ORMEC SYSTEMS CORP

PROGRAMMABLE MOTION CONTROLLER

(PMC-900, PMC-901, PMC-902)

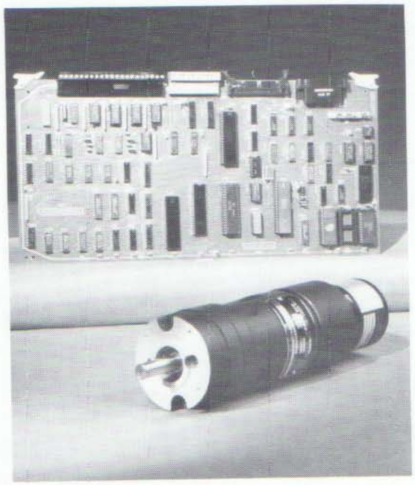
ORMEC's Programmable Motion Controller (PMC) is a single-board, microprocessor-based controller which combines with either a DC or AC servomotor system to provide high performance motion control.

It gives OEMS, as well as end users, a unique capability for integrating robotic-like features into equipment designs without requiring the intensive in-house technical support that custom microprocessor-based feedback control systems require. And it is ideal for retrofit applications, because the advanced automation technology of the PMC can improve speed and performance of existing production equipment.

The PMC operates as an intelligent slave precisely controlling velocity and position in response to high level ASCII commands.

In a typical application, the PMC interfaces with a servodrive, a servomotor, a DC tachometer and an incremental position encoder with quadrature outputs to create a closed loop digital positioning servo that accurately controls the position and velocity of a mechanical load.

The system offers outstanding performance including position stepping rates to 384 kHz and encoder resolution in excess of 100,000 counts per revolution if you need it for your application.



Interfacing the PMC with a servomotor, a servodrive, a tachometer and a digital position encoder is a fast, easy method for creating high performance motion applications.

PMC based systems can perform variable indexes at rates greater than 3000 indexes per minute.

The PMC closes both the velocity and position loops, providing load independent positioning accuracy and predictability. And it eliminates the need for potentiometer adjustments by utilizing gain and compensation data stored in the microprocessor's non-volatile memory. Field installation is simplified by the polarized removable terminal strips and industry standard connectors.

Doing sophisticated motion control applications with the PMC is as easy as programming a calculator. ORMEC's Motion Programming Language (MPL) offers system designers a wide choice in how to configure their system and a full set of programming commands, from indexing a specified distance or jogging at a predetermined velocity to decelerating on a machine sensor or coordinating I/O points with the motion. These commands can be combined with branching and subroutine calls in a general way to provide solutions for complex motion control applications.

