot arm provided a third signal called the "Nest" signal, which indicated that a limit switch had closed at the end of the arm's travel. The Puma 560 arm did not provide this signal. Instead, there was a third signal, which could be used as an "interrupt" signal. That is, if the arm moved, a pulse was activated, indicating that the arm was in motion. The DSI bus cards required a signal that indicated the end of travel for a joint. And the users needed to know the absolute position of each joint.

This could have been done by installing an analog to digital (A/D) converter for each joint in the Mark II controller, to read each potentiometer setting. Much more circuitry and cabling would have been required. A simpler approach was chosen. For each joint, a simple comparator circuit was constructed, which provided a one bit digital signal. As the arm moved, a triggering potentiometer voltage was reached, and the comparator would switch from its low state to its high state. Thus, the comparator circuits allowed the user to determine the absolute position of each joint, with a minimum of complexity. The comparator signals were labelled "Nest" signals on the Position Information Socket. These Nest signals had to be read by the host computer. The signals were routed through a 7407 TTL open collector buffer and connected to the input pins of a spare parallel port on the back of the PC/AT host computer.

Finally, the DSI bus cards required a signal that indicated the end of travel for a joint (a reset pulse for each up/down counter circuit). Six output pins (one for each DSI bus card) on the spare parallel port of the PC/AT were used to reset the counter circuits for each DSI bus card.

On system power up, the calibration procedure for the Puma 560 robot arm is as follows. Each joint is driven by the digital to analog converter (mentioned next) until the "CAL" position is reached. This is determined by the change of state of the comparator signal. The DSI input card for that joint is then set to a known state. This process repeats until all six joints are initialized.

The DSI system works very well for monitoring the exact position of all joints of the Puma 560 arm. In fact, using a stock PC/AT (with a 6 MHz system clock), and no tasks active, update rates of several kilohertz are realized. Velocity feedback is much improved as well, since the DSI cards can be used to compute velocity from the position information. The DSI allows the researcher a much improved platform for real time control. It should be noted that the DSI Communication Prototype card could easily be modified for other computing platforms as well. Future plans may include a Macintosh II or Sun version.

### *C. The Digital to Analog Card*

The digital to analog card used in the host computer for the modified Mark II controller was the Metrabyte DDA-06. It is a six channel, twelve bit converter with voltage outputs selected for plus or minus ten volt range. The analog output signals were connected through a thirty seven pin D type connector to a prototype adapter card. This adapter card at first, was merely used to convert the signals to two 16 pin IDE sockets, which plugged into cables from the Mark II power amplifiers.

The stock power amplifiers showed signs of instability with the new digital to analog card driving them. For this reason a set of six 1 kHz low pass filters were used between the digital to analog card and the power amplifiers. This limited the output step rate to just under one kilohertz, but cleared up the problem of instability in the power amplifiers. Since there was a need for an unmodified Puma at times, this problem was not investigated further. Also, in a practical situation, an output update rate of just under one kilohertz would seem more than adequate.

## IV. Conclusions

Both the Mitsubishi RM-501 and PUMA-560 Robot Manipulator control hardware structures have been sufficiently modified to facilitate manipulator motion control research. In the case of the RM-501, the Movemaster Controller was completely replaced by hardware based on the Driver-Sensor Interface Bus (DSIBus). For the PUMA system, it was desired to use as much of the MARK II controller as possible. The resulting system utilized the motor drivers of the MARK II controller along with the DSIBus for positioning encoding of the joint quadrature signals.

Both of the modified systems have been tested. The modified Mitsubishi system was used to perform successful studies in adaptive manipulator motion control. The modified PUMA system was used by International Submarine Engineering in tests with Control System Probe, ISE's propriatory real time, object oriented control system software.

Presently, our research group is configuring an advanced computer system based on the MACINTOSH IIfx to control the modified PUMA-560 or RM-501 systems. The MACINTOSH computer will serve as the user interface. LabVIEW, a powerful graphical programming language, will be used as the development environment and the system will use the TMS320C30 32-bit DSP for number crunching engine.

## References


[1] Instruction Manual, Move Master II, RM-501 Model, 1984.

[2]] W.-S, Lu *Laboratory Manual for ELEC 475 Robotics: II*, University of Victoria, 1988.

[3] M. Erlic and W.-S., Lu, "Computer Independent Driver-Sensor Interface (DSI) for Closed Loop Control of a Mechanical Manipulator,"*Canadian Conference on Electrical and Computer Engineering*, Montreal, PQ, 1989.

[4] J. R., Drummond "Three Bus Interface Designs for the PC,"*Byte, The Small Systems Journal*, no. 12, pp. 225 – 245, 1987.

[5] The TTL Data Book for Design Engineers, Texas Instruments Inc., 2, 1976.

[6] Microsystem Components Handbook, Microprocessor Volume II, Intel Corporation.