JULY/AUGUST 1982 VOL. 4 NO. 4 $3.00 USA/$3.50 CANADA

THE JOURNAL OF INTELLIGENT MACHINES

ROBOTICS AGE™

THE MICROBOT TEACHMOVER

John W. Hill, Ph.D., Vice President
and
Clement M. Smith, Research Engineer
Microbot, Inc.
453-H Ravendale Drive
Mountain View, California 94043

Until now, operating any of the low-cost tabletop robot arms on the market required the use of an external computer. If you didn't already own a computer, this increased your initial cost. And even if you did already own a computer, there was another drawback. Programs for robot arms had to be written in terms of motor steps, rather than actual arm motions. For example, to command the arm to reach for an object, the user couldn't simply specify where in space the arm should go, but had to figure out in advance which arm joints had to be rotated and by exactly how much. This constraint has proved awkward for many a user, thus defeating one of the major purposes of these low-cost arms: robotics education and investigation.

The TeachMover, designed and manufactured by Microbot Inc., is a tabletop robot arm which needs no host computer. It comes with a built-in microprocessor and a hand-held teach control which lets the user operate the arm and record arm motions without computer programming (photo 1). The teach control itself (photo 2) has virtually all the capabilities common to the control boxes used on large industrial robots costing upwards of $40,000.

The TeachMover allows the user inexpensive, hands-on experience with little initial training. The basic operations are simple. The unit comes with a complete user manual and with a demonstration program stored permanently in system firmware. The TeachMover is exceptionally safe to use, even by novices, since it operates at low power and since there is virtually nothing the user can do to damage the unit by using the controls incorrectly.

Control by an external computer does, of course, have its place in industrial robotics. We have incorporated two RS-232C asynchronous serial communication lines so that the TeachMover arm can be controlled by a host computer or computer terminal as an alternative to the hand-held teach control. Even this mode of operation is unusually simple; only six commands are needed to accomplish all possible program options and arm motions when the TeachMover is in the serial interface mode. Additional commands allow programs to be downloaded from a host computer to the robot memory, or uploaded from the robot to a host. The latter capability means the user can develop arm motion programs on the teach control and then save them on disk.

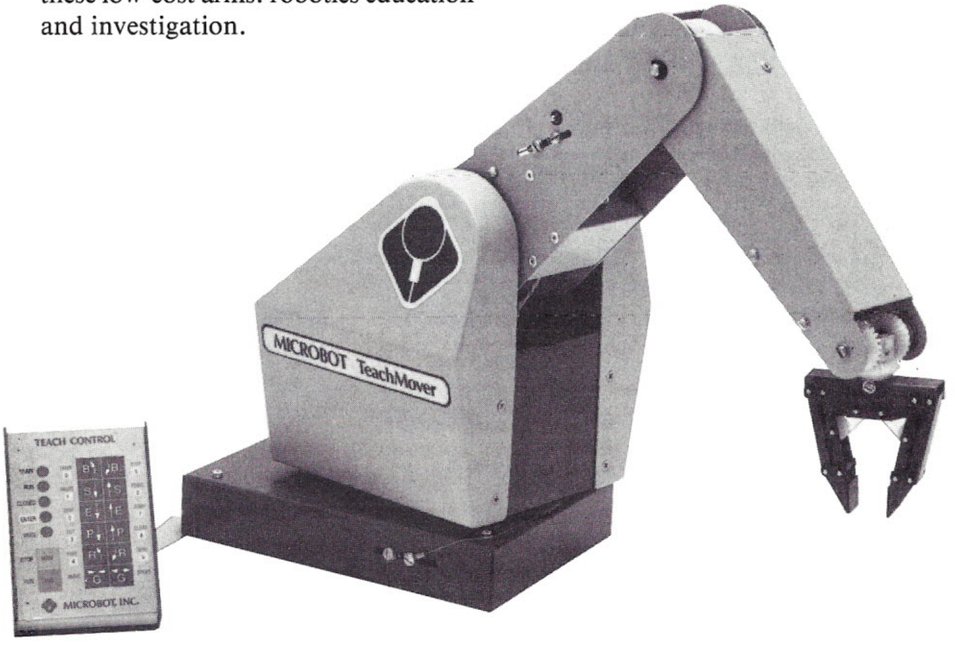


Photo 1: The TeachMover consists of a hand-held control box and a five-jointed, cable-driven mechanical robot arm. The arm, which has a microprocessor in its base, can also be controlled from an external computer using six simple commands.

On-Board Computer and Interface: The base of the TeachMover arm houses a 6502A microprocessor, which is an eight-bit, 2 MHz chip used to coordinate joint motions of the arm and handle all input and output. The 6502 is the microprocessor used in the Apple, Atari, and PET computers.

All internal system software programs are stored in 4K bytes of ROM. User-generated arm motion programs are stored in 1K of programmable memory; these programs can be up to

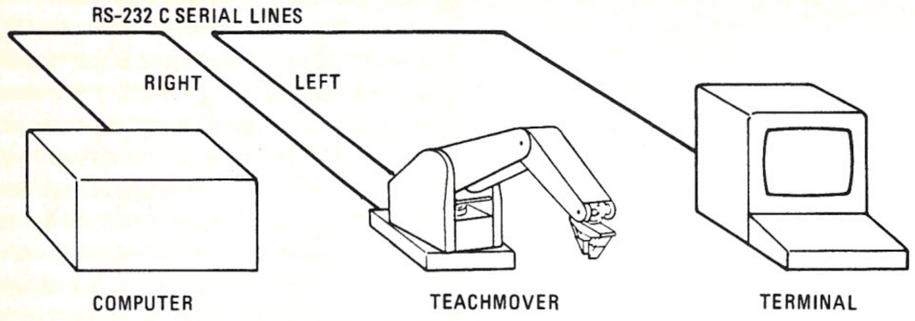


Figure 1: The TeachMover can be placed in series with a host computer and a terminal or printer. Signals from the terminal or printer to the host computer pass through the TeachMover unchanged. Signals passing in the other direction also pass unchanged, unless they are preceded by an @ sign. These latter signals are interpreted as TeachMover commands.

53 steps long. Memory can be expanded to extend the maximum program length to 126 steps.

The right-hand serial port in the base of the arm is used for interfacing with a host computer. The left-hand port is for a terminal or printer. Signals from the left-hand port always pass through to the right-hand port unchanged. The same is true for signals passing from the right port to the left port, unless these signals are preceded by an "at" sign (@), in which case, the signals are interpreted as arm commands.

