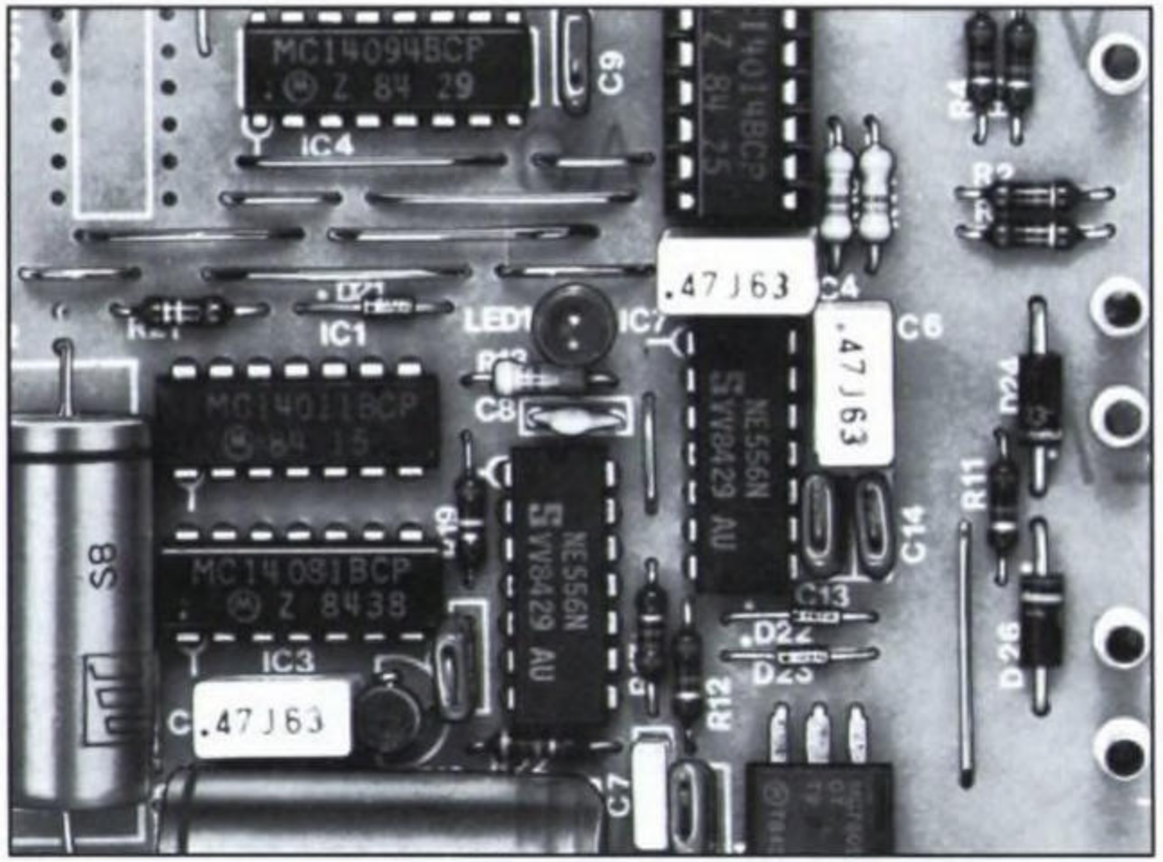
fischertechnik® COMPUTING

Programming/Kit-Building Instructions



Contents 2

Dear Friend of fischertechnik:

The fischertechnik Robotic Computing Kit has been developed to extend the range of applications for your computer by enabling it to control various models. With it you can explore the world of robotics and computer control of mechanical equipment, or solve actual problems of real-world industry by developing prototypes and solutions for production and process systems.

The kit contains two motors and a magnet, providing your models with arms, legs, and hands—the robot models discussed in this manual will show you what we mean. There are three lamps to indicate what actions are being undertaken by the computer, and eight switches and two potentiometers to give your models 'senses' to go with their limbs. The switches also serve to establish instruction arrays as your models 'learn' various tasks.

In addition to the robots, a wide variety of other models may be built. At first, you should attempt the simpler models like the traffic light, machine tool, and elevator. They will introduce you to the fischer building system and provide a step-by-step introduction into the world of fischertechnik computer control. Simple and clearly defined BASIC programs assist the newcomer, while providing a source of inspiration for the more experienced programmer.

The more extensive models have also been provided with sample programs so that they can be used immediately. The Plotter model and program will allow you to draw charts from graphic input, while the Graphic Panel will take input from your actions on the tablet and deliver this information to the computer as data. With a suitable graphics program, you can do paintings and drawings, graphic constructions, or set up a menu and select programs from the tablet.

The Solar Tracking System will allow you to keep a cell aimed at the sun for maximum energy output, while the Sorting System will differentiate between building blocks 30 and 15 and sort them into different bins.

In addition to the fischertechnik Robotic Computing Kit, you may use all of the fischertechnik components in building your models, giving much wider range to the possibilities you can create.

I am sure that fischertechnik computing will inspire you to perform further experiments on your own, and that it will extend your knowledge and experience in the fascinating world of computer control and modeling. I am also sure that it will provide endless hours of fun.

Yours sincerely,

Artur Fischer

fischertechnik Robotic Computing Kit—Parts List

31393 Turntable assembly

32126 Disk
32125 Disk
32124 Disk
32123 Disk
32122 Disk

31357 2-wire cable—
red/green

31229 Aluminum section 180

32119 Ribbon cable with
plug—20 wire

38069 Base plate—
259x187
(Not to scale)

38259 Mounting plate—
4 peg 30x30
38246 Mounting plate—
15x15
38241 Mounting plate—
2 peg 15x30
38242 Mounting plate—
2 peg 15x45
38244 Mounting plate—
2 peg 15x75

31006 Double-ended block 15
31005 Building block 15
31003 Building block 30
31004 Building block 30
with hole

31015 Flat hub
32117 Toothed ring
32116 Flat block 30
with keyhole

38240 Building block
V-15
38423 Angle block
10x15x15
31981 Angular block 15°
38309 Block 15—60°
31010 Angle block—
equilateral

31016 Winch drum
37636 Roller bearing
37468 Building block 7.5
37237 Building block 5

32121 Fiber pen

36443 Mini-screwdriver

31198 Axle 50
38413 K-axle 30
38414 K-axle 40
35404 V-axle 17
31982 Spring cam
31060 Link 15
31061 Link 30
31330 Link 45

Light cover
- 31316 Red
- 31317 Yellow
- 31318 Green

38217 Light holder
37875 Bulb
35796 Compression spring
37679 Spring ring
31647 Disk 4

32235 Potentiometer
31324 Electromagnet
31325 Rear end plate
35787 Small wheel

Socket
- 31253 Green
- 31254 Red

Plug
- 31336 Green
- 31337 Red

31394 Worm drive

31062 Mini-motor/6V
37780 Switch
37457 Rack & pinion 30
37351 Rack & pinion 60

37268 Mini-motor rack
37263 Mini-motor rack without gears
31068 Mini-motor gear box

Kit-Building Tips

Assembling the models in the fischertechnik Robotic Computing Kit will be easy if you follow the step-by-step pictorial instructions on the following pages. Each project has been broken down into a series of subassemblies which make up the final model. A wiring diagram is provided to aid you in connecting your model to the computer, and additional notes and callouts will help you to clarify any steps which are not clearly shown. Your work will be easier if you remember these simple hints:

- Begin with the first project in the book and work your way through the kit one project at a time. In this way, you will become familiar with construction techniques and programming theory that will help with the more complicated models.
- As you begin each step, select the parts you need as shown in the pictorial parts list(s) on each page. Set these aside in a small group so your construction can proceed without the distraction of searching for the parts you need. Use the metric scales and other hints on the pictorials to be sure you have the right parts.
- Always build the models in the sequence shown, or you may find that you have to disassemble earlier work to add new modules.
- DO NOT CUT the ribbon cable between the models and the computer once you have it set up as shown on page 8. If you are not using particular wires for a project, bend them out of the way, keeping the tips separated to avoid short circuits. If you cut the cable, you will not have the wires you need for other projects. Use the short wires you cut from the cable during its setup to make short jumper wires when needed.
- If you want to modify the computer programs, be SURE you have made a backup copy of the original disk first. A backup is always good procedure, even if you do not plan to change the programs.
- The parts in your kit are made from durable, precision-molded nylon that will give you years of kit-building fun. Remember that the parts interlock by sliding together...do not pry or twist them apart or use anything to force them together.
- Use fischertechnik power supply, Part No. 30173A, or damage to your interface or kit may result.
- Remember that your Robotic Computing Kit uses electricity. Although the voltage at the end of the cable is low, be sure that unused wire ends or connectors are separated or insulated with tape to prevent short circuits which could damage the interface, motors, or lamps. Unplug the power supply from the wall outlet when your kit is not being used.

Assembling the Cables

Before beginning the construction of the fischertechnik computing models, you should assemble the 20-wire flat ribbon cable. The wires Black 2 through Yellow 1 (see diagram) are shortened at the right side (opposite the connector) by approximately 50cm (17 in). Do not throw away the cut-off piece; short jumper cables are made from it. Now separate the wires of the cable into separate strands, again starting at the right end (the one you cut) and leaving approximately 50cm from the left end (the connector end) still attached to each other. The wire pairs Black 2/White 2, Grey 2/Violet 2, Blue 2/Green 2, and Yellow 2/Orange 2 can be kept together as pairs.

The cable you have just created will mate with all of the fischertechnik computing models. If some of the wires are not used by a particular model, DO NOT cut them off...just bend them gently out of the way, being sure that wire ends do not touch each other.

Next, remove some of the insulation from each of the wire ends, being careful not to damage the fine wire strands. Plugs should be installed on the ends of the flat cable wires. Plugs are also installed on the potentiometer wires. To make these installations, bend the ends of the wires that you have stripped back towards the insulation (see illustration 2) and insert these ends into the plugs or sockets. Use the small screwdriver to tighten the connector screws, being careful not to overtighten them as this may squeeze the wire out of the connector.

The flat cable has a connector that will mate with your fischertechnik computing interface. Just plug the connector into the interface, and all wires will be properly connected.

The potentiometers are mounted in their brackets by using a spacer and a small nut (see illustration 3).

When all of these steps have been carried out, you should perform a test of the cable assembly and the interface. Perform the test as follows:

— Connect a mini-motor at output M1 (use the interface circuit diagram on the back of the interface manual as a guide).
— Connect a switch between input E1 and +5 volts.
— Connect the two potentiometers between EX and EY and +5 volts.
— Connect the power supply to the interface (observe proper polarity).
— Load the DIAGNOSTIC program from the disk or tape (see the interface manual for additional information).
— Start diagnostic routine (RUN).

Be sure the red LED on the interface is illuminated. If it is not, check the polarity of the power supply connection, and be sure the power cable is correctly plugged in. The program will display the status of the ten input channels and the four outputs on the screen.

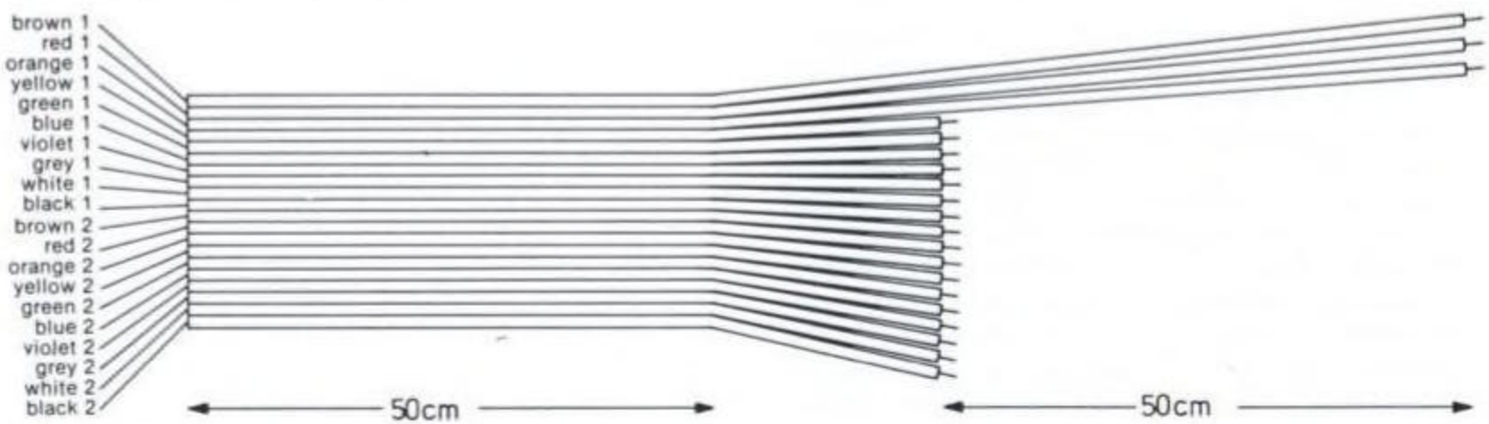
If you activate the switch connected to E1, the status shown should change. You can check E2...E8 the same way. Then rotate the shaft of the potentiometers connected to EX and EY. The values shown should change within the range of 0 to 255, although the very ends of the range may not be reached.

In case of any differences from the expected results, check the wiring carefully to be sure that the switches or potentiometers are actually connected as you think they are.

As indicated by the program, you can start the motor, cause it to run both clockwise and counterclockwise, and then shut it off. If motor output M1 operates correctly, you can switch the motor to the wires for each of the other motor lines (M2, M3, M4) and repeat the test.

If the motor does not work on any of the circuits, connect it directly to the power supply to be sure the motor itself is not defective.

Assembly of flat strip cable

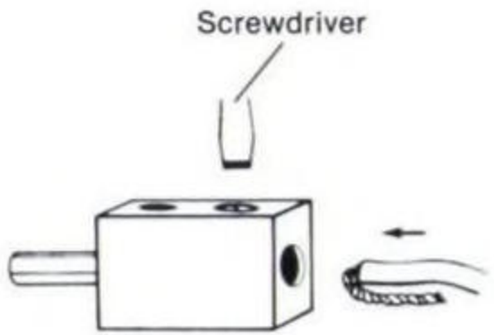


(Illustration 1)

The following general principles may be used as a guide during the test: If some of the inputs work and others do not, or if motors operate clockwise but not counterclockwise, it is reasonable to assume that the interface is defective. If, however, nothing runs at all, then you should check the entire installation very carefully, starting with the installation of the interface in or to the computer, the wiring, and that devices are connected to the proper lines. Finally, you can disconnect the cables from the interface and from the devices and check them for continuity. Use a continuity tester or ohmmeter for this test (see illustration 4).

One other note regarding wiring your models: As far as possible, always use the same color wires for the same functions. While some of the models will not use all of the switches and other devices pictured on the full cable assembly, the software always assumes that switches are connected where they are shown, potentiometers on their circuits, etc.

Mounting of plug



(Illustration 2)

Potentiometer mounting

(Illustration 3)

Continuity tester

(Illustration 4)

fischertechnik Computing Programming

Anyone who has thought about the idea of controlling appliances or models with the computer knows that this is not the easiest of tasks. You have to be familiar with the input/output structure of the computer, with the requirements of a suitable interface, and with computer programming in order to make it all work. Until now, that is. The fischertechnik Robotic Computing Kit provides the proper interface for your computer, and with it, the software (programs) you need to make it all work smoothly and easily.

If you have not yet connected the interface to your computer, we suggest that you do so now. BE SURE THE COMPUTER IS TURNED OFF WHEN YOU INSTALL THE INTERFACE.

Now load the BASIC program DRIVER.(...) for your computer model from the disk or tape. See the manual that comes with the interface for more information. When the program has loaded, type RUN. After a few moments, the computer will show READY. Although nothing seems to have happened, your computer now contains some new commands which were not there before. These commands have been designed to mate the fischertechnik computing interface to the computer so that the operation will be smooth and simple for you. Instead of detailed programming knowledge, the few BASIC commands we'll describe here will give you complete control over your fischertechnik computing models.

The motor output M1 is controlled as follows:

SYS M1, CW	**SYS M1, CCW**
(Clockwise)	(Counterclockwise)

The same commands offer similar control over outputs M2, M3, M4. Remember that a command to turn an output on always results in rotation to the right.

The status of the ten input lines is checked by means of the BASIC USR function. The value of USR (E1) = 1 if the line carries +5 volts; otherwise, the function returns a value of 0. In the same way, the status of the other digital inputs can be checked by the functions USR (E2)...USR (E8).

The analog inputs EX and EY are connected to +5 volts through one potentiometer (4.7K ohms) each. The functions USR (EX) and USR (EY) will have a range of values from 0 to 255, depending on the position of the potentiometer. Using this information, the motion of a robot arm or other turning device can be measured if it is connected in such a way as to turn the potentiometer, and if the program repeatedly calls the function to measure the value.

The printed programs which appear in this manual have been written in Commodore 64 BASIC. Some program lines may have to be changed for use with other computer systems. These are marked in the printed listings with an asterisk (*). The disk which you received with your interface for the Apple II or other computer has already been changed, so you need not make further changes if you use the programs supplied on the disk. If you program for yourself, using the printed materials for reference, remember to make the appropriate changes in command or syntax for your system.

Enough theory for the moment. Let's try a few practical experiments to familiarize you with the commands. Take a motor from the kit and attach it to the wires for the M1 output line. Then type the following command:

SYS M1, CW < RETURN >

The motor will run for a short time and then stop. As you know from reading the interface instructions, the interface monitors the data flow, and when no further commands are received, it shuts off all outputs.

Next, try the other output commands to see how they work. Then try the input lines by typing the command:

PRINT USR (E1) < RETURN >

Depending on whether the switch connected to E1 was open or closed when you pressed RETURN, the screen will show a 0 or 1. At the same time, if any motor was connected, but had stopped operating because no command had been given for a while, this command on the input line will also reactivate the outputs.

If you are not using a floppy disk and have loaded the startup routine by typing it into your computer, you should now store it onto tape. You will need this program each time you start to use your computer and fischertechnik computing interface; you must load it into your computer before doing anything else, since it installs the operating commands we discussed above, which are needed to operate your models.

With this preliminary training under our belts, let's try the first project, a computer-controlled traffic light installation.

Traffic Light

We'll start with the simplest model in the kit. Following the pictorial instructions on the next page, take the few parts needed, assemble them according to the illustration, and your traffic light is ready.

The wiring diagram is shown on page 16. Connect the green lamp to M1, the yellow lamp to M2, and the red to M3. Then, if you are using a disk drive, load the startup program, if needed, and next, the program Traffic Light.(...). If you have copied the programs to tape, proceed with the same routine.

If you are going to type in the program from the listings, remember that you need the startup program first. If you saved it on tape as suggested, load it in first, then type the Traffic Light program. If you do not have the startup program on tape, you will have to type it in first, as it must be loaded for the other programs to work.

The point of contact between the two programs is the line:

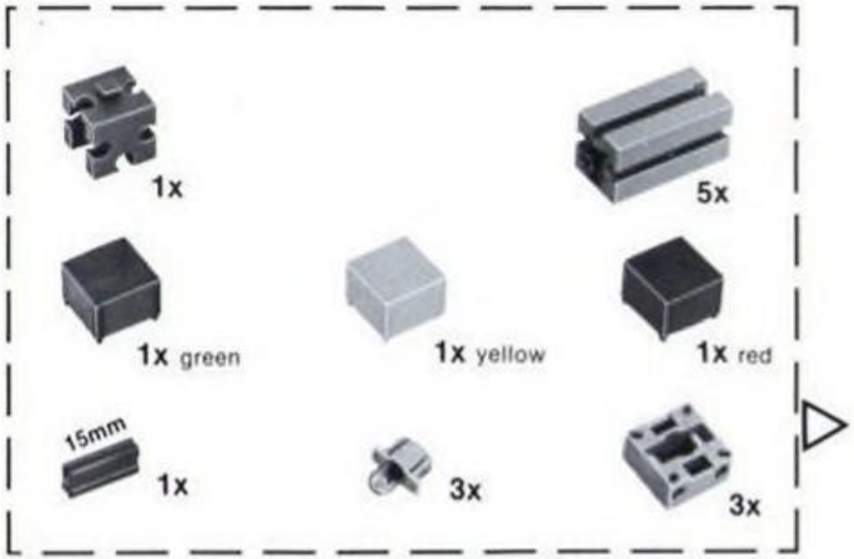
500 SYS INIT

Let's take a look at the program. Use the printed copy for reference, although you can also LIST the program on your screen using the appropriate procedure for your computer.

Lines 600 through 820 are REM lines which contain the titles and the setup of the various interface lines. Using this listing, you can be sure your model is correctly connected to the interface.

The actual program begins with line 900. You will find four blocks of programming, each of which defines the parameters of one phase of the traffic light's operation. In these blocks, the switching status of the lamps and the timing are indicated. The timing and switching are actually performed by the subroutine which begins at line 2000. As you can see from the example, the switching status of the outputs need not be used only in the form shown earlier in the SYS command. You can also define these by variable names; in this example, the names of the colors are used to represent ON (CW) and OFF.

```
•500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM TRAFFIC LIGHT
 640 REM
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 730 REM ASSIGNMENTS FOR THE INTERFACE
 740 REM
 750 REM GREEN=M1
 760 REM YELLOW=M2
 770 REM RED=M3
 780 REM
 800 REM FUNCTION
 810 REM TRAFFIC LIGHT WITH CHANGING GREEN, YELLOW AND
     RED PERIODS
 820 REM
•900 PRINT CHR$(147)
 910 PRINT"FISCHERTECHNIK"
 920 PRINT"COMPUTING"
 930 PRINT
 940 PRINT"TRAFFIC LIGHT"
 950 PRINT:PRINT
1000 REM
1010 REM GREEN PERIOD
1020 LET GREEN=CW
1030 LET YELLOW=OFF
1040 LET RED=OFF
1050 LET DELAY=1000
1060 PRINT"{UP}";
1070 PRINT"GREEN PERIOD "
1080 GOSUB 2000
1090 REM
1100 REM YELLOW PERIOD
1110 LET GREEN=OFF
1120 LET YELLOW=CW
1130 LET RED=OFF
1140 LET DELAY=200
1150 PRINT"{UP}";
1160 PRINT"YELLOW PERIOD"
1170 GOSUB 2000
1180 REM
1190 REM RED PERIOD
1200 LET GREEN=OFF
1210 LET YELLOW=OFF
1220 LET RED=CW
1230 LET DELAY=1000
1240 PRINT"{UP}";
1250 PRINT"RED PERIOD   "
1260 GOSUB 2000
1270 REM
1280 GOTO 1020
2000 REM DELAY LOOP
2010 FOR T=0 TO DELAY
•2020 SYS M1,GREEN
•2030 SYS M2,YELLOW
•2040 SYS M3,RED
 2050 NEXT T
 2060 RETURN
```



0 15mm

Traffic Light with Push Button

It's all well and good to be able to control the traffic light with a computer. But turning the lights on and off is just a beginning. In fact, we really wouldn't need a computer at all if we just wanted to turn the lights on and off at some predetermined rate. In the real world, a predetermined rate is, at best, a compromise. At some times of the day, the cycle will be too long for the traffic load. At other times, it will be too short. This is where the computer comes in. By using sensors such as induction loops in the roadway or push buttons for pedestrians, the computer can be used to analyze the input information and adjust the traffic flow to meet the conditions of the moment.

With just a simple modification, we can equip our traffic light with a pedestrian push button. Connect one of the switches between lines E8 and the +5 volts wire.

Now the traffic programmer must arrange for a changeover when the button is pushed. This change must not happen abruptly, but with a small time lag to let the traffic come to a gradual halt.

In the sample program, this has been accomplished by changing the timing loop when an operation of the push button is detected (line 2070). This shortening of the timing loop causes the lights to change more quickly, modifying the preset cycle. As you can see from line 2050, this action can take place only if the light is green; changing the timing of red or yellow would leave us with a strange pattern.

Checking the value of the USR function tells the system if the switch is open or closed. Its value is 1 if the switch is closed, allowing the voltage to flow in the circuit; the value is 0 if no voltage is present (open switch).

```
•500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM TRAFFIC LIGHT WITH A PUSHBUTTON FOR PEDESTRIAN
 640 REM CROSSING
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 730 REM ASSIGNMENTS FOR THE INTERFACE
 740 REM
 750 REM GREEN=M1
 760 REM YELLOW=M2
 770 REM RED=M3
 780 REM
 790 REM E8=PUSHBUTTON
 800 REM
 810 REM FUNCTION
 820 REM A TRAFFIC LIGHT WITH CHANGING GREEN, YELLOW AND RED
 825 REM PERIODS AND A PUSHBUTTON FOR PEDESTRIANS
 830 REM
•900 PRINT CHR$(147)
 910 PRINT"FISCHERTECHNIK"
 920 PRINT"COMPUTING"
 930 PRINT
 940 PRINT"TRAFFIC LIGHT WITH PUSHBUTTON"
 945 PRINT "FOR PEDESTRIANS"
 950 PRINT
1000 REM
1010 REM GREEN PERIOD
1020 LET GREEN=CW
1030 LET YELLOW=OFF
1040 LET RED=OFF
1050 LET DELAY=1000
1060 PRINT"GREEN PERIOD "
1070 GOSUB 2000
1080 REM
1090 REM YELLOW PERIOD
1100 LET GREEN=OFF
1110 LET YELLOW=CW
1120 LET RED=OFF
1130 LET DELAY=200
1135 PRINT"(UP)";
1140 PRINT"YELLOW PERIOD"
1150 GOSUB 2000
1160 REM
1170 REM RED PERIOD
1180 LET GREEN=OFF
1190 LET YELLOW=OFF
1200 LET RED=CW
1210 LET DELAY=1000
1215 PRINT"(UP)";
1220 PRINT"RED PERIOD   "
1230 GOSUB 2000
1240 PRINT"(UP)";
1250 REM
1260 GOTO 1020
2000 REM DELAY LOOP
2010 FOR T=0 TO DELAY
•2020 SYS M1,GREEN
•2030 SYS M2,YELLOW
•2040 SYS M3,RED
2050 IF GREEN <>CW THEN GOTO 2090
2060 REM IF NOT IN GREEN PERIOD, DON'T SCAN THE KEY
•2070 IF USR(E8)=1 THEN LET T=DELAY-10
2080 REM IF PUSHBUTTON WAS PRESSED, SHORTEN GREEN PERIOD
2090 NEXT T
2100 RETURN
```


1x

Circuit Layout—Traffic Lights

E1 brown

E2 red

EX orange

EY yellow

+5V green

E3 blue

E4 violet

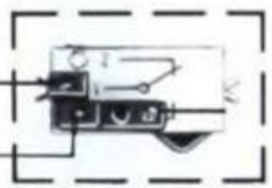
E5 grey

E6 white

E7 black

E8 brown

+5V red



Add switch
for Traffic Light
with Push Button

M1 orange

M1 yellow

M2 green

M2 blue

M3 violet

M3 grey

M4 white

M4 black

Machine Tool

Now let's put some life and motion into our models. A small machine tool will be a good project.

First, an explanation of what a "machine tool" is. In the processing of metals into manufactured parts, operations such as drilling, milling, turning, punching, and embossing are often required to turn raw material into a useful product. In many cases, these operations are performed by automated machinery. Sometimes the work is done by a series of machines arranged around a central rotating worktable. The material to be processed is placed on the table, and the turntable moves in preset steps, bringing the work into position in front of each of the machines. After all steps are completed, the finished piece is removed.

This process has been simplified a bit in designing the model we are about to build. There is no insertion or removal of the workpiece; the four blocks representing the product being machined rotate continuously on the turntable. There is also only one machine in our setup—a small model of a drilling machine.

The turntable rotates until a switch is operated by one edge of the workpiece. Then the drill moves down, with its lowest position also being marked by a switch closure. The drill arm stops its motion, and drilling action is indicated by the yellow lamp. Then the drill moves up until it reaches the top of its movement, indicated by closing yet another switch. At this time, the cycle begins again. Note that the actual cycle should always start with a check that the arm is in its highest position. Otherwise, there is a slight chance that the arm could be caught in the workpiece when the rotation starts, and damage to the modules could result.

For controlling the motors, we use the following commands:

SYS M1, CW
SYS M1, CCW
SYS M1, OFF

for clockwise and counterclockwise rotation, and for switch OFF.

Please note that clockwise rotation of the motor does not always mean that the part of your model to which it is connected will also rotate clockwise. The direction may be affected by your use of gears in the construction and by the fact that clockwise, or left and right, can be confused because of the angle at which you view your model.

To test this out for the machine tool model, enter the following command in the direct command mode (from the READY prompt):

SYS M2, CW

If the drill arm moves up slightly, then your choice of directions and your wiring are correct. If it moves down, however, reverse the two plugs at the motor and try again. If nothing moves at all, or if the other motor moves, check all of your wiring. After completing the test, enter the following command in the direct mode:

SYS INIT

The DIAGNOSTIC routine is a big help in testing your wiring. With it, you can check all switches and the two potentiometers. You can control the motors manually just by pressing a key on the computer.

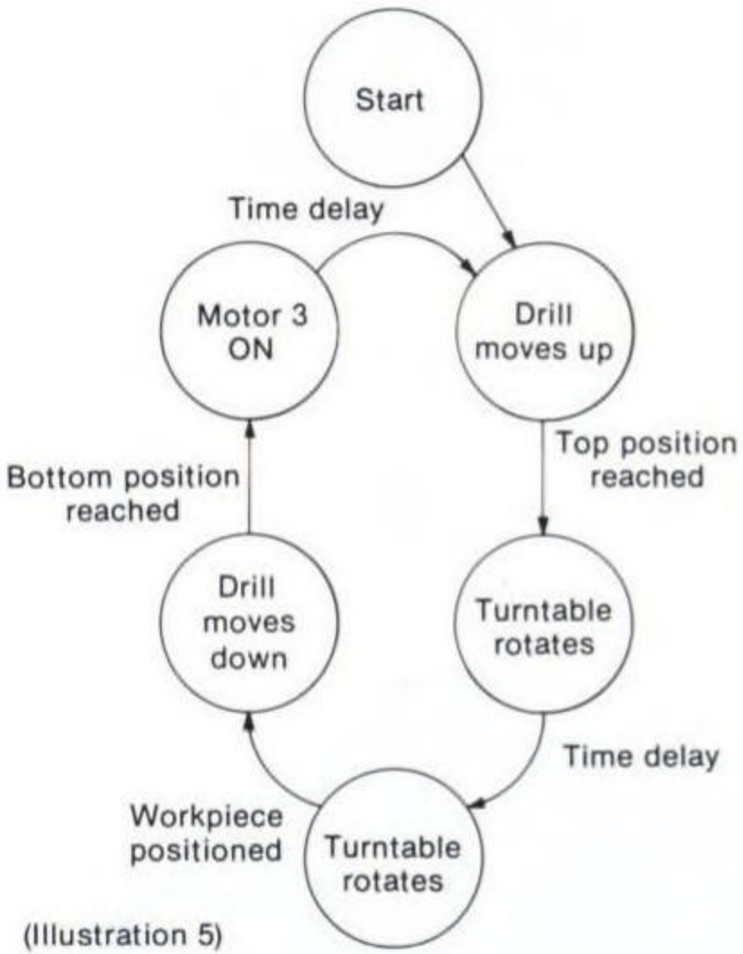
Let's look now at some of the features of the program. In line 1010, we find that the computer "loops" back on itself, waiting for an input from the motion limit switch. Until that signal is received, the computer marks time. Since the checking of the input line is an active command, the motor continues to run. When the limit switch is activated, the computer proceeds to line 1020, switching off the motor. Similar looping actions occur in lines 1070 and 1100.

Lines 1130 through 1150 create a pause in processing and activate M3 for signaling the processing procedure. The FOR-NEXT loop structure in lines 1040-1060 serves another purpose. It delays the start of activity at a new position to allow for the movement from the previous turntable position.

The diagram shown in illustration 5 graphically illustrates the flow of work and programming for this project. The activities in the circles indicate that the machine is doing something; this action continues until the condition indicated at the output (arrow to the next step) has been met; at that time, the model proceeds to the next action.

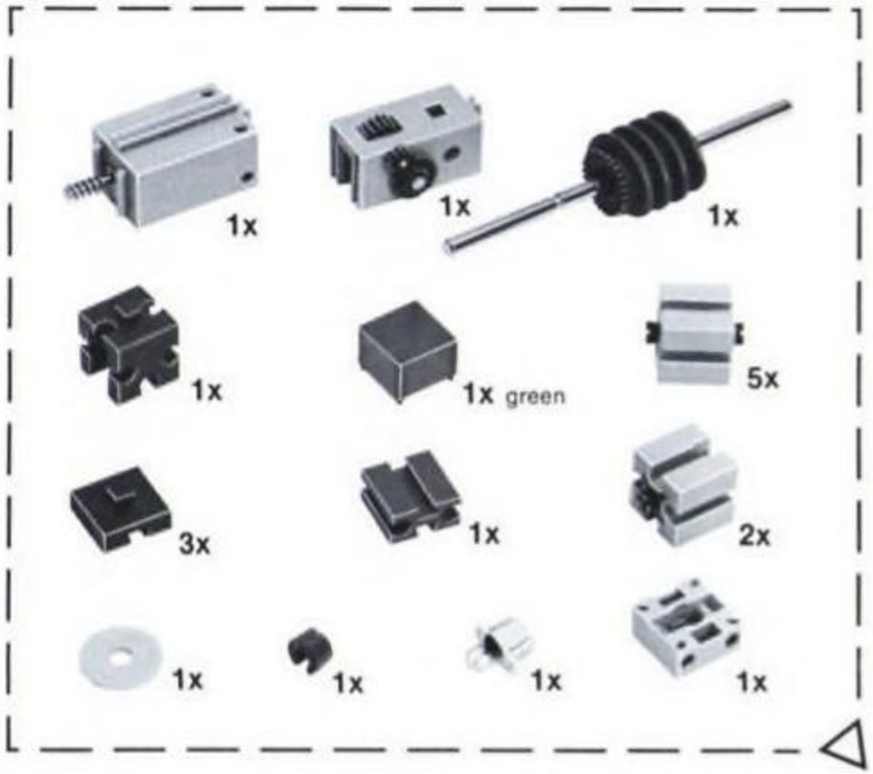
Graphic representations like this, called flowcharts, can be a big help in understanding how something works or the way in which a program should flow. By breaking a complex activity down into manageable steps, you simplify the task of understanding what is going on or of programming the computer to perform the actions you want.