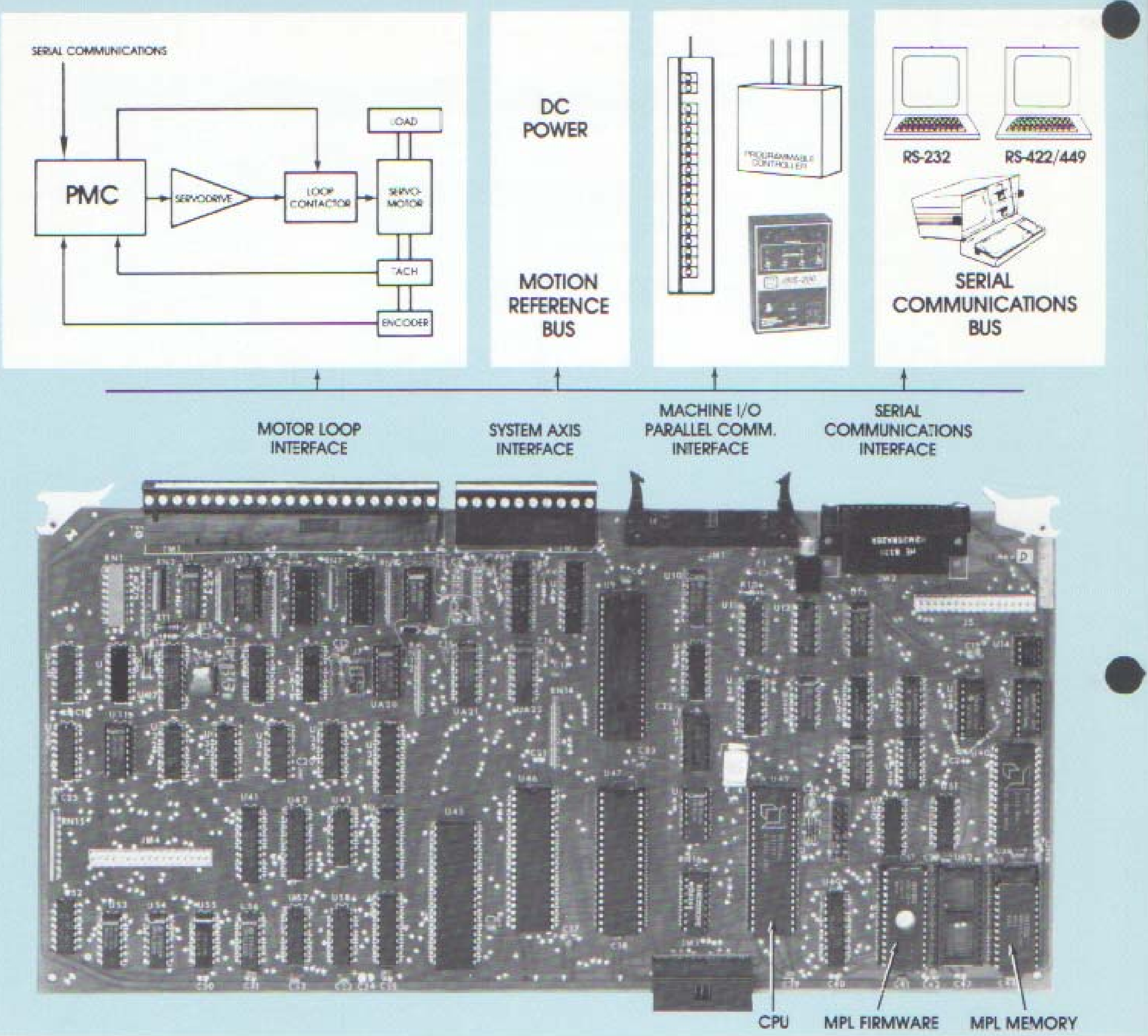
The PMC can be controlled by virtually any host computer system or programmable controller. Or it may be operated in a stand-alone mode, since it can coordinate commanded motion with machine inputs (limit switches and sensors) and machine outputs (solenoids, indicators, etc.).

When interfaced to the host system through the Serial Communications Interface, the host can interact with the PMC, even while it is controlling motion, to monitor position, velocity or acceleration.

The PMC provides a consistent, cost effective approach to motion control, whether it is used to control one axis or combined for use in multi-axis systems.

BENEFITS OF A PMC

- Executes high level Motion Programming Language commands
- Stores 95 unique motion control routines in memory
- Creates a closed loop, digital position servo
- Coordinates motion with Machine Inputs/Outputs
- Operates stand-alone, or as slave of computer or PLC
- Eliminates troublesome potentiometer adjustments
- Simplifies debug/maintenance with on-board diagnostics
- Interfaces through quality, industry standard connectors



FEATURES OF THE PROGRAMMABLE MOTION CONTROLLER

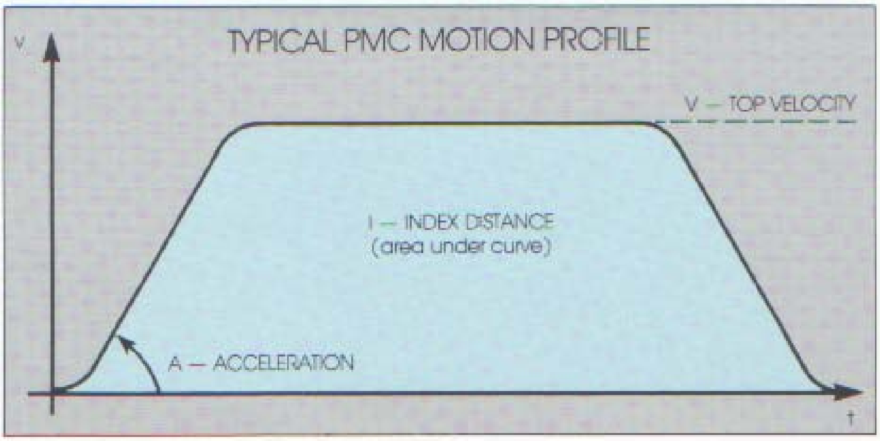
The Programmable Motion Controller (PMC) has been carefully architected and thousands of engineering manhours have created a general purpose, programmable motion controller that is easy to use in a wide variety of complex motion control applications.