The left-hand port is useful when the host computer itself has only one serial port. In such cases, the computer can still interface with the TeachMover without losing its ability to run a remote printer or terminal. This is accomplished simply by placing the TeachMover in series with the other peripheral (figure 1).

The rate of serial transmission may be set to 110, 150, 300, 600, 1200, 2400, 4800, or 9600 bps by means of switches located on the computer card inside the base of the TeachMover (photo 3).

Also included in the base of the unit is an auxiliary parallel I/O port which permits the user to interface the TeachMover to external equipment via a 16-conductor flat ribbon cable. Five user output bits may be set to logical 1 and cleared to logical 0 under program control to turn other equipment on or off when a given motion is complete. Seven user input bits may be read to initiate an arm sequence when a given external condition is met. These inputs can come from external switch closures or from TTL logic signals generated by a host computer.

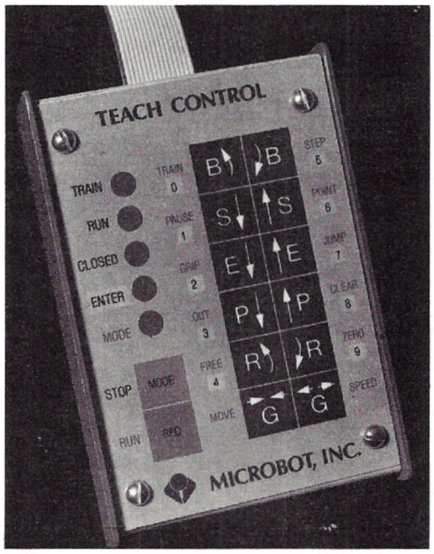


Photo 2: Using only 14 keys and five indicator lights, the teach control allows the user to independently move all arm joints; compose, step through, and edit arm motion programs; enter constants for speed of arm and duration of pauses; turn external switches on or off; and perform conditional branching based upon the state of the grip switch or specifiable external inputs.

Cable-Driven Design: The TeachMover arm (figure 2) is virtually identical to the arm on Microbot's MiniMover-5 robot (see "Introducing MiniMover-5," *Robotics Age*, Summer 1980, Vol. 2, No. 2). There are six drive motors: one each for base, shoulder, and elbow joint; one for hand opening; and one each for the left and right wrist gears. The wrist mechanism is a differential. When left and right wrist gears move in the same direction, the wrist motion is that of "pitch." When they move in opposite directions, the motion is "roll."

Many robot arms are designed with several of the drive motors mounted on the extension members (hand, forearm, and so on). This places an extra load on these members and increases their inertia. This, in turn, necessitates that the motors which move these members be stronger and more expensive than they would otherwise need to be.

In designing the arm for the MiniMover-5 and TeachMover, we decided to place all the drive motors in the body and to use a system of cables to manipulate the members. Our design represents an adaptation and refinement of the "tendon technology" used in aircraft, high-speed printers, plotters, and manipulators used in handling nuclear materials in "hot cells." To further reduce the number of mechanical parts in the arm, we placed all six drive gears on the same shaft. Putting all the motors in the body gave us an added bonus. The arm is very stable and does not have to be bolted down to keep it from tipping.

The drive gears are all operated by stepper motors which are driven under microprocessor control by digital power ICs (integrated circuits). We decided on stepper motors because of their low cost and the ease with which they can be controlled from a computer. We designed the drive system so that one motor step produces an incremental motion of .011 inch (0.25 mm) or less at the tip of the fingers. This incremental motion is hardly visible.

The cabling in the TeachMover is designed so that rotation of the shoulder or elbow joints does not change the orientation of the hand in space. Thus, if the hand is holding a container of fluid and the shoulder or elbow rotates, the fluid won't spill. Manually rotating the elbow does, however, affect the hand opening. Preventing this would require some complex and unwieldy cabling. We instead compensated for the effect by designing the firmware to automatically uncouple hand opening from elbow rotation. In the serial interface mode, the user needs to program the uncoupling, but this is accomplished through a simple formula.

A hand closure switch interfaced to the robot computer permits intelligent robot behavior. The cable that operates the mechanical hand passes over a

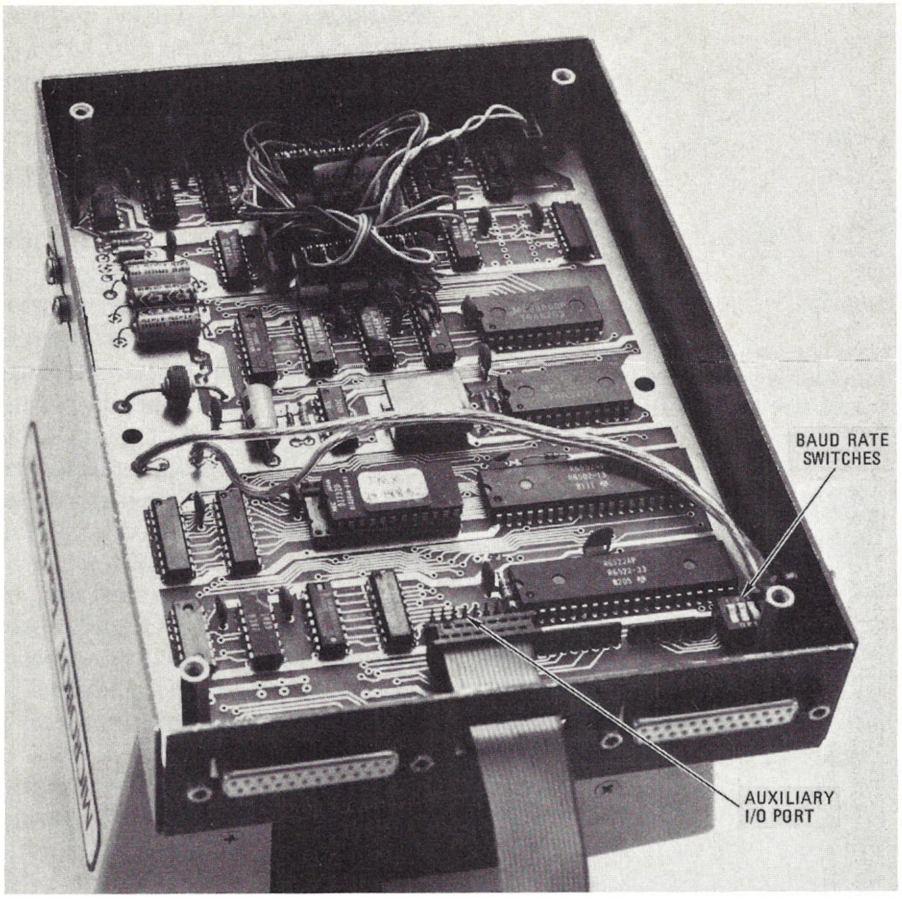


Photo 3: The computer card inside the base of the TeachMover contains the connector for the 16-conductor I/O parallel port cable and a set of switches which allow the serial transmission rate to be set to any of eight standard values from 110 to 9600 bps.