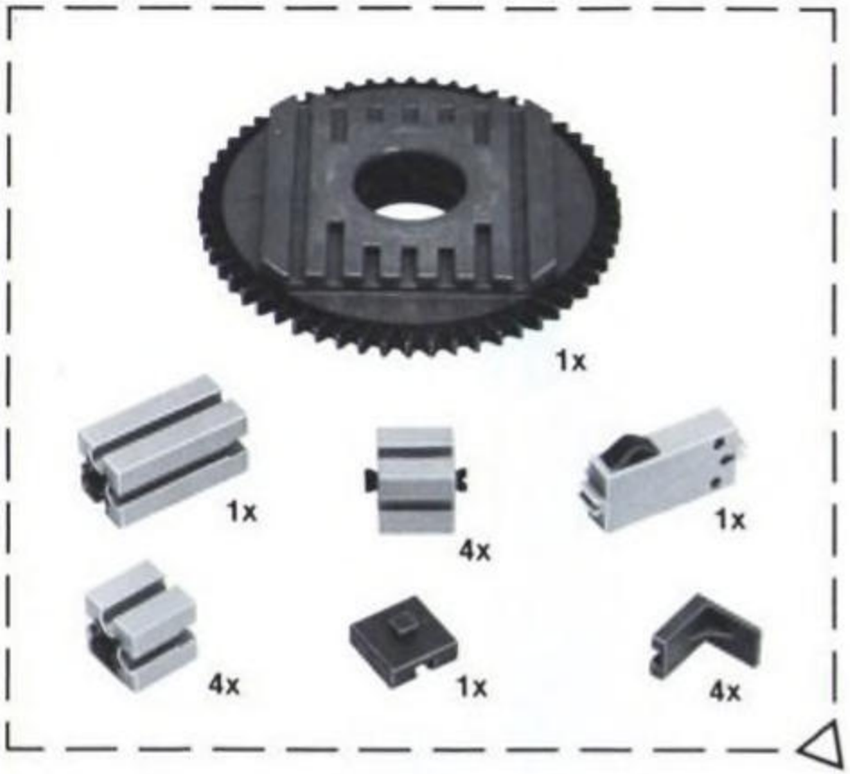
Machine Tool

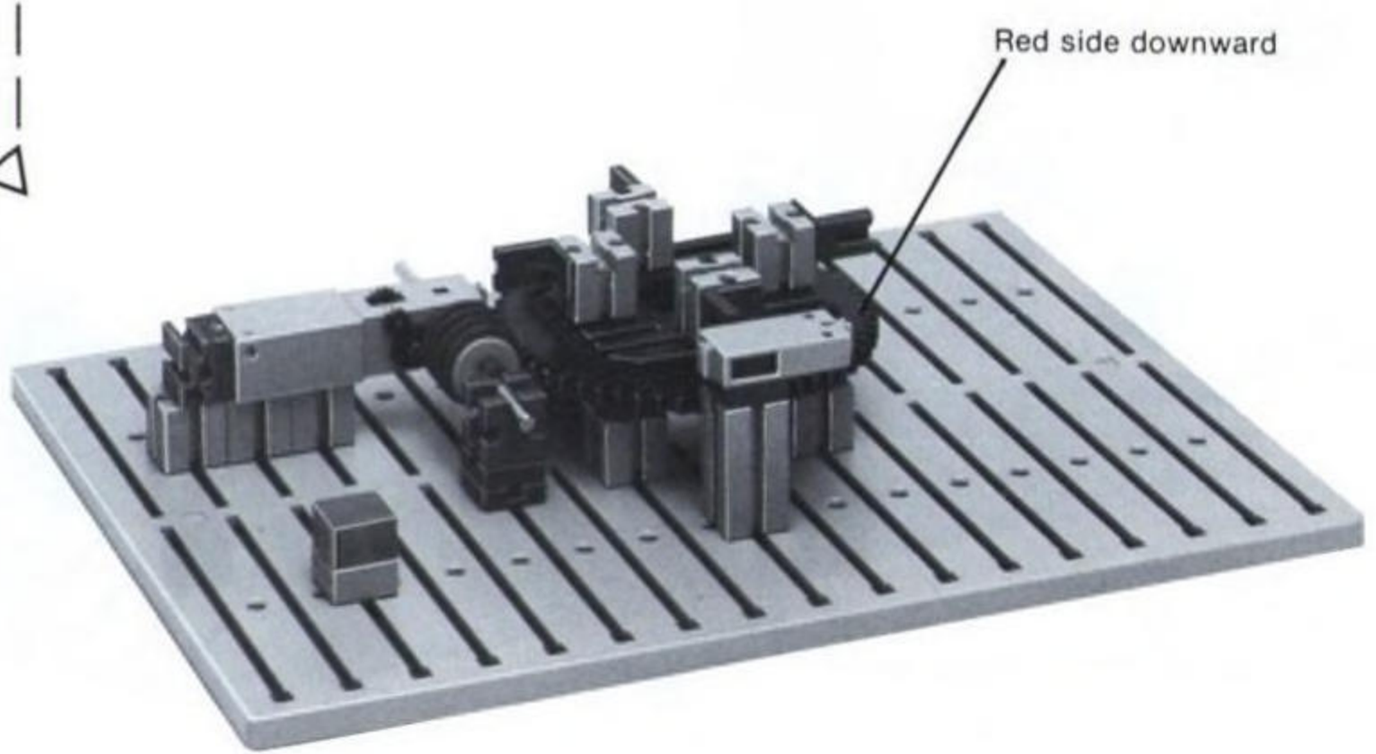


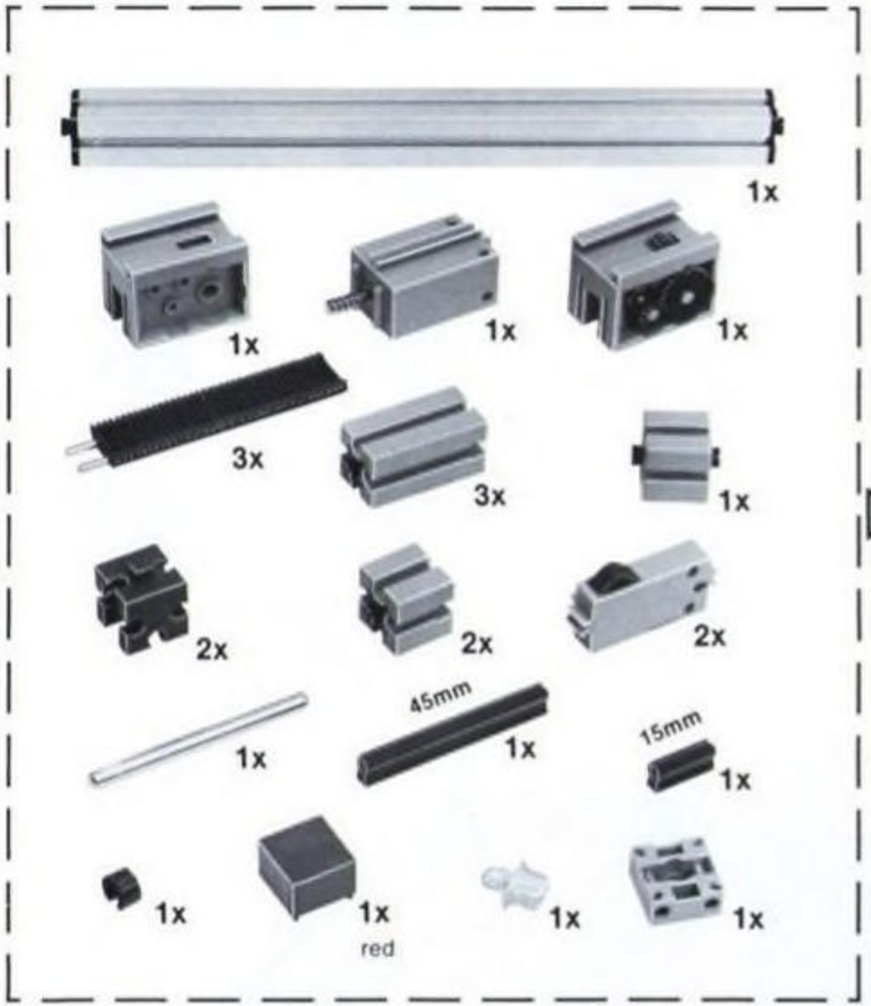
(Illustration 5)

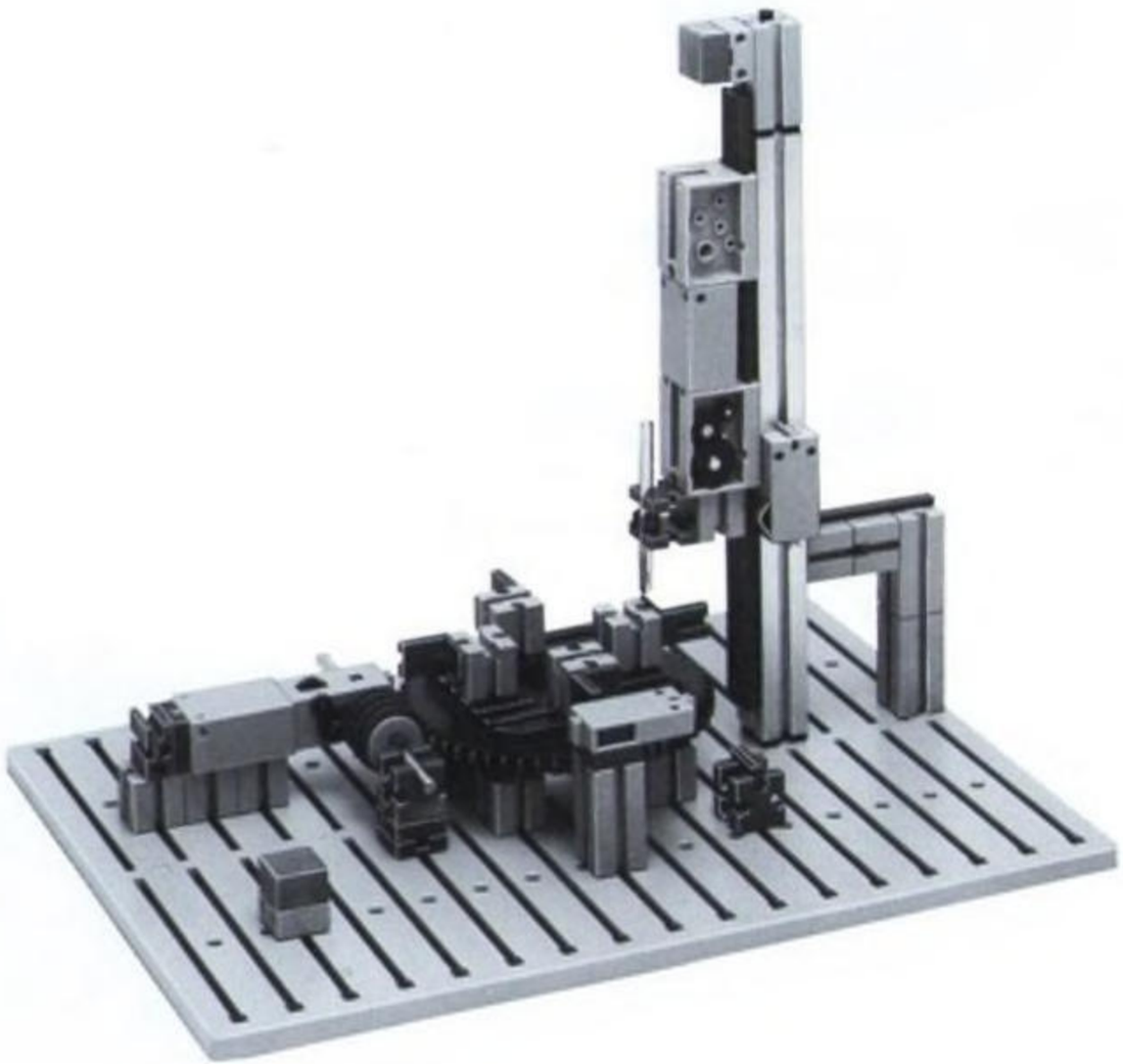
```
•500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM MACHINE TOOL
 640 REM
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 730 REM ASSIGNMENTS FOR THE INTERFACE
 740 REM
 745 REM INPUT
 750 REM E3= STOP POSITION OF TURN TABLE
 760 REM E4= BOTTOM POSITION OF THE DRILLING MACHINE
 770 REM E5= TOP POSITION OF THE DRILLING MACHINE
 780 REM
 800 REM OUTPUT
 810 REM
 820 REM M1=TURN TABLE MOTOR
 830 REM M2=DRILLING MACHINE MOTOR
 835 REM M3=BUSY INDICATOR
 840 REM
 842 REM
 844 REM FUNCTION DESCRIPTION :
 846 REM CYCLIC CONTROL OF TURN TABLE
 848 REM AND DRILLING MACHINE
•850 PRINT CHR$(147)
 900 PRINT"FISCHERTECHNIK"
 910 PRINT"COMPUTING"
 920 PRINT
 930 PRINT"MACHINE TOOL"
 940 REM
 950 PRINT
 990 PRINT"MOVE THE DRILLING MACHINE UP"
•1000 SYS M2,CW:REM MOVE THE DRILL UP
•1010 IF USR(E5)=0 THEN GOTO 1010:REM CHECK IF DRILL IS IN
      THE TOP POSITION
•1020 SYS M2,OFF:REM SWITCH OFF DRILL
 1030 PRINT"{UP}";
 1040 PRINT"NEXT WORKPIECE                "
 1050 REM DELAY LOOP FOR TURN TABLE
 1060 FOR I=1 TO 200
•1070 SYS M1,CW
 1080 NEXT I
•1090 IF USR(E3)=0 THEN GOTO 1090:REM DRILLING POSITION
      REACHED ?
•1100 SYS M1,OFF:REM SWITCH OFF TURN TABLE MOTOR
 1110 PRINT"{UP}";.
 1120 PRINT"MOVE DRILLING MACHINE DOWN  "
•1130 SYS M2,CCW:REM MOVE DRILL DOWN
•1140 IF USR(E4)=0 THEN GOTO 1140:REM DRILLING MACHINE IN
      BOTTOM POSITION ?
•1150 SYS M2,OFF:REM SWITCH OFF DRILL
 1160 PRINT"{UP}";
 1170 PRINT"MACHINE WORKPIECE             "
 1180 REM DELAY FOR DRILLING
 1190 FOR I=1TO500
•1200 SYS M3,CW:REM SWITCH ON BUSY INDICATOR
 1210 NEXT I
•1220 SYS M3,OFF:REM SWITCH OFF BUSY INDICATOR
 1230 PRINT"{UP}";
 1240 GOTO 990
```


1x
1x
1x
1x
1x green
5x
3x
1x
2x
1x
1x
1x
1x

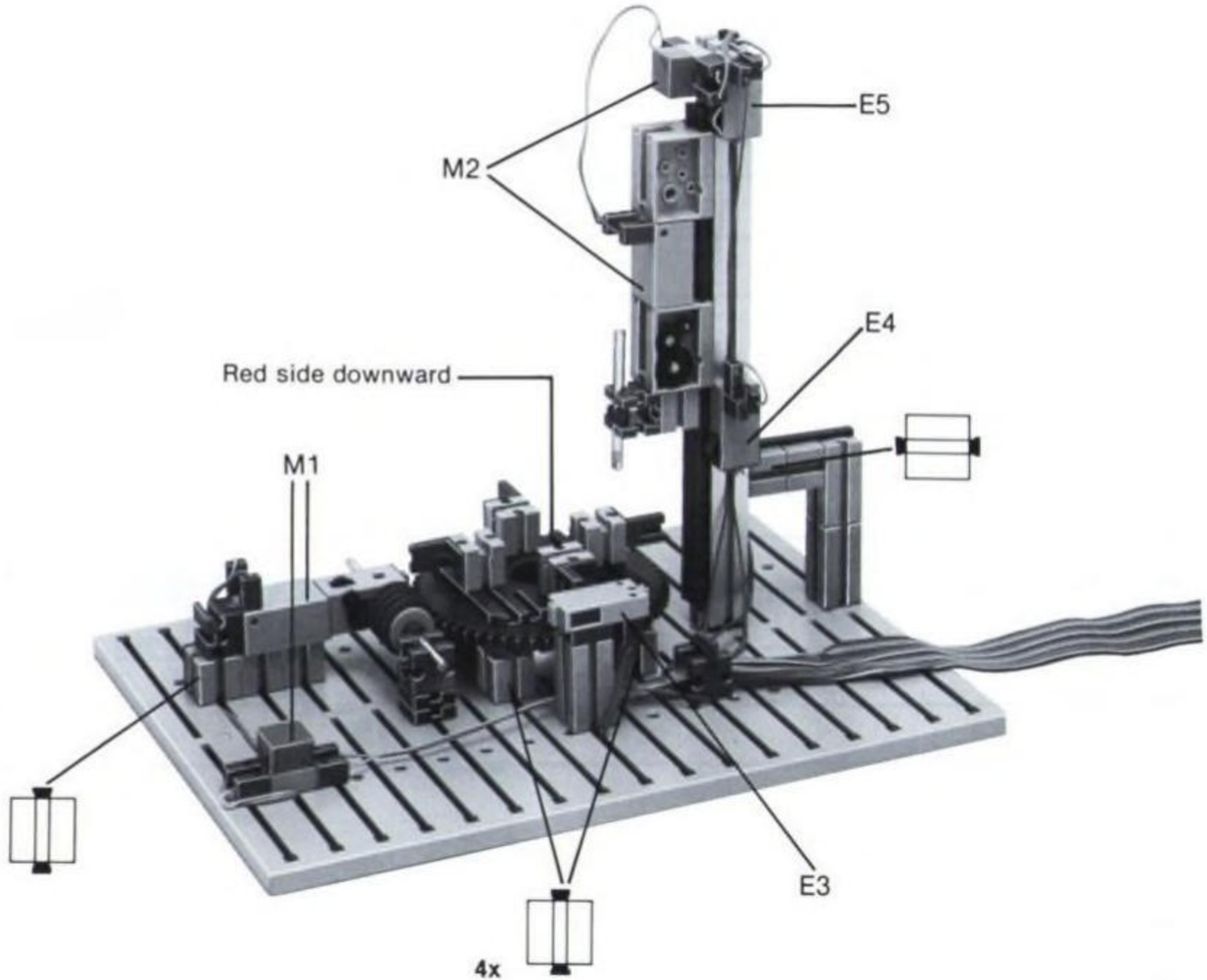

1x
1x
4x
1x
4x
1x
4x


Red side downward


1x
1x
1x
1x
3x
3x
1x
2x
2x
2x
1x
45mm
1x
15mm
1x
1x
1x
red
1x
1x




0
15
45mm


E5
M2
E4
Red side downward
M1
E3
4x

Circuit Layout—Machine Tool

E1 brown

E2 red

EX orange

EY yellow

+5V green

E3 blue

E4 violet

E5 grey

E6 white

E7 black

E8 brown

+5V red

M1 orange

M1 yellow

M2 green

M2 blue

M3 violet

M3 grey

M4 white

M4 black

Lift

The lift model provides ample opportunity for developing a more sophisticated program based on the starting point provided by the program supplied with the kit.

From a simple elevator (lift) that just goes up and down repeatedly, we can create a system that will remember which floor has requested the elevator and stop there, and go on from there to a system that gives priority to calls from a particular floor. The "real world" applications are obvious.

The wiring plan for this project can be taken from the illustration in the assembly manual. Three switches are used to indicate the position of the elevator; three more will be used to request the elevator at a particular floor; a seventh switch may be installed to the right of the elevator "door" on the middle floor. We'll discuss this extra switch later.

The basic form of the elevator is not difficult to program. From the "calling" switches, a value for the variable SP is determined, indicating at which floor the elevator is wanted. In this case, 0 means ground floor, 1 the first floor, 2 the second. The variable is set as the program cycles through lines 1100-1130.

The next three lines of the program set the variable IP, which reflects the position of the cage as indicated by the three position switches. From here on it's simple...if the desired location (SP value) is smaller than the actual position (IP value), the elevator has to move down; if the value is higher, the elevator moves up. And if the two values are equal, the car is where it is wanted, the motor shuts off, and the floor level is displayed on the screen.

At the beginning of the program, the first few lines move the car to the ground floor. This serves to establish a clear point of reference for the rest of the program. We can use this motion to test our wiring. Input in direct command mode (at the READY prompt):

SYS M1, CCW

If the car moves downwards, everything is in order. Once you have the program operating properly, you should try varying the program to achieve different results. Try to figure out, for example, what would happen if the following lines were changed to read:

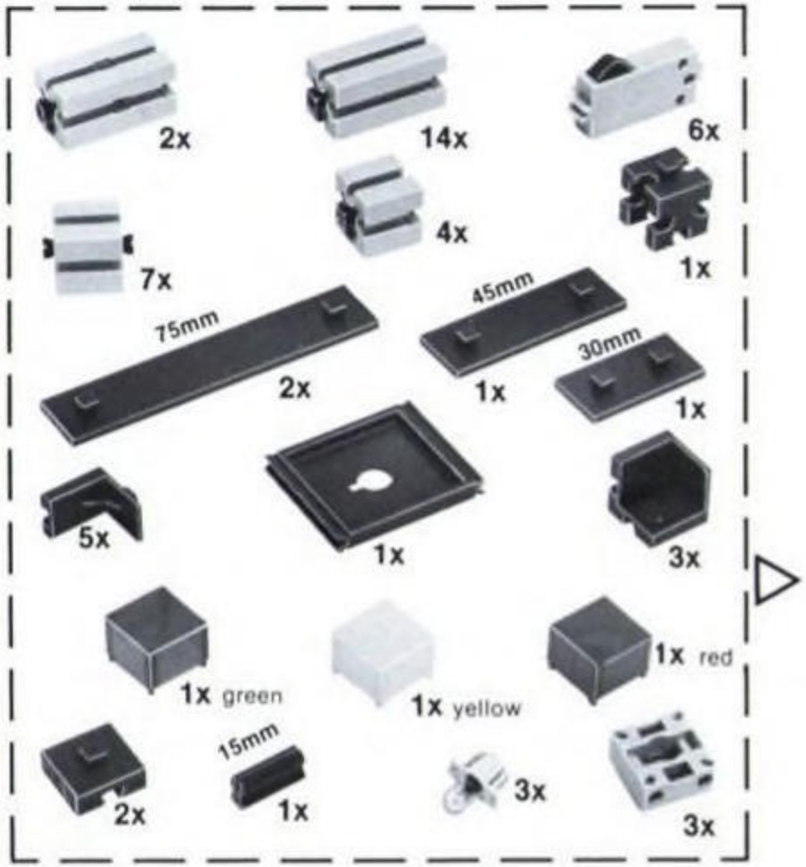
1230 GOTO 1110
1270 GOTO 1110

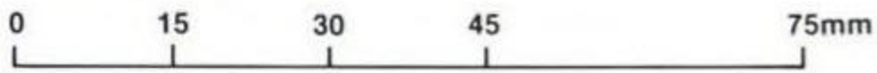
For another variation, try programming an elevator with storage of the desired floor call instead of the need to wait for the car to come to rest before entering a new value. You might also want to try to improve your elevator system with a priority control. This is where you get to use the extra switch we installed at the second floor. Set up the program so that one switch is a call to move upwards, and the other key is a request to go down, just like a real elevator. Now rewrite the program so that the elevator will not stop at the second floor unless it is moving in the desired direction of travel. That is, the elevator should not stop if it is going up and the call is down, and vice versa.

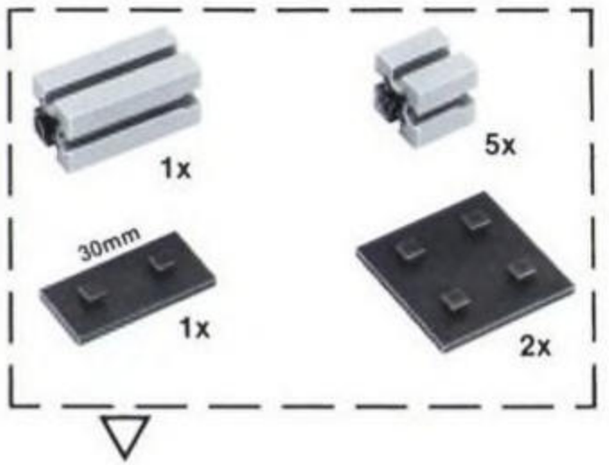
If you want a further programming exercise, set things up so the computer keyboard can be used to simulate the floor selection buttons usually found inside the elevator. Remember not to change the programs on your disk until a backup copy has been made to preserve the original.

As you can see, there's plenty of room to give your imagination full play.

```
*500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM LIFT
 640 REM
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 670 REM ASSIGNMENTS FOR THE INTERFACE
 680 REM
 690 REM INPUT
 700 REM E3= KEY FOR THIRD FLOOR
 710 REM E4= KEY FOR SECOND FLOOR
 720 REM E5= KEY FOR FIRST FLOOR
 730 REM E6= KEY TO CALL THE LIFT TO THE SECOND FLOOR
 740 REM E7= KEY TO CALL THE LIFT TO THE FIRST FLOOR
 750 REM E8= KEY TO CALL THE LIFT TO THE GROUND FLOOR
 760 REM
 770 REM OUTPUT
 780 REM M1=MOTOR FOR LIFT
 790 REM M2=LIFT IS MOVING UP
 800 REM M3=LIFT IS MOVING DOWN
 810 REM
 820 REM FUNCTION DESCRIPTION :
 830 REM LIFT WITH THREE FLOORS
 840 REM THE LIFT IS POSITIONED BY PRESSING
 850 REM THE KEY BELONGING TO THE DESIRED FLOOR
*860 PRINT CHR$(147)
 870 PRINT"FISCHERTECHNIK"
 880 PRINT"COMPUTING"
 890 PRINT
 900 PRINT"LIFT"
 910 PRINT:PRINT
 920 REM
 1000 LET SP=0 :REM NOMINAL POSITION
 1010 LET IP=0 :REM ACTUAL POSITION
 1020 LET AZ=0 :REM PRINTER FLIP FLOP
*1030 SYS M1,CCW :REM LIFT DOWN
*1040 IF USR(E5)=0 GOTO 1030:REM IS LIFT DOWN ?
*1050 SYS INIT :REM SWITCH ALL DEVICES OFF
 1060 IF AZ=1 THEN GOTO 1120
 1070 PRINT"(UP)";
 1080 IF IP=0 THEN PRINT"FIRST FLOOR "
 1090 IF IP=1 THEN PRINT"SECOND FLOOR"
 1100 IF IP=2 THEN PRINT"THIRD FLOOR "
 1110 LET AZ=1
*1120 IF USR(E6)=1 THEN SP=2 :REM GO TO THIRD FLOOR
*1130 IF USR(E7)=1 THEN SP=1 :REM GO TO SECOND FLOOR
*1140 IF USR(E8)=1 THEN SP=0 :REM GO TO FIRST FLOOR
*1150 IF USR(E3)=1 THEN IP=2 :REM LIFT IS IN THE THIRD FLOOR
*1160 IF USR(E4)=1 THEN IP=1 :REM LIFT IS IN THE SECOND FLOOR
*1170 IF USR(E5)=1 THEN IP=0 :REM LIFT IS IN THE FIRST FLOOR
 1180 IF SP<IP THEN 1210:REM MOVE LIFT DOWN
 1190 IF SP>IP THEN 1250:REM MOVE LIFT DOWN
 1200 GOTO 1050
*1210 SYS M1,CCW :REM LIFT DOWN
*1220 SYS M2,CW :REM LAMP FOR MOVING DOWN ON
 1230 LET AZ=0
 1240 GOTO 1150
*1250 SYS M1,CW :REM LIFT UP
*1260 SYS M3,CW :REM LAMP FOR MOVING UP ON
 1270 LET AZ=0
 1280 GOTO 1150
```

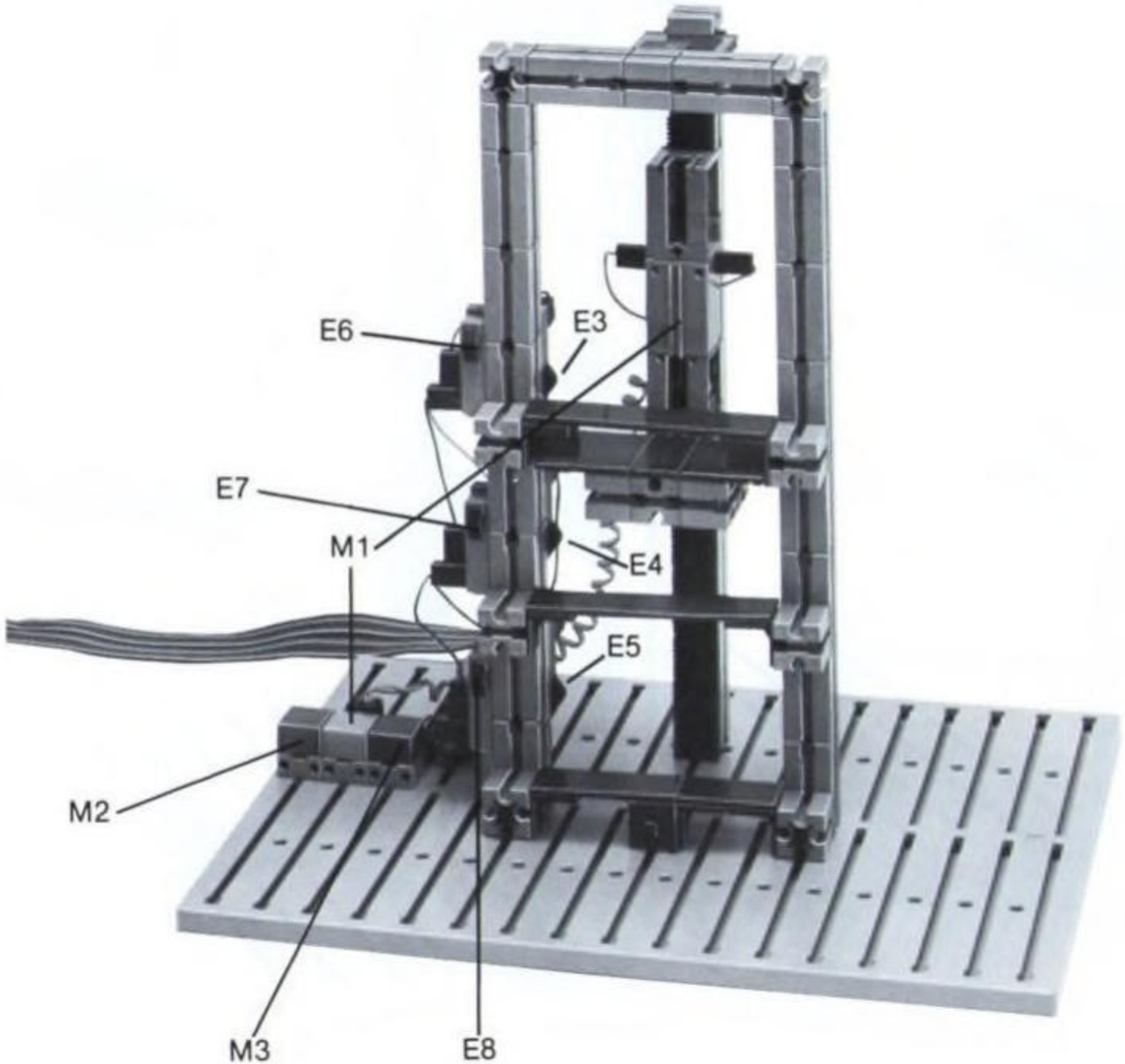

2x
14x
6x
4x
1x
7x
75mm
2x
45mm
1x
30mm
1x
5x
1x
3x
1x green
1x yellow
1x red
2x
15mm
1x
3x
3x


0
15
30
45
75mm

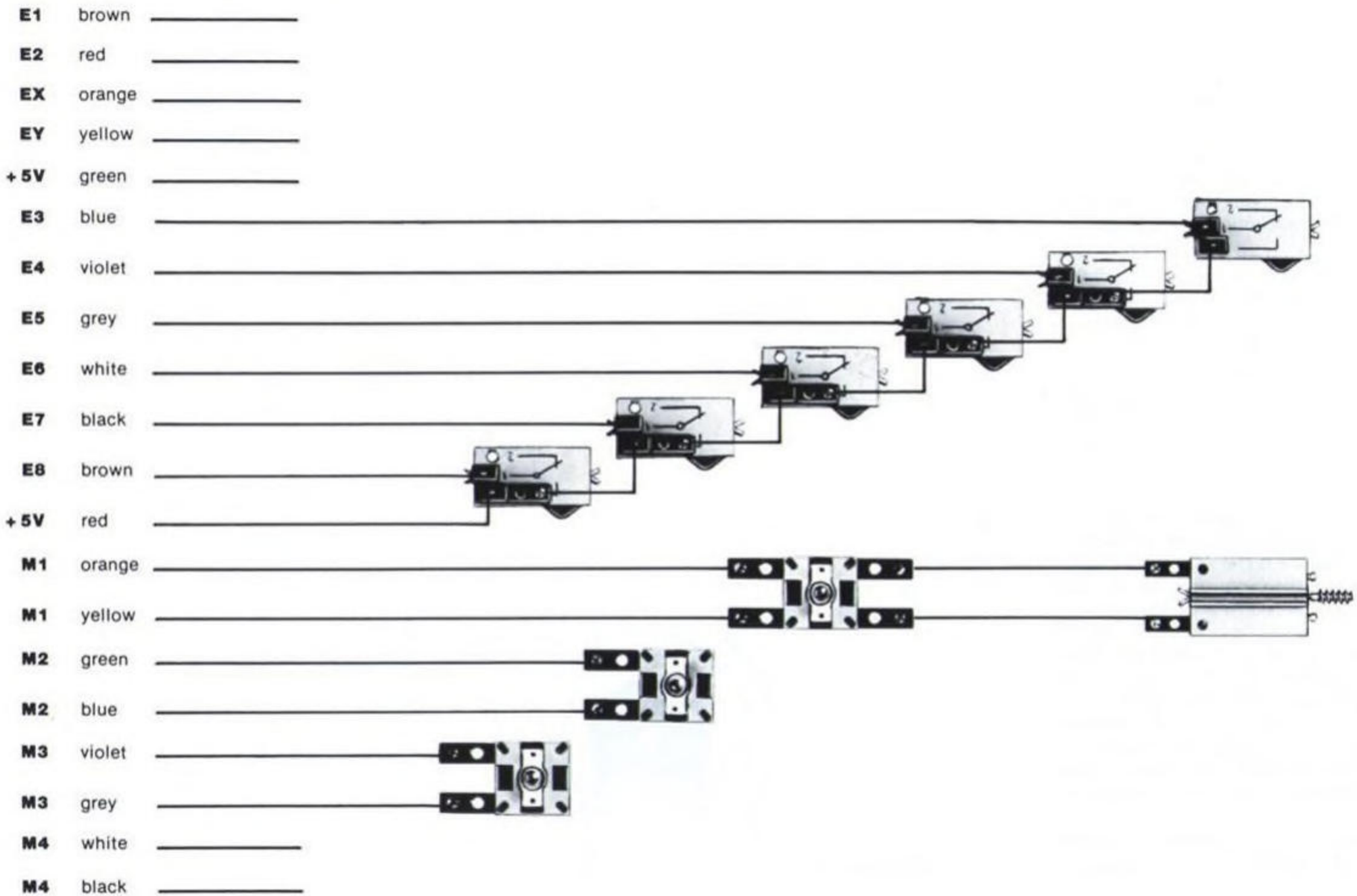


Underside shown

Rear view


E6
E3
E7
M1
E4
E5
M2
M3
E8

Circuit Layout—Lift



Continuous Position Indicator

Included with your fischertechnik Robotic Computing Kit are two potentiometers which can be used for giving continuous indication of the position of a movable part of your model, instead of the step-by-step indication we get by using the switches.

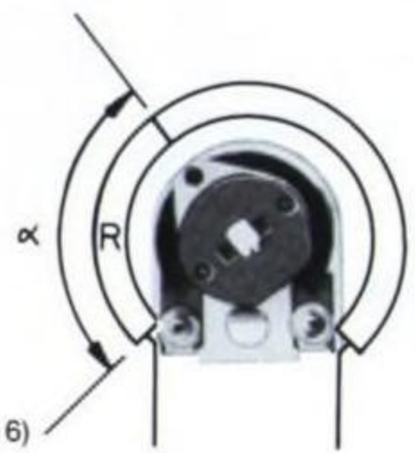
A potentiometer consists of a resistance element, along which a slider moves when we turn an axle. At the ends of the track and the slider, contacts have been placed so we can add wires. Illustration 6 shows the internal construction.

Depending on the position of the slider on the track, we will get a different resistance value which is directly related to that position, and which can therefore be used to indicate the position of an object attached to the axle.

One of the tasks the interface performs for us is to measure and give a value to this resistance. We connect the potentiometer to the lines EX and/or EY and +5 volts. The position of the potentiometer (the value of the resistance) is then checked by the functions USR (EX) or USR (EY).

Although the interface and programs for different computers operate in the same fashion, the results of the USR function may be different. In particular, you should be careful not to request a value from a line which is not connected to a potentiometer. The results of this will range from a slow flashing of the interface LED (Commodore 64) to an OVERFLOW AT 255 error (Apple II) to a lockup of the computer requiring a reset and restart (VIC 20). In addition, the values returned by the USR function may vary, although they should be within the range 0 to 255. In some cases, the lowest value reached will be around 20...this is normal and no cause for concern.

In order to become more familiar with the practical use of this concept, let's move on to the next project.



(Illustration 6)

Aerial Rotor

The aerial rotor project is designed to position the antenna correctly to a preset direction. The base of the antenna is equipped with a potentiometer for indicating its position. A second potentiometer, with a marker pin, is used to indicate the desired aiming point. If you remember the elevator project, the concept is similar. Here, too, there are desired and actual positions (values), except that in this case, the choice of position is much more widely variable since we are dealing with the potentiometer instead of the switches, which mark a single position only. The motor's activity and direction are controlled by the difference between the values of the desired and actual positions. When they match, the motor is switched off.

Take a look at the sample program printed here. You can see that the relation of this program to the elevator control is clear, at least through line 1050. Then we find something new.

Imagine, if you will, that when the values of the desired and actual positions match, we simply shut off the motor and jump to the beginning of the loop. What would happen is that the drive motor would not stop immediately, but would coast to a halt. As small as this motion might be, the value of the resistance of the potentiometer would change, and the motor would be driven in the opposite direction. In some cases, it is possible that the rotor would never actually stop; it would constantly seek back and forth on each side of the desired stopping point. This action is known as a "regulating oscillation."

To prevent this, the motor control cycle has been programmed so that it approaches the desired point in a controlled fashion. If the difference in values is greater than ten units, the motor runs at full speed. But if the difference is less than ten, the motor runs at reduced speed to allow it to stop at the right point without drifting past. Since we cannot change the voltage supplied to the motor as a control, we switch the motor on and off rapidly to simulate a slowdown. The on-time is the result of the FOR-NEXT loop at lines 1070 and 1080. The turn-OFF time is controlled by the duration of the remainder of the loop that checks the potentiometer for the next value.

In this process, we use another technique known as feedback. We use the difference in value between the desired and actual positions as the length of the delay loop, so the closer we get to the target point, the shorter the switch-on time is.

Give the program a try to better familiarize yourself with the theory by actual practice. As in previous projects, check the direction of rotation by giving a SYS M1, CW command in direct mode. If the antenna does not rotate and tries to keep pushing against the stop at one end of its travel, the program should be stopped immediately and the wires to the motor reversed to change the direction of rotation.

Unusual results may also appear at the very ends of the potentiometer range. In addition, because of the variations which may occur during manufacture, it is possible that the values of the two potentiometers may never quite coincide. If this should happen, pick a new set point in the center of the range rather than at the ends. The values will then reach a match.