Let's look at the PMC's unique features. Refer to the diagram above, and you'll see how our overall understanding of servo control systems has resulted in a single-board, programmable motion controller that is simple to use, but has power and flexibility for your most demanding applications.

PMC ARCHITECTURE

CLOSES THE VELOCITY AND POSITION LOOPS

The PMC interfaces with a servodrive, a servomotor, an



A typical PMC Motion Profile using "s-curve" acceleration is shown above. Users may also choose linear, parabolic or some other acceleration shape.

incremental position encoder and a tachometer to create a closed loop digital position servo. This closed loop operation results in load independent positioning accuracy and predictability, and creates a reliable feedback control system.

MICROPROCESSOR BASED

The heart of each PMC is an Intel 8085 microprocessor, giving each axis of motion control the processing power of a personal computer.

The CPU processes instructions stored in its firmware (EPROM), allowing the PMC to interpret the high level set of user oriented motion control commands of the Motion Programming Language (MPL).

MPL operates interpretively, like the BASIC programming language, enabling commands to be executed directly to parametize or initiate high performance motion. The same commands can then be combined with program flow and timing commands in "motion control routines" to easily implement complex motion control applications.

NON-VOLATILE MPL MEMORY

For the PMC-901, motion control programs are stored in up to 2k bytes of non-volatile memory (EEPROM). The PMC-902 uses RAM memory, and the PMC-900 is designed for use under the direct control of a host computer.

MPL ARCHITECTURE

Because MPL uses intuitive single character commands, programming the PMC is as simple as programming a calculator. Simple commands such as A, V, and I are used to establish motion parameters such as Acceleration, Velocity and Index distance.

In addition to being easy to learn, MPL's terseness and regularity speeds program execution and simplifies host communications.

The PMC is architected so that MPL operates independent of the motion it creates. This feature allows the MPL program to continue to execute, even while a motion is in progress; machine outputs can be operated durng the motion, or future motions can be parametized.

Specific commands are provided for synchronization of the program and motion in progress.

MOTOR LOOP INTERFACE

ENCODER ASSURES POSITION ACCURACY AND PERFORMANCE

Encoders with "TTL compatible phase quadrature" outputs, displaced in phase by 90 degrees, are used in ORMEC positioning systems.

Using this interface provides flexibility since it is common to many position encoders from inexpensive optical incremental encoders to exotic and accurate transducers such as laser interferometers.

HIGH NOISE IMMUNITY

The PMC has a unique quadrature decoder circuit which, while interpreting the input channels to determine the load direction and distance travelled, rejects noise pulses which may occur on the signals. Both quadrature channels of the encoder must pass a minimum time criteria to be accepted by the digital circuitry, and if a long duration noise pulse occurs, it is decoded by the interface as a combination of a forward and a reverse encoder pulse.

In addition, this circuit utilizes "4X multiplication circuitry" which, because four encoder distance units are derived from each encoder cycle, increases the effective resolution of the encoder to four times its stated line count.

NO POTENTIOMETERS

Both the position and velocity loops are closed in hardware, for maximum performance and reliability. Gain and compensation circuitry for both loops are under direct digital control of the on board microprocessor which provides initial setup convenience and prevents field adjustments from becoming a future problem.

The PMC is configured with a 48 db gain adjustment for both the velocity and position servo loops. Integral + proportional compensators with adjustable "zeros" are implemented. These adjustments are made using an MPL command, and can be changed dynamically during system operation.

Since both position and velocity loops are closed by the PMC, it will operate with virtually any servodrive, relying on it only as a power amplifier.

ADVANTAGES OF A TACHOMETER

A PMC-based positioning system utilizes a tachometer as a loop compensation device to provide motion control performance not otherwise possible. To provide speed control accuracy beyond the limits of analog systems, the tach is not used as the final authority on speed, which is reserved for the digital position transducer.

An analog velocity feedforward signal is included in the microprocessor controlled servo reference generation circuitry, and applied directly to the velocity loop through a PMC adjustable gain. This feature allows a PMC based system to operate with minimal position error, even during hard acceleration and deceleration and at variable rates of speed.