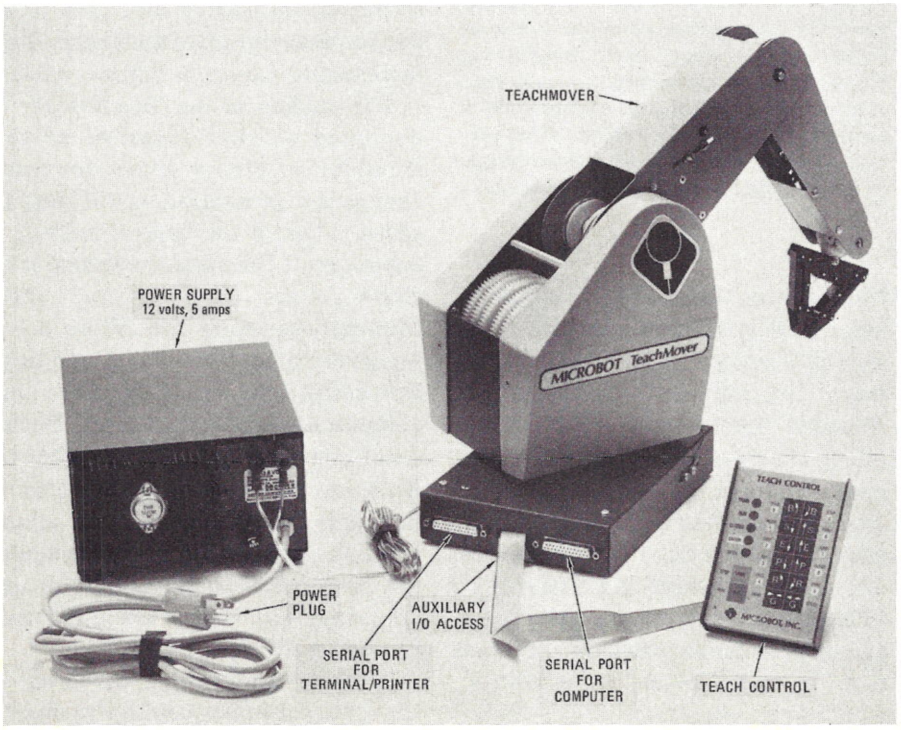


Photo 4: The base houses a 6502A microprocessor, two RS-232C serial communication interfaces, an auxiliary input/output parallel port, and connectors for the teach control and power supply.

switch located in the upper arm member (figure 3). This switch senses cable tension when the hand has closed either on an object or on itself. This switch allows the user to write programs to measure the size of objects, differentiate between objects of different sizes, and determine the presence of objects in the gripper. User control via the grip switch is supported by both the teach control (through a conditional branch command) and the RS-232C serial port.

Color-Coded Controls: Industrial teach controls vary in sophistication from devices which can only rotate the various joints to those with alphanumeric displays, diagnostic messages, and keypads that let the user specify arm motion via Cartesian coordinates. After weighing all the tradeoffs between features and costs, we decided to include the following capabilities:

- independent movement of each joint by means of labeled control keys
- recording of movement positions to create a step-by-step, repeatable program
- ability to move the arm through a program one step at a time
- program editing
- specification of speed of arm and duration of pauses
- conditional branching based on the state of the grip switch or other external inputs
- ability to turn external outputs on or off (for example, to control other machinery)
- ability to manually position arm when power is off

In addition, we wanted to implement all these functions with a minimum number of control keys, again to keep product cost as low as possible. We decided to use three colored "overlays" so that the same keys could be used (1) to select functions, (2) to input numeric values, and (3) to position the arm. Rather than use an expensive alphanumeric display to indicate which overlay is in use, we developed a simple system of color-coded indicator lights and key labels.

When the red MODE light is on, the words printed in red apply to the keys, and the user can select from TRAIN, STEP, PAUSE, RUN, and other functions. When the yellow ENTER light is on, the yellow numerals next to the keys apply, and the user can enter numerical values. The labels printed on the keys themselves apply when the teach control is in the TRAIN mode (or in the MOVE mode, as explained later).

Control Functions: The TeachMover incorporates 13 different control functions. They work as follows:

TRAIN and *RUN*: When the TeachMover is first powered up, the teach control is in the TRAIN mode, and the green TRAIN light is on. In the TRAIN mode, each of the key pairs B, S, E, P, R, and G control one of the joint motions of the arm (B = base, S = shoulder, E = elbow, P = pitch of wrist, R = roll of wrist, and G = grip). The keys in the right-hand column cause these joints to move in the direction indicated on the key face, while those in the left-hand column cause them to move in the opposite direction.

When the user presses one or more of these keys to move the arm to a desired position, he can record this position as a program step by then pressing the REC key. Up to 53 steps may be programmed; these steps are internally numbered 0-52. Pressing the REC key overwrites the current program step and then increments an internal sequence pointer so that the TeachMover memory is ready to record the next step. To run a program, the user first presses the MODE key (to exit from TRAIN) and then presses RUN.

PAUSE: The user can program pauses into a program by pressing the MODE key, if the MODE light is off, and then the PAUSE key. Once PAUSE is pressed, the yellow ENTER light comes on, and the two rows of keys may be used to enter numerals according to the yellow labels. The user simply enters a number (0 to 255) equal to the length of the desired pause in seconds, then presses the MODE key again. The PAUSE command is saved as a program step, and the sequence pointer is incremented. When the program runs, the arm pauses for the desired number of seconds.