Try some experiments to find the best position for your model and to try the effects of variations in the basic program. For a test, eliminate line 1020 and see what happens. Or change line 1070 to read:

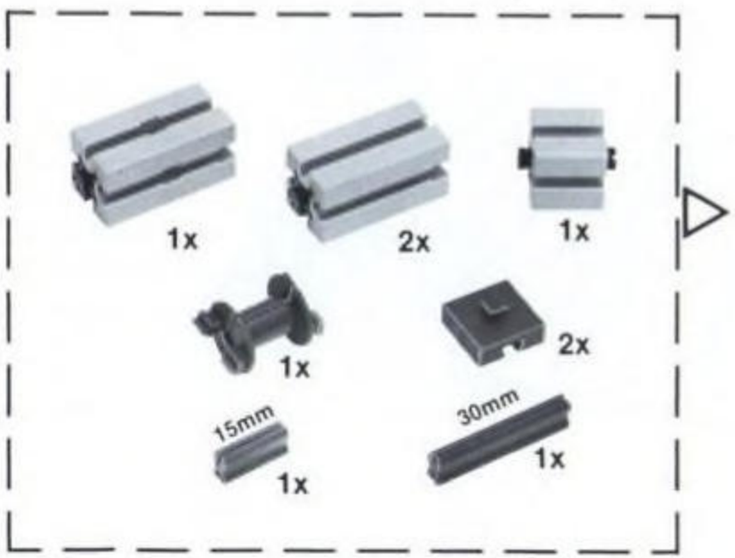
1070 FOR I = 0 TO 2 * D

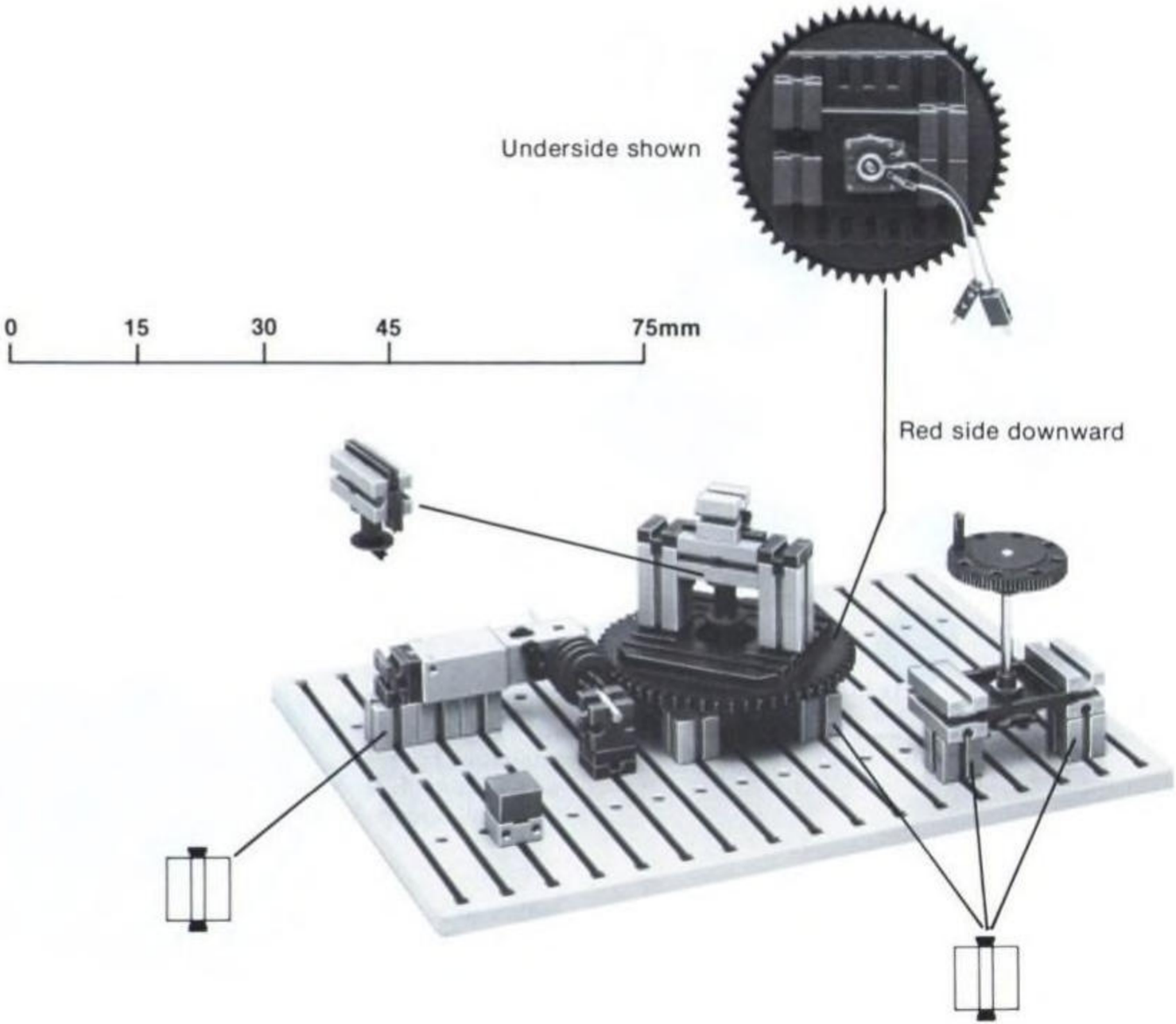
Also, try changing the range (10 in our example) in which braking action occurs (line 1060) and see what happens.

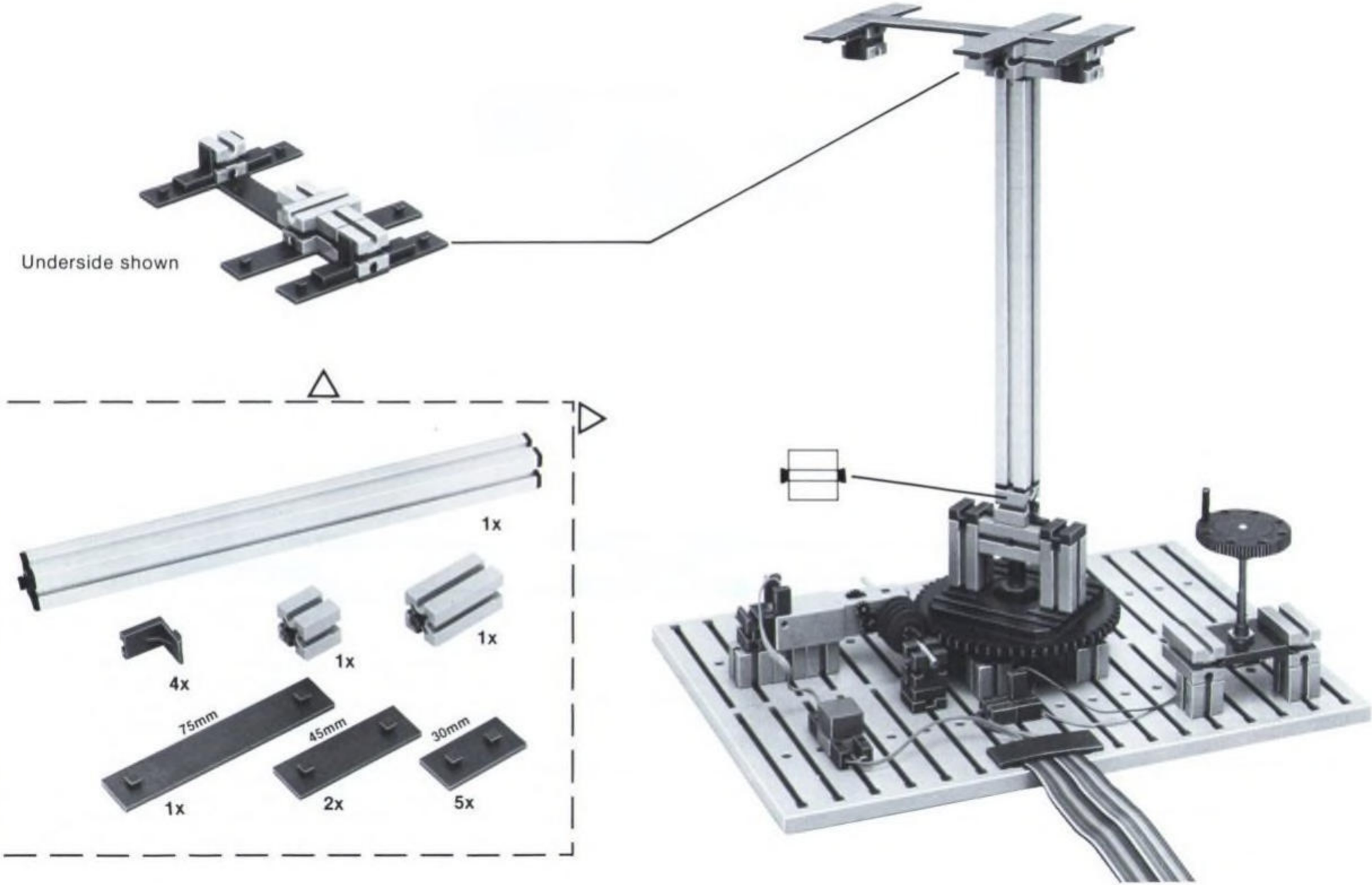
```
*500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM AERIAL ROTOR
 640 REM
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 730 REM ASSIGNMENTS FOR THE INTERFACE
 740 REM
 750 REM INPUT
 760 REM EX= POTENTIOMETER FOR CONTROL WHEEL
 770 REM EY= POTENTIOMETER FOR ROTOR
 780 REM
 790 REM OUTPUT
 800 REM M1=MOTOR FOR ROTOR
 810 REM
 820 REM FUNCTION DESCRIPTION :
 830 REM ROTOR FOLLOWS THE MOVEMENT OF THE CONTROL WHEEL
 840 REM
*900 PRINT CHR$(147)
 910 PRINT"FISCHERTECHNIK"
 920 PRINT"COMPUTING"
 930 PRINT
 940 PRINT"AERIAL ROTOR"
 950 PRINT
*1000 LET X=USR(EX) :REM POSITION OF CONTROL WHEEL
*1010 LET Y=USR(EY) :REM POSITION OF THE ROTOR
 1020 PRINT"                              "
 1030 PRINT"(UP)";
 1040 PRINT"NOMINAL=";X;"    ACTUAL=";Y
 1050 PRINT"(UP)";
 1060 LET D=ABS(X-Y)
*1070 IF Y>X THEN SYS M1,CW
*1080 IF Y<X THEN SYS M1,CCW
 1090 IF D>10 THEN 1000
 1100 FOR I=0 TO D
 1110 NEXT I
*1120 SYS M1,OFF
 1130 GOTO 1000
```


1x
1x
1x
7x
1x
2x
1x
1x
1x
1x
1x
3x
1x
2x
1x
red
1x
1x


1x
1x
1x
4x
4x


1x
2x
1x
1x
2x
15mm
1x
30mm
1x


Underside shown
0
15
30
45
75mm
Red side downward


Underside shown
1x
1x
1x
4x
75mm
45mm
30mm
1x
2x
5x

Circuit Layout—Aerial Rotor

E1 brown

E2 red

EX orange

EY yellow

+5V green

E3 blue

E4 violet

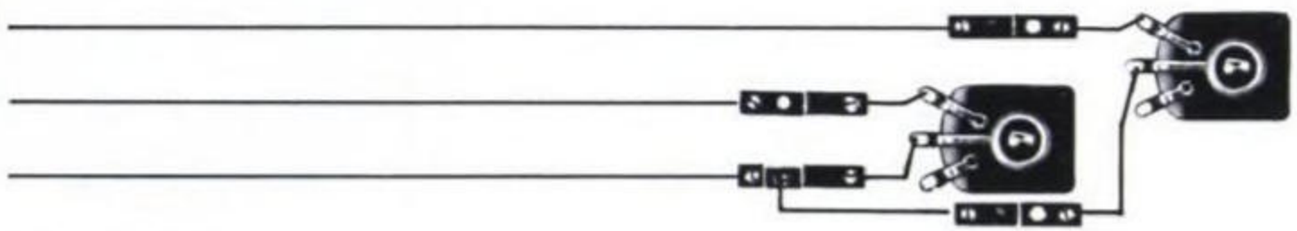
E5 grey

E6 white

E7 black

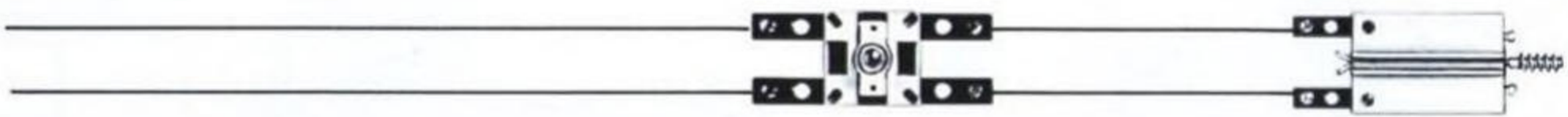
E8 brown

+5V red



M1 orange

M1 yellow



M2 green

M2 blue

M3 violet

M3 grey

M4 white

M4 black

Sorting System

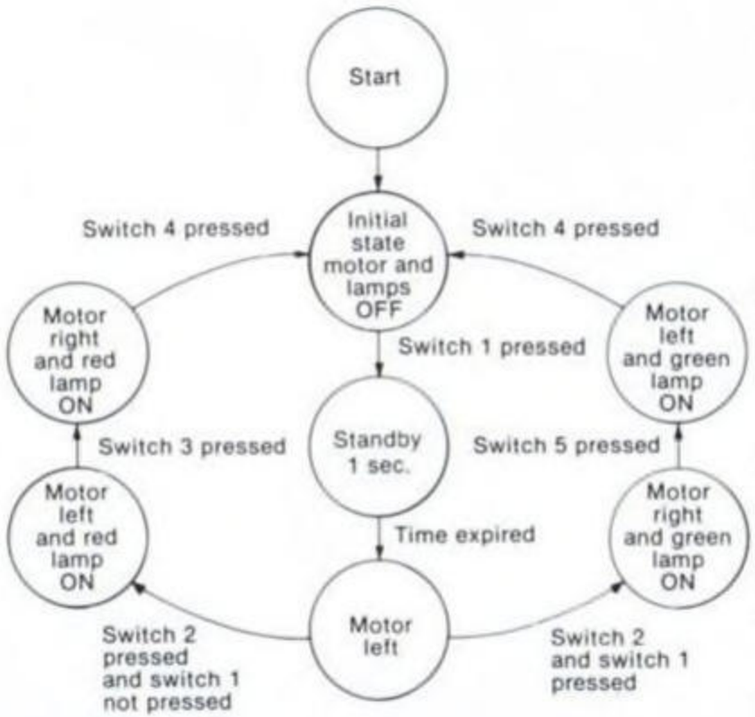
Metering and sorting are two basic tasks which occur over and over again in industry, whether the job is sorting eggs by weight or eliminating scrap from a production line. We will build a model which is able to differentiate between blocks 15 and 30 from the kit. While this may not sound impressive to everyone, it is an important principle to be studied. Besides, building the model and getting it to work will be fun!

As we said, the task consists of two basic actions: metering—or measuring in this case—the length of the blocks; then sorting the parts into two bins at either side of the model.

The part is first conveyed over the angled chute at the front of the model into the measuring area. It is then pushed to the left past the start switch into the measuring channel. Measuring really means comparing, and here we compare the unknown length of the part to the distance between the start switch and the measuring switch. If both switches are activated simultaneously by the part, it must be a block 30, since a 15 will activate only one switch.

When the length of the part has been recognized, sorting begins. In one case, the carrier which puts the piece into the measuring channel continues its motion and ejects the part into the bin at the left. Otherwise, the direction of the carrier is reversed, and the part is carried to the right-hand bin. Three switches along the track signal the arrival of the carrier at either end of its travel range, and also indicate its arrival at the center position where it begins a new cycle. You might also want to use a red and green lamp to add a visual signal to the sorting activity.

Once again, the flowchart helps us to understand the step-by-step actions of the process. In this case, we have two possible start positions indicated in the middle. Depending on which status is found on either side of the bottom circle, the procedure will then follow in that direction. As you can see, the path to the right indicates a block 30, while the left path is for a block 15. In the BASIC program, the decision is made in lines 1070-1090.



(Illustration 7)

By now you should have gathered quite a bit of experience with controlling your models. The direction of motion of the sorting carrier can be checked quickly with the commands:

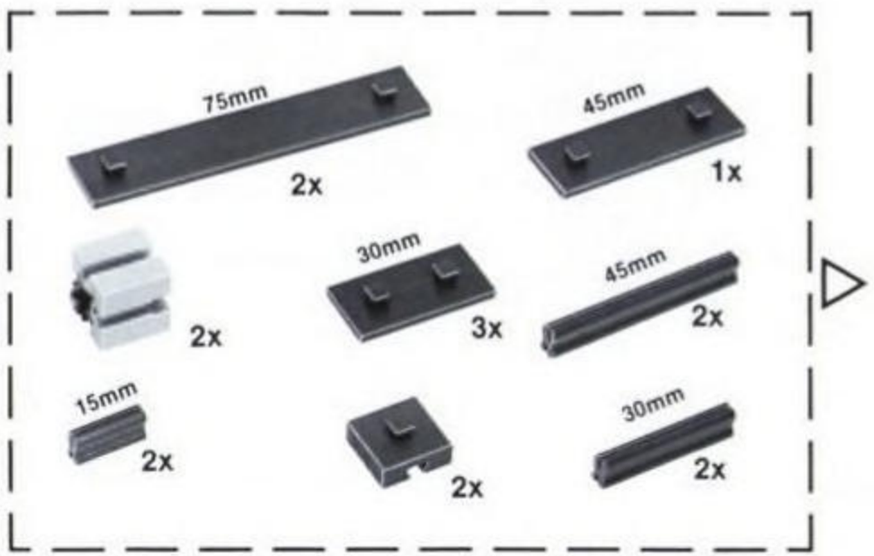
SYS M1, CW and
SYS M1, CCW

which should move the carrier in the indicated directions.

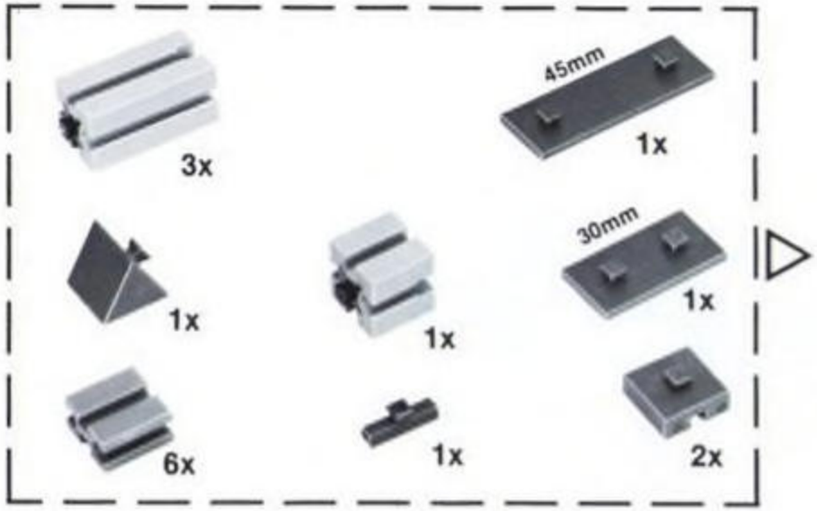
```
•500 SYS INIT
 600 REM
 610 REM FISCHERTECHNIK COMPUTING
 620 REM
 630 REM SORTING SYSTEM
 640 REM
 650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 660 REM
 730 REM ASSIGNMENTS FOR THE INTERFACE
 740 REM
 750 REM INPUT
 760 REM E1=START KEY
 770 REM E2=KEY FOR MEASUREMENT
 780 REM E3=SORT OUT SHORT PIECES
 790 REM E4=START POSITION
 800 REM E5=SORT OUT LONG PIECE
 810 REM
 820 REM OUTPUT
 830 REM M1=CARRIAGE
 840 REM M2=LAMP FOR SHORT BRICK
 850 REM M3=LAMP FOR LONG BRICK
 860 REM
 870 REM FUNCTION DESCRIPTION :
 880 REM SORTING SYSTEM SORTS OUT THE
 885 REM SHORT AND LONG FISCHERTECHNIK BRICKS
 890 REM
•900 PRINT CHR$(147)
 910 PRINT"FISCHERTECHNIK"
 920 PRINT"COMPUTING"
 930 PRINT
 940 PRINT"SORTING SYSTEM"
 950 PRINT
 960 PRINT"SYSTEM READY"
 970 PRINT
•1000 SYS INIT
•1010 IF USR(E1)=1 THEN GOTO 1010 :REM WAIT FOR START KEY
 1020 REM DELAY LOOP
 1030 FOR I=1 TO 500
 1040 NEXT I
•1050 SYS M1,CCW :REM GET PIECE FOR MEASUREMENT
•1060 IF USR(E1)=0 AND USR(E2)=0 THEN GOTO 1150
      :REM MEASUREMENT OF LONG BRICK
•1070 IF USR(E1)=1 AND USR(E2)=1 THEN GOTO 1090
      :REM MEASUREMENT OF SHORT BRICK
 1080 GOTO 1060
 1090 PRINT"SHORT BRICK"
•1100 SYS M2,CW :REM SWITCH ON INDICATOR FOR SHORT BRICK
•1110 IF USR(E3)=0 THEN GOTO 1110
•1120 SYS M1,CW :REM CARRIAGE RETURN
•1130 IF USR(E4)=0 THEN GOTO 1130
 1140 GOTO 1000
 1150 PRINT"LONG BRICK"
•1160 SYS M1,CW :REM SORT OUT
•1170 SYS M3,CW: REM SWITCH ON INDICATOR FOR LONG BRICK
•1180 IF USR(E5)=0 THEN GOTO 1180
•1190 SYS M1,CCW: REM CARRIAGE RETURN
•1200 IF USR(E4)=0 THEN GOTO 1200
 1210 GOTO 1000
```


1x
3x
3x
3x

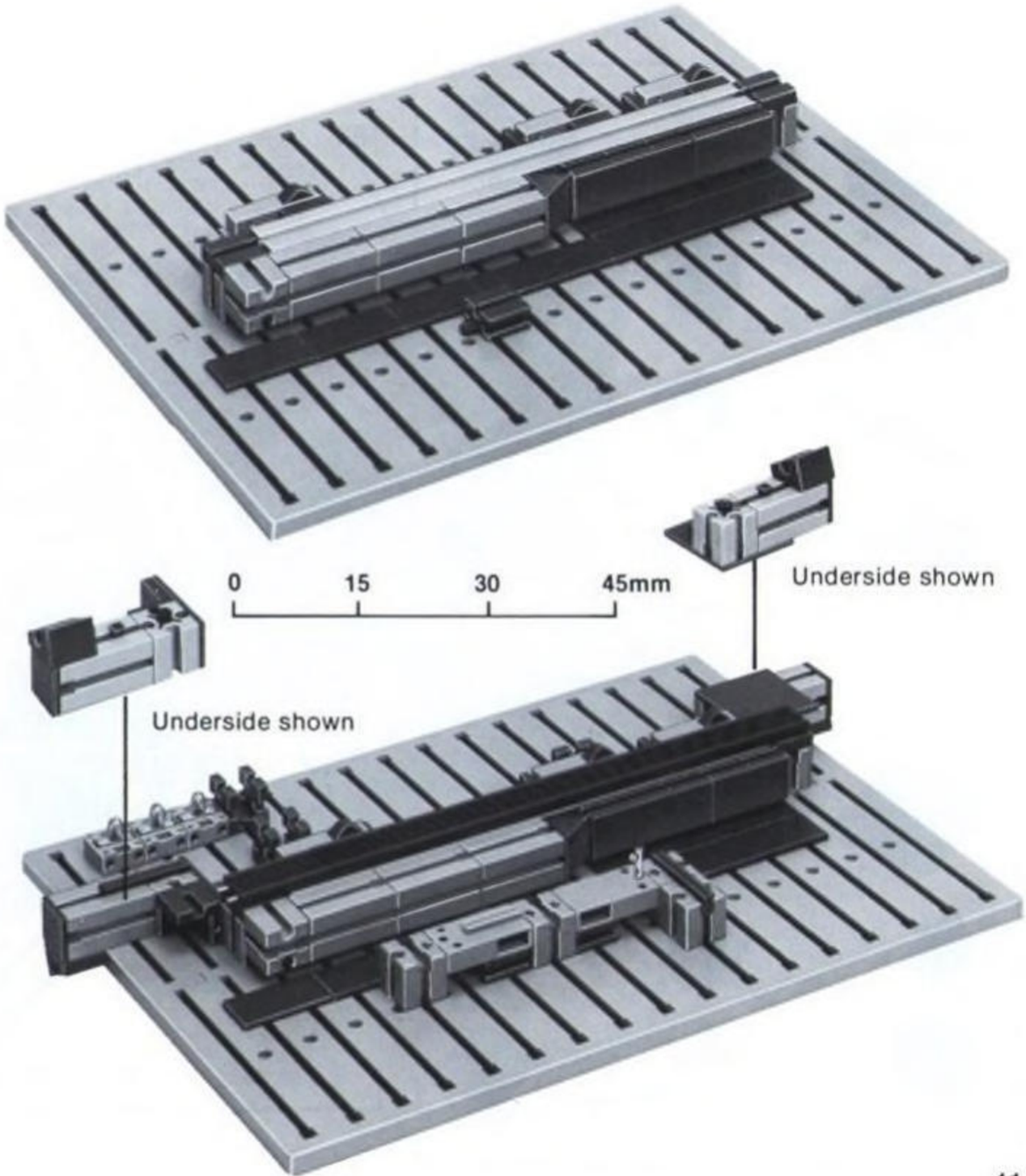

0
15mm
30
45
75mm


75mm
2x
45mm
1x
2x
30mm
3x
45mm
2x
15mm
2x
2x
30mm
2x

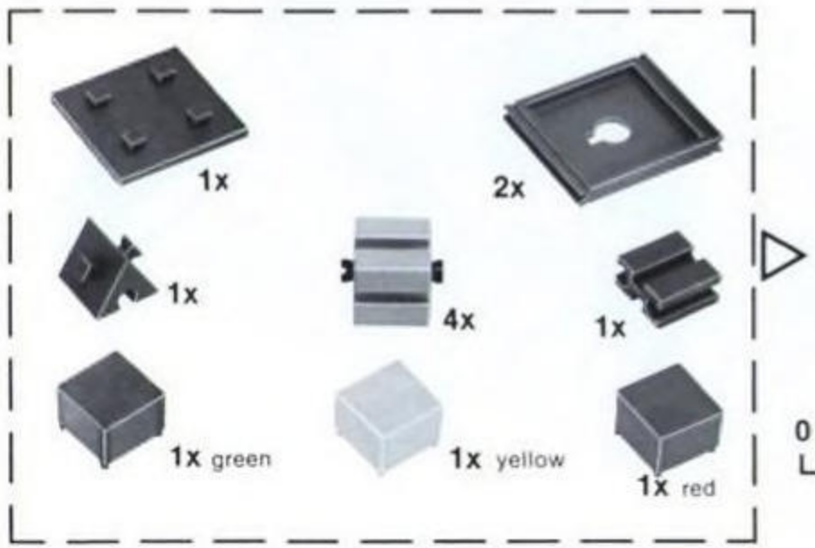

Underside shown

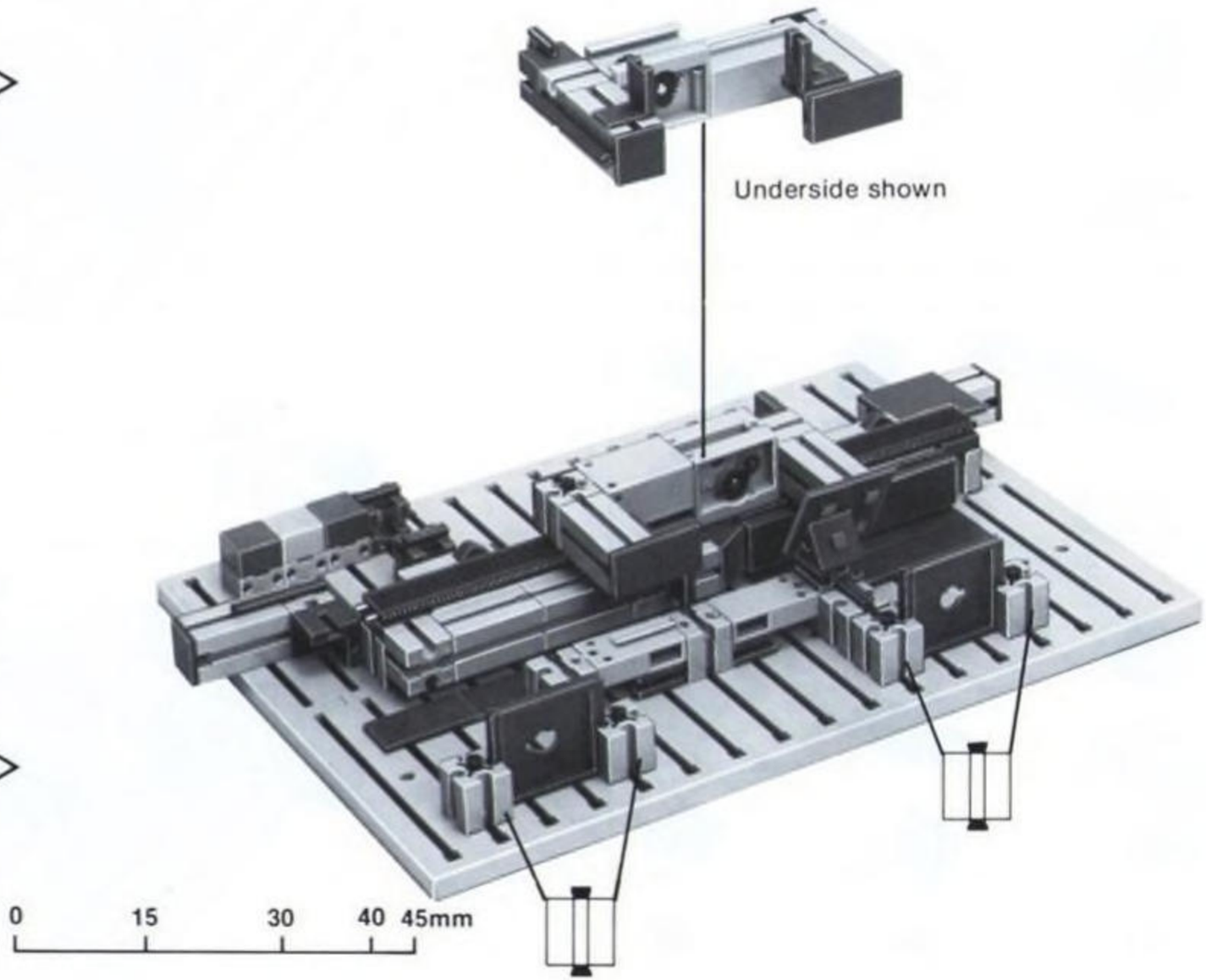

3x
45mm
1x
1x
1x
30mm
1x
6x
1x
2x

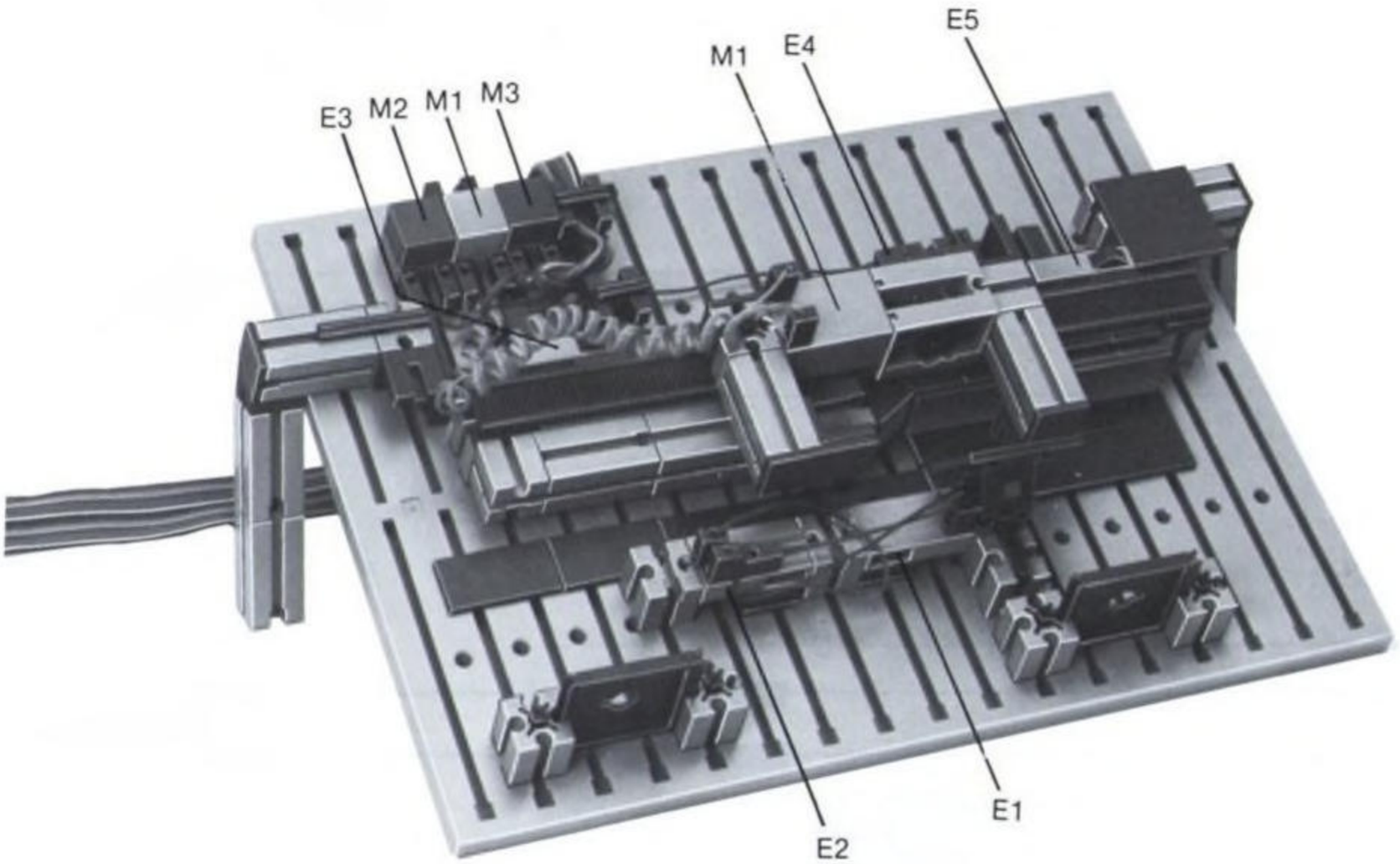

3x
3x
1x
2x
6x
2x
1x
1x
1x
30mm
2x
30mm
1x
3x
15mm
3x
2x
2x
1x
3x
2x


0
15
30
45mm
Underside shown
Underside shown

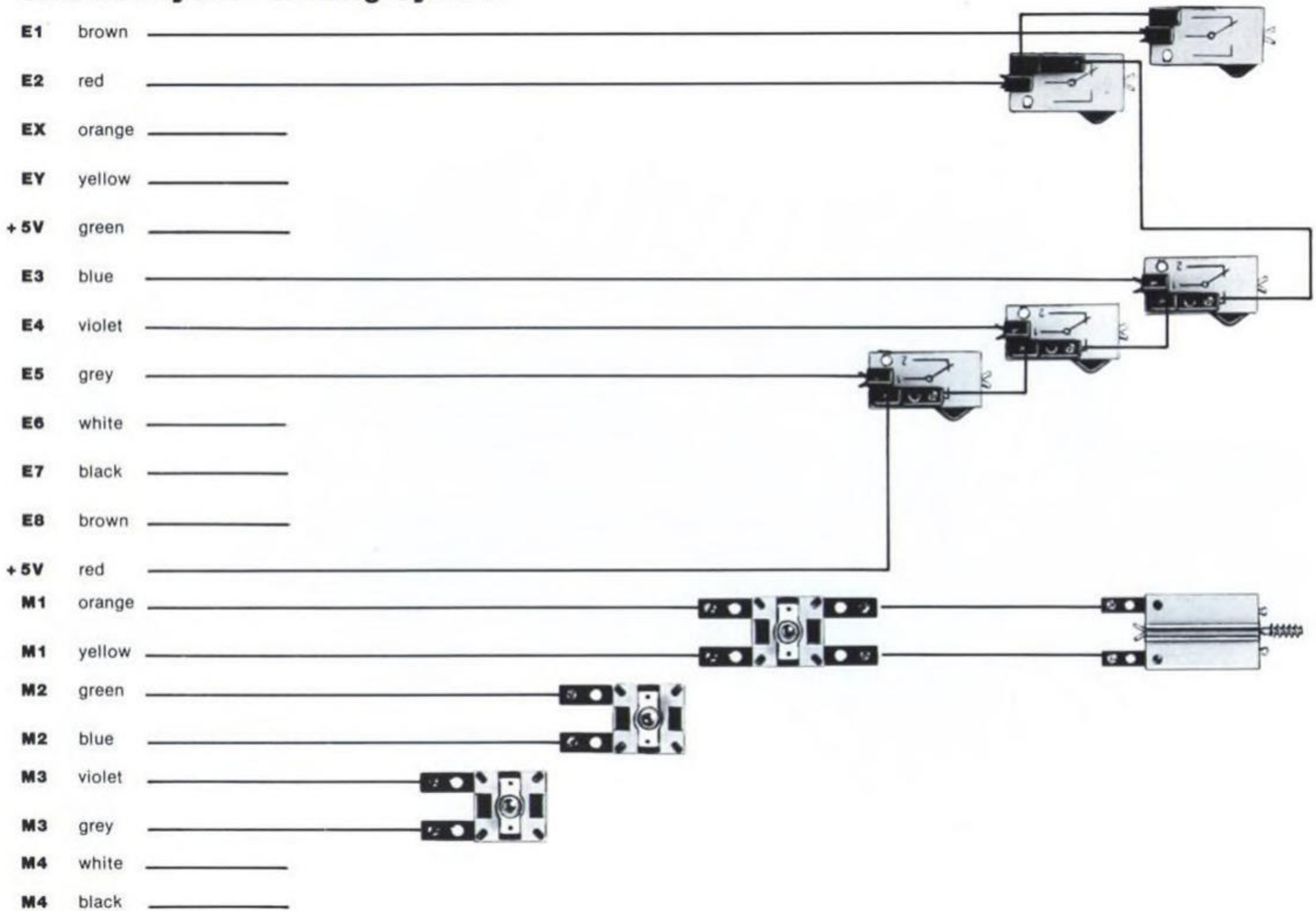

1x
1x
30mm
1x
2x
2x
45mm
1x
2x
2x
2x
40mm
1x
15mm
1x


1x
2x
1x
4x
1x
1x green
1x yellow
1x red


Underside shown
0
15
30
40
45mm


E3
M2
M1
M3
M1
E4
E5
E2
E1

Circuit Layout—Sorting System



The Towers of Hanoi

At last we come to our first robot. This fellow will be able to rotate around his vertical axis and to move his arm up and down. Using a magnet as fingers, he will be able to pick up ferrous metal parts and deposit them in another place. The circular metal plates from the kit are ideal for this purpose, although other magnetic items may be used. For our robot's actions, we have selected a sequence of motion based on an old puzzle.

According to ancient texts, the monks of certain Buddhist monasteries are to solve a puzzle that requires extreme measures of patience. There are three slender posts on a board, made of copper, silver, and gold. The copper rod holds a pile of 100 disks, positioned in size order with the smallest on top. The job is to transfer all of the disks to the golden rod, observing the following rules:

1. Only one disk may be moved at a time.
2. A larger disk may never be placed on top of a smaller one.
3. At each move, the disk must be placed on one of the three rods.

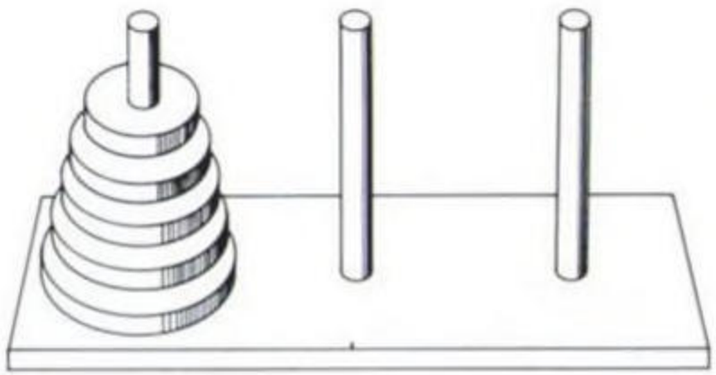
The legend also says that when the problem has been solved and all the disks are transferred, the world will come to an end!