Velocity feedforward is particularly valuable for maintaining wide dynamic system response when the position encoder is attached to the system remote from the servomotor. Examples of this include an encoder measuring web position in a winding application or a linear encoder attached to a ball screw-driven positioning table.

SAFETY FEATURES

The Loop Contactor, which can control an optional dynamic braking resistor, is architected into the PMC as a safety feature. In normal operation, the PMC provides an output which may be used to enable either a loop contactor or the servodrive. Should the PMC sense excessive position error, it disables this output, rendering the servomotor powerless. This output is controllable from an MPL program, allowing the positioning system to be placed in the IDLE (disabled) mode.

MACHINE I/O INTERFACE

16 INPUT/OUTPUT POINTS

The PMC has 16 I/O points to handle limit switches, provide user programmable digital I/O points and provide a convenient interface for controlling a PMC from a programmable controller or directly from hardware switches.

SIMULATING INPUTS/OUTPUTS SIMPLIFIES SET-UP/DEBUGGING

ORMEC's Machine Interface Simulator (MIS-200) speeds system development of PMC-based systems. This inexpensive unit allows the user to simulate Machine Inputs with switches and Machine Outputs with LEDs in a lab or office environment.

SIMPLE TO INTERFACE FOR STAND-ALONE OPERATION

Since the PMC-901 is equipped with its own Machine I/O and nonvolatile memory, it can be operated in a stand-alone mode to control a small machine or machine module directly.

Any ASCII terminal, such as ORMEC's Portable Programming Terminal, can be used to create MPL programs which are stored in the PMC's non-volatile memory.

The PMC has a "startup program" feature that allows it to operate a user program upon powerup, so that a terminal is not required for operation. This user programmable startup program is also used to initialize the tuning parameters for the servo loops.

ACCESS MPL ROUTINES FROM A PROGRAMMABLE CONTROLLER

The user can access up to 32 of the 95 possible motion control routines by placing a binary address in parallel at the MIO interface. When the READY' output indicates that the PMC is "ready" for a new command, the user starts the selected routine by asserting the EXECUTE' input.

The PMC can be interrupted at any time by asserting the STOP' input, which stops system motion in addition to interrupting program execution.

This straightforward interface allows convenient integration of PMCs with a wide range of systems, such as switches and relays, discrete logic systems, computers and programmable controllers.

SERIAL COMMUNICATIONS INTERFACE

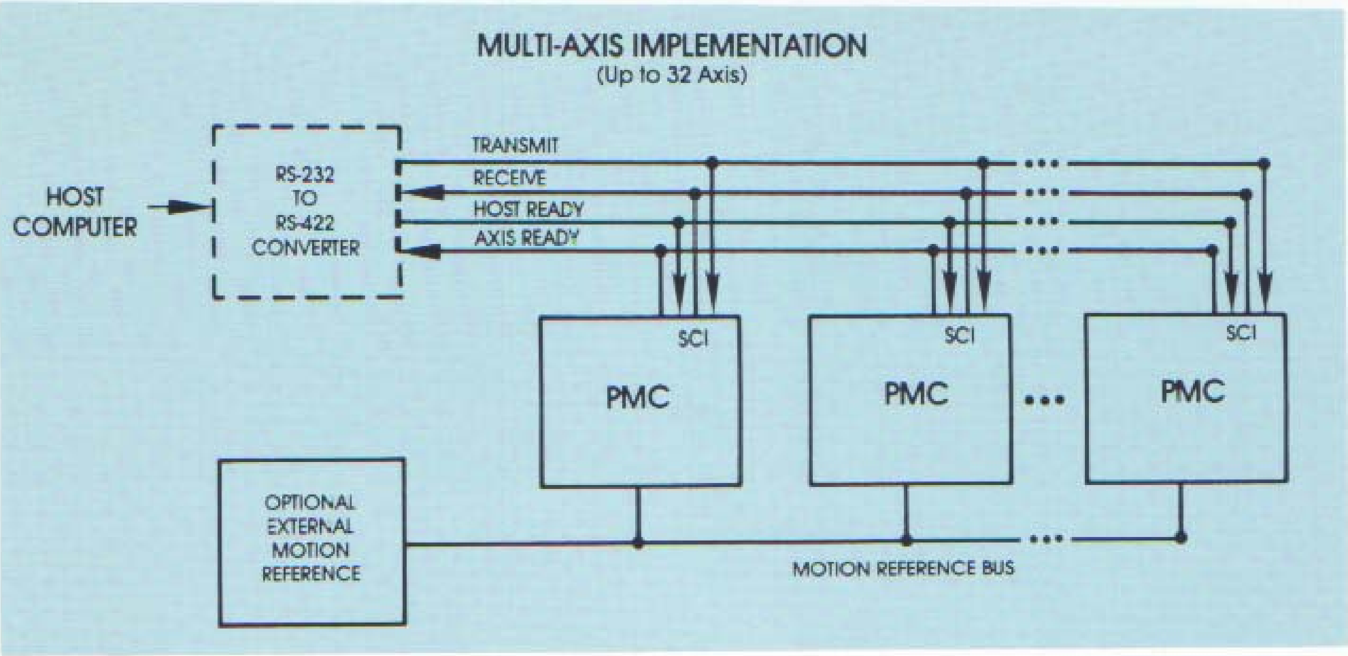
INTERFACES WITH RS-232, RS-422/449 DEVICES