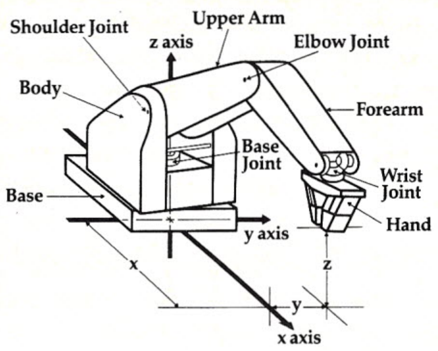


Figure 2: The TeachMover arm consists of a base, body, upper arm, forearm, and hand, separated by joints labeled as shown. To reduce the load at the extremities, all motors are mounted in the body.

SPEED: This command allows the user to change the speed of the arm. The SPEED command is not saved as a program step nor does it cause the sequence pointer to change. Issuing this command simply causes all subsequent recorded steps and manual motions to be executed at the commanded speed.

After the SPEED key is pressed, the yellow ENTER light comes on, and the user may enter any number from 0 to 15. Zero is the slowest speed, and 15 is the fastest. The TeachMover is always initialized to speed 6 when it is turned on. The correspondence between speed numbers and steps per second of the drive motors is given in a table in the user manual.

STEP: This command is useful during program development, for it allows the user to move the arm through a program one step at a time.

JUMP: This command allows the user to write highly sophisticated programs, for it provides for conditional branching. When the JUMP key is pressed, the yellow ENTER light comes on and the user enters two numeric values, pressing the MODE key in between. The first value represents the jump condition, and the second is the sequence pointer number (step number) to jump to if the jump condition has been met. The jump conditions are as follows:

- condition 0: grip switch is open
- condition 1-7: user input bit 1-7 is on (that is, set to 1)
- condition 8: never
- condition 9: always

For example, to cause the arm to go to step 5 in a program on the condition that the grip switch is open, one would key in: JUMP 0,5. Input bits 1-7 can be set in a variety of ways — for example, by sensor switches the user might mount on the TeachMover's fingers, by switches on external machinery, or by signals generated in a host computer program.

POINT: This frequently used command is similar to an unconditional jump. POINT 6, for example, means to go to step 6 in the program and proceed from there. Unlike the JUMP com-

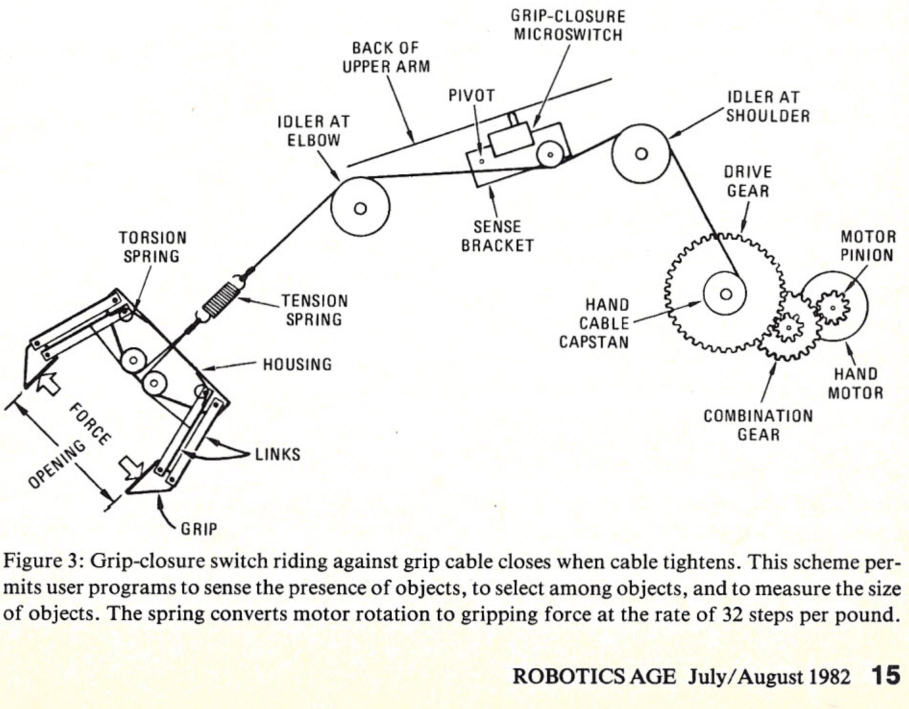


Figure 3: Grip-closure switch riding against grip cable closes when cable tightens. This scheme permits user programs to sense the presence of objects, to select among objects, and to measure the size of objects. The spring converts motor rotation to gripping force at the rate of 32 steps per pound.

mand, however, POINT does not create a program step. POINT is used simply to move to a given step in an existing program; it can be invoked even in the middle of program execution by simply pressing the MODE key first.

The POINT command is especially useful during program editing. To change a program step, the user need only POINT to the step in question, press the TRAIN key, use the joint control keys to achieve the desired new position, and then press REC. The user can also STEP through the program until the arm attains the position to be modified, then use TRAIN as before, except that stepping to a JUMP executes the JUMP.

The POINT command can also be used to execute multiple programs stored in memory. Program 2 in figure 4 for example, can occupy memory steps 21 through 31, with step 31 being an unconditional JUMP back to step 21. Program 2 can be executed simply by issuing the command POINT 21 prior to pressing the RUN key. Program 3 can be executed in a similar fashion. The POINT command is useful to run the demonstration program that is permanently stored in TeachMover firmware at location 126.

The POINT command has still another function. In creating a program, it is sometimes desirable to copy a program step; that is, key in a program step which causes the arm to move back to a previously achieved position. To do this, the user need only STEP through the program until the desired position is achieved. POINT to the program step to which that position is to be copied, then press TRAIN and REC. This duplicates the desired position at the desired program step.

CLEAR: This command is activated by first pressing the MODE key and then, while holding down the MODE key, pressing the CLEAR key. This clears all recorded positions and operations in program memory and sets the sequence pointer to step 0.

GRIP: This command causes the gripper to close by proceeding 32 motor steps past the point at which the grip closure switch is activated. This builds up about 1 pound of gripping force.

MOVE: This activates the joint control keys used in TRAIN mode but does not change the internal position registers or allow a position to be recorded. The MOVE command is useful in moving the arm back to a known position in the event of motor slippage or mechanical interference from an external obstacle.

FREE: This turns off all motor currents, allowing the arm to be positioned manually. Positioning the arm by hand is often faster than using the keys.