In order not to hasten the end of the world with our experiments, we have modified the problem by using only 19 disks.

For stacking them, we have one post each at the front and rear of the model, as well as one at the center. The arm of the robot is adjustable in length, so that the magnet can operate over various radii.

You will find that trying to solve the problem with even five disks will soon prove hopeless. It would be better for you to think your way through the following explanation:

To bring the lowest (largest) disk from the copper post to the gold one, you first of all have to move all the disks from above it onto the silver post *somehow*, keeping in mind the rules of the game. Then all you have to do is move the rest of the disks from the silver post onto the gold. Sound too easy? Right you are. This is the tricky part, but you're close to the solution.



(Illustration 8)

Let's assume that we're working with five disks. We already know how to move the bottom disk of this pile by stacking the other four disks alternately from one post to the other. This leaves us with a pile of four to move. Follow the same alternating process, and then there are three, and then two, and then one. Surely we know how to move one little disk? Extend the same process, and we can claim to be able to solve the Towers of Hanoi for any number of disks. The computer program operates in exactly this way.

Just a word at this point regarding the operating time of the program, lest you be tempted to try it with too large a pile of disks. If we review the action of the recursive program (one that calls itself in a circular fashion to repeat a calculation using the previous result), we find that the number of moves required for N disks is $2^n - 1$. Thus, the robot must perform 31 moves for only five disks. With 100 disks, you would have

126,765,060,000,000,000,000,000,000,000

moves. Even the quickest monk, working at one move per second, would have to work for four hexillion years in order to cause the end of the world. Isn't *that* a relief?

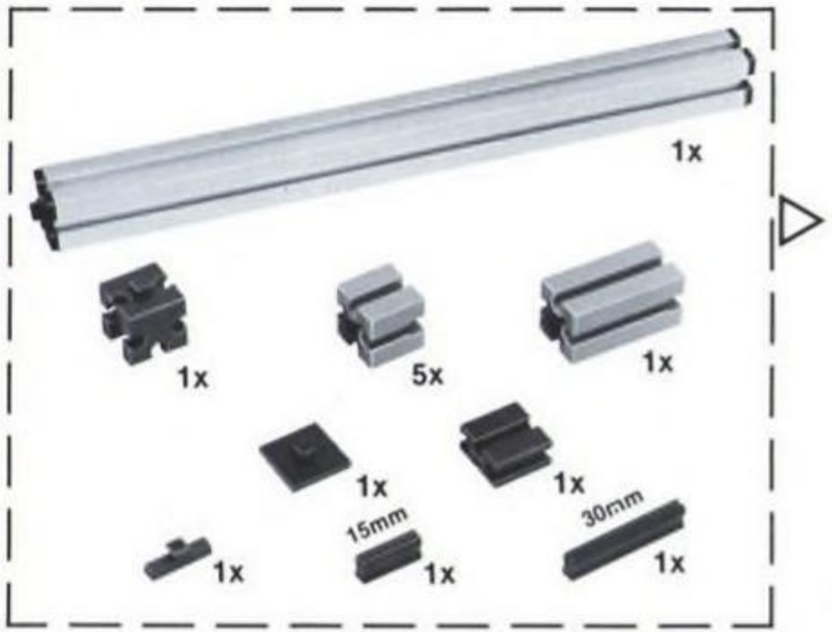
The BASIC program itself is pretty clear. The first part of it from line 1000 to line 1850 contains the general program with the initial conditions, and the recursion routine from line 1690. This part may be used alone for other projects, since the control program for the robot begins at line 1780. This procedure can be used to convert other games, like NIM, with the robot as a partner.

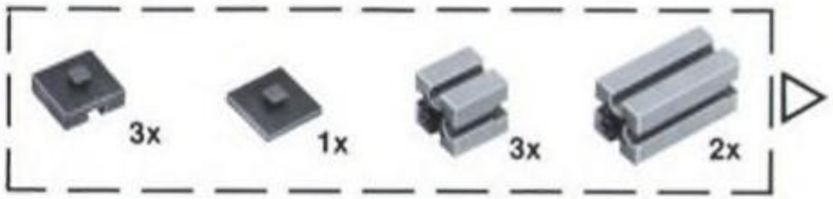
In the robot control routines, you will find familiar elements, like the direction control routine that starts at line 2200 and controls the rotation of the robot.

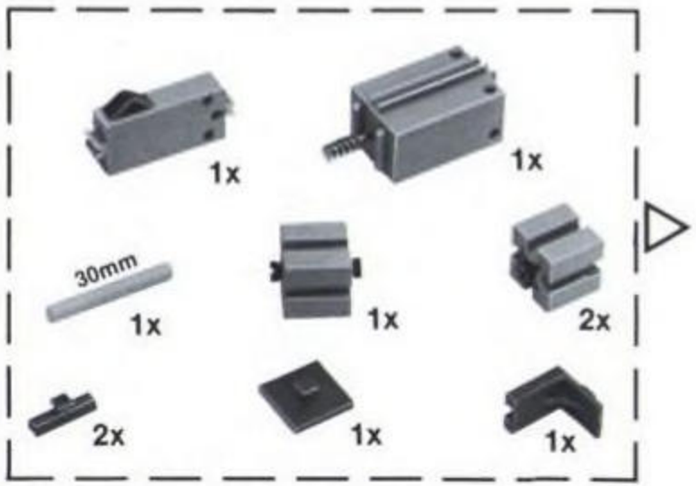
Actually, the desired position is defined by the column positions set in lines 1010-1040. Before beginning the program, you should determine these settings by use of the DIAGNOSTIC program. The same program can also test the operation of the limit switch for vertical motion.

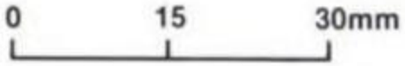
The subroutine that switches off the magnet has a special feature. Before the magnet shuts off, it operates for a brief moment with a reversed current flow. In this way, residual magnetization of the metal disks is avoided. You may have to adjust the force of the magnet by sticking a bit of tape on the pole pieces until you find the parts are gripped safely, yet release again when desired. Once you test the direction of the drive motors to be sure it is as expected, you can start the sorting process.

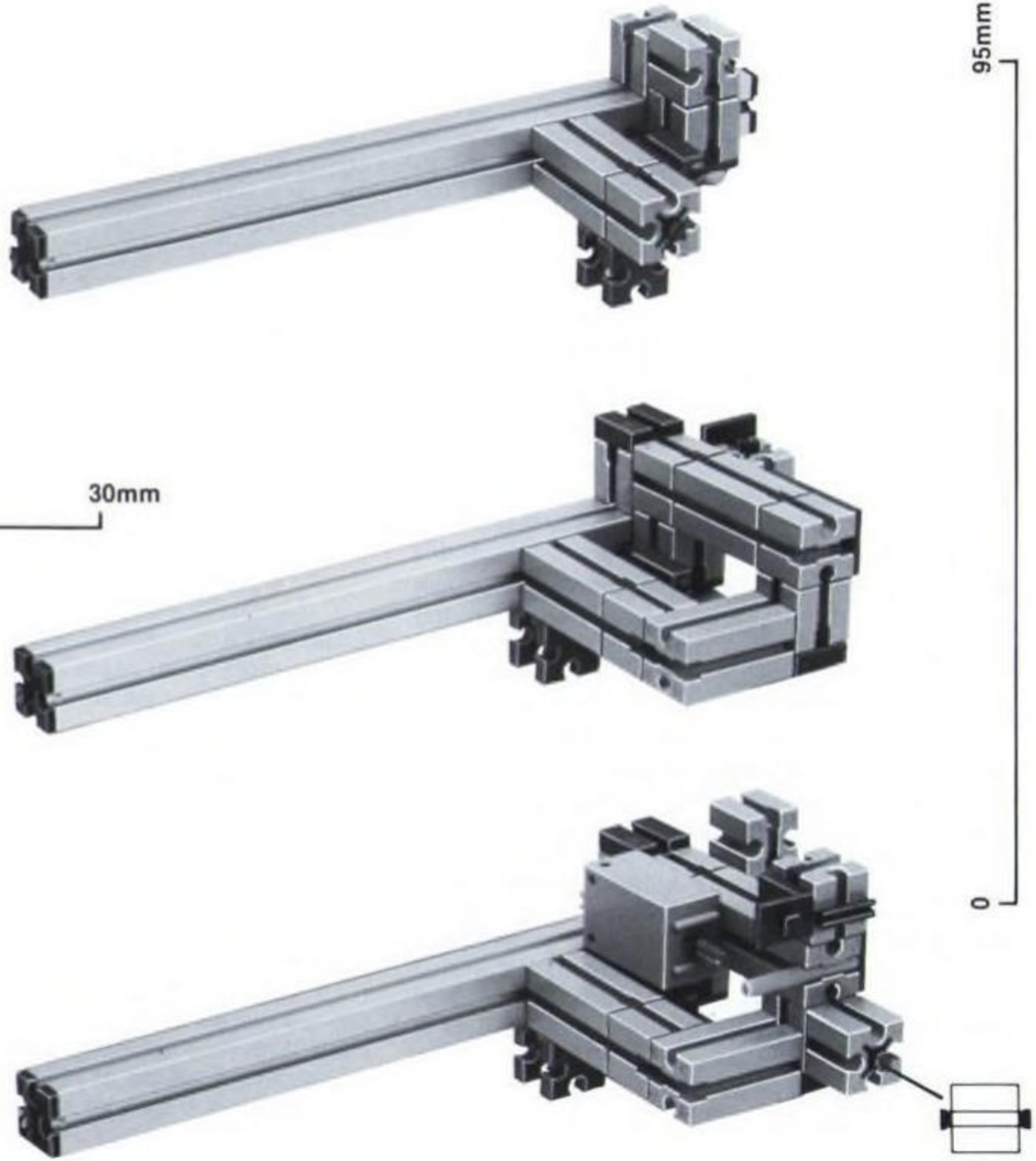
```
• 500 SYS INIT
  600 REM
  610 REM FISCHERTECHNIK COMPUTING
  620 REM
  630 REM TOWER OF HANOI
  640 REM
  650 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
  660 REM
  730 REM ASSIGNMENTS FOR THE INTERFACE
  740 REM
  750 REM INPUT
  760 REM E1=BOTTOM
  770 REM E2=TOP
  780 REM EY=ANGLE POSITION
  790 REM
  800 REM OUTPUT
  810 REM M1=BODY
  820 REM M2=ARM
  830 REM M3=MAGNET
  840 REM
  850 REM FUNCTION DESCRIPTION :
  860 REM SORTING OF THE PILE OF DISCS FOLLOWING THE
  870 REM PUZZLE ORIGINATING FROM BUDDHIST TRADITION
  880 REM
 1000 DIM X(19),Y(19),Z(19)
 1010 REM POSITION OF PILES (EMPIRICAL VALUES)
 1020 LET PILE(1)=85
 1030 LET PILE(2)=110
 1040 LET PILE(3)=135
• 1500 PRINT CHR$(147)
 1510 PRINT"FISCHERTECHNIK"
 1520 PRINT"COMPUTING"
 1530 PRINT
 1540 PRINT"TOWER OF HANOI"
 1550 PRINT
 1600 INPUT"HOW MANY DISCS HAS GOT THE TOWER";N
 1610 IF N<1 THEN END
 1620 LET M=0
 1630 LET X(M)=N:LET Y(M)=1:LET Z(M)=3
 1640 GOSUB 1700
 1650 END
 1690 REM PSEUDORECURSIVE ROUTINE
 1700 IF  X(M)=0 THEN RETURN
 1710 LET M=M+1
 1720 LET X(M)=X(M-1)-1
 1730 LET Y(M)=Y(M-1)
 1740 LET Z(M)=6-Y(M-1)-Z(M-1)
 1750 GOSUB 1700 :REM RECURSION
 1760 LET M=M-1
 1770 PRINT"DISC";X(M);"FROM PILE";Y(M);" TO PILE ";Z(M)
 1780 GOSUB 2000 :REM ROBOT ROUTINE
 1790 LET M=M+1
 1800 LET X(M)=X(M-1)-1
 1810 LET Y(M)=6-Y(M-1)-Z(M-1)
 1820 LET Z(M)=Z(M-1)
 1830 GOSUB 1700 :REM RECURSION
 1840 LET M=M-1
 1850 RETURN
 2000 REM ROBOT ROUTINE
 2010 LET NOMINAL=PILE(Y(M))
 2020 GOSUB 2200 :REM TURN BODY
 2030 GOSUB 2310 :REM LOWER ARM
• 2040 SYS M3,CW  :REM SWITCH MAGNET ON
 2050 GOSUB 2350 :REM LIFT ARM
 2060 LET NOMINAL=PILE(Z(M))
 2070 GOSUB 2200 :REM TURN BODY
 2080 GOSUB 2310 :REM LOWER ARM
 2090 GOSUB 2120 :REM SWITCH MAGNET OFF
 2100 GOSUB 2350 :REM LIFT ARM
 2110 RETURN
 2120 REM MAGNET-OFF ROUTINE
• 2130 SYS M3,CCW :REM REVERSE MAGNET CURRENT FOR
      DEMAGNETIZATION
• 2140 SYS M3,OFF :REM SWITCH MAGNET OFF
 2150 RETURN
 2200 REM POSITIONAL CONTROL ROUTINE
• 2210 LET D=USR(EY)-NOMINAL
• 2220 IF D>0 THEN SYS M1,CW
• 2230 IF D<0 THEN SYS M1,CCW
 2240 IF D=0 THEN RETURN
 2250 LET D=ABS(D)
 2260 IF D>5 THEN GOTO 2210
 2270 FOR I=0 TO D
 2280 NEXT
• 2290 SYS M1,OFF
 2300 GOTO 2210
• 2310 SYS M2,CCW :REM LOWER ARM
• 2320 IF USR(E1)=0 THEN 2310
• 2330 SYS M2,OFF
 2340 RETURN
• 2350 SYS M2,CW :REM LIFT ARM
• 2360 IF USR(E2)=0THEN 2360
• 2370 SYS M2,OFF
 2380 RETURN
```