The PMC interfaces with any RS-232 or RS-422/449 device. It is shipped from the factory strapped for RS-232 DCE with no handshake to work with a standard terminal. This interface may be used for manual programming only or for control under a host system.

The PMC automatically adjusts its baud rate over a range from 19.2k baud to 300 baud when communications is initialized.

SIMPLE TO INTERFACE WITH A HOST COMPUTER SYSTEM

A PMC-based system operates



as an intelligent slave serving a host computer when it is interfaced through the Serial Communications Interface.

To communicate with the PMC, the host sends a byte (character) to it and waits for that character to be processed. When the host completes a command by sending a command terminator, the PMC executes that command and returns a prompt character to the host signifying its completion.

Therefore, simply by sending ASCII characters to the PMC, the host can execute PMC positioning commands directly. Only high level commands and status requests are required from the host computer, since the PMC isolates it from the real-time servicing requirements of the servo system.

This allows a single microcomputer host to manage several high performance servomotor systems with ease.

With a PMC-901 or 902, which include programming, the host can also download a motion control program to the PMC or branch to a previously defined MPL routine in the PMC's motion program memory.

MULTI-AXIS SYSTEMS ON SINGLE SERIAL PORT

The PMC's RS-422/449 serial communications bus is unique and has a number of significant advantages for the customer who needs more than one axis of control. The bus allows PMCs to be separated by distances of more than 1,000 ft. (especially useful for distributed multi-axis systems) and up to 32 devices can be conveniently connected to the bus with simple, twisted pair or mass termination cable.

With this approach, any host computer implementing a serial communications channel with RS-422 interface can connect directly to the bus, or a host with an RS-232 serial channel can be connected to the bus with an inexpensive serial communications bus interface card. This is a true serial bus, not a daisy chain, and offers the interconnection, reliability and throughput advantages of a bus structure.

SYSTEM AXIS INTERFACE

MOTION REFERENCE BUS

This interface allows a PMC to output its position reference information for use by other "slave" PMCs. Slave PMCs can create motion which is "referenced" to these signals. This feature allows flexible and precise coordination of multiple axis systems.

Examples of these applications include registration systems, electronic cams, electronic lineshafts, flying shears, and digital speed draw systems.

DIAGNOSTICS

The PMC has a two color LED which is used as an output device for on-board diagnostics. Each time a PMC is reset, it signals the results of on-board diagnostics using the LED. During use, normal CPU operation is indicated by high frequency toggling of the LED between red and green.

FIELD INTERCONNECTION

Polarized, industry standard connectors create an effective interface for Machine I/O and Serial Communication. Field wiring to the encoder, power supplies, and servoamplifier is simplified by convenient, removable terminal block connectors.