ZERO: This command is activated by first pressing the MODE key and then, while holding down the MODE key, pressing the ZERO key. This sets the TeachMover's six internal position registers (one for each motor) to zero and sets the sequence pointer to zero. The ZERO command can be used for program initialization.

OUT: This command allows the user to output a binary logic 0 or 1 on up to five output lines or to turn lights on the teach control box on and off, based on arm positions achieved or conditions met. The OUT command must be followed by two numerical entries; the first is an output number, and the second is a 0 or 1 (off or on, respectively, in the case of the teach control lights). The output numbers are as follows:

- output 0: the MODE light
- output 1-5: user outputs on the I/O connector
- output 6: the TRAIN light
- output 7: the RUN light
- output 8: the ENTER light

TeachMover

Description	Includes teach control, power supply and RRA-2 user manual
Configuration	Five revolute axes and integral hand
Drive	Electrical stepper motors — open loop control
Controller	Microprocessor with 4K bytes of EPROM and 1K bytes of RAM located in base of unit
Interface	Dual RS-232C asynchronous serial communications interfaces (data rate is switch selectable between 110, 150, 300, 600, 1200, 2400, 4800, and 9600 bps)
Program Languages	ARMBASIC through serial port
Teach Control	13-function keyboard
External I/O	5 output and 7 input bits under computer control
Power Requirements	12 to 14 volts, 4.5 amps DC (cont.)
Cable	Teach control cable length 3 ft. (900 mm)
Payload	1 lb. (454 gm) max. at full extension
Resolution	0.011 in. (0.25 mm) max. on ea. axis
Gripping Force	3 lbs. (13N) max.
Reach	17.5 in. (444 mm)
Static Load Force	4 lbs. (18 N) max.
Arm Weight	8 lbs. (4 kg)
Positioning Accuracy	±.030 in.
Speed	(0-6.5 ips) 16 selectable movement rates via teach control
Gripper Opening	3 in. max. (0-75 mm)
Control Method	PTP (Point to Point)
Programming Capacity	53 stored positions
Base Motion	±90°
Shoulder Motion	+144°, −35°
Elbow Motion	+0°, −149°
Wrist Roll	±180°
Wrist Pitch	±90°
Gripper	Included
Power Supply	Included. 105-125 VAC input, 13.8 VDC output, 5 amps. 220 VAC also available.
Application Manual	Included with purchase, also sold separately
Plastic Carrying Case	Optional

Table 1: TeachMover performance characteristics.

Join the robotics revolution now!

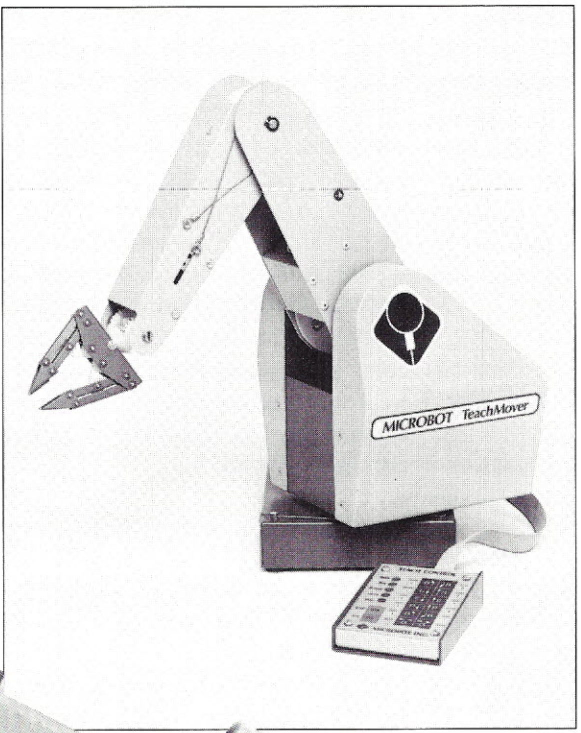
With TeachMover, the low-cost, self-contained developmental robotic arm.

Microbot's new TeachMover robotic arm brings unique freedom to robotics development with its exclusive 13-mode "teach control." This simple set of buttons lets you program complex routines with ease—without having to learn languages or enter long commands. No other comparable unit has anything like it.

Priced at only $2,395, the TeachMover is ideal for engineers, educators, and hobbyists alike. It has:

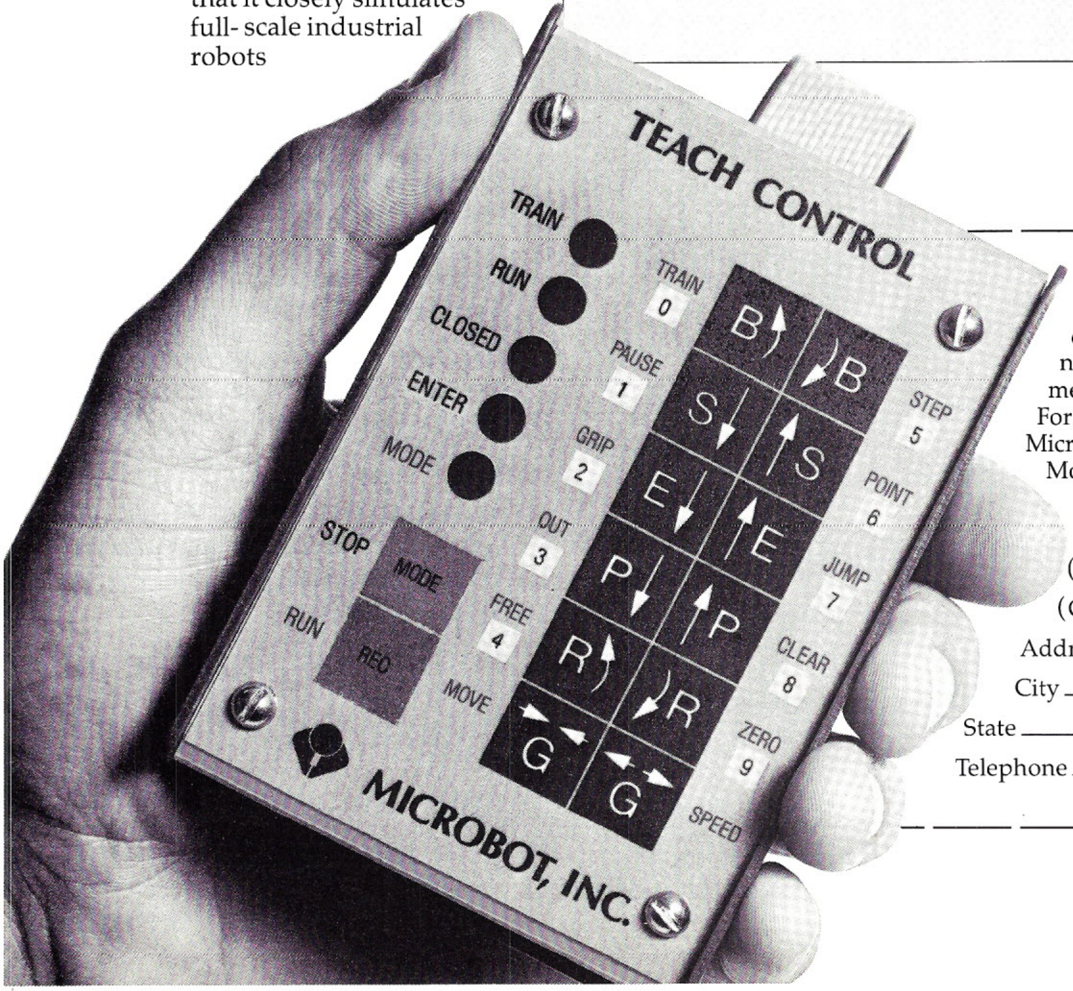
- Five axes of movement so that it closely simulates full- scale industrial robots
- An onboard microprocessor with RS232C interface to allow operation with most computers
- A built-in "intelligent" gripper that can sense objects and their size