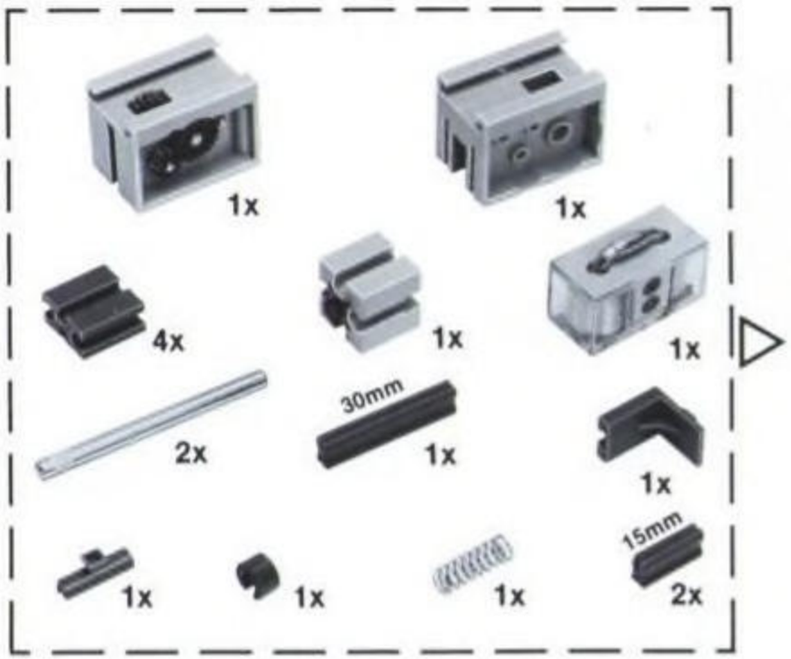

1x
1x
5x
1x
1x
1x
1x
15mm
1x
30mm
1x

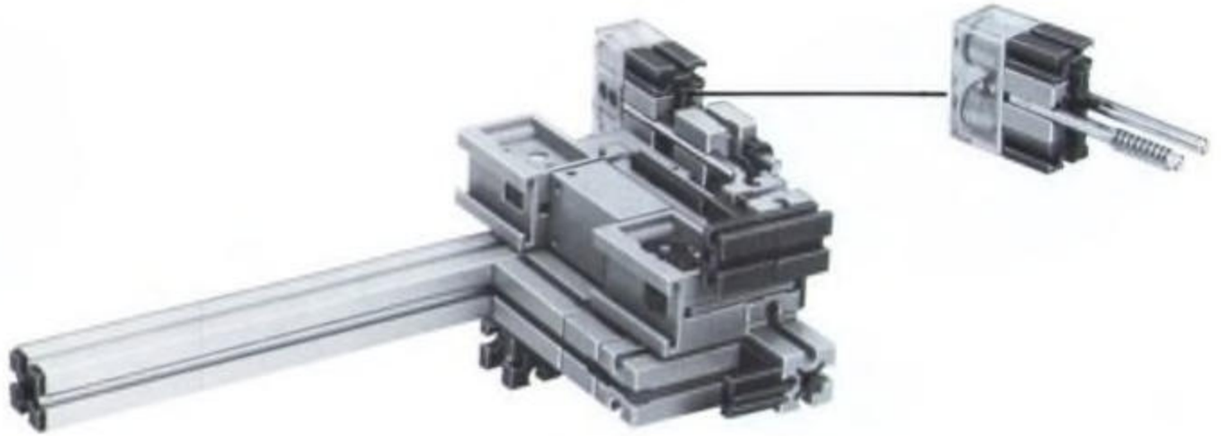

3x
1x
3x
2x

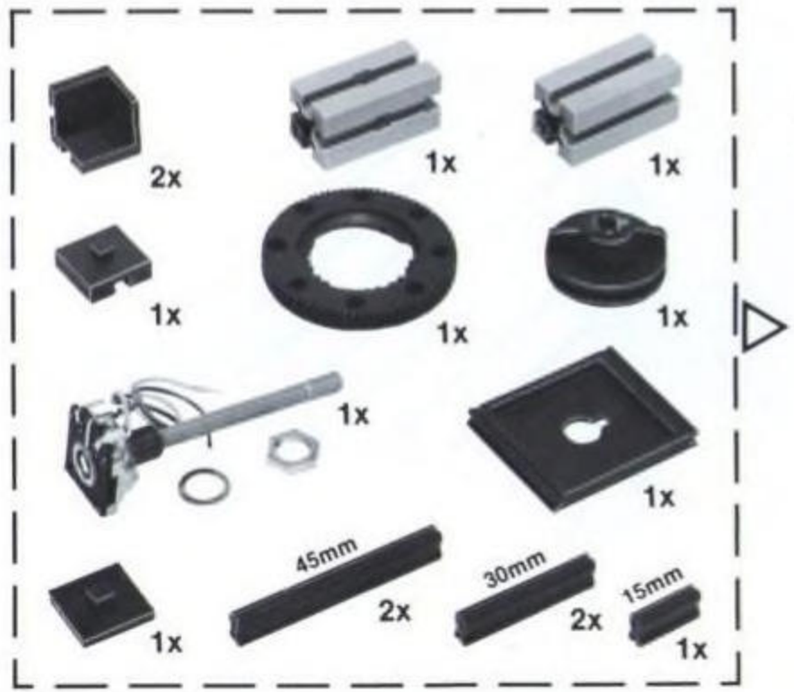

1x
1x
30mm
1x
1x
2x
2x
1x
1x

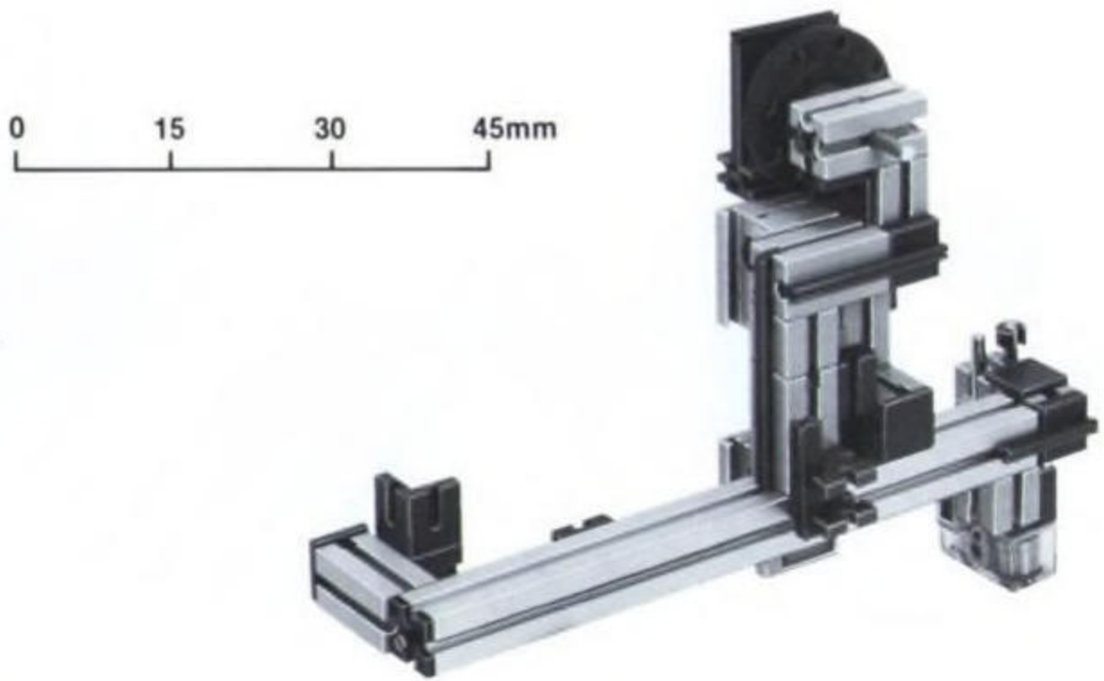

0
15
30mm


95mm
0

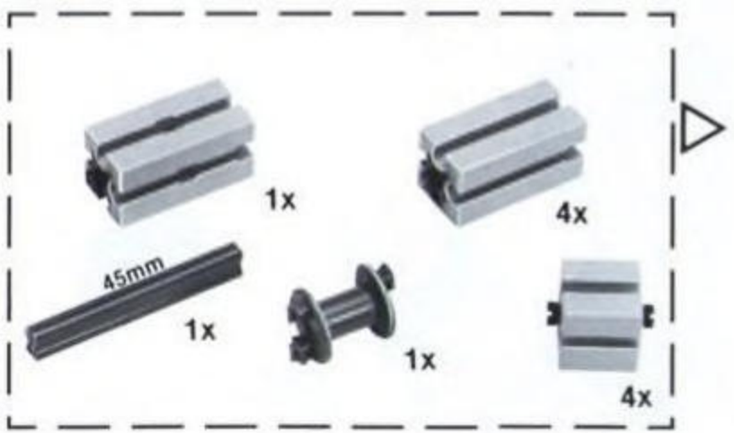

1x
1x
4x
1x
1x
2x
30mm
1x
1x
1x
1x
1x
15mm
2x



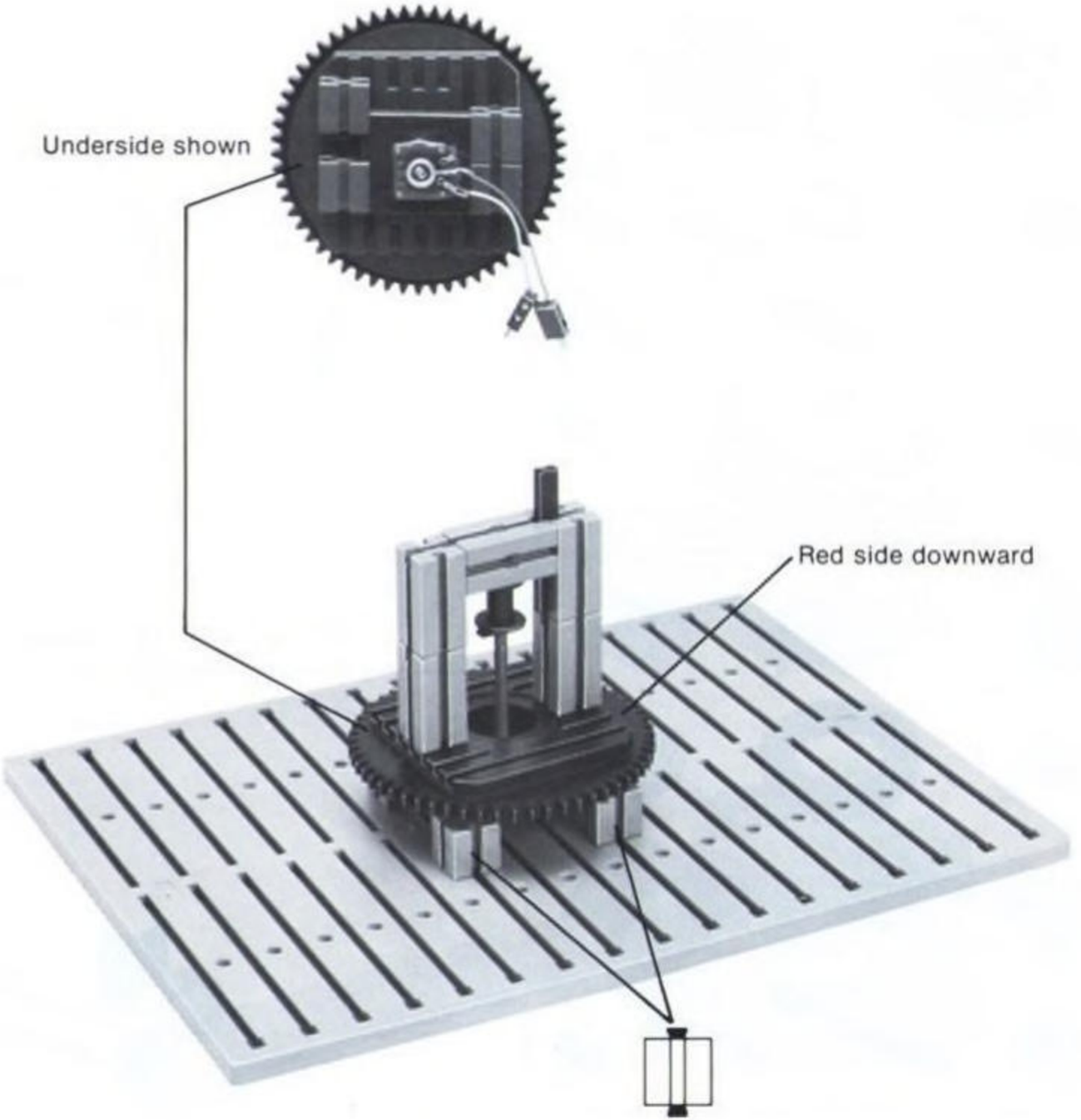

2x
1x
1x
1x
1x
1x
1x
1x
1x
45mm
2x
30mm
2x
15mm
1x

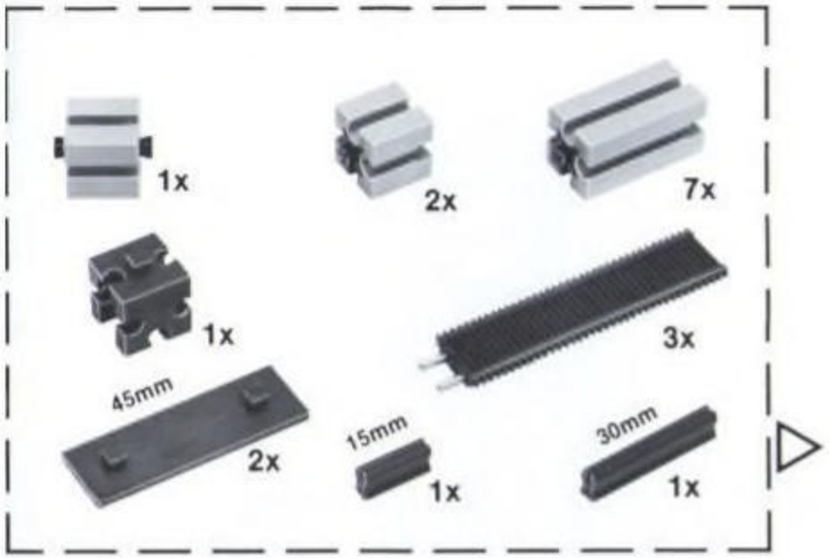

0
15
30
45mm


1x
1x
1x
4x
4x

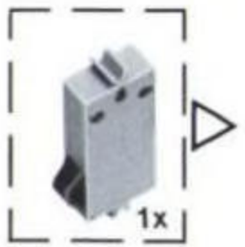

1x
4x
45mm
1x
1x
4x

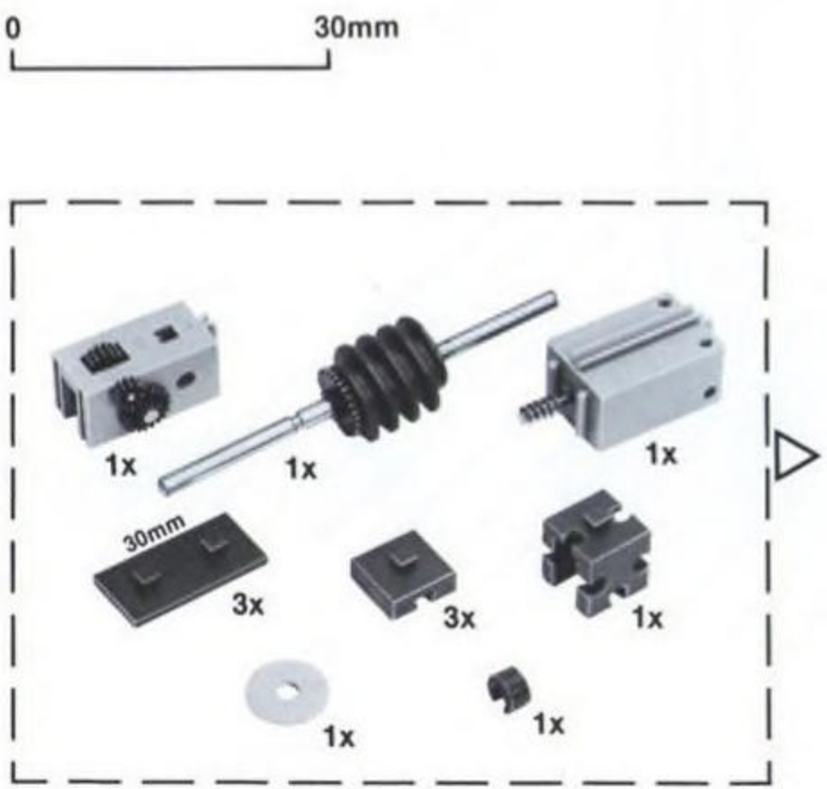

0
45mm

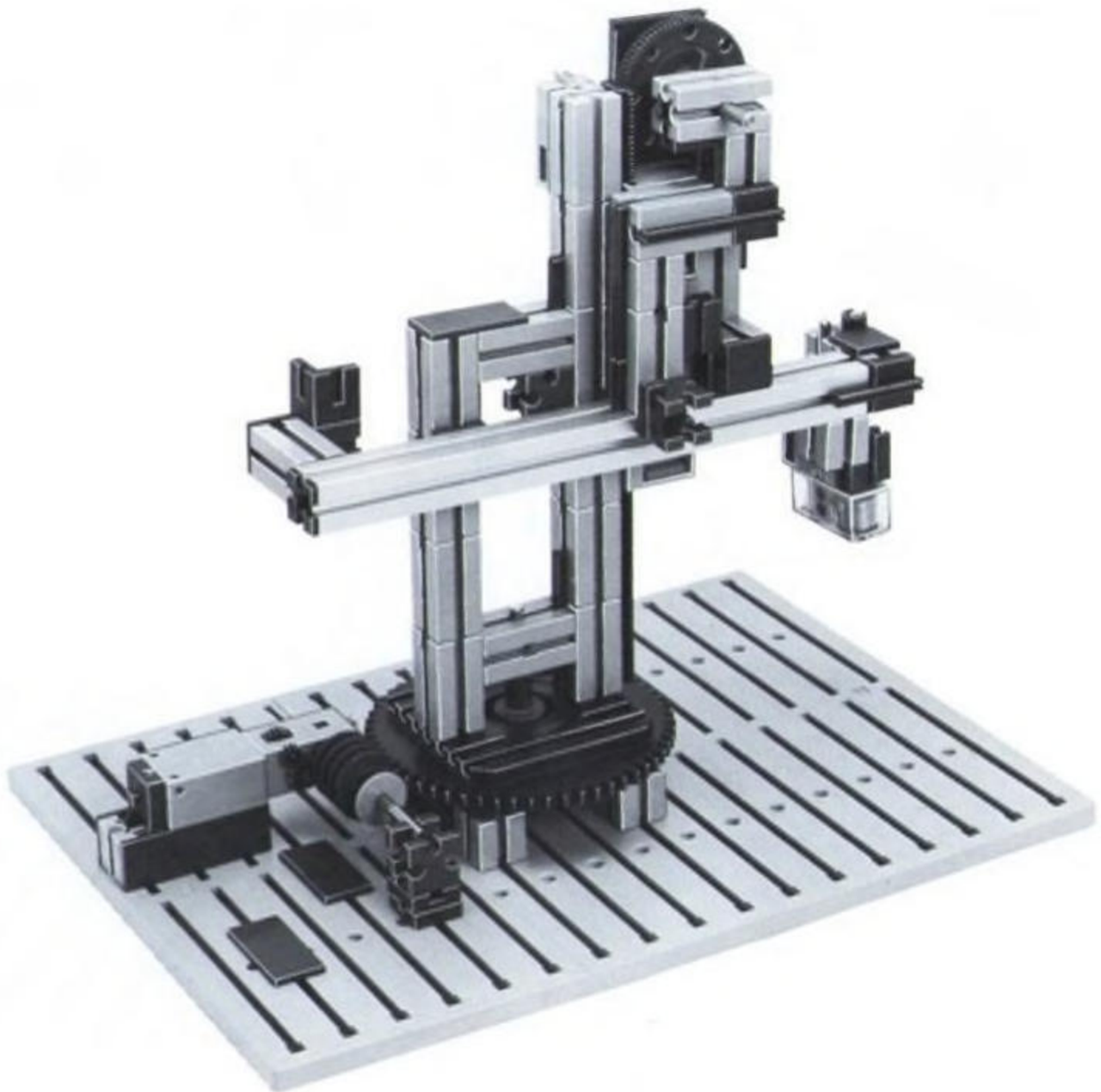

Underside shown
Red side downward

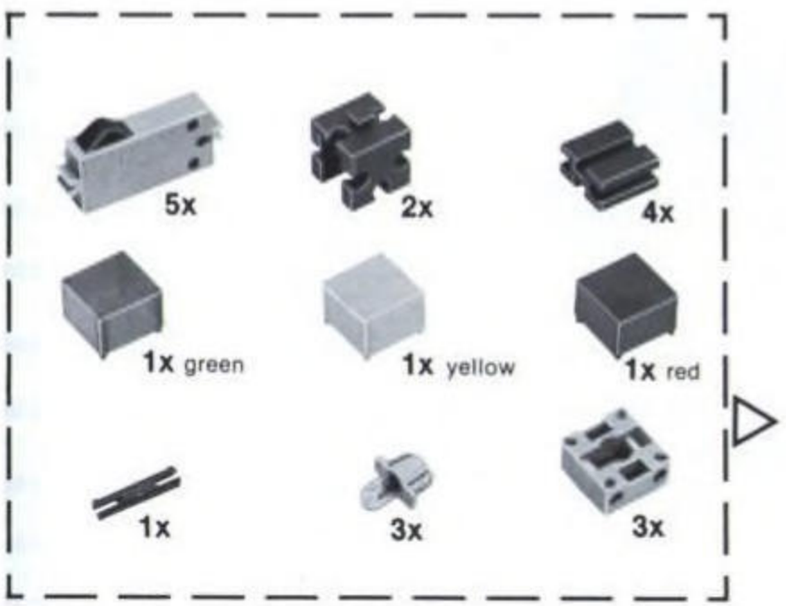

1x
2x
7x
1x
3x
45mm
2x
15mm
1x
30mm
1x

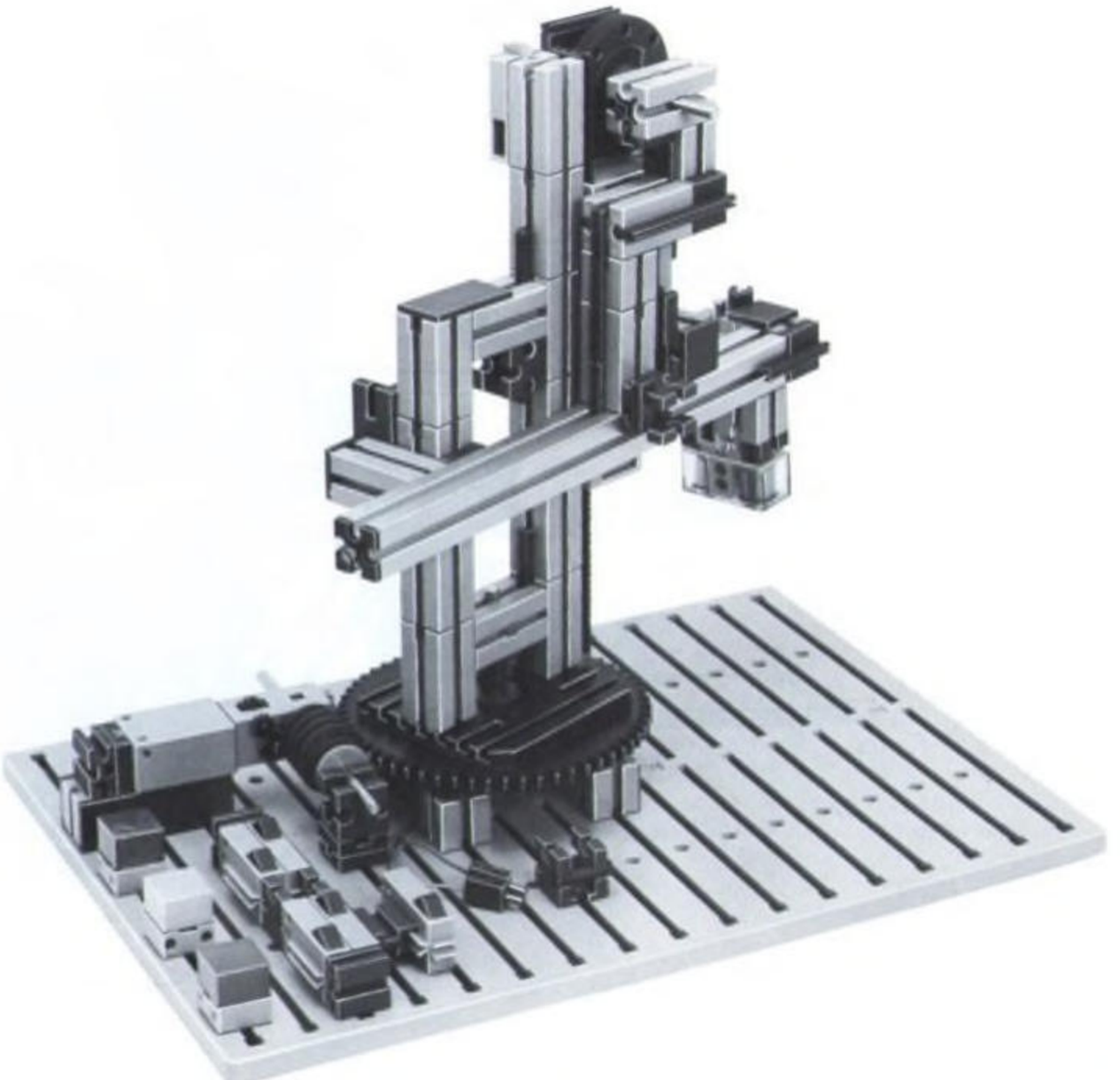

0
15
30
45mm

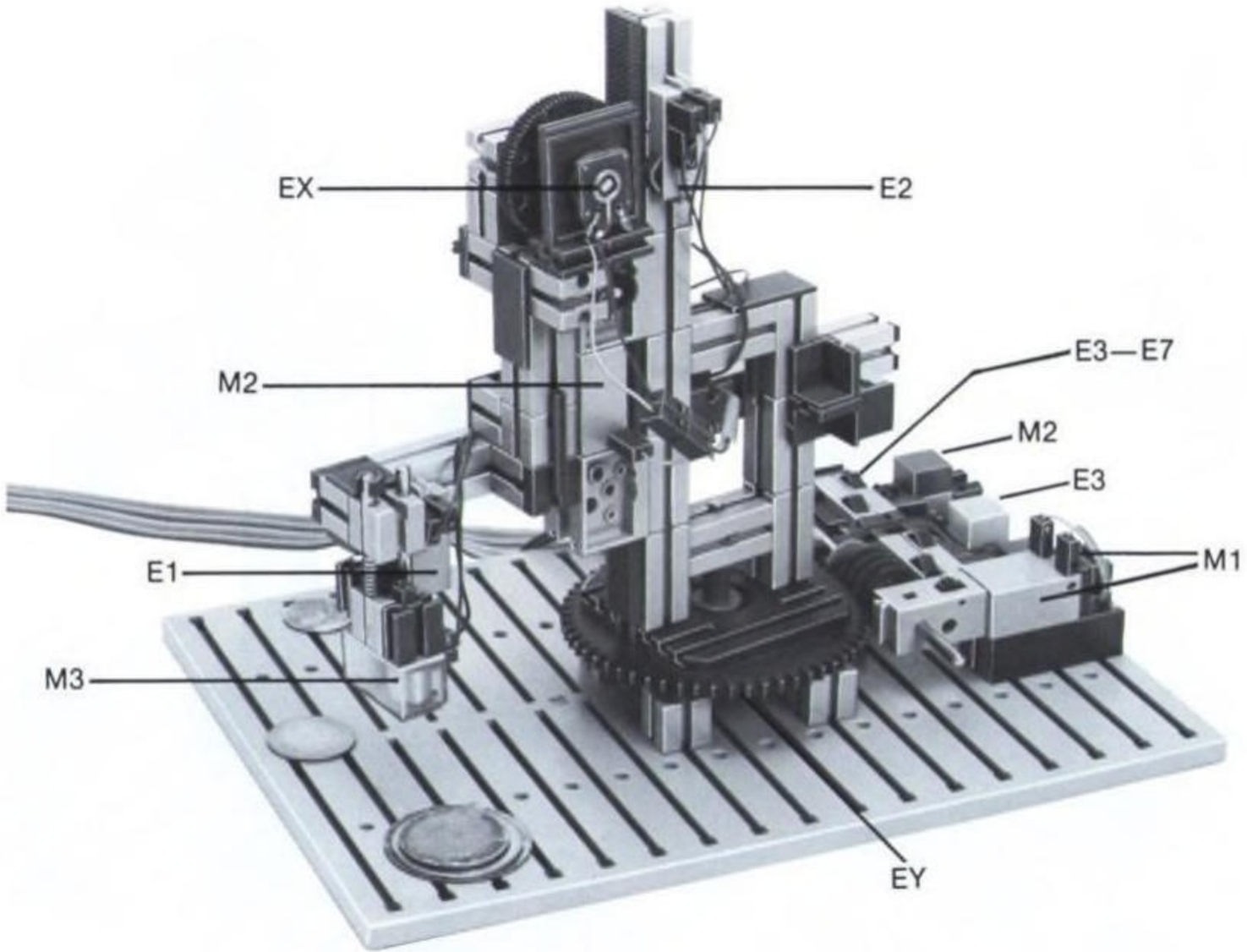

1x


0
30mm
1x
1x
1x
30mm
3x
3x
1x
1x
1x

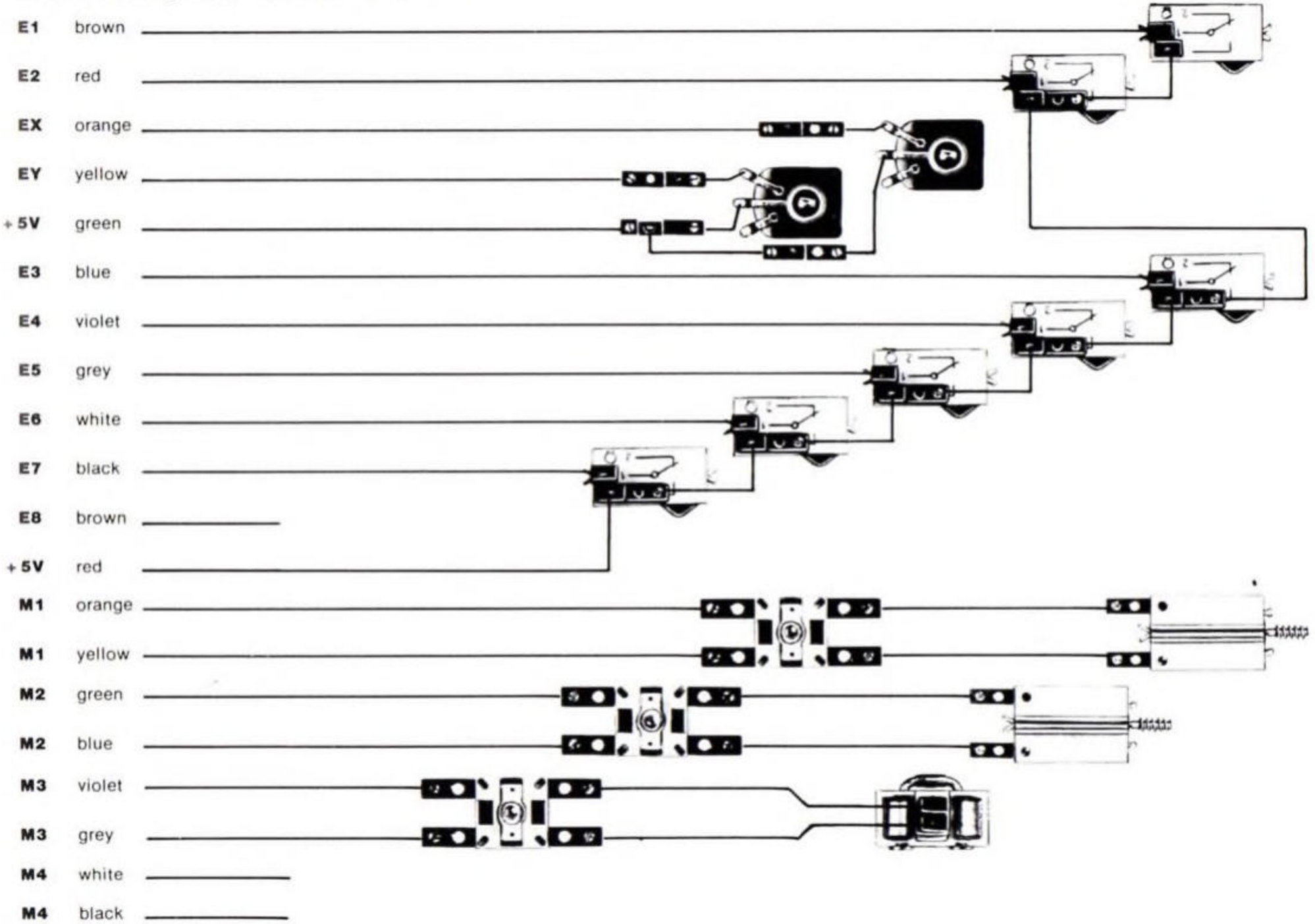



5x
2x
4x
1x green
1x yellow
1x red
1x
3x
3x




EX
E2
M2
E3—E7
M2
E3
M1
E1
M3
EY

Circuit Layout—Towers of Hanoi



A Teachable Robot

Industrial robots are most often used in situations where a sequence of actions must be performed again and again without operator fatigue and with unfailing accuracy. In addition, this work must often be done under environmental conditions that would be unhealthy for human workers—radiation areas, heat, dust, paint spray, etc. Another usual requirement for the use of robots is that the job to be done changes from time to time. If this were not the case, then a dedicated machine, rather than a robot, would most likely be used. This leads us to a definition of the industrial robot—its sequence of actions can be programmed and that program changed when required.

Two programming methods are commonly used. In one, the programmer sits at his or her desk and plans the program from beginning to end. When the program is written, it is loaded into the computer that controls the robot and then tested and modified as needed until the desired result is obtained. This method is time consuming, but it is used in those cases where the movements of the robot can be defined with mathematical calculation.

More frequently, however, the so-called "teaching" procedure is being used. In this case, the robot is manually controlled by an operator using the command keyboard. The program in the computer records the step-by-step instructions, and once the programming is complete, will repeat these motions and perform the task over and over until a new program is entered. The obvious advantage is the direct programming of the robot at the workplace, which can be carried out even by someone without knowledge of computers and programming. The limitation to this method is that it is restricted to simple unbranching sequences of motion.

Let's explore this teachable robot concept further. Our model will be controlled by six switches on the control panel. Their assignments are:

E1 – arm up
E2 – arm down
E3 – turn left
E4 – turn right
E5 – magnet ON
E6 – magnet OFF

Two additional switches have been given special functions. The switch E7 = LEARN and causes the computer to store the position of the robot when it is pressed. Switch E8 = Start/Stop, and it either starts or stops the movements of the robot.

As you might guess from the explanation, the program consists of two parts: the LEARN module for recording the sequence of motions, and the PERFORMANCE module for repeating what has been learned.

The LEARN module covers the first part of the program from lines 1000-1680, with the program testing all eight switches of the control panel over and over. If one of the motor control switches has been pressed, the motor is driven in the desired direction of rotation. The switch will be checked again and again until it is released, so it is possible to position the robot quite precisely without the need to press the switch repeatedly. In the case of the magnet, the activation of the switch is stored in variable MG$, and the magnet remains on until an OFF command is noted.

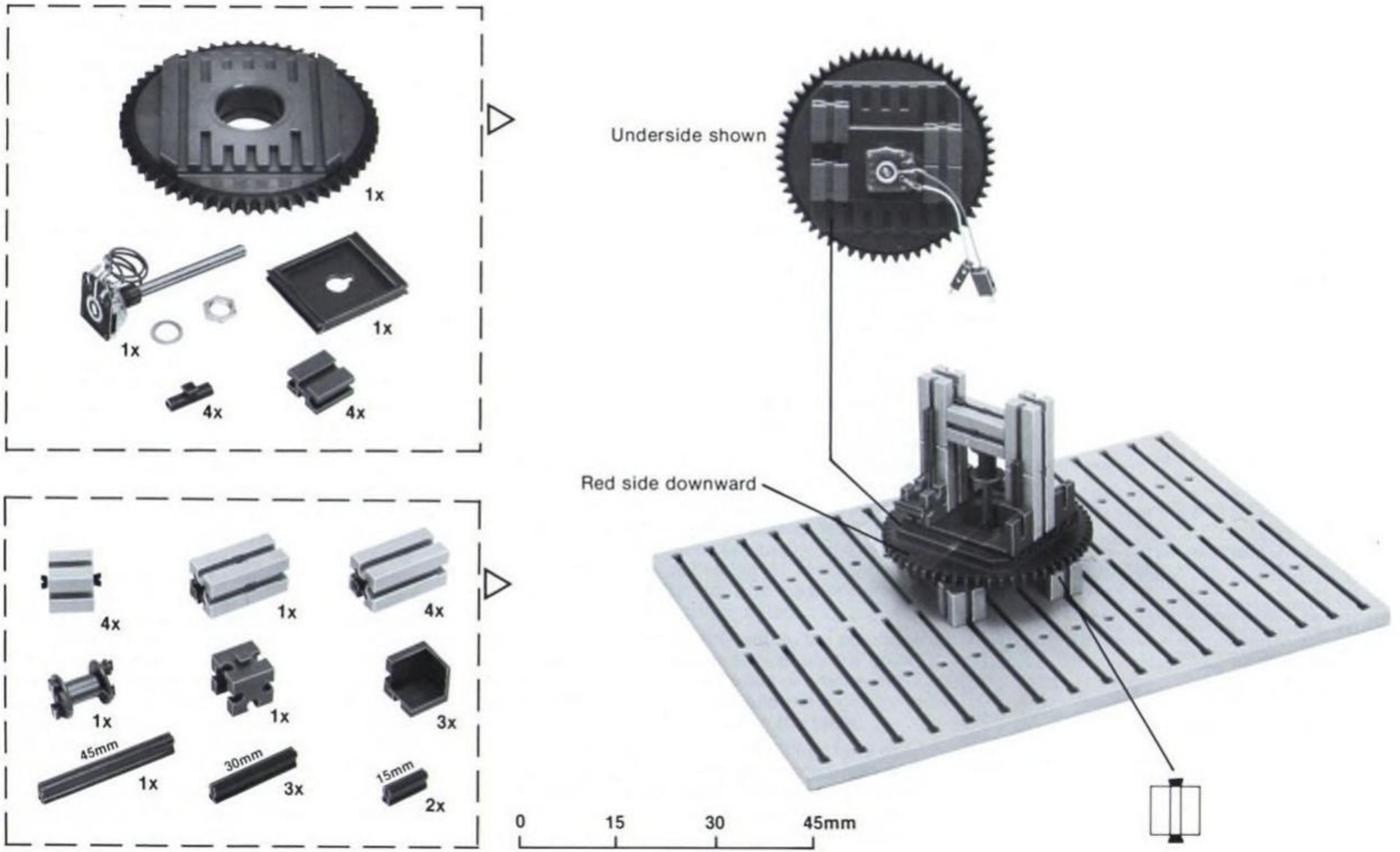
When you push the LEARN switch, the positions of the two potentiometers as well as the on/off state of the magnet are recorded in a table (1620-1660), printed on the screen, and the indicator is advanced to the next position in the table.

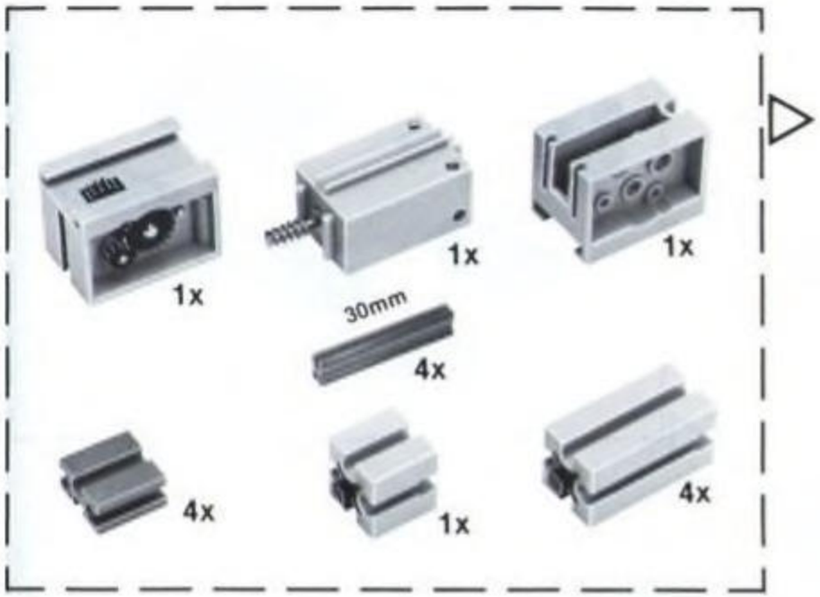
Note that the program checks the lines for a position when the LEARN switch is released, not when it is pushed. Why? Because if the program did not wait until the switch was released, it would check and record the same position over and over again, thus interfering with the smooth action of the model. You can create pauses, however, by pushing the LEARN switch several times in a row.

Pressing the START switch branches into the second part of the program, the PERFORMANCE module. The first thing the program does is check this switch again, since it also functions as the stop switch. Next, the table pointer is placed at the first item stored. These table values now serve as aim points for direction routines similar to the ones we explored in the two previous projects. One controls the up-and-down motion, the other the rotary motion. In the third part of the program module, the magnet is switched on and off. When the end of the table is reached, the pointer returns to the first item, and the cycle which our robot has learned is repeated.

```
•500 SYS INIT
 550 REM
 560 REM FISCHERTECHNIK COMPUTING
 570 REM
 580 REM TEACH IN ROBOT
 590 REM
 600 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 610 REM
 620 REM TEACH TABLE
 630 DIM HEIGHT(100),ANGLE(100),MAGNETS(100)
 640 REM CONSTANTS
 650 LET GX=30 :REM BRAKE PERIOD ARM
 660 LET GY=4 :REM BRAKE PERIOD ANGLE
 670 LET ID=-1 :REM POINTER TO TEACH TABLE
 680 LET MG$="OFF" :REM START WITH MAGNET OFF
 690 REM ASSIGNMENTS FOR THE INTERFACE
 700 REM
 710 REM INPUT
 720 REM E1=UP
 730 REM E2=DOWN
 740 REM E3=RIGHT
 750 REM E4=LEFT
 760 REM E5=MAGNET ON
 770 REM E6=MAGNET OFF
 780 REM E7=LEARN
 790 REM E8=START
 800 REM EX=POSITION OF ARM
 810 REM EY=POSITION OF ANGLE
 820 REM
 830 REM OUTPUT
 840 REM M1=TURN ROBOT
 850 REM M2=UP/DOWN
 860 REM M3=MAGNET
```

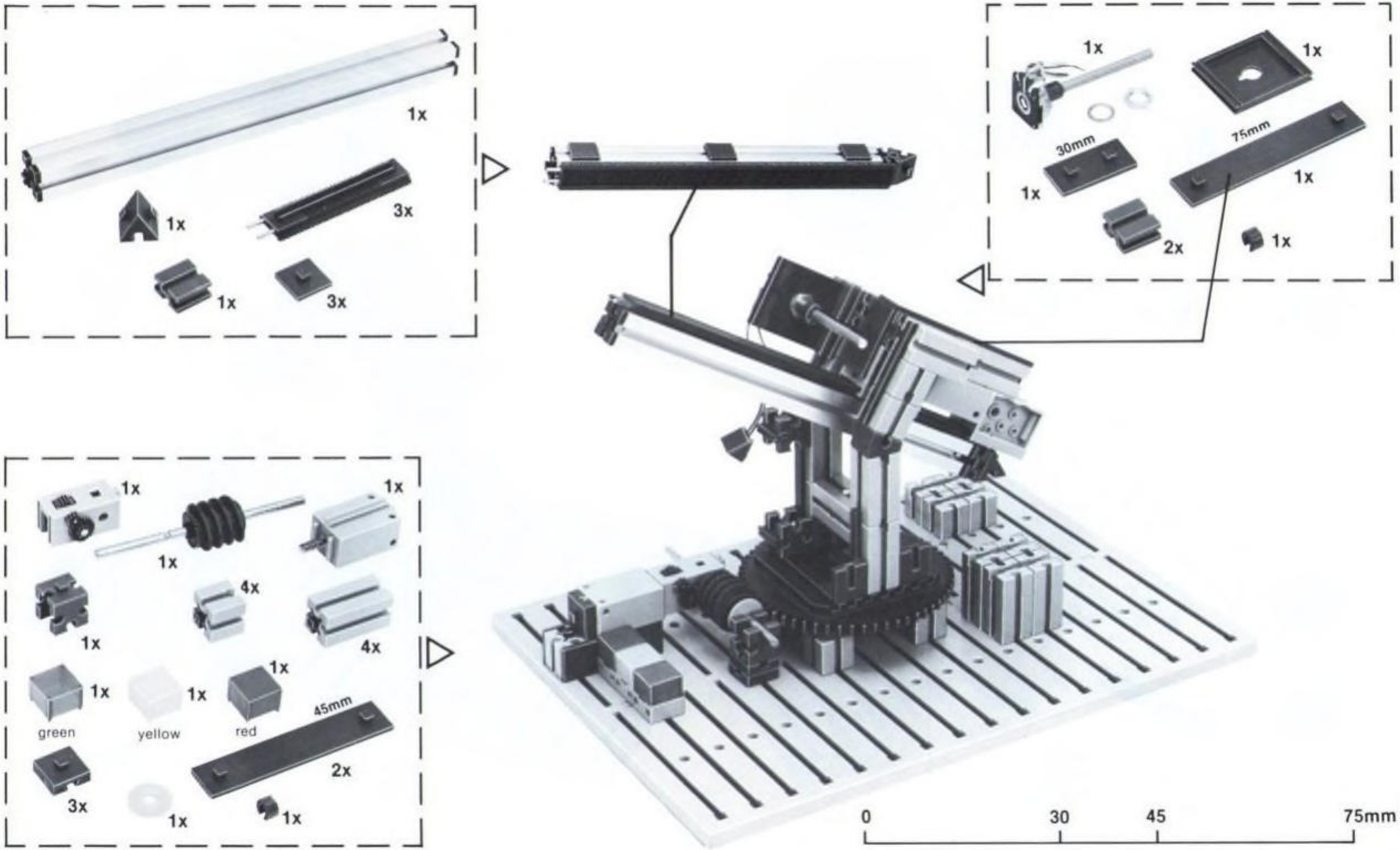
```
870 REM
880 REM FUNCTION DESCRIPTION :
890 REM WHEN YOU START THE PROGRAM IT IS IN LEARN MODE.
900 REM THE ROBOT IS CONTROLLED WITH THE KEYS E1-E6.
910 REM WITH E7 YOU CAN LATCH THE CURRENT POSITION OF
      THE ROBOT.
920 REM WITH E8 YOU ENTER THE REPEAT MODE. THE ROBOT DOES
      NOW THE MOVEMENT
930 REM YOU TAUGHT IT BEFORE.
940 REM THIS MOVEMENTS ARE REPEATED UNTIL YOU PRESS KEY
      E8 AGAIN.
950 REM
960 GOSUB2900
1000 REM SCANNING LOOP
1100 REM KEY 'UP'
•1110 IF USR(E1)=0 THEN GOTO 1140
•1120 SYS M2,CW
 1130 GOTO 1100
•1140 SYS M2,OFF
 1200 REM KEY 'DOWN'
•1210 IF USR(E2)=0 THEN GOTO 1240
•1220 SYS M2,CCW
 1230 GOTO 1200
•1240 SYS M2,OFF
 1300 REM KEY 'RIGHT'
•1310 IF USR(E3)=0 THEN GOTO 1340
•1320 SYS M1,CW
 1330 GOTO 1300
•1340 SYS M1,OFF
 1400 REM KEY 'LEFT'
•1410 IF USR(E4)=0 THEN GOTO 1440
•1420 SYS M1,CCW
 1430 GOTO 1400
•1440 SYS M1,OFF
 1500 REM KEY 'MAGNET ON'
•1510 IF USR(E5)=0 THEN GOTO 1540
•1520 SYS M3,CW
 1530 LET MG$="ON"
 1540 REM KEY 'MAGNET OFF'
•1550 IF USR(E6)=0 THEN GOTO 1580
•1560 SYS M3,OFF
 1570 LET MG$="OFF"
 1580 REM KEY 'LEARN'
•1590 IF USR(E7)=0 THEN GOTO 1670
 1600 REM WAIT UNTIL KEY IS NO LONGER DEPRESSED
•1610 IF USR(E7)=1 THEN GOTO 1610
 1620 LET ID = ID +1
 1630 LET HEIGHT(ID)=USR(EX)
 1640 LET ANGLE(ID)=USR(EY)
 1650 LET MAGNET$(ID)=MG$
 1660 PRINT ID;" ";HEIGHT(ID);" ";ANGLE(ID);" ";MAGNET$(ID)
 1670 REM KEY 'START REPEAT MODE'
•1680 IF USR(E8)=0 THEN GOTO 1100
 1690 GOSUB 2900
 1700 IF USR(E8)=1 THEN GOTO 1700
 1710 LET ID=ID+1
 1720 LET HEIGHT(ID)=-1
 2000 REM EXECUTION OF MOVEMENT
 2010 LET ID = -1
 2100 LET ID =ID + 1
 2110 PRINT ID ;" ";HEIGHT(ID);" ";ANGLE(ID);" ";MAGNET$(ID)
 2120 IF HEIGHT(ID)=-1 THEN GOTO 2000
 2140 REM ADJUST ARM
•2150 IF USR(E8)=1 THEN GOTO 4000
•2200 D=USR(EX)-HEIGHT(ID)
•2210 IF D>0 THEN SYS M2,CW
•2220 IF D<0 THEN SYS M2,CCW
 2230 IF D=0 THEN GOTO 2400
 2240 LET D=ABS(D)
 2260 IF D>GX THEN GOTO 2150
•2300 SYS M2,OFF
 2310 GOTO 2120
 2400 REM ADJUST ANGLE
•2410 IF USR(E8)=1 THEN GOTO 4000
•2420 D=USR(EY)-ANGLE(ID)
•2430 IF D>0 THEN SYS M1,CW
•2440 IF D<0 THEN SYS M1,CCW
 2450 IF D=0 THEN GOTO 2600
 2460 LET D=ABS(D)
 2470 IF D>GY THEN GOTO 2410
•2520 SYS M1,OFF
 2530 GOTO 2410
 2600 REM SWITCH MAGNET
•2610 IF MAGNET$(ID)="ON" THEN SYS M3,CW
 2620 FOR I=0TO 100
•2630 IF USR(E8)=1 THEN 4000
 2640 NEXT
 2650 IF MAGNET$(ID)="OFF"THEN GOSUB 2800
 2700 GOTO 2100
 2800 REM MAGNET-OFF ROUTINE
• 2810 SYS M3,CCW :REM REVERSE MAGNET CURRENT FOR
      DEMAGNETIZATION
•2820 SYS M3,OFF :REM SWITCH OFF MAGNET
 2830 RETURN
 2900 REM DISPLAY TITLE
•2910 PRINT CHR$(147)
 2920 PRINT"FISCHERTECHNIK"
 2930 PRINT"COMPUTING"
 2940 PRINT
 2950 PRINT"TEACH IN ROBOT"
 2960 PRINT
 2970 PRINT"TEACH IN TABLE"
 2980 PRINT
 2990 PRINT" #  HEIGHT  ANGLE  MAGNET"
 3000 RETURN
 4000 REM STOP ROBOT
•4010 SYS INIT
```

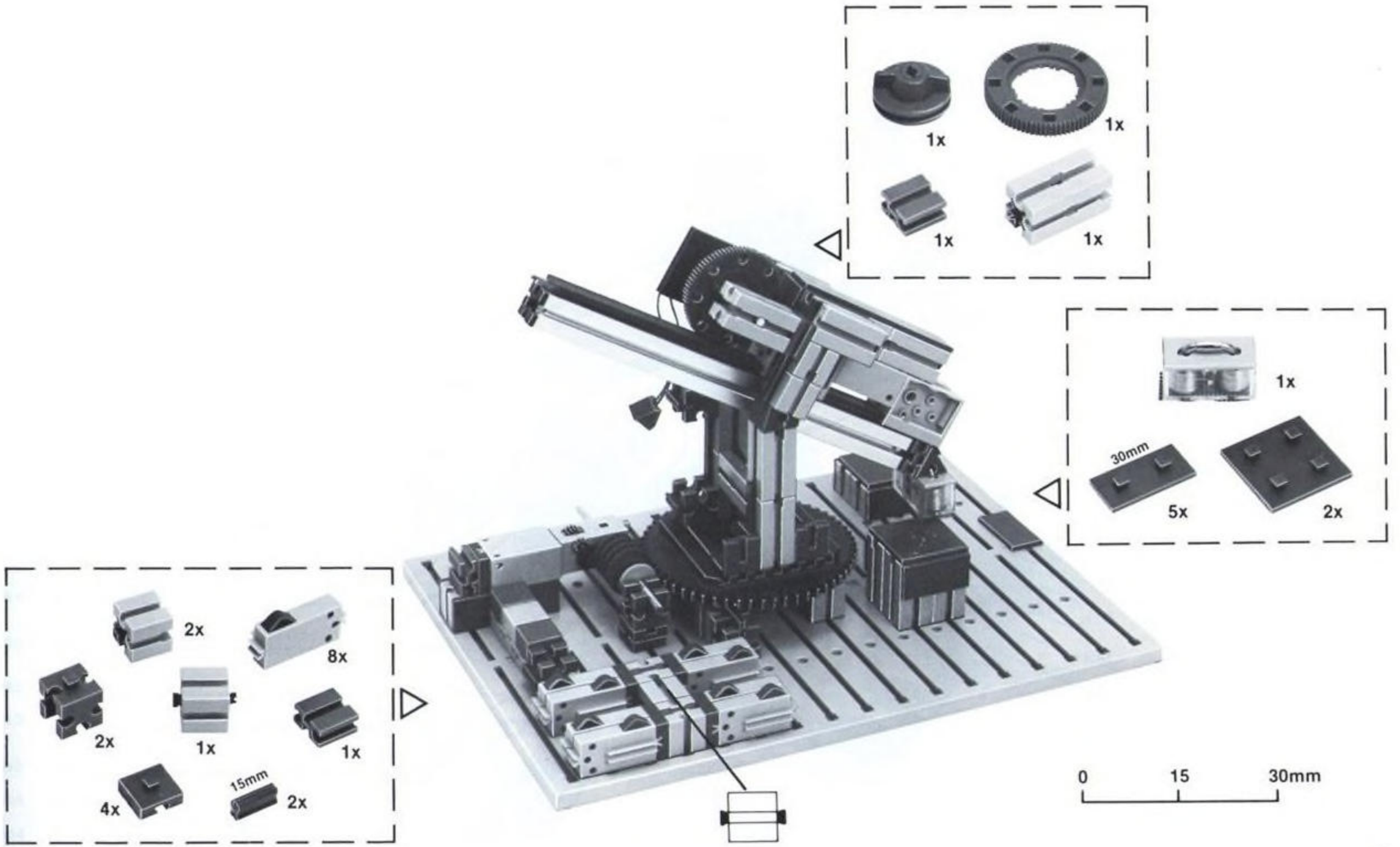

1x
1x
1x
4x
4x
4x
1x
4x
1x
1x
3x
45mm
1x
30mm
3x
15mm
2x
Underside shown
Red side downward
0
15
30
45mm

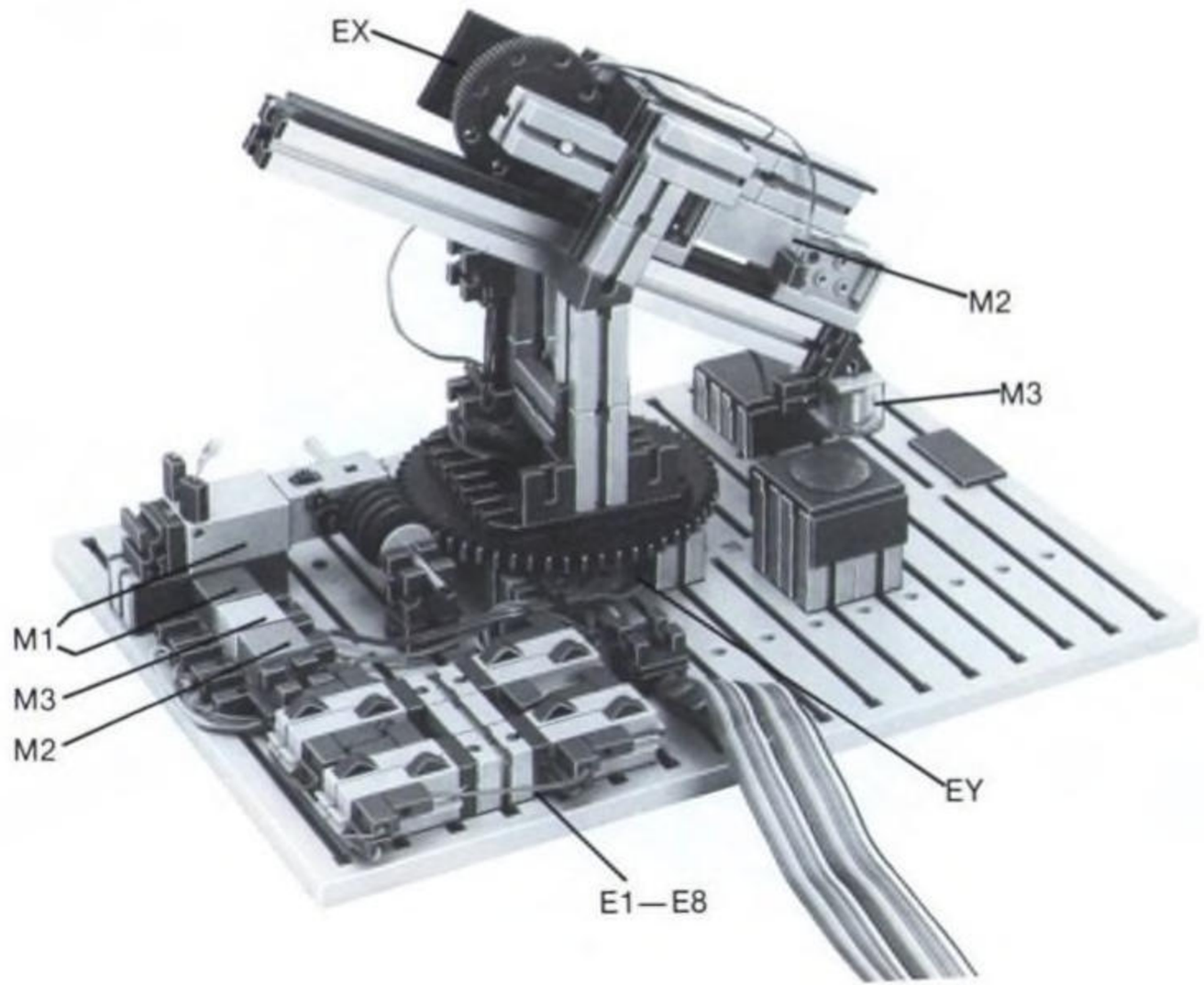

1x
1x
1x
30mm
4x
4x
1x
4x


0
15
30
45mm

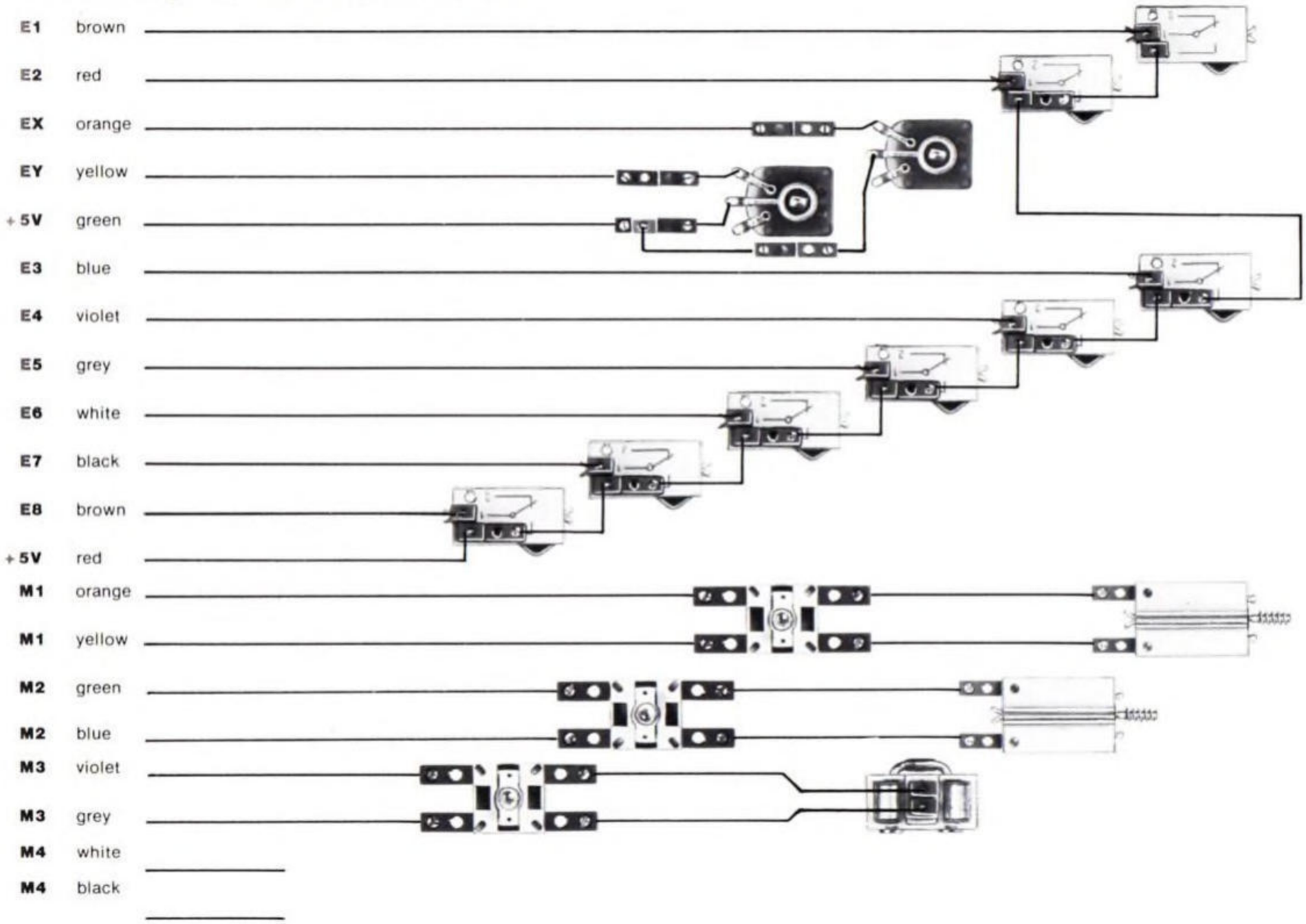

1x
3x
4x
45mm
2x
2x
15mm
1x
3x
3x


1x
1x
3x
1x
1x
3x
1x
1x
30mm
1x
75mm
1x
2x
1x
1x
1x
1x
4x
4x
1x
1x
1x
green
1x
yellow
1x
red
45mm
2x
3x
1x
1x
0
30
45
75mm


1x
1x
1x
1x
1x
30mm
5x
2x
2x
8x
2x
1x
1x
4x
15mm
2x
0
15
30mm


EX
M2
M3
M1
M3
M2
EY
E1—E8

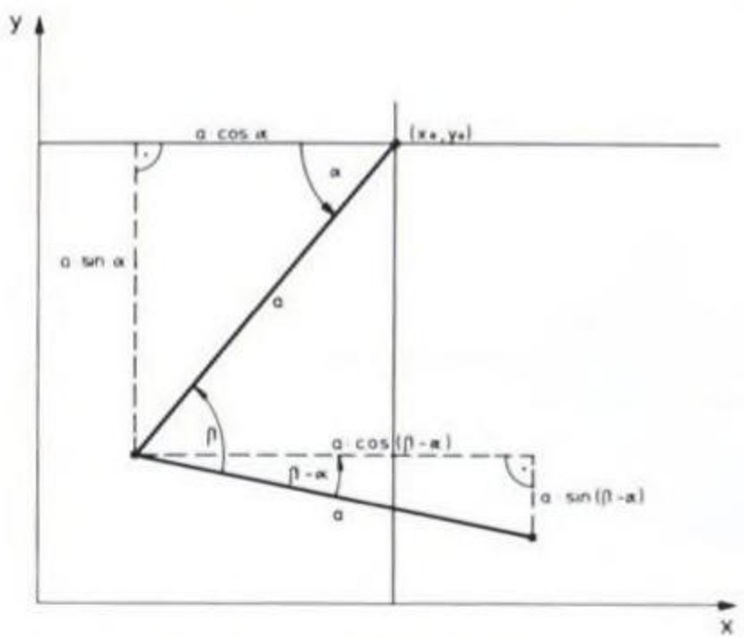
Circuit Layout—Teachable Robot



Graphic Panel

CAD—Computer Aided Design—is one of the newest and fastest growing areas in computer technology. It exchanges the drawing board for the computer screen, and the creation of a design can be accomplished by the movements of a pencil or pointer on the graphic panel. Although these systems can perform the most complicated tasks, for our purposes we will build and study a simplified design. It consists of a flexible arm which can be guided over the drawing surface, with potentiometers at the pivot points.