MOTION PROGRAMMING LANGUAGE (MPL)		
Function	**Description**	**Commands**
SYSTEM SETUP COMMANDS	Selecting System Options	**N**ormalize **S**et/Show **T**une servo loops
MOTION PARAMETER COMMANDS	Defining Motion Parameters	**V**elocity **A**cceleration **I**ndex **J**og **H**ome
MOTION ACTION COMMANDS	Creating Motion	**J**og **G**o **I**ndex **H**ome
PROGRAM BUFFER COMMANDS	Entering or Editing A Program	**P**rogram **@** Labelling Routines
PROGRAM CONTROL COMMANDS	Utilizing Subroutines & Creating Complex Motion Control Applications	**B**ranching **L**ooping **F**unction Call **E**xit
SYNCHRONIZATION/ INTERFACE COMMANDS	Synchronizing Motion	**D**elay **U**ntil **,** & **;** Terminators
AUXILIARY OUTPUT COMMAND	Manipulating Machine Outputs	**O**utput

MOTION PROGRAMMING LANGUAGE (MPL)

ORMEC's Motion Programming Language (MPL) provides the systems designer with a versatile tool for writing programs that create high performance motion applications. A manageable, yet complete, number of motion commands can be combined to create complex motion—in much the same way that a calculator can be programmed to compute complicated mathematics. (See the language architecture diagram for MPL's range of programming options.)

Single letter commands (**A**,**V**,**I**,**J**, etc.) set motion parameters such as **A**cceleration rate, top **V**elocity, **I**ndex distance and **J**og speed. These parameters are stored in a buffer for use when a motion is initiated.

MPL utilizes variations of the same commands to create motion: **I**ndex a distance relative to the current position; **G**o to an absolute position; **J**og at the current jog speed; and move at **H**ome speed to the nearest encoder reference or machine sensor.

These commands may be executed interactively by the user from a programming terminal, or by a host computer which is directing the activity of the PMC. Alternatively, the same commands can be combined in step-by-step routines to configure a sophisticated motion control solution.

The PMC can create motion while continuing to process MPL commands. This powerful feature allows the PMC to synchronize output signals with motion or configure the next motion before the current one is complete. Commands are provided for synchronizing the MPL program to the motion or **D**elaying a specified time interval.

Other commands are provided for: delaying **U**ntil a specific input condition is true before executing the next command; conditionally **B**ranching to a program in the program buffer; **L**ooping to a program segment a specified number of times; and calling a **F**unction (subroutine) from the program buffer.

In addition, commands are provided for enabling several advanced features such as: changing the speed and acceleration range and resolution; synchronizing a PMC to external motion such as a variable speed machine; synchronizing a PMC's motion (acceleration, deceleration or stop) to a high speed machine sensor; choosing an alternate acceleration curve such as "s-curve for softer starts and stops; and reconfiguring host communications to binary or ASCII hexadecimal.

As the program is written or edited, it is stored in motion control memory. MPL efficiently utilizes this 2k byte memory, allowing each individual application productive use of program space. For example, a routine to create 10 equal indexes can use as little as 16 bytes.

Turn to the back page to see how MPL programming is applied in a high performance Laser Milling application.

TECHNICAL SPECIFICATIONS

POWER SUPPLIES

+5VDC	1.6A max.
±12VDC	0.15A

SERIAL COMMUNICATIONS INTERFACE STANDARDS

EIA RS-232C or RS-422. standard strapping is for RS-232C (DCE) with no flow control

Autobauds at rates of 19.2K, 9600, 4800, 2400, 1200, 600, or 300 baud

ENVIRONMENTAL SPECIFICATIONS

Operating Temp: 0 to 70 degrees C
Storage Temp: -25 to 125 degrees C
Relative Humidity: 0 to 90% (w/o condensation)

MECHANICAL SPECIFICATIONS

Max Size: 14.5" x 7.0" x 0.8"
Max Weight: one pound

CENTRAL PROCESSING UNIT

type	8085A
speed	3.072 MHz

MPL STORAGE (optional)

PMC-901	2K bytes EEPROM
PMC-902	2K RAM

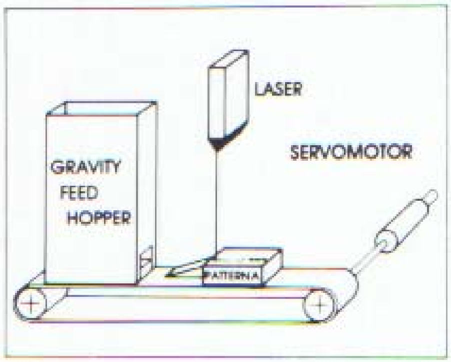
ORDERING INFORMATION

MODEL NO.	DESCRIPTION
PMC-900	P.C. board (6.9" x 14.5") with standard indexing firmware but no programming option (for use under direct host computer control only); requires regulated +5VDC at 1.6A; ±12 VDC at .15A if the RS-232 communications is used.
PMC-901	PMC-900 with Motion Programming Language (MPL) and 2k EEPROM non-volatile memory for stand-alone operation. This model operates in three configurations: computer hosted, programmable controller hosted and stand alone.
PMC-902	PMC-900 with standard indexing firmware and Motion Programming Language (MPL) and 2k bytes RAM memory. For use under direct host computer control only.