Joining the robotics revolution is easy with the TeachMover from Microbot. For more information on the TeachMover and other Microbot products, call toll-free (800) 227-8909 for the name of the Microbot representative nearest you. In California or outside the continental U.S., call collect (415) 968-8911.



MICROBOT

Microbot, Inc.
453-H Ravendale Drive
Mountain View, CA 94043



Join the Microbot user group!

Share your applications, software development projects, and ideas in a newsletter offered free through no-cost membership in the Microbot user group. For details, fill out and mail this coupon to Microbot, Inc., 453-H Ravendale Drive, Mountain View, CA 94043.

Name ______________________

(Title) ______________________

(Company) ______________________

Address ______________________

City ______________________

State ____________ ZIP ____________

Telephone ______________________

RA

Sample Programs: A "pick and place" program which uses just the joint-control keys is shown in listing 1. Once the user masters the basic arm motions through creating simple programs like this, he can progress to more complex tasks. One such task is flowcharted in figure 5. Here, two different blocks are placed in predetermined locations. (P1 and P2 as shown in figure 6.) The arm does not know in advance which block is larger. The task is for the arm to locate the larger block, move it to a new location (P3), and stack the smaller block on top of it (location P4). Listing 2 shows how this is accomplished. Note how the JUMP command is used for conditional branching.

Serial Interface Commands: When the TeachMover is linked to a host computer over serial transmission lines, six commands can be issued to allow complete control of arm functions. These commands are:

@STEP: This causes all six stepper motors to move simultaneously. The syntax of the @STEP command is:

@STEP (SP), (J1), (J2), (J3), (J4), (J5), (J6), (OUT)

(SP) is a value which gives the speed of motion, (J1) through (J6) are the number of steps that each of the six motors are to be moved (J1 = base joint, J2 = shoulder, J3 = elbow, J4 = right wrist, J5 = left wrist, J6 = hand), and (OUT) is a decimal number which translates into the binary pattern the user wishes to appear at the user outputs mentioned above. (Note: the speed values, SP, are not the same as those used with the teach control, but are related to the latter by a simple formula.) For example, to rotate the base joint 20 steps at a speed value of 50 without rotating any of the other joints or changing any of the user outputs, the command would be:

@STEP 50, 20, 0, 0, 0, 0, 0

or simply:

@STEP 50, 20

When the @STEP command is given, the TeachMover returns a 1 or a 0 to the host computer, depending on whether the command has been executed or not. The host computer can easily be programmed to test for this and to print INVALID COMMAND in the event of a 0.

Programming the TeachMover on a host computer does require knowledge of how rotation of the various motors affects motion of the arm itself. In addition, pitch and roll cannot be programmed directly, but only by specifying the proper coordinated motion of the left and right wrists, (J4) and (J5). A command to move the elbow joint affects the hand opening unless the user compensates by adding the elbow position to the hand position when the command is issued. Thus, if the desired number of steps for the base, shoulder, elbow, pitch, roll, and gripper are represented by B, S, E, P, R, and G, respectively, the correct motion command would be:

@STEP (SP), B, S, E, $(P+R)/2$, $(P-R)/2$, $E+G$

In designing the TeachMover, we programmed the firmware so that motion of the six motors is simultaneous rather than sequential. If unequal numbers of steps are specified for different motors, the firmware automatically coordinates the timing to produce smooth motion.

@CLOSE: This command causes the hand to close just until the grip switch is activated. The syntax is simply:

@CLOSE (SP),

The optional (SP) determines the speed of closing. The arm responds with 0 if there is a syntax error or with 1 if the gripper has closed.

@SET:This puts the arm into the TRAIN mode, activating the keys on the hand-held teach control. The syntax is:

@SET (SP)

This mode is terminated when the REC or MODE key is pressed on the teach control. If the REC key is pressed after