The geometric concepts behind the project are shown in illustration 9. In order to store the information in the computer and to make the drawings appear on the screen, we need the coordinates of the point we wish to create on the horizontal and vertical directions, X and Y. The mathematical theory of the Cartesian coordinate system (named after the French mathematician Descartes), which forms the basis of the tracking system we will use, is shown below the diagram.



$$x = x_0 - a \cdot \cos \alpha + a \cdot \cos(\beta - \alpha)$$

$$y = y_0 - a \cdot \sin \alpha - a \cdot \sin(\beta - \alpha)$$

(Illustration 9)

At the point (x_0, y_0), we define the center of the radius of our first angle of rotation. At the end of the first arm, we have the second rotation point; potentiometers are placed at both of these points to provide feedback regarding the angle of the arm. At the end of the second arm section is our pen or pencil. Using the formulas shown on the diagram, we can obtain the position of the pencil on the graphic panel. Luckily, the program does the calculations for us. Next, the position on the panel is translated into a position on the screen.

First of all, your computer must have a graphic system. Most of the machines for which interfaces are available have such a system built in (Apple II), or one may be added. This is the case with Simon's BASIC for the Commodore; our examples and printed programs are based upon this addition to the computer. If you are using another computer with your fischertechnik computing kit, the programs on the disk supplied will have already been corrected for your computer model. Please note that Simon's BASIC is not included on the disk we supply for Commodore; you must obtain it and load it into your computer before loading our programs.

Next, you have to determine the source or center point of the graphic screen in your system. Some of the systems may place this point at the upper left corner of the screen rather than in the center. You must also determine the graphic resolution available. To simplify matters, you should be willing to accept a slight distortion of your drawn figures on the screen. Use this formula to determine screen adjustment factors:

f = (pixel resolution in x direction)/260mm
g = (pixel resolution in y direction)/187mm

This leads to the following screen coordinates:

X (screen) = f * x (tablet)
Y (screen) = g * y (tablet)
Y (screen) = (pixel resolution y) − g * y

This last formula concerns the conversion of the function values obtained from USR (EX) and USR (EY)—the potentiometers—to angular degrees.

We calibrate the graphic panel at the beginning of the program. To do this, we move the pencil to the upper left corner of the drawing surface and record the numerical values A1 and B1 of the two potentiometers. These values, and those taken from the table of values shown just before the program listing, can be entered in the following formulas if you wish to do your own programming and conversion of coordinates.

$$\alpha_n = \alpha_1 + \left(\frac{\alpha_1 - \alpha_2}{A1 - A2} \right) * (AN - A1)$$

$$\beta_n = \beta_1 + \left(\frac{\beta_1 - \beta_2}{B1 - B2} \right) * (BN - B1)$$

Our program uses the switch at the right border of the drawing surface (E1) to inform the computer when the pencil is positioned. The other switches control the graphic functions produced as the pencil is moved. To expand the range of controls, switch E2 is used as a shift key, allowing double the number of functions. These are:

E3 – draw a dot
E4 – draw a line
E5 – draw a triangle
E6 – draw a rectangle
E7 – draw a circle
E8 – erase the screen
E3 + E2 – transfer letter from keyboard

E4 + E2 – draw any line
E5 + E2 – draw full triangle
E6 + E2 – draw full rectangle
E7 + E2 – draw full circle
E8 + E2 – fill the field

To explain a bit further, to draw a triangle on the screen, you would first press the switch E5 to tell the program you wish to create a triangle. Next, you would move the arm and pointer to where you wanted one corner of the triangle, and you would then press the switch at the upper right corner of the panel (E1) to mark the spot. In the same way, you would mark the desired locations of the other two corners of the triangle. When you've marked the third point, the computer program will draw a triangle between the three points you've marked. For a circle, you mark the center point and then the outside rim (radius); for a rectangle, only the two opposite corners need be marked. The combination E3 + E2 will let you add one letter or number from the keyboard to your drawing at the point marked by the pointer. This can be repeated to add full words if you wish.

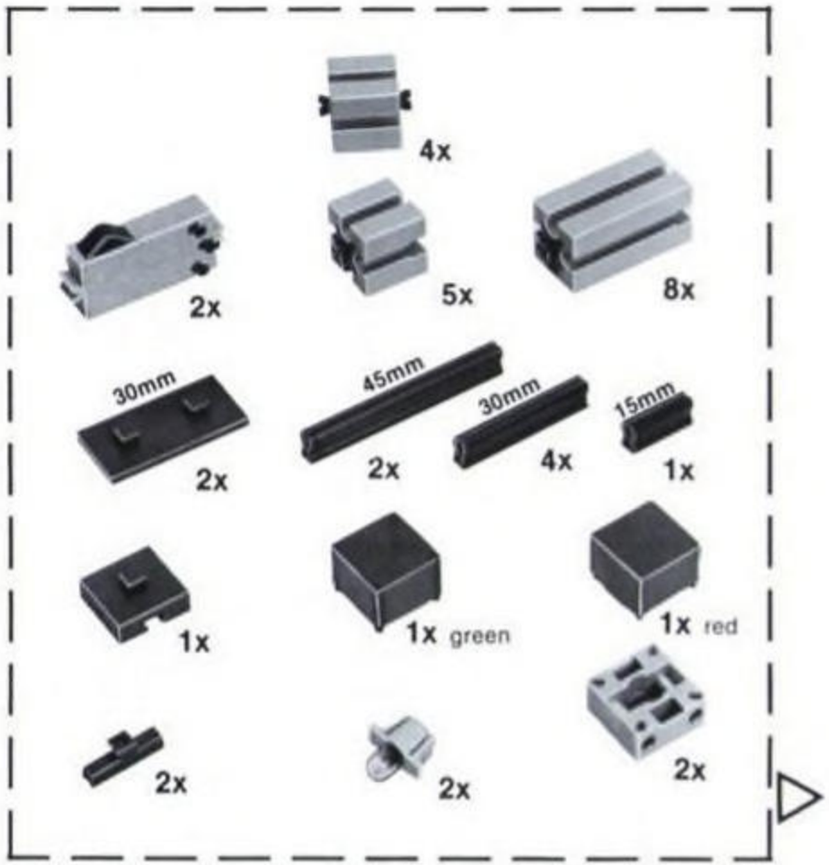
Finally, here's a suggestion for a program you might want to try yourself. Draw a keyboard of a piano on the tablet, and then use the x and y values to turn on the tone generator of your computer. If you program frequencies on the horizontal axis, and duration on the vertical, you should be able to play music or write your own compositions. We know you'll have fun drawing or composing with your panel, and that you will find lots of new and interesting projects as you use it.

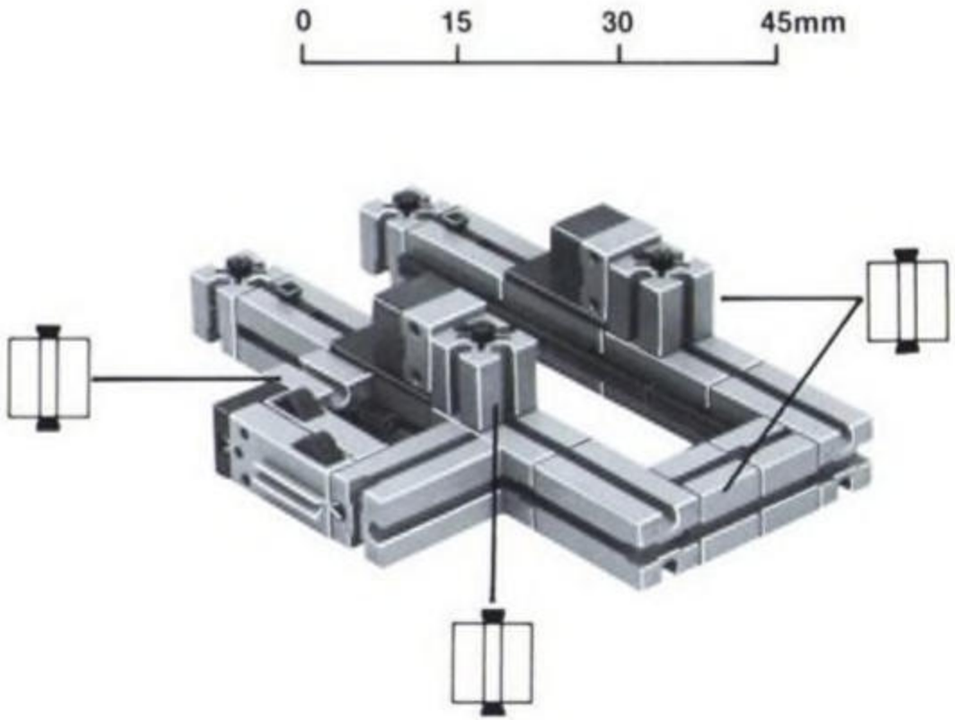
Table of geometric data for graphic panel

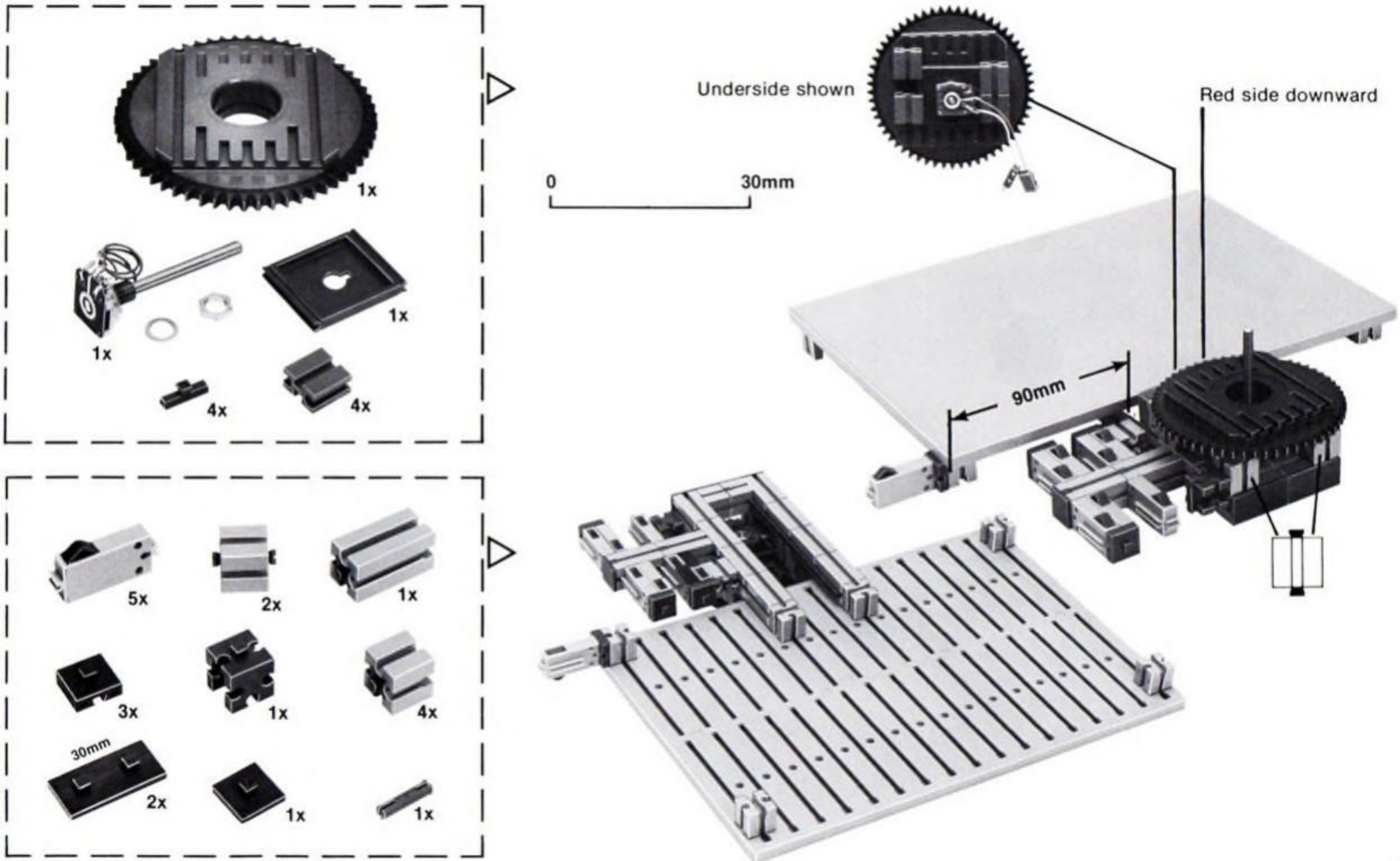
$\alpha_1 = -39°$	$\beta_1 = 53.7°$
$\alpha_2 = 82.5°$	$\beta_2 = 112.2°$
$x_o = 137$ mm	$a = 166$ mm
$y_o = 247$ mm	

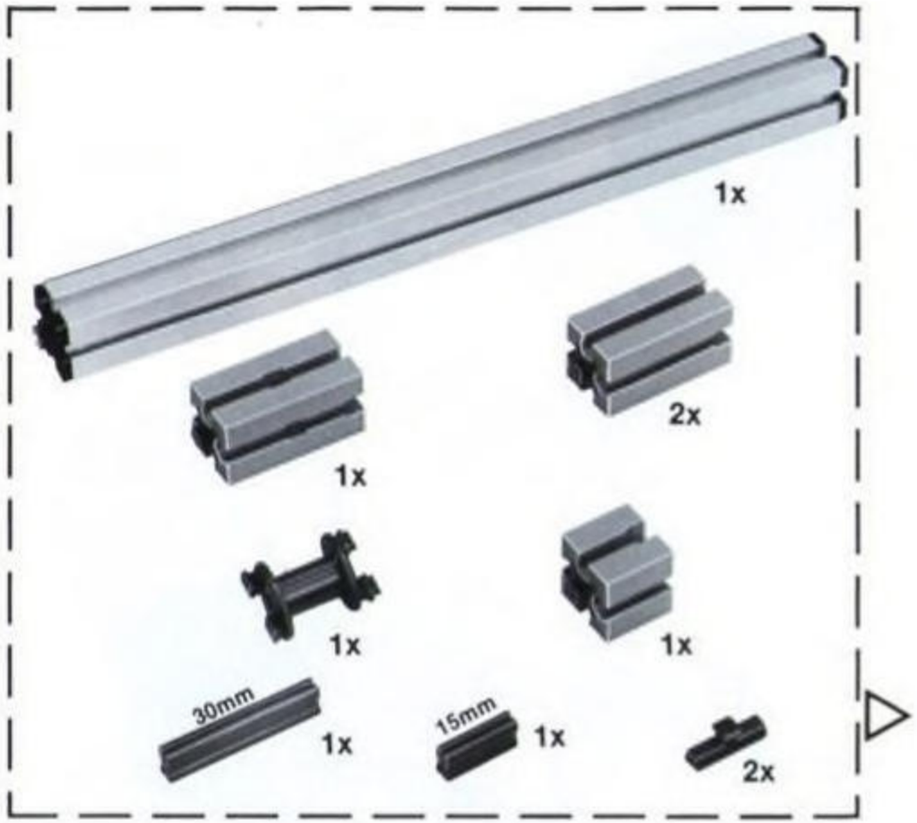
```
• 500 SYS INIT
  510 REM
  520 REM FISCHERTECHNIK COMPUTING
  530 REM
  540 REM GRAPHIC PANEL
  550 REM
  560 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
  570 REM
  580 REM ASSIGNMENTS FOR THE INTERFACE
  590 REM
  600 REM INPUT
  610 REM E1=READ IN ANGULAR VALUE
  620 REM E2=SHIFT KEY
  630 REM E3=PLOT A PIXEL / PRINT CHARACTER
  640 REM E4=DRAW A LINE (FIXED COORDINATES)
  645 REM     / DRAW OFF HAND LINE - END-POINTS ARE FIXED BY
      PRESSING E1
  650 REM E5=TRIANGLE / FILLED TRIANGLE
  660 REM E6=RECTANGLE / FILLED RECTANGLE
  670 REM E7=CIRCLE / FILLED CIRCLE
  680 REM E8=CLEAR SCREEN / FILL ANY GIVEN AREA
  690 REM EX=BETA
  700 REM EY=ALPHA
  710 REM
  720 REM OUTPUT
  730 REM M1=GRAPHIC ACTIVE
  740 REM M2=READY FOR INPUT
  750 REM
  760 REM FUNCTION DESCRIPTION :
  770 REM TRANSFER A PICTURE ONTO THE
  780 REM GRAPHIC SCREEN OF THE
  790 REM COMPUTER.
  800 REM
  810 REM GEOMETRICAL DATA
  820 LET WA=-0.68068 :REM ANGLE ALPHA TOP LEFT
  830 LET WB=0.93724 :REM ANGLE BETA TOP LEFT
  840 LET A=166 :REM LENGTH OF ARM
  850 LET X0=137 :REM ORIGIN OF CO-ORDINATE SYSTEM
  860 LET Y0=61
  870 LET SX=1.2364 :REM SCALE FACTOR X-AXIS (PIXELS/MM)
  880 LET SY=1.0699 :REM SCALE FACTOR Y-AXIS (PIXELS/MM)
  890 REM
• 900 PRINT CHR$(147)
  910 PRINT"FISCHERTECHNIK"
  920 PRINT"COMPUTING"
  930 PRINT
  940 PRINT"GRAPHIC PANEL"
  950 PRINT
  960 PRINT"THIS PROGRAM REQUIRES SIMONS BASIC PROVIDING GRAPHIC
      COMMANDS."
  970 PRINT
 1000 PRINT"PLEASE CALIBRATE"
 1010 PRINT
 1020 PRINT"POSITION PEN"
 1030 PRINT"IN THE LEFT UPPER CORNER"
 1040 PRINT"DEPRESS READ-IN KEY !"
• 1050 IF USR(E1)=0 THEN GOTO 1050:REM WAIT FOR ACCEPT KEY
• 1060 LET A1=USR(EY)
• 1070 LET B1=USR(EX) :REM PEN POSITION
• 1080 IF USR(E1)=1 THEN GOTO 1080:REM WAIT UNTIL KEY IS NO LONGER
       DEPRESSED
 1090 PRINT
 1100 PRINT"POSITION PEN"
 1110 PRINT"IN RIGHT LOWER CORNER"
 1120 PRINT"DEPRESS READ-IN KEY !"
• 1130 IF USR(E1)=0 THEN GOTO 1130:REM WAIT FOR READ-IN KEY
• 1140 LET A2=USR(EY)
• 1150 LET B2=USR(EX) :REM PEN POSITION
• 1160 IF USR(E1)=1 THEN GOTO 1160:REM WAIT UNTIL KEY IS NO LONGER
       DEPRESSED
 1170 LET AF=2.12058/(A2-A1)
 1180 LET BF=1.02102/(B2-B1)
• 1190 HIRES7,6 :REM HI-RES GRAPHICS (YELLOW/BLUE)
• 1200 SYS INIT :REM ACTIVE INDICATOR OFF
• 1210 SYS M2,CW :REM READY INDICATOR ON
 1220 FOR I=2 TO 7 :REM SCANNING LOOP
• 1230 S=USR(E2)*7 :REM SHIFT-KEY
 1240 LET EJ=2^I
• 1250 IF USR(EJ)=0 THEN GOTO 1200
 1260 LET S=S+I-1
 1270 LET I=7 :REM EXIT LOOP
 1280 NEXT I
 1290 IF(S=0)OR(S=7) THEN GOTO 1220
• 1300 SYS INIT :REM READY INDICATOR OFF
• 1310 SYS M1,CW :REM ACTIV INDICATOR ON
 1320 ON S GOTO 1330,1360,1420,1440,1520,1190,1200,1540,1600,1670,
      1720,1780,1810
 1330 GOSUB 3000 :REM CO-ORDINATES
• 1340 PLOT X,Y,1 :REM PLOT PIXEL
 1350 GOTO 1200
 1360 GOSUB 3000 :REM CO-ORDINATES
 1370 LET X1=X
 1380 LET Y1=Y :REM SAVE CO-ORDINATES
 1390 GOSUB 3000
• 1400 LINE X1,Y1,X,Y,1 :REM DRAW LINE
 1410 GOTO 1200
 1420 GOSUB 5010:REM DRAW TRIANGLE
 1430 GOTO 1200
 1440 GOSUB 3000 :REM CO-ORDINATES
 1450 LET X1=X
 1460 LET Y1=Y :REM SAVE CO-ORDINATES
 1470 GOSUB 3000 :REM CO-ORDINATES
 1480 LET H=Y-Y1 :REM HEIGHT
 1490 LET B=X-X1 :REM WIDTH
• 1500 REC X1,Y1,B,H,1 :REM DRAW RECTANGLE
 1510 GOTO 1200
 1520 GOSUB 6010:REM DRAW CIRCLE
 1530 GOTO 1200
 1540 GOSUB 3000 :REM CO-ORDINATES
 1550 GET C$ :REM READ CHARACTER
 1560 IF C$="" THEN GOTO 1550
• 1570 C=ASC(C$)-64 :REM SCREEN CODE
• 1580 CHAR X,Y,C,1,1 :REM DISPLAY CHARACTER
 1590 GOTO 1200
 1600 GOSUB 3000 :REM START SIGNAL
• 1610 IF USR(E1)=1 THEN GOTO 1200
 1620 GOSUB 4000:REM CO-ORDINATES
• 1630 LINE X1,Y1,X,Y,1
 1640 LET X=X1 :REM SAVE CO-ORDINATES
 1650 LET Y=Y1
 1660 GOTO 1610
 1670 GOSUB 5010
 1680 LET XM=(X1+X2+X)/3
 1690 LET YM=(Y1+Y2+Y)/3
• 1700 PAINT XM,YM,1 :REM FILL ANY GIVEN AREA
 1710 GOTO 1200
 1720 GOSUB 3000 :REM CO-ORDINATES
 1730 LET X1=X
 1740 LET Y1=Y :REM SAVE CO-ORDINATES
 1750 GOSUB 3000 :REM CO-ORDINATES
• 1760 BLOCK X1,Y1,X,Y,1 :REM DRAW BLOCK
 1770 GOTO 1200
 1780 GOSUB 6010:REM DRAW CIRCLE
• 1790 PAINT X1,Y1,1 :FILL CIRCLE
```

```
 1000 GOTO 1200
 1010 GOSUB 3000 :REM CO-ORDINATES
•1020 PAINT X,Y,1 :REM FILL ANY GIVEN AREA
 1030 GOTO 1200
 3000 REM READ CO-ORDINATES
•3010 IF USR(E1)=0 THEN GOTO 3010
•3020 LET ALPHA=(USR(EY)-A1)*AF+WA
•3030 LET BETA=(USR(EX)-B1)*BF+WB
 3040 REM READ ANGLE
 3050 LET X=INT(SX*(X0-A*COS(ALPHA)+A*COS(BETA-ALPHA)))
 3060 LET Y=INT(SY*(A*SIN(ALPHA)+A*SIN(BETA-ALPHA)-Y0))
 3070 IF X<0 THEN LET X=0
 3080 IF X>319 THEN LET X=319
 3090 IF Y<0 THEN LET Y=0
 3100 IF Y>199 THEN LET Y=199
•3110 IF USR(E1)=1 THEN GOTO 3110
 3120 RETURN
 4000 REM READ CO-ORDINATES WITHOUT WAITING FOR READ-IN KEY
•4010 LET ALPHA=(USR(EY)-A1)*AF+WA
•4020 LET BETA=(USR(EX)-B1)*BF+WB
 4030 REM READ ANGLE
 4040 LET X1=INT(SX*(X0-A*COS(ALPHA)+A*COS(BETA-ALPHA)))
 4050 LET Y1=INT(SY*(A*SIN(ALPHA)+A*SIN(BETA-ALPHA)-Y0))
 4060 IF X1<0 THEN LET X1=0
 4070 IF X1>319 THEN LET X1=319
 4080 IF Y1<0 THEN LET Y1=0
 4090 IF Y1>199 THEN LET Y1=199
 4100 RETURN
 5000 REM SUBROUTINE TRIANGLE
 5010 GOSUB 3000 :REM CO-ORDINATES
 5020 LET X1=X
 5030 LET Y1=Y :REM SAVE CO-ORDINATES
 5040 GOSUB 3000 :REM CO-ORDINATES
 5050 LET X2=X
 5060 LET Y2=Y :REM SAVE CO-ORDINATES
 5070 GOSUB 3000 :REM CO-ORDINATES
•5080 LINE X1,Y1,X2,Y2,1 :REM DRAW TRIANGLE
•5090 LINE X2,Y2,X,Y,1
•5100 LINE X,Y,X1,Y1,1
 5110 RETURN
 6000 REM SUBROUTINE CIRCLE
 6010 GOSUB 3000 :REM CO-ORDINATES
 6020 LET X1=X
 6030 LET Y1=Y :REM SAVE CO-ORDINATES
 6040 GOSUB 3000 :REM CO-ORDINATES
 6050 LET R=SQR((X1-X)↑2+(Y1-Y)↑2)
•6060 CIRCLE X1,Y1,R,A,1 :REM DRAW CIRCLE
 6070 RETURN
```

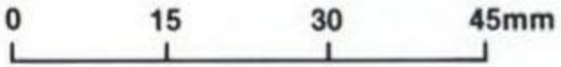

4x
2x
5x
8x
30mm
2x
45mm
2x
30mm
4x
15mm
1x
1x
1x green
1x red
2x
2x
2x

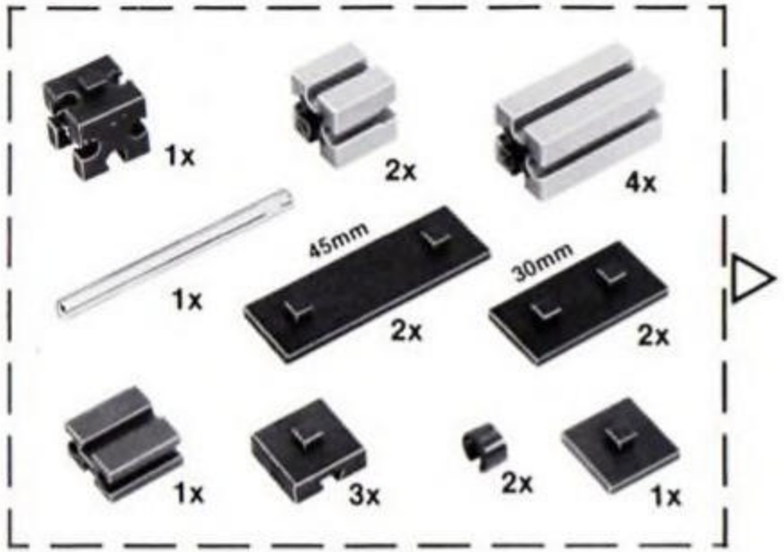

0
15
30
45mm


1x
1x
1x
4x
4x
5x
2x
1x
3x
1x
4x
30mm
2x
1x
1x
Underside shown
Red side downward
0
30mm
90mm



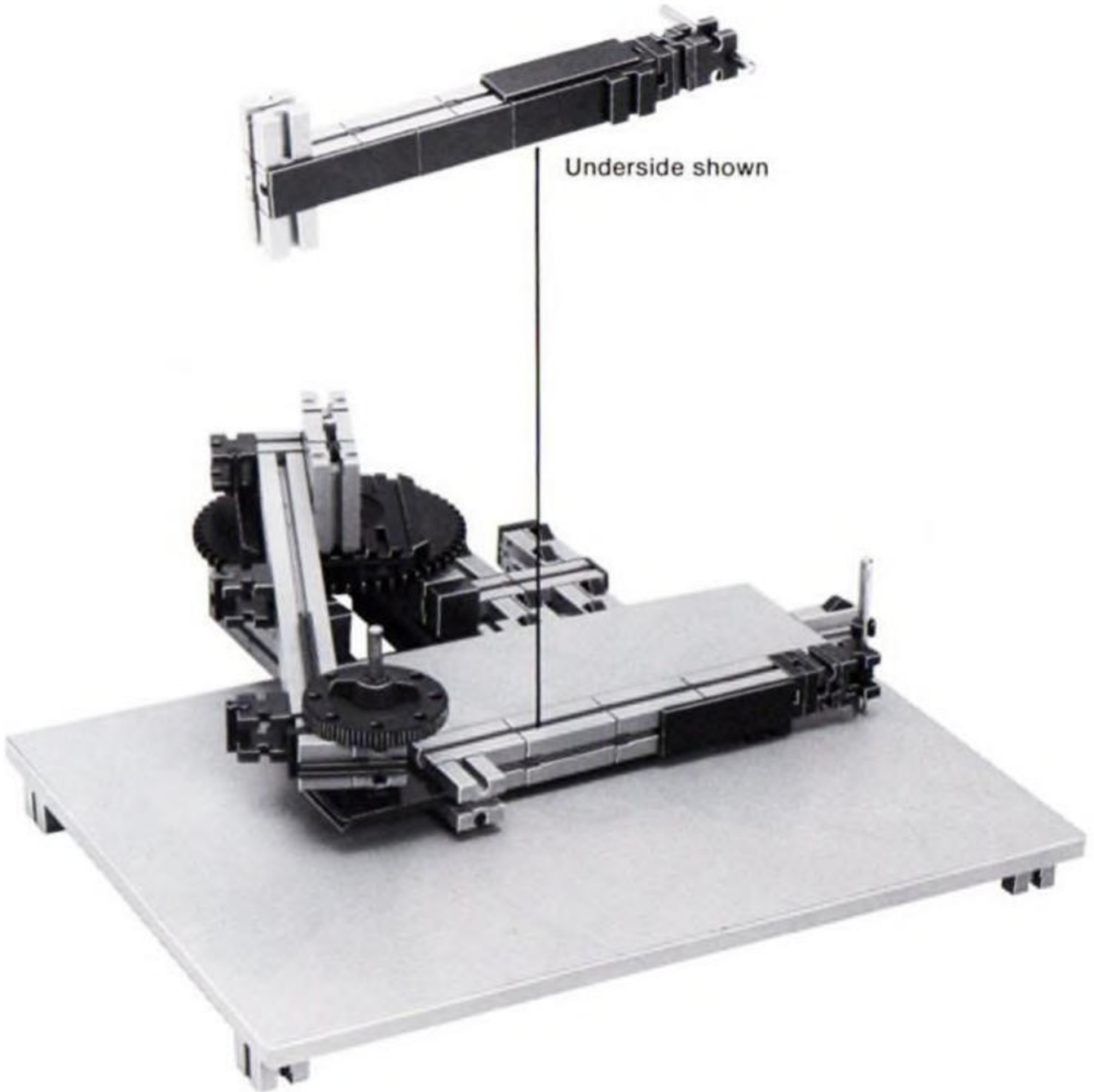
Underside shown

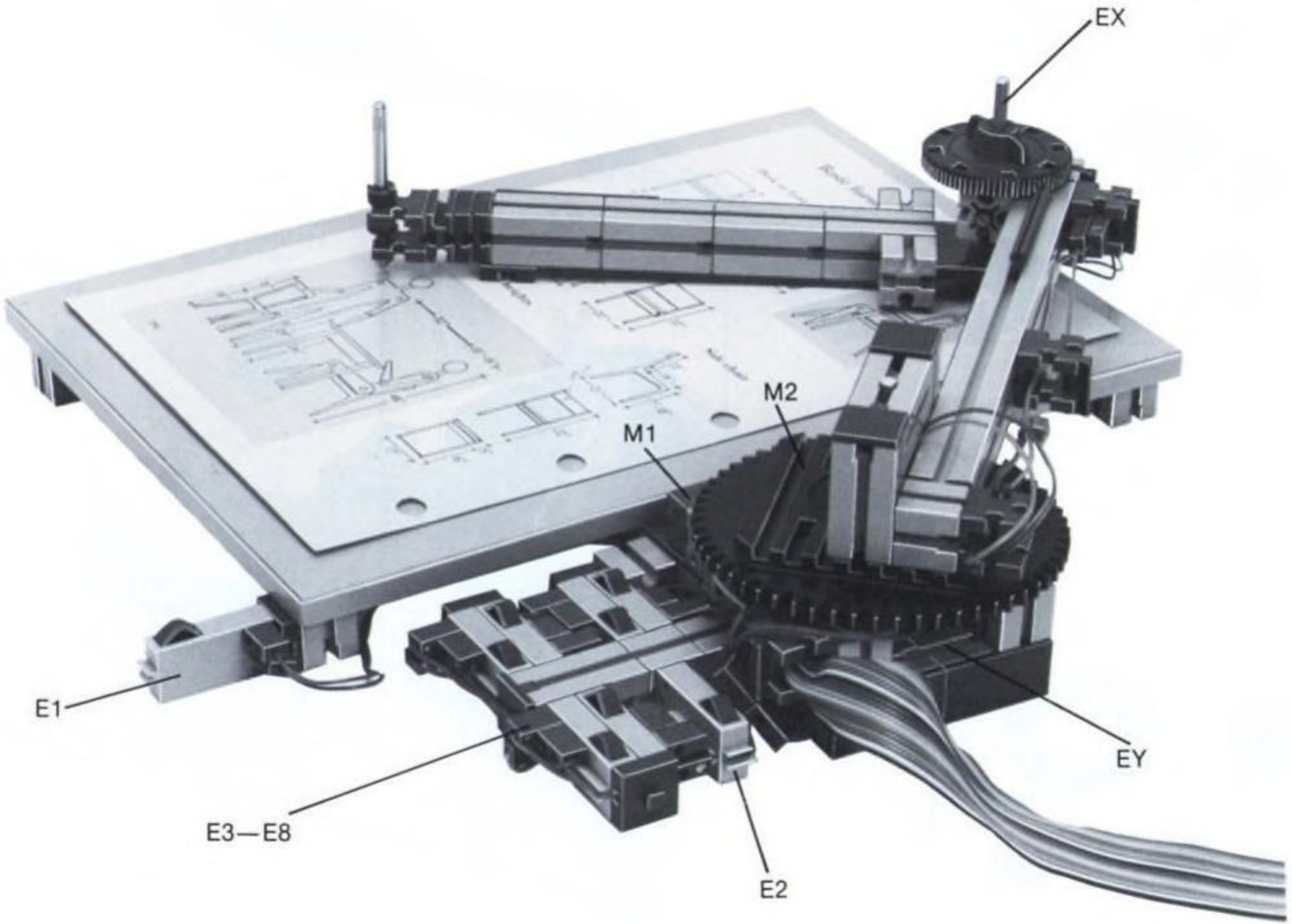



1x
2x
4x
1x
45mm
2x
30mm
2x
1x
3x
2x
1x

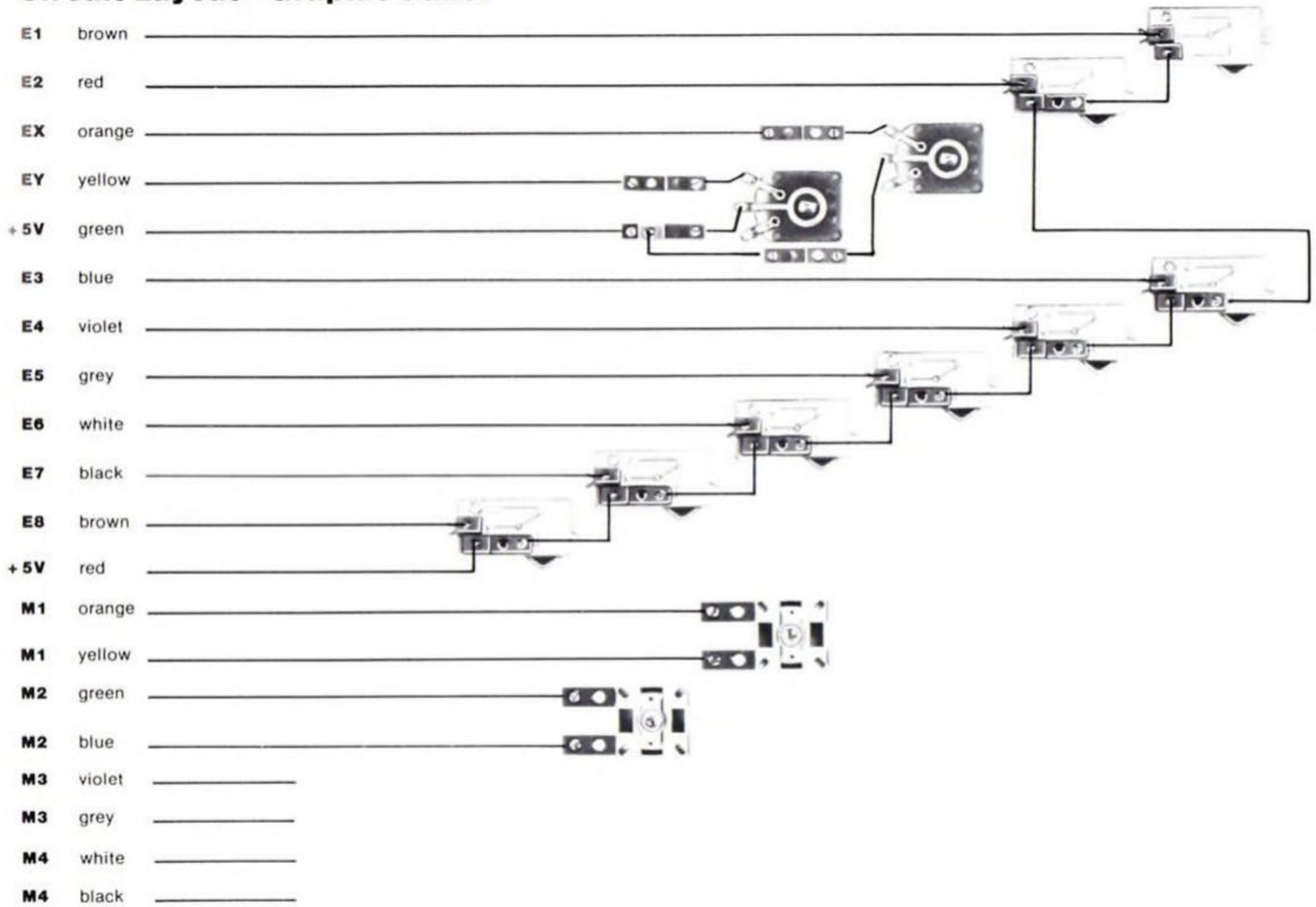

0
15
30
45mm


1x
1x
3x
2x
15mm
2x


Underside shown


EX
M2
M1
E1
E3—E8
E2
EY

Circuit Layout—Graphic Panel



Plotter

Here's another project that uses some of the things we have already learned to expand the uses of our computer and to explore new directions in computing and model building.