Phone or write your ORMEC representative for information on other modules or complete motion control systems.

ORMEC SYSTEMS CORP. • 19 Linden Park • Rochester, NY 14625 • 716-385-3520

LASER MILLING APPLICATION



The power and flexibility of ORMEC's Motion Programming Language (MPL) becomes readily apparent when we investigate an actual application.

XYZ Company has products A and B which are laser milled with two distinct patterns. Pattern A includes three equally spaced holes, a 1/2" slot and three more holes, while Pattern B has both sets of three holes but omits the slot. Pattern selection should be fully programmable, allow simple changeover from Pattern A to B and operate with a minimum of operator assistance.

An MPL routine for this application offers a straightforward approach to achieving these objectives.

The program moves the bar stock 1.500" from the material hopper to the starting position, and calls Function "X" which mills the first three hole pattern. The PMC activates the laser, milling three holes precisely .250" apart.

Using an input from a switch or sensor, the program selects either Function "Y" or "Z" to continue the milling operation.

Function "Y" reduces the velocity and activates the laser to mill the .500" center slot. Function "Z" omits the slot and moves to the location for the final three holes.

After recalling Function "X" to repeat the process of drilling the three-hole pattern, the system returns for the next piece. This application demonstrates how MPL, with its terse, intuitive program command structure, provides the ability to easily combine command functions in a general manner to provide a wide variety of non trivial motion control solutions.

This particular program illustrates the PMC's outstanding performance. The .250" moves require only 32 milliseconds and the speed regulation while milling the slot is within .1%. Since this program requires only 150 bytes, the PMC's 2048 byte program space is adequate for significantly more complex applications. The power of MPL is further illustrated by the capability of changing the 3 hole pattern to 100 holes by adding only one character to program space.

MPL	COMMENT
@Laser	Label Laser milling routine "L"
V30.0	Set Velocity to 30.0 in/sec (30.0 kHz)
A1000	Set Accel rate to 1000 in/sec^2 (1000 Hz/msec)
U1	Wait Until Run Switch (Input 1) is on
G1500+	Go to absolute position +1.500" (+1500 counts)
D,	Delay until conveyor stops
FX	Mill 3 hole pattern by calling Function "X"
FY2	Mill slot for Product A by calling Function "Y" if Input 2 is on
FZ-2	Skip slot for Product B by calling Function "Z" if Input 2 is off
FX	Mill 3 hole pattern by calling Function "X"
GO+	Return for next piece (Go to absolute position O)
D,	Delay until conveyor stops
BL	Branch to "L" to repeat routine
@X	Label Function "X"
O1	Activate laser by asserting Output 1
D100	Delay 100 milliseconds for drilling operation
O0	Deactivate laser by clearing Output 1
V15.0	Set Velocity to 15.0 in/sec (15.0 kHz)
I250+,	Index forward .250" (250 counts); wait until conveyor stops
LX2	Loop to Function "X" twice
E	Exit "X" and continue with routine "L"
@Y	Label Function "Y"
I750+,	Index forward .750" (750 counts); wait until conveyor stops
V2.0	Set Velocity to 2.0 in/sec (2.0 kHz)
O1	Activate laser by asserting Output 1
I500+,	Index forward .500" (500 counts); wait until conveyor stops
O0	Deactivate laser by clearing Output 1
V15.0	Set Velocity to 15.0 in/sec (15.0 kHz)
I750+,	Index forward .750" (750 counts); wait until conveyor stops
E	Exit "Y" and continue with routine "L"
@Z	Label Function "Z"
V30.0	Set Velocity to 30.0 in/sec (30.0 kHz)
I2000+,	Index forward 2.000" (2000 counts); wait until conveyor stops
E	Exit "Z" and continue with routine "L"

*A PMC can be programmed from any ASCII device