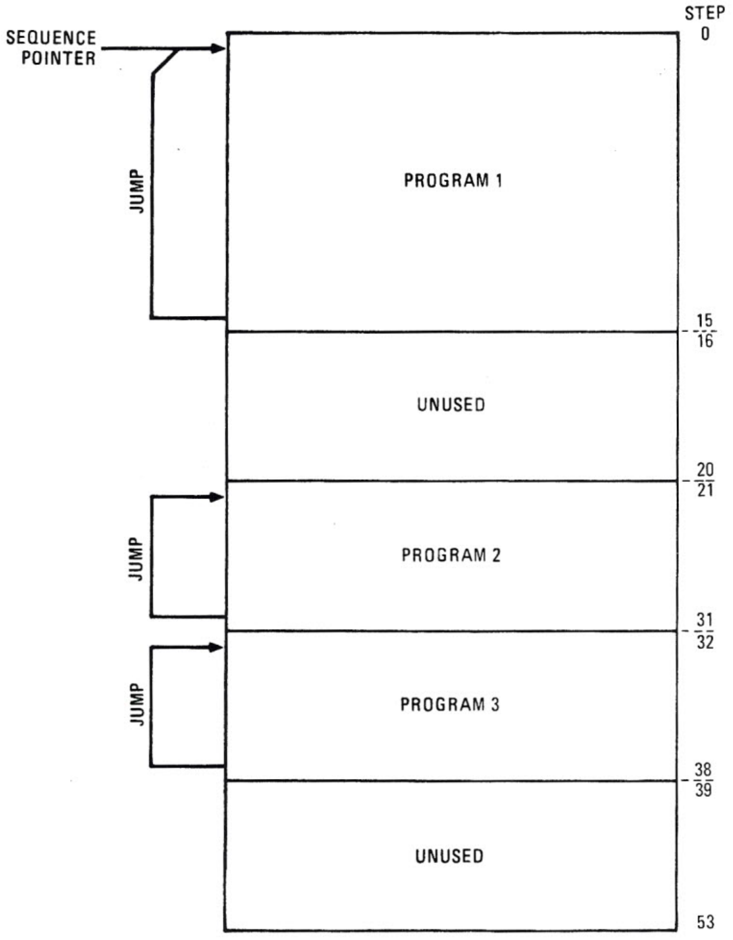


Figure 4: JUMP and POINT instructions can be used to store and execute several isolated programs within the TeachMover's 53-step memory.

an @SET command, no step is recorded. The @SET command gives the user the option of programming arm positions via the teach control even in the course of execution of a host-driven program. After terminating this command, the arm responds with a 1, or a 0 if there was a syntax error.

Three additional commands allow entire programs to be transferred between the host computer and the TeachMover:

- *@QWRITE (SS)*: Downloads a sequence step to the TeachMover.
- *@RUN (SS)*: Starts running a recorded sequence at step (SS).
- *@QDUMP*: Uploads a program to the host, after first converting the program to ASCII format. Once uploaded, the program can, of course, be stored on disk. Thus, the user can develop programs on the hand-held teach control and save them for later use and investigation.

Room to Experiment: The TeachMover allows the user to try simple experiments at first, then progress to more complex investigations as he gains experience. One interesting experiment involves connecting one of the user input bits to one of the output bits. This provides the user, in effect, with a one-bit memory: A condition can be read at one time and acted upon at a later time.

Sensors may be installed in the fingers of the robot and connected to the input bits to obtain more intelligent operation. Microswitches or optical (phototransistor-LED) sensing modules will permit the robot to sense objects before grasping them. An optical retro reflector will permit the robot to scan bar codes or to align itself using a

STEP	DESCRIPTION
0	Home position (gripper open)
1	Move right
2	Move down
3	Close gripper
4	Move up
5	Move left
6	Move down
7	Open gripper
8	Move up (to clear object)
9-52	(Null)

Listing 1: A simple "pick and place" program. With the teach control in the TRAIN mode, the user moves the arm from one position to the next, using one or more of the arm motion keys. The REC key is pressed after the arm achieves each desired position. When the program RUNs, the arm cycles through all the steps repeatedly until another command is given.

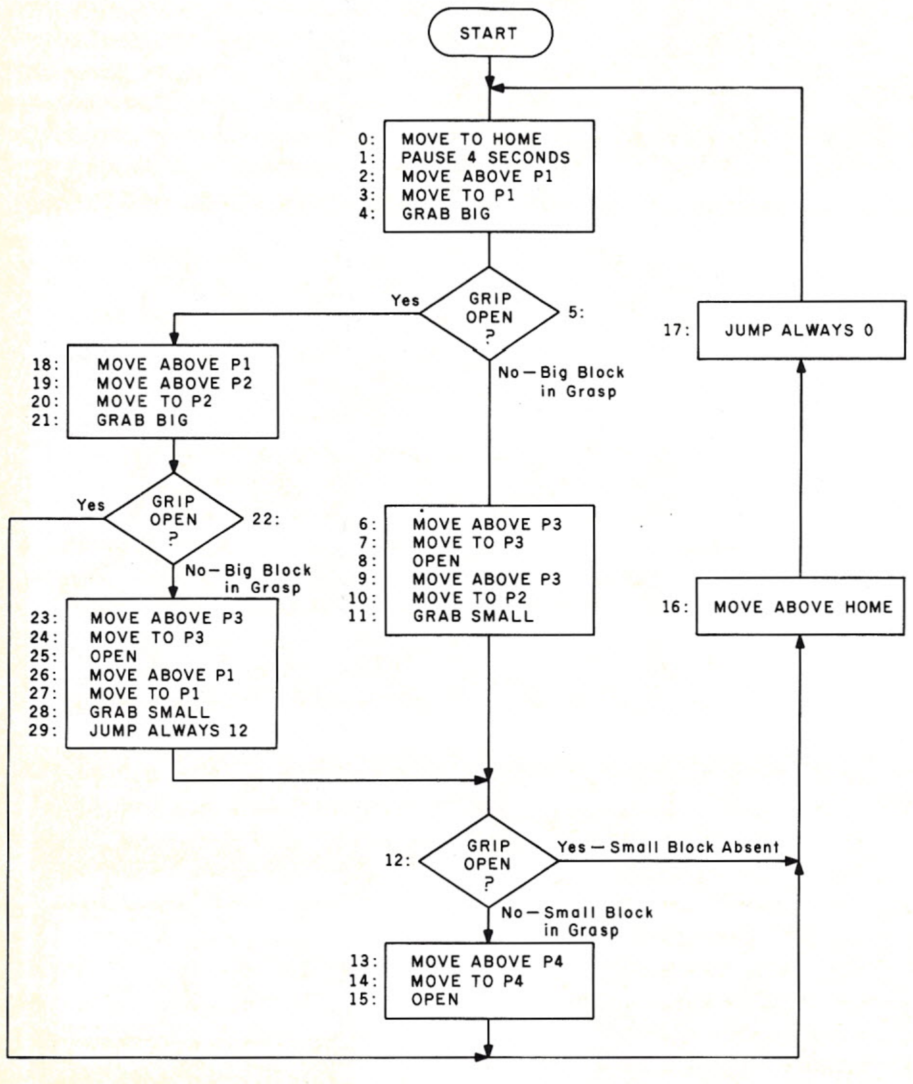


Figure 5: Flowchart for a block-stacking program. The arm reaches for one of two different blocks and determines whether it is the smaller or the larger one. If it is the larger block, the arm moves the block to a new location and then stacks the smaller block on top.