A plotter is very often found in use with a computer. It provides a means of displaying in graphic form the results of computer calculations. While the plotter you can build with the computing kit is not of the highest precision (a separate kit is available to build a high-precision plotter), the model shown here will be of sufficient accuracy to demonstrate some principles you may wish to explore further.

Using the components in the computing kit, we can build a polar coordinate plotter. With this type of device, the positioning of the pen above the paper is achieved by the combined movement of the pen head across the paper on the stationary arm and by rotation of the head. Writing is controlled by a magnet; when it is activated, the pen is pulled down against the paper.

When programming the plotter, we have to consider the conversion of the coordinates. Usually, as with the graphic panel, we think in terms of rectangular coordinates according to the Cartesian system. Our plotter, however, operates in a polar coordinate system using the factors of rotation and radius to define its points. These are normally indicated by the notation ϕ or r. Conversions between both coordinate systems are possible:

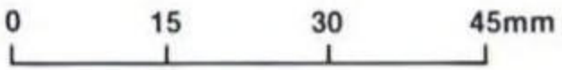
$$x = r * \cos(\phi)$$
$$y = r * \sin(\phi)$$

For a sample program, we will prepare a pie chart (which will not require the conversion). You are familiar with this type of graph, in which a circle is divided into unequal sections, or slices, to represent the various shares of the data being graphed. After setting up the operating range of the plotter, the program requests the number of segments and their values. The computer will take care of calculating the percentage share each item represents of the total. When input is complete, the plotter will draw the frame and then the borders of the segments.

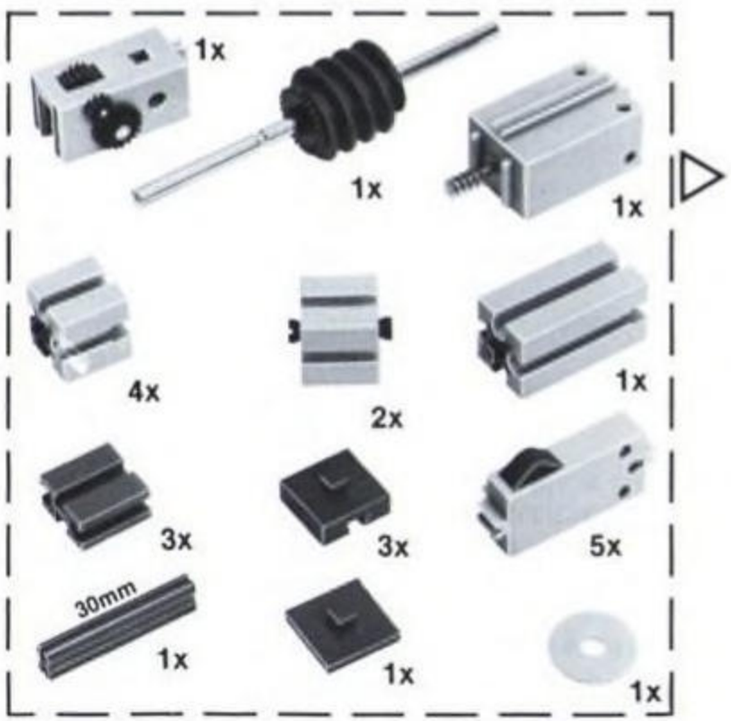
Like all fischertechnik projects, the basic structure of the plotter can be adapted for other purposes. Using just the magnet, without the pen unit, the plotter can be used as a limited area transporter to move any small magnetic objects from one side to another, as an overhead crane might in a factory. Perhaps you can use this to program a game robot for yourself.

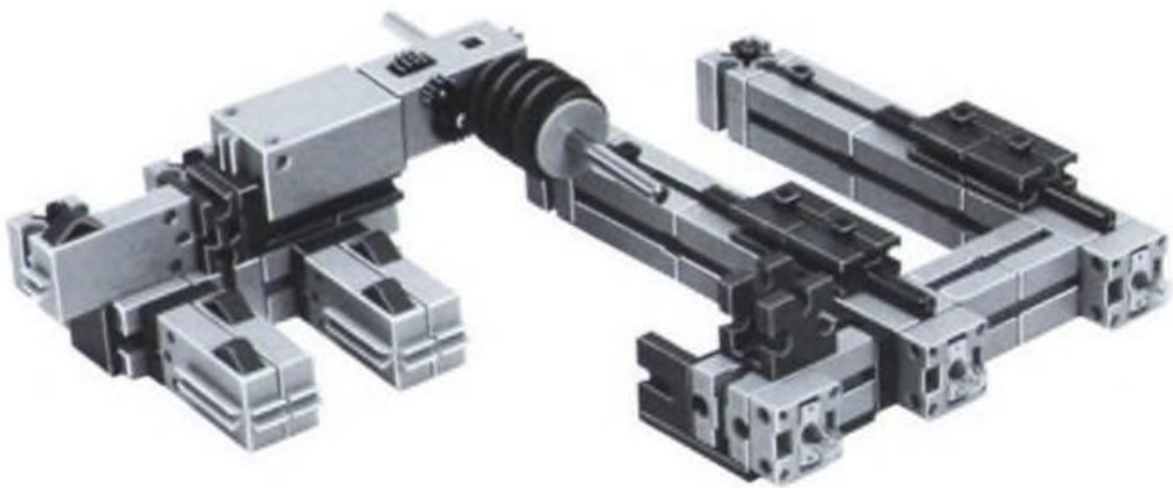
```
•500 SYS INIT
 510 REM
 520 REM FISCHERTECHNIK COMPUTING
 530 REM
 540 REM PLOTTER
 550 REM
 560 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 570 REM
 580 REM SEGMENT TABLE
 590 DIM SEG(9)
 600 REM CONSTANTS
 610 LET GR=30 :REM BRAKE PERIOD R
 620 LET GA=5 :REM BRAKE PERIOD A
 630 REM ASSIGNMENTS FOR THE INTERFACE
 640 REM
 650 REM INPUT
 660 REM E3=TAKE VALUES GIVEN BY POTENTIOMETER
 670 REM E4=RIGHT
 680 REM E5=LEFT
 690 REM E6=FORWARD
 700 REM E7=BACKWARD
 710 REM EX=RADIAL POSITION
 720 REM EY=ANGLE POSITION
 730 REM
 740 REM OUTPUT
 750 REM M1=AZIMUTHAL MOVEMENT
 760 REM M2=RADIAL MOVEMENT
 770 REM M3=MAGNET
 780 REM
 790 REM FUNCTION DESCRIPTION :
 800 REM THE PROGRAM CONTROLS THE PLOTTER WHICH DRAWS A
       GRAPHIC CONSISTING
 810 REM OF SEGMENTS OF A CIRCLE. AFTER THE START OF THE
       PROGRAM THE PLOTTER
 820 REM IS CALIBRATED. THEREAFTER YOU INPUT THE NUMBERS AND
 830 REM THE GRAPHIC IS PLOTTED.
 840 REM
•850 PRINT CHR$(147)
 860 PRINT"FISCHERTECHNIK"
 870 PRINT"COMPUTING"
 880 PRINT
 890 PRINT"PLOTTER"
1000 REM
1010 LET A$="PLEASE POSITION PLOTTER VIA THE KEYS TO : "
1020 PRINT
1030 PRINT A$
1040 PRINT"RIGHT - OUTER POSITION"
1050 GOSUB 3000
1060 LET RM=R
1070 LET AM=A
1080 PRINT A$
1090 PRINT"LEFT - INNER POSITION
1100 GOSUB 3000
1110 LET R0=R
1120 LET A0=A
1130 REM INPUT OF DATA
1140 PRINT
1150 INPUT"NUMBER OF SEGMENTS";SG
1160 IF SG=0 THEN END
1170 LET SS=0
1180 FOR S=1 TO SG
1190 PRINT"AMOUNT #";S;
1200 INPUT SEG(S-1)
1210 LET SS=SS+SEG(S-1)
1220 NEXT S
1230 REM STANDARDIZE VALUES
1240 FOR S=1 TO SG
1250 LET SEG(S-1)=SEG(S-1)/SS
1260 NEXT S
1270 PRINT"PLEASE INSERT PAPER !"
1280 PRINT"DEPRESS E3 WHEN READY !"
•1290 IF USR(E3)=0 THEN GOTO 1290
1300 REM PLOT
1310 REM I BOUNDARY
1320 LET R=R0
1330 LET PEN=OFF
1340 GOSUB 4000 :REM R-MOVEMENT
1350 LET A=A0
1360 LET PEN=OFF
1370 GOSUB 5000 :REM A-MOVEMENT
1380 LET A=AM
1390 GOSUB 6000 :REM PEN DOWN
1400 GOSUB 5000
1410 LET R=RM
1420 LET PEN=CW
1430 GOSUB 4000 :REM R-MOVEMENT
1440 LET A=A0
1450 GOSUB 6000 :REM PEN DOWN
1460 GOSUB 5000 :REM A-MOVEMENT
1470 LET R=R0
1480 GOSUB 6000 :REM PEN DOWN
1490 GOSUB 4000 :REM R-MOVEMENT
1500 REM II PLOT BORDERS OF SEGMENTS
1510 LET SS=0
1520 FOR S=1 TO SG-1
1530 LET SS=SS+SEG(S-1)
1540 LET A=A0+INT((AM-A0)*SS+.5)
1550 LET PEN=OFF
1560 GOSUB 5000 :REM A-MOVEMENT
1570 LET R=RM
1580 GOSUB 6000 :REM PEN DOWN
1590 GOSUB 4000 :REM R-MOVEMENT
1600 LET R=R0
1610 LET PEN=OFF
1620 GOSUB 4000 :REM R-MOVEMENT
1630 NEXT S
1640 GOTO 1140
1650 END
3000 REM TAKE THE VALUES
```

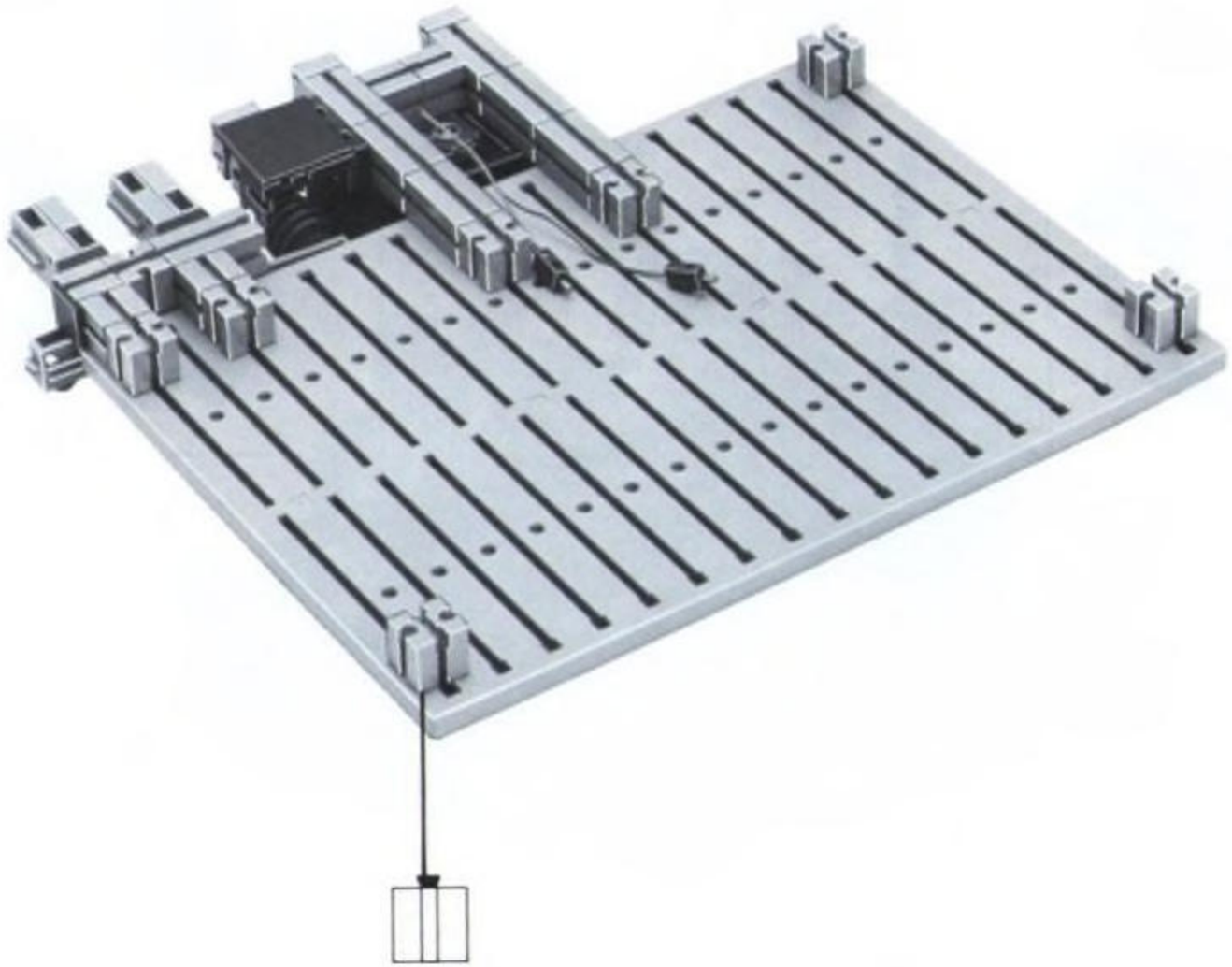
```
•3010 SYS INIT :REM ALL MOTORS OFF
•3020 IF USR(E4)=0 THEN GOTO 3050
•3030 SYS M1,CW
 3040 GOTO 3020
•3050 SYS M1,OFF
•3060 IF USR(E5)=0 THEN GOTO 3090
•3070 SYS M1,CCW
 3080 GOTO 3060
•3090 SYS M1,OFF
•3100 IF USR(E6)=0 THEN GOTO 3130
•3110 SYS M2,CW
 3120 GOTO 3100
•3130 SYS M2,OFF
•3140 IF USR(E7)=0 THEN GOTO 3170
•3150 SYS M2,CCW
 3160 GOTO 3140
•3170 SYS M2,OFF
•3180 IF USR(E3)=0 THEN GOTO 3010
•3190 LET R=USR(EX)
•3200 LET A=USR(EY)
•3210 IF USR(E3)=1 THEN GOTO 3210
 3220 REM WAIT UNTIL KEY IS NO LONGER DEPRESSED
 3230 RETURN
 4000 REM R-MOVEMENT
•4010 SYS M1,PEN
•4020 LET D=USR(EX)-R
•4030 IF D>0 THEN SYS M2,CW
•4040 IF D<0 THEN SYS M2,CCW
 4050 IF D=0 THEN RETURN
 4060 LET D=ABS(D)
 4070 IF D>GR THEN GOTO 4020
 4080 FOR I=0 TO D
 4090 NEXT I
•4100 SYS M2,OFF
 4110 GOTO 4020
 5000 REM A-MOVEMENT
•5005 SYS M1,PEN
•5010 LET D=USR(EY)-A
•5020 IF D>0 THEN SYS M1,CW
•5030 IF D<0 THEN SYS M1,CCW
 5040 IF D=0 THEN RETURN
 5050 LET D=ABS(D)
 5060 IF D>GA THEN GOTO 5010
 5070 FOR I=0 TO D
 5080 NEXT I
•5090 SYS M1,OFF
 5100 GOTO 5010
 6000 REM PEN DOWN
 6010 LET PEN=CW
 6020 FOR I=0 TO 100
•6030 SYS M1,PEN :REM DELAY LOOP
 6040 NEXT
 6050 RETURN
```

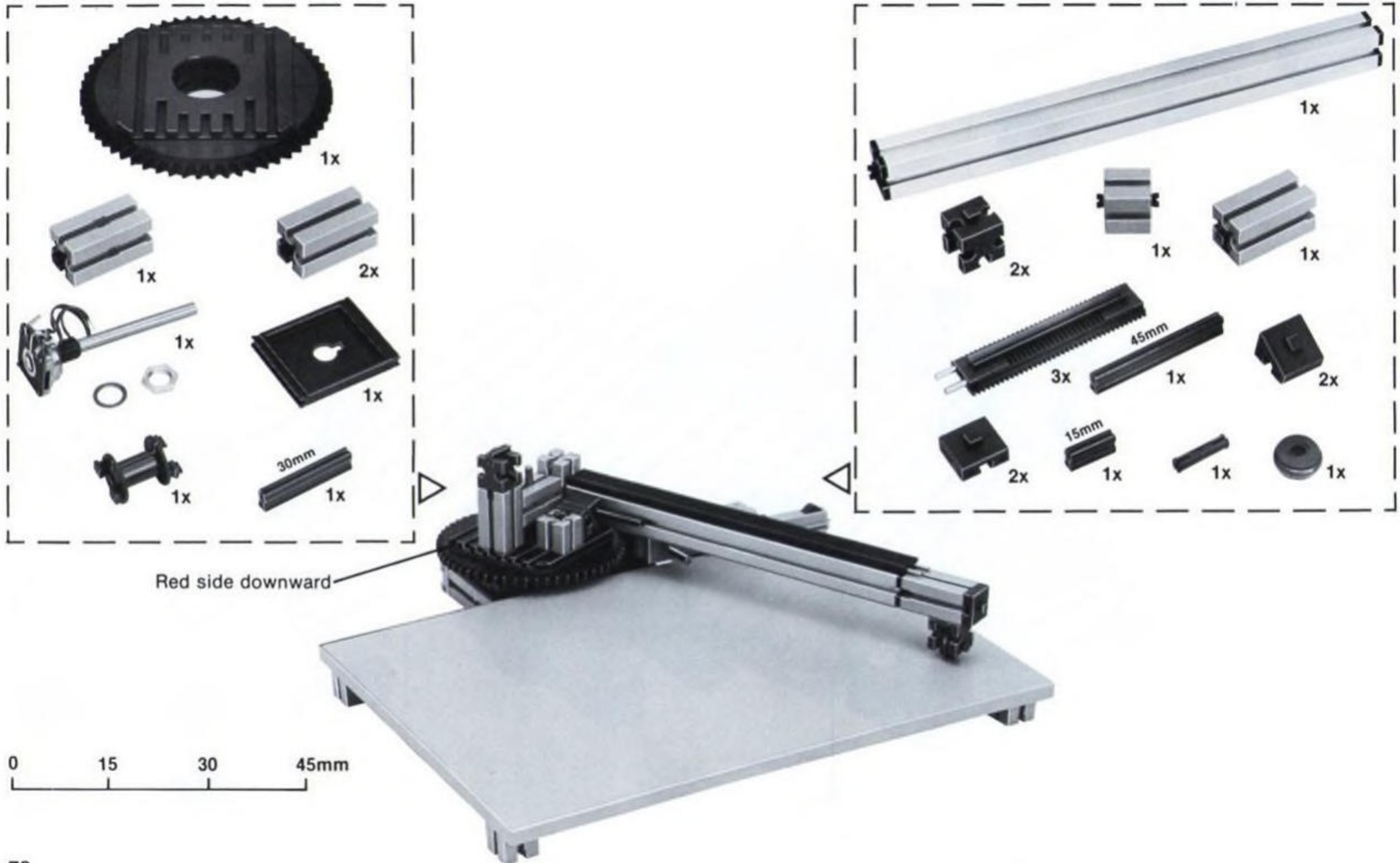

0
15
30
45mm

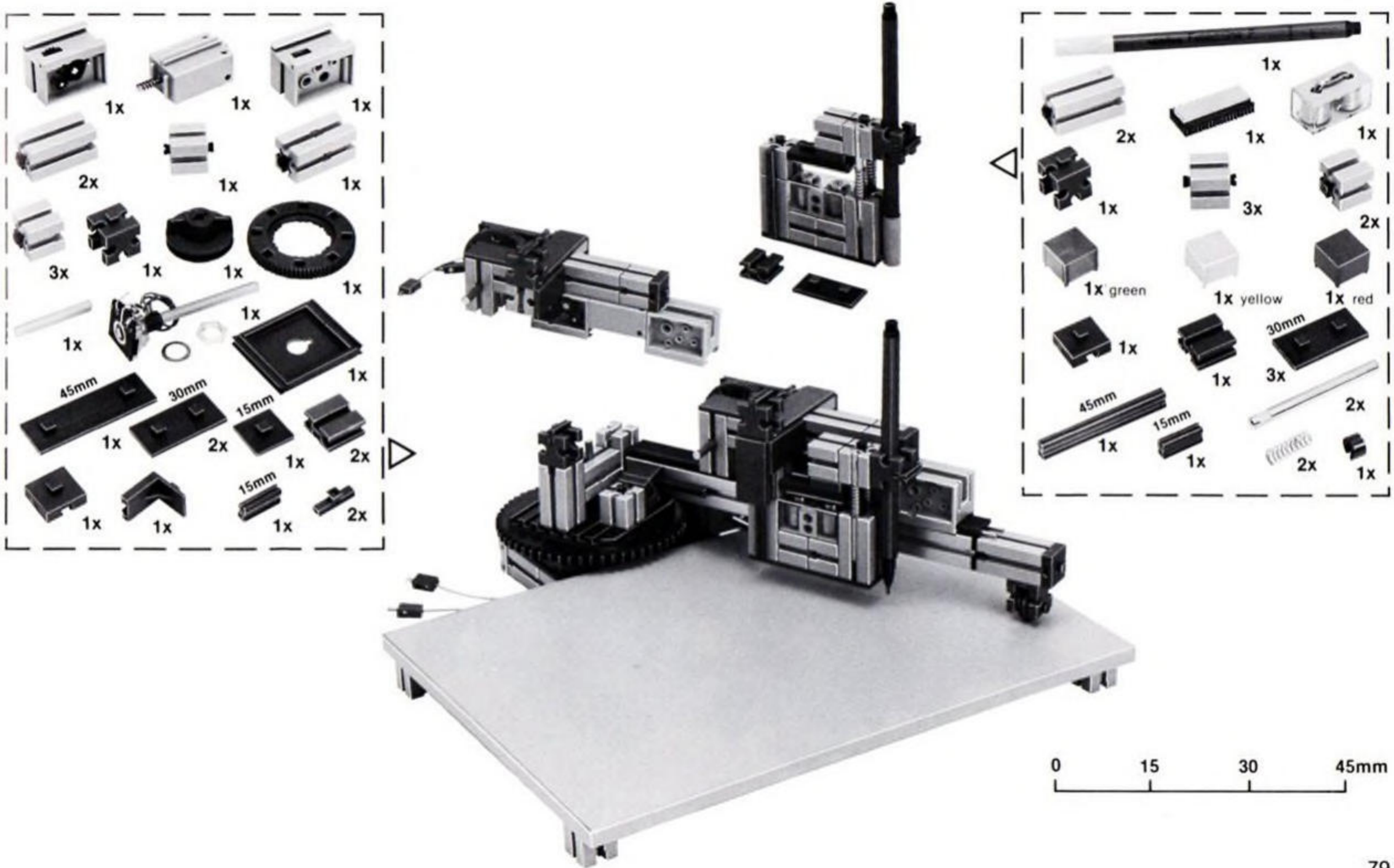

45mm
2x
1x
7x
3x
1x
2x
1x
2x
3x
15mm
2x
6x
3x
3x

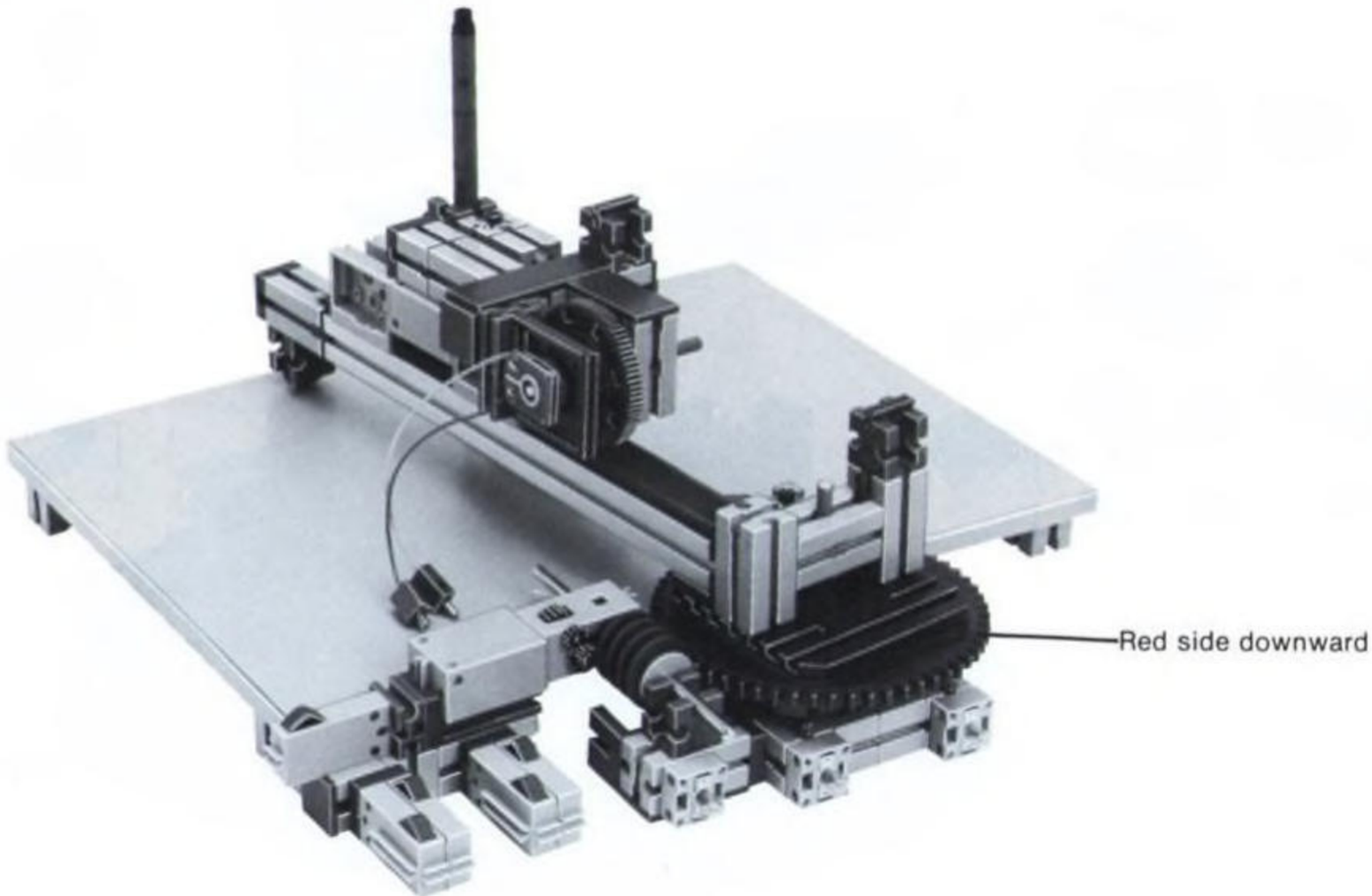

1x
1x
1x
4x
2x
1x
3x
3x
5x
30mm
1x
1x
1x

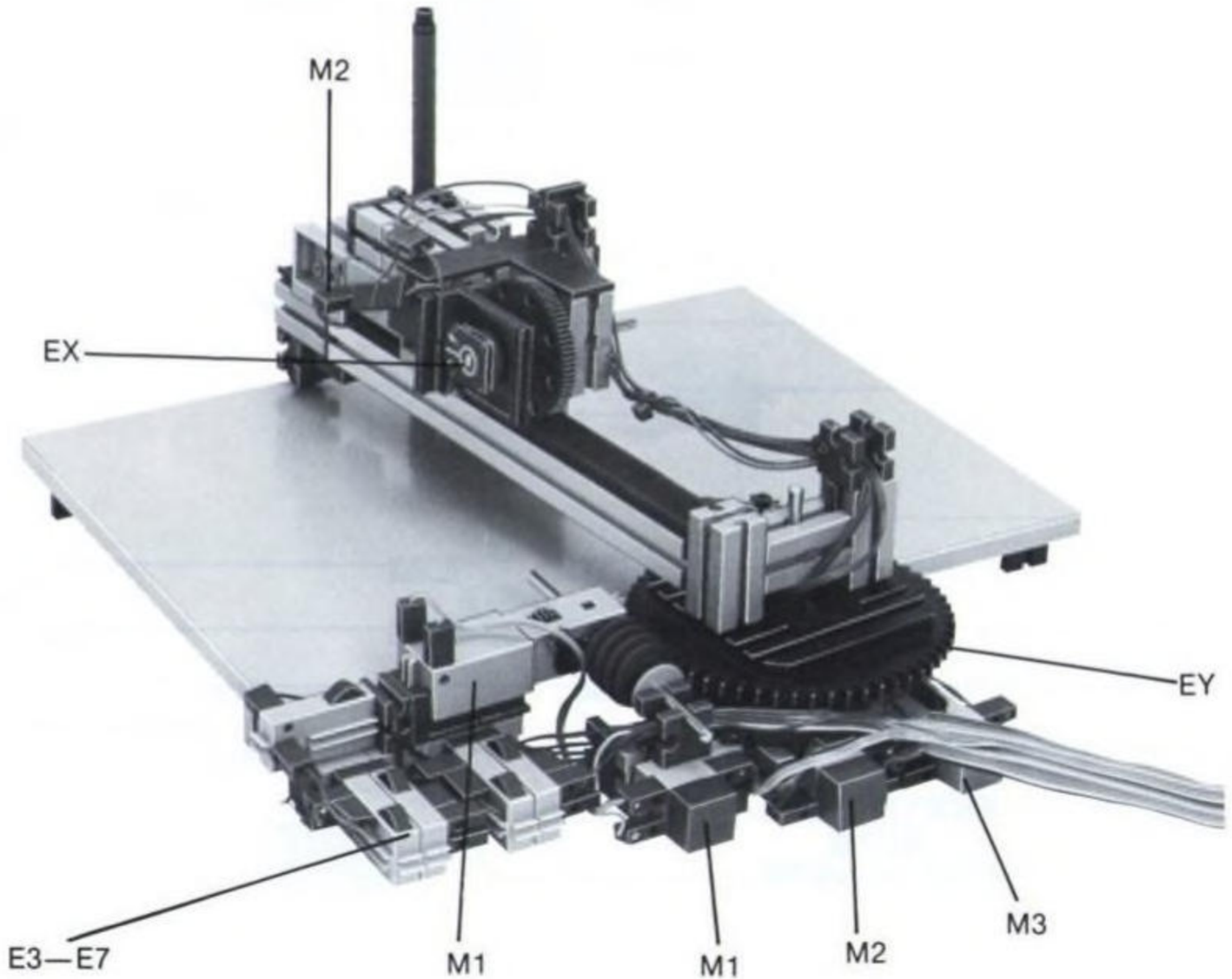




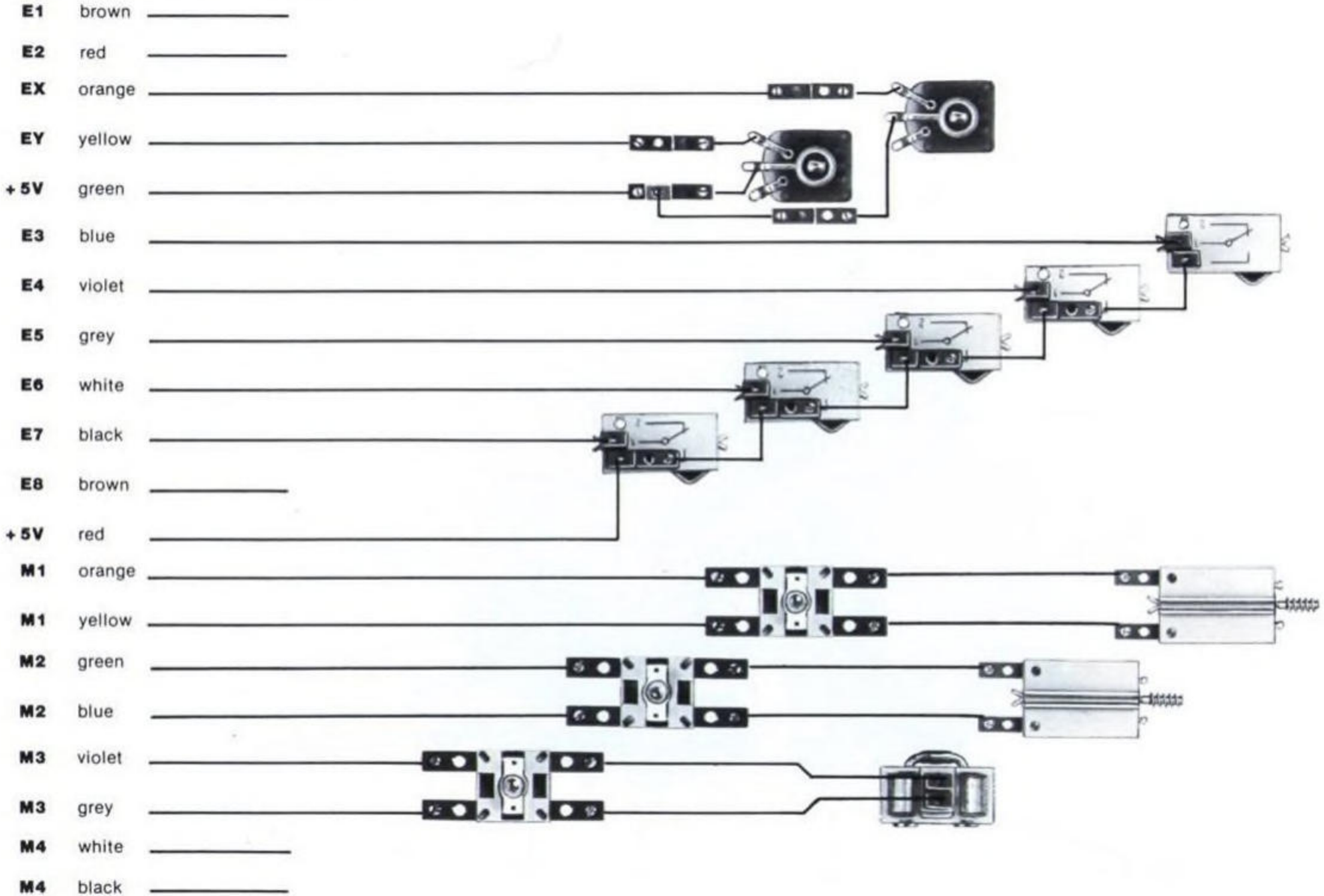

1x
1x
2x
1x
1x
1x
1x
30mm
1x
1x
2x
1x
1x
3x
45mm
1x
2x
2x
15mm
1x
1x
1x
Red side downward
0
15
30
45mm


1x
1x
1x
2x
1x
1x
3x
1x
1x
1x
1x
1x
45mm
1x
30mm
2x
15mm
1x
2x
1x
1x
15mm
1x
2x
1x
2x
1x
1x
1x
3x
2x
1x green
1x yellow
1x red
1x
1x
30mm
3x
45mm
1x
15mm
1x
2x
2x
1x
0
15
30
45mm


Red side downward


M2
EX
EY
E3—E7
M1
M1
M2
M3

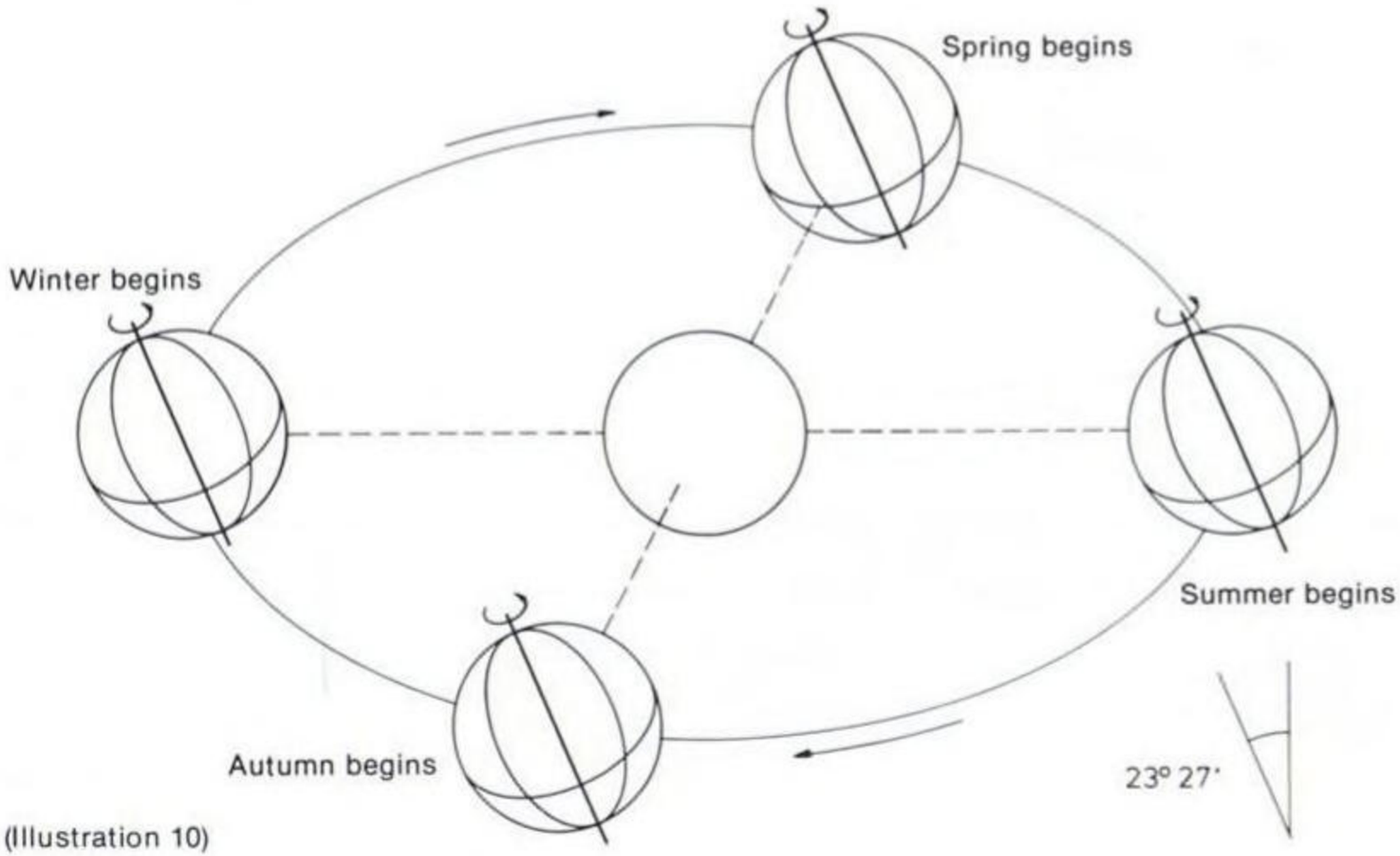
Circuit Layout—Plotter



Solar Tracking System

In certain regions on the earth, combustion or nuclear power plants may someday be replaced by solar cell stations. For our discussion, the important factor to be considered is that the cell produces maximum energy output when it is perpendicular to the incident solar radiation. This defines our programming goal: to maintain an aim point towards the sun, even though the earth rotates on its own axis and moves around the sun at changing angles of inclination (illustration 10).

These motions are the basis for our daily motion of the sun in the sky, as well as for our seasonal variations. These, in turn, are influenced not only by time and date, but by our position on the earth as well, defined by our latitude, or position north or south of the equator. At the polar latitudes, there is one day each year when the sun does not set and one on which the sun does not rise. These occur on the 21st of June and the 21st of December, respectively. Other important dates for us are the beginning of Spring (March 20) and the beginning of Autumn (September 23).



(Illustration 10)

You can easily see that the formulas to describe these motions are most complicated, involving spherical trigonometry. But before going further, an even more basic question must be answered. What time is it? No problem, you might say, and offer the time with the accuracy of the quartz-timed watch on your wrist. But watch time fails to take into account the variations between our daily time and the real time of the sun; it is *this* difference that we make up every four years with leap year. Furthermore, our time is divided into uniform time zones around the earth, while the actual sun time on an astronomical basis is calculated more precisely on the longitude of the place of observation. Since the earth travels around the sun in an elliptical, rather than circular, orbit, this leads to even more complicated calculations.

Luckily, this has all been taken into account by the BASIC program provided. The accuracy of the program regarding the orbital positions of earth and sun is better than 1°, which is far more accurate than the mechanical precision of the model and the definition of angle possible with the potentiometer. If you are an amateur astronomer, however, you may find this accuracy to be a great help in tracking celestial events.

To summarize, this is what the program will require in input:

Indication of time: year, day of year, hour, minute, second, and time zone, including daylight saving time in summer. Indication of place in latitude and longitude. If a map with these details is not readily available, your town or city engineer or the geology department of the nearest college should be able to help find the data you need.

The program provides you with the angles of the two axes of motion of the tracking system: the azimuth indicating the deviation from the south and the inclination or height above the horizontal.

As we did before with the graphic panel, potentiometer values are converted to angular degrees by calibration. If you desire more accurate data, the calibration should be performed at several points of the tracker's line of travel. These values (potentiometer value against angle setting) are plotted on a chart. By connecting the single points with a line, you will have a graphic representation of the linearity (which may vary from one potentiometer to the other) of your potentiometer. This chart can be saved and used for conversion purposes.

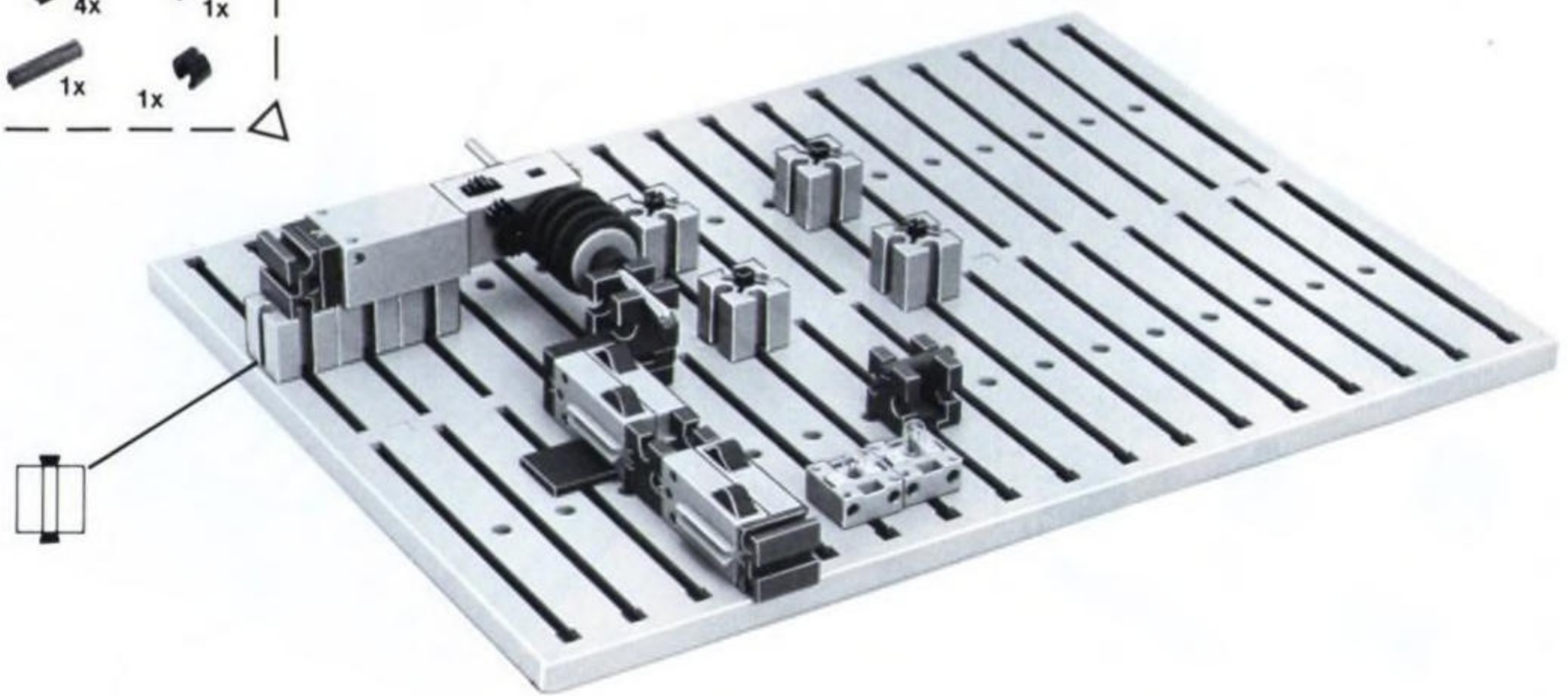
The control of the motors for this model is similar to that of the rotor models. To avoid the oscillation around the desired point, we suggest you program the motor outputs to provide very short on pulses as the rotor nears the desired setting. There is no need for speed here; repositioning approximately every 10 minutes will be more than sufficient. An attempt at an accelerated cycle, such as having one day pass in two minutes real time, will very quickly reach the capacity of the BASIC interpreter to handle the calculations required.

```
•500 SYS INIT
 510 REM
 520 REM FISCHERTECHNIK COMPUTING
 530 REM
 540 REM SOLAR CELL TRACKING
 550 REM
 560 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 570 REM
 580 REM ASSIGNMENTS FOR THE INTERFACE
 590 REM
 600 REM INPUT
 610 REM E3=MANUAL CONTROL UP/RIGHT
 620 REM E4=MANUAL CONTROL DOWN/LEFT
 630 REM E5=MOTOR-SELECT
 640 REM E6=ACCEPT KEY
 650 REM EX=ELEVATION
 660 REM EY=AZIMUTH
 670 REM
 680 REM OUTPUT
 690 REM M1=AZIMUTHAL DRIVE MOTOR
 700 REM M2=ELEVATION DRIVE MOTOR
 710 REM
 720 REM FUNCTION DESCRIPTION :
 730 REM THE FOLLOWING PROGRAM CALCULATES THE POSITION OF THE
 740 REM SUN AND CONTROLS THE SOLAR CELL TRACKING
 750 REM INPUT PARAMETERS ARE :
 760 REM THE YEAR, THE DAY IN THE YEAR,
 770 REM STARTING WITH 1 ON JANUARY 1ST WITH THE EXCEPTION OF
 780 REM LEAP YEARS WHERE COUNTING STARTS WITH 0.
 800 REM HOURS, MINUTES AND SECONDS OF YOUR LOCAL TIME ZONE
 810 REM THE TIME ZONE RELATIVE TO GREENWICH UNIVERSAL TIME.
 820 REM COUNTING IS POSITIVE TO THE WESTAND NEGATIVE TO THE
     EAST.
 840 REM THE LOCAL LATITUDE "POSITIVE TO THE NORTH"
 850 REM THE LOCAL LONGITUDE WEST TO GREENWICH.
 860 REM THE FORMULAS ARE TAKEN FROM
 870 REM R.WALRAVEN IN SOLAR ENERGY,VOL20(1978), P.393-397
 880 REM
 890 REM DEFINE THE FUNCTION ARC SINE
 900 DEF FNAS(X)=ATN(X/SQR(1-X*X))
 910 REM
 920 REM CONSTANTS 2*\ AND \/180
 930 DATA 6.2831853,0.017453293
 940 READ TWOPI,RD
 950 LET AL=20 :REM MIN. VALUE OF EY
 960 LET AR=200 :REM MAX.VALUE OF EY
 970 LET EM=20 :REM MIN. VALUE OF EX
 980 LET ER=200 :REM MAX. VALUE OF EX
 990 LET EA=EM
 995 AA=AL
1000 LET GA=5 :REM BRAKE PERIOD FOR AZIMUTH MOTOR
•1050 PRINT CHR$(147)
1060 PRINT"FISCHERTECHNIK"
1070 PRINT"COMPUTING"
1080 PRINT
1090 PRINT"SOLAR CELL TRACKING"
1110 PRINT
1120 INPUT"YEAR";YEAR
1130 INPUT"DAY OF YEAR";DAY
1140 INPUT"TIME ZONE";TZ
1150 INPUT"DEGREE OF LATITUDE";LAT
1160 INPUT"DEGREE OF LONGITUDE";LO
1170 REM CALIBRATION
1180 PRINT"PLEASE CALIBRATE SOLAR CELL"
1190 LET A$="MOVE SOLAR CELL EXACTLY"
1200 PRINT A$+" EAST"
1210 PRINT"FACING THE HORIZON"
1220 GOSUB 1990
1230 LET Y0=Y
1240 LET E0=X
1250 PRINT A$+" WEST"
1260 PRINT "FACING STRAIGHT UPWARDS"
1270 GOSUB 1990
1280 LET A0=(Y+Y0)/2
1290 LET FA=(Y0-Y)/180
1300 LET FE=(X-E0)/90
1310 PRINT"INPUT TIME"
1320 INPUT"HOUR : ";H$
1330 LET H$=RIGHT$("00"+H$,2)
1340 INPUT"MINUTE : ";M$
1350 LET M$=RIGHT$("00"+M$,2)
1360 INPUT"SECOND : ";S$
1370 LET S$=RIGHT$("00"+S$,2)
•1380 LET TI$=H$+M$+S$:REM SET REAL TIME CLOCK
1390 LET ZR=0
•1400 IF TI$="000000" THEN LET ZR=1
1410 FOR I=0 TO 2000
1420 REM WAIT AT LEAST ONE SECOND
1430 NEXT
•1440 IF TI$=H$+M$+S$ THEN LET ZR=1
1450 REM THERE EXISTS NO CLOCK WHEN TI$ IS UNCHANGED
1460 LET DF=YEAR-1980
1470 LET LEAPYEAR=INT(DF/4)
1480 LET HOUR=VAL(H$)
1490 LET MIN=VAL(M$)
1500 LET SE=VAL(S$)
1510 LET TTIM=HOUR+(MIN+SE/60)/60+TZ
1520 LET TM=DF*365+LEAPYEAR+DAY-1+TTIM/24
1530 IF DF=4*LEAPYEAR THEN LET TM=TM-1
1540 IF (DF<0) AND (DF<>4*LEAPYEAR) THEN LET TM=TM-1
1550 LET THETA=(TWOPI*TM/365.25)
1560 LET G=-0.031271-(4.5196E-7)*TM+THETA
1570 LET EL=4.900968+(1.67474E-7)*TM
1580 LET EL=EL+(0.033434-(2.3E-9)*TM)*SIN(G)
1590 LET EL=EL+(3.49E-4)*SIN(2*G)+THETA
1600 REM LENGTH OF THE SUN
1605 LET SEL=SIN(EL)
1610 LET CL=COS(EL)
1620 LET EPS=0.40914-(6.2149E-9)*TM
1625 REM ECLIPTIC APPROX. 23.5 DEGREES
1630 LET CP=COS(EPS)
1631 LET RA=ATN(SEL*CP/CL)
1632 LET RL=0
1634 IF SEL*CP<0THEN LET RL=TWOPI/2
1635 IF SEL*CP*CL<0 THEN LET RL=RL+TWOPI1/2
1636 LET RH=RL*TWOPI/4
1638 IF RA<RL THEN LET RA=RA+TWOPI/2:GOTO 1638
1640 IF RA>RH THEN LET RA=RA-TWOPI/2:GOTO 1640
1660 LET DECL=FNAS(SEL*SIN(EPS))
1670 REM DECLINATION
1680 LET SID=1.759335*TWOPI*(TM/365.25-DF)+(1.694E-7)*TM
1690 IF SID>=TWOPI THEN LET SID=SID-TWOPI
1700 REM SIDERIAL TIME
1710 LET S=SID+(TTIM*15-LO)*RD
1720 IF S>=TWOPI THEN LET S=S-TWOPI
1730 REM LOCAL SUN TIME
1740 LET H=RA-S
1750 REM HOUR ANGLE
1760 LET PHI=LAT*RD
1770 LET E=FNAS(SIN(PHI)*SIN(DECL)+COS(PHI)*COS(DECL)*COS(H))
1780 LET A=FNAS(COS(DECL)*SIN(H)/COS(E))/RD
1790 IF SIN(E)>=SIN(DECL)/SIN(PHI) THEN GOTO 1820
1800 IF A<0 THEN A=A+360
1810 LET A=180-A
1820 LET E=E/RD
1830 PRINT H$;":";M$;":";S$
1840 PRINT "A=";A;"DEG  E=";E;" DEG"
1850 GOSUB 3000:REM CONTROL ROUTINE
1860 LET MIN=MIN+10
1870 IF MIN<60 THEN GOTO 1900
1880 LET MIN=0
1890 LET HOUR=HOUR+1
1900 IF HOUR<24 THEN GOTO 1930
1910 LET HOUR=0
1920 LET DAY=DAY+1
1930 LET M$=STR$(MIN)
1940 LET H$=STR$(HOUR)
1950 IF ZR=1 THEN GOTO 1510
1960 LET T=100*(MIN+100*HOUR)
•1970 IF VAL(TI$)<T THEN GOTO 1970
1980 GOTO 1510
1990 REM ACCEPT CALIBRATED VALUES
2000 PRINT"IF DIRECTION IS OK"
2010 PRINT"PRESS KEY E6"
2020 M=M1
•2030 IF USR(E5)=1 THEN M=M2
•2040 IF USR(E3)=0 THEN GOTO 2070
•2050 SYS M,CW :REM UP/RIGHT
2060 GOTO 2040
•2070 SYS M1,OFF
•2080 SYS M2,OFF
```

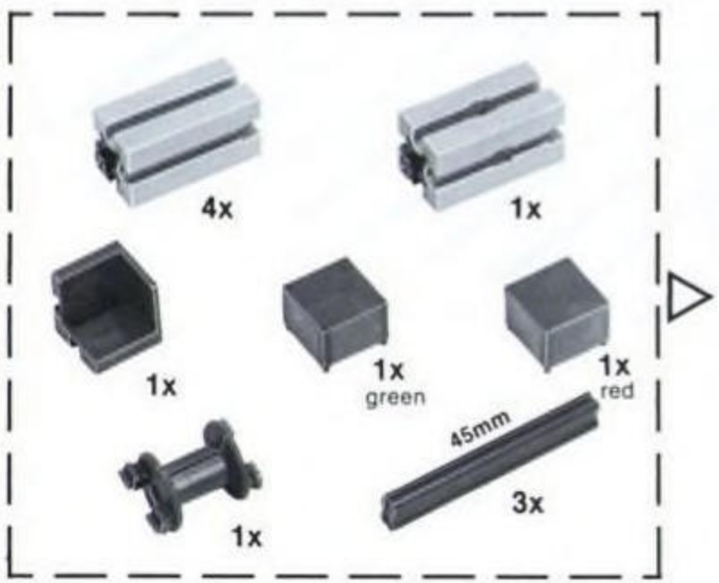
```
•2090 IF USR(E4)=0 THEN GOTO 2120
•2100 SYS M,CCW :REM DOWN/LEFT
 2110 GOTO 2090
•2120 SYS M1,OFF
•2130 SYS M2,OFF
•2140 IF USR(E6)=0 THEN GOTO 2020
•2150 LET X=USR(EX) :REM ELEVATION
•2160 LET Y=USR(EY) :REM AZIMUTH
•2170 IF USR(E6)=1 THEN GOTO 2170
 2180 RETURN
 3000 REM MOVEMENT ROUTINE
 3005 IF E<0 THEN RETURN
 3010 LET AS=INT(A*FA+A0+0.5)
 3020 LET ES=INT(E*FE+E0+0.5)
 3030 IF AS<AL THEN LET AS=AL
 3040 IF AS>AR THEN LET AS=AR
 3050 IF ES<EM THEN LET ES=EM
 3060 IF ES>ER THEN LET ES=ER
 3070 IF ES=EA THEN GOTO 3160
 3080 REM CONTROL OF ELEVATION
•3090 LET D=USR(EX)-ES
•3100 IF D>0 THEN SYS M2,CW
•3120 IF D<0 THEN SYS M2,CCW
 3140 IF D=0 THEN GOTO 3160
•3145 SYS M2,OFF
 3150 GOTO 3090
 3160 LET EA=ES
 3170 IF AS=AA THEN GOTO 3270
 3180 REM CONTROL OF AZIMUTH
•3190 LET D=USR(EY)-AS
•3200 IF D>0 THEN SYS M1,CW
•3210 IF D<0 THEN SYS M1,CCW
 3220 IF D=0 THEN GOTO 3270
 3230 LET D=ABS(D)
 3240 IF D>GA THEN GOTO 3190
•3250 SYS M1,OFF
 3260 GOTO 3190
 3270 LET AA=AS
 3280 RETURN
```

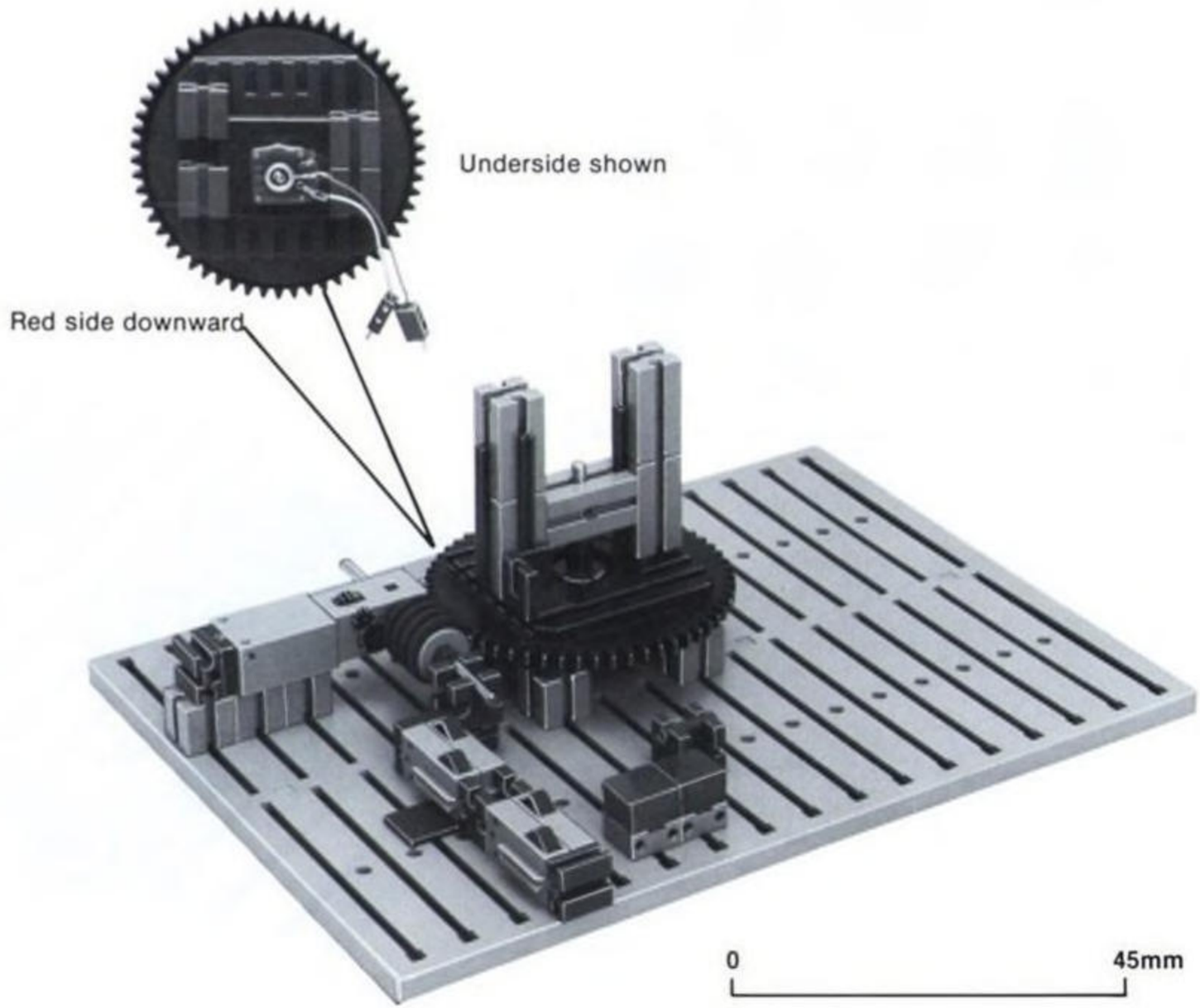
Note: An actual solar cell is not necessary to complete model.

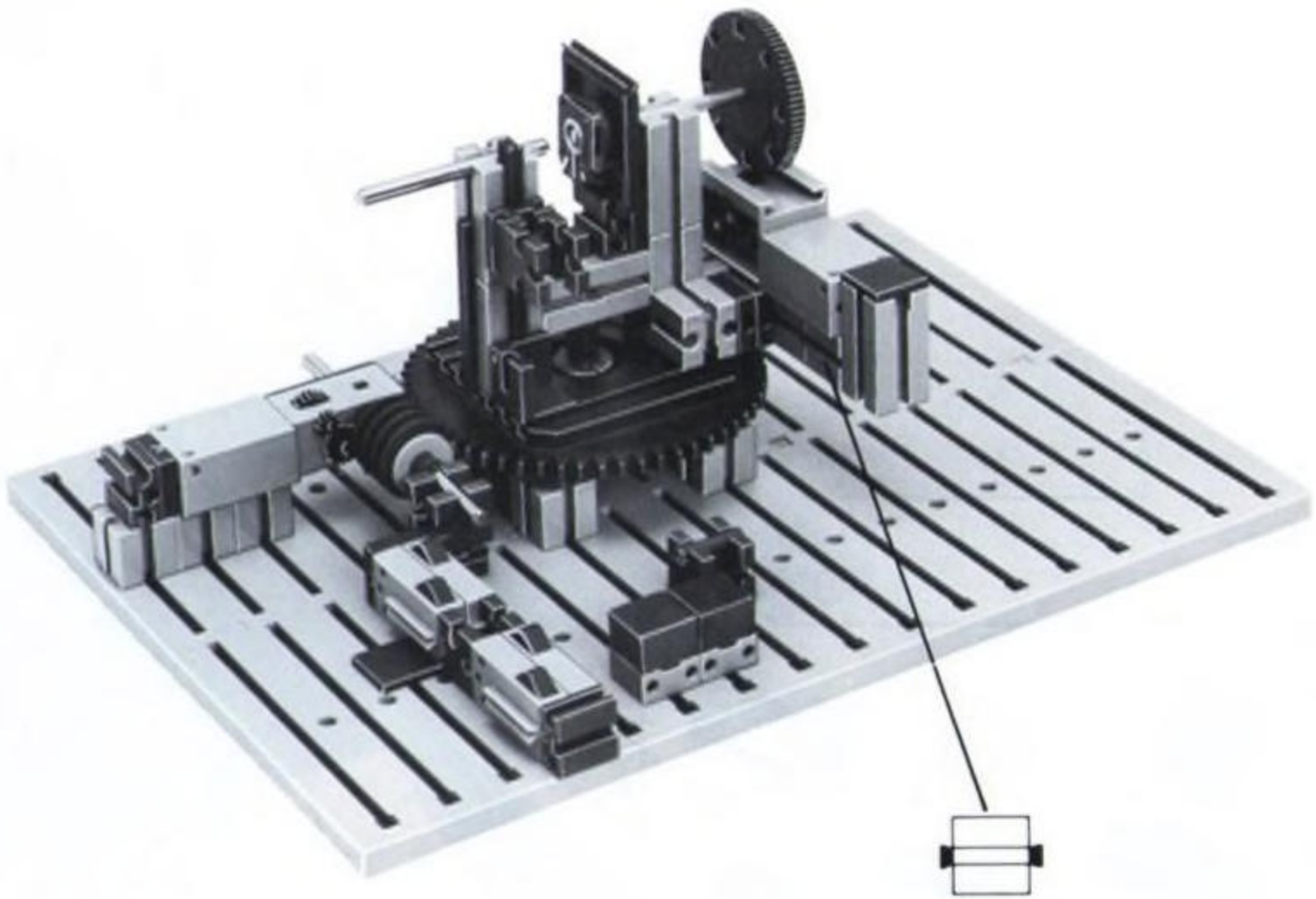

1x
1x
1x
2x
5x
2x
4x
2x
2x
4x
1x
2x
1x
1x
1x



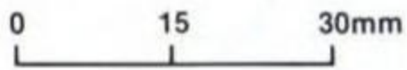

1x
1x
1x
4x
4x

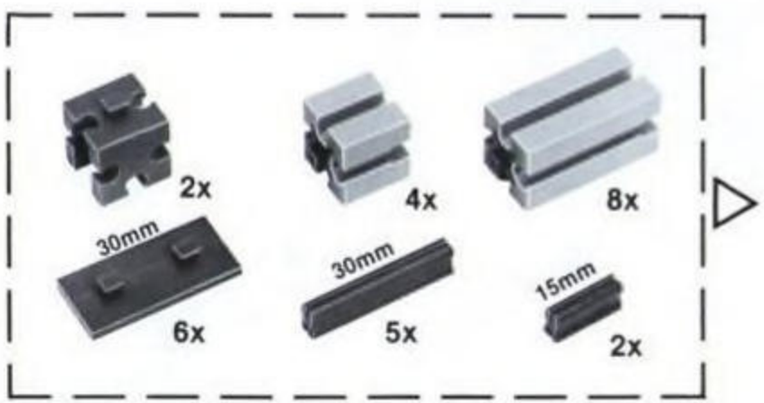

4x
1x
1x
1x
green
1x
red
45mm
3x
1x

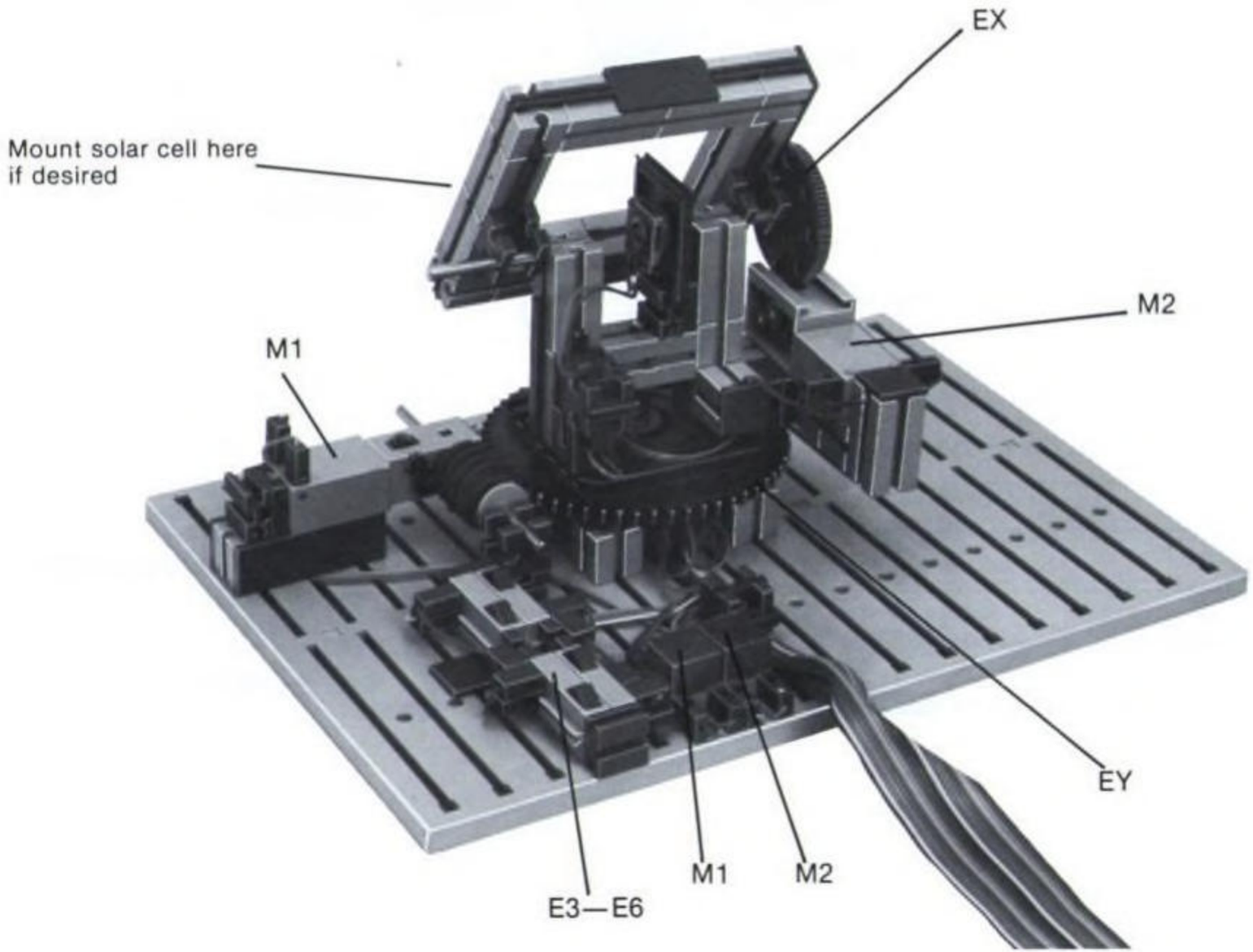

Underside shown
Red side downward
0
45mm



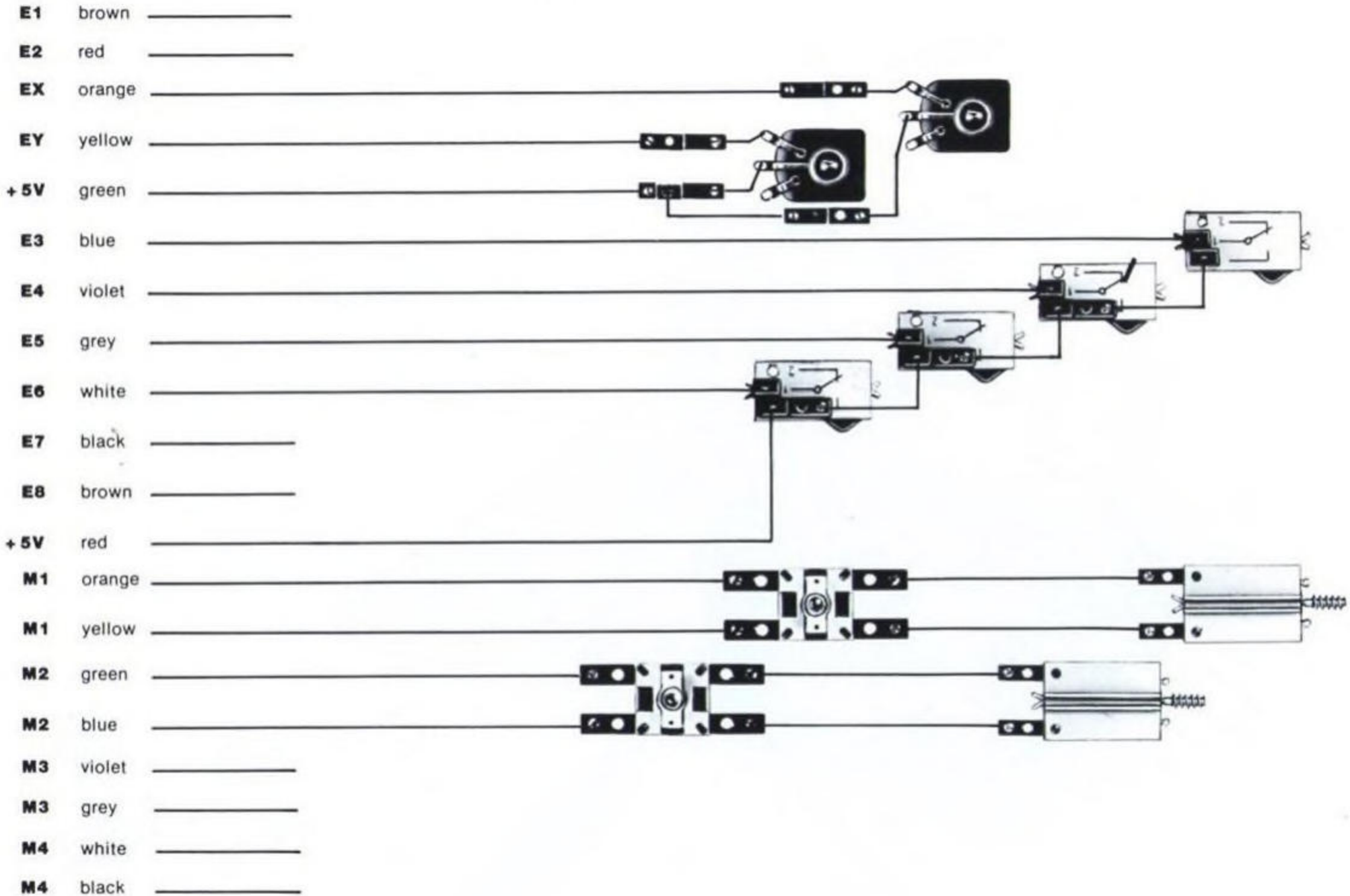
0 15mm


0
15
30mm


2x
4x
8x
30mm
6x
30mm
5x
15mm
2x


Mount solar cell here
if desired
EX
M2
M1
EY
E3—E6
M1
M2

Circuit Layout—Solar Tracking System



fischertechnik Computing—A Summary