STEP	DESCRIPTION
0	Move to home
1	Pause four seconds
2	Move above P1
3	Move to P1
4	Grab big
5	Jump if grip open to 18
6	Move above P3
7	Move to P3
8	Open
9	Move above P3
10	Move to P2
11	Grab small
12	Jump if grip open to 16
13	Move above P4
14	Move to P4
15	Open
16	Move above home
17	Jump always 0
18	Move above P1
19	Move above P2
20	Move to P2
21	Grab big
22	Jump if grip open to 16
23	Move above P3
24	Move to P3
25	Open
26	Move above P1
27	Move to P1
28	Grab small
29	Jump always 12

Listing 2: This program accomplishes the task described in figures 5 and 6. The necessary conditional branching is accomplished by means of the JUMP instruction.

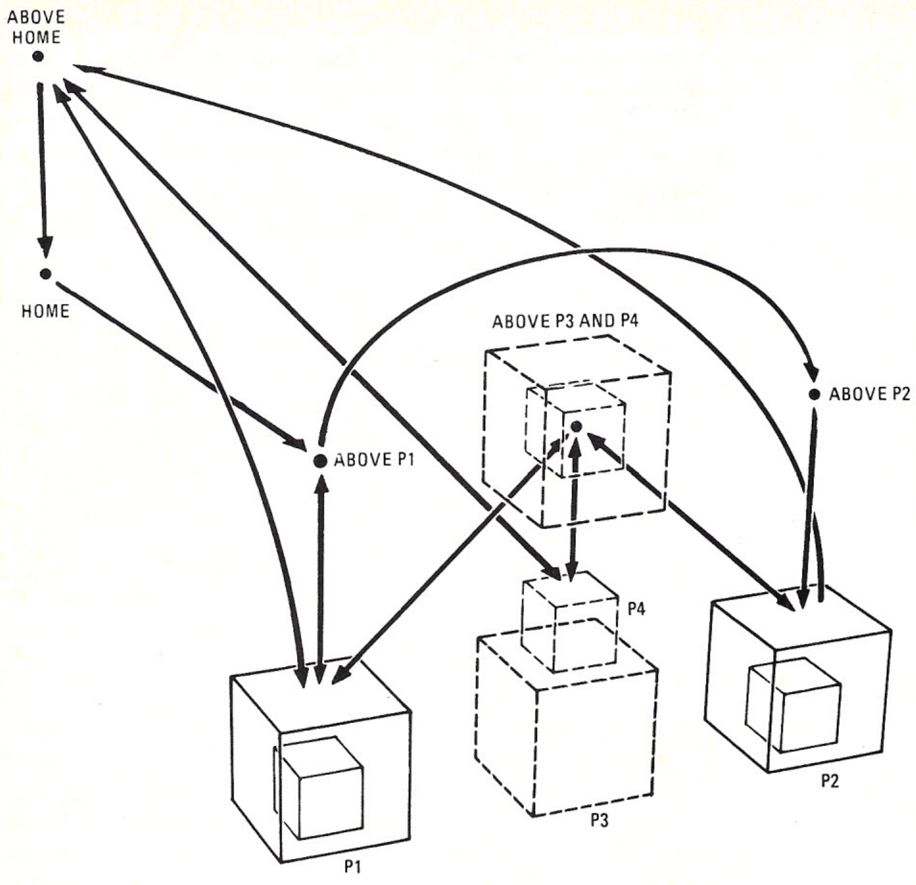


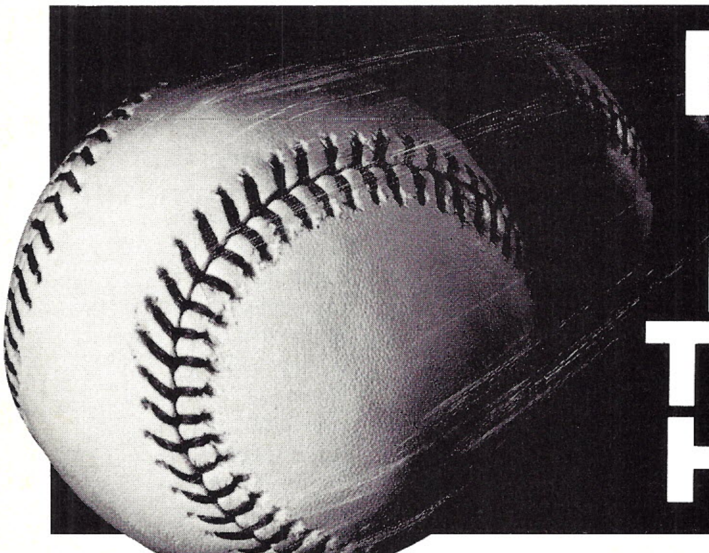
Figure 6: Definition of position referred to in block-stacking program with motion shown by arrows.

reflection target. A light beam from finger to finger also offers interesting possibilities.

Connecting two TeachMover arms together with a 16-conductor flat cable offers another possibility for investigation. Conditional branches can be used, for example, to cause one arm to take an object from the other arm.

Other Products: The TeachMover is available now from Microbot, Inc., 453-H Ravendale Drive, Mountain View, California 94043; telephone 415/968-8888.

Microbot is in the exclusive business of developing and manufacturing robots. In addition to robots for education and experimentation, Microbot is initiating manufacturing of small industrial robots that are compatible with the TeachMover. Soon to be released in a test marketing program is the Microbot Alpha, made entirely of industrial-grade parts. As compared with the standard TeachMover, the Alpha model will have both greater speed and greater lifting power. □



BEI'S H-25. THE ENCODER THAT PLAYS HARDBALL.

The machine tool environment is so harsh, many encoders never make it past first base. This means costly downtime and replacement.

The H-25 is specifically designed for abusive applications. Impervious to normal shock, vibration and temperature changes. Sealed against oil and water splash. Rugged cast housing. Heavy duty shaft. Class 7 bearings and more.

FOUR WEEK SHIP DATE on standard models means no change-ups or curves, just fast ball delivery. Two week priority service available. Competitive pricing makes it easy to have this star on your team. Call BEI about the H-25, the encoder built to play the toughest game in town.

BEI Electronics, Inc.

BEI

The Encoder Company.™

Industrial Encoder Division
7230 Hollister Avenue
Goleta, CA 93117 • (Tel) 805-968-0782

CIRCLE 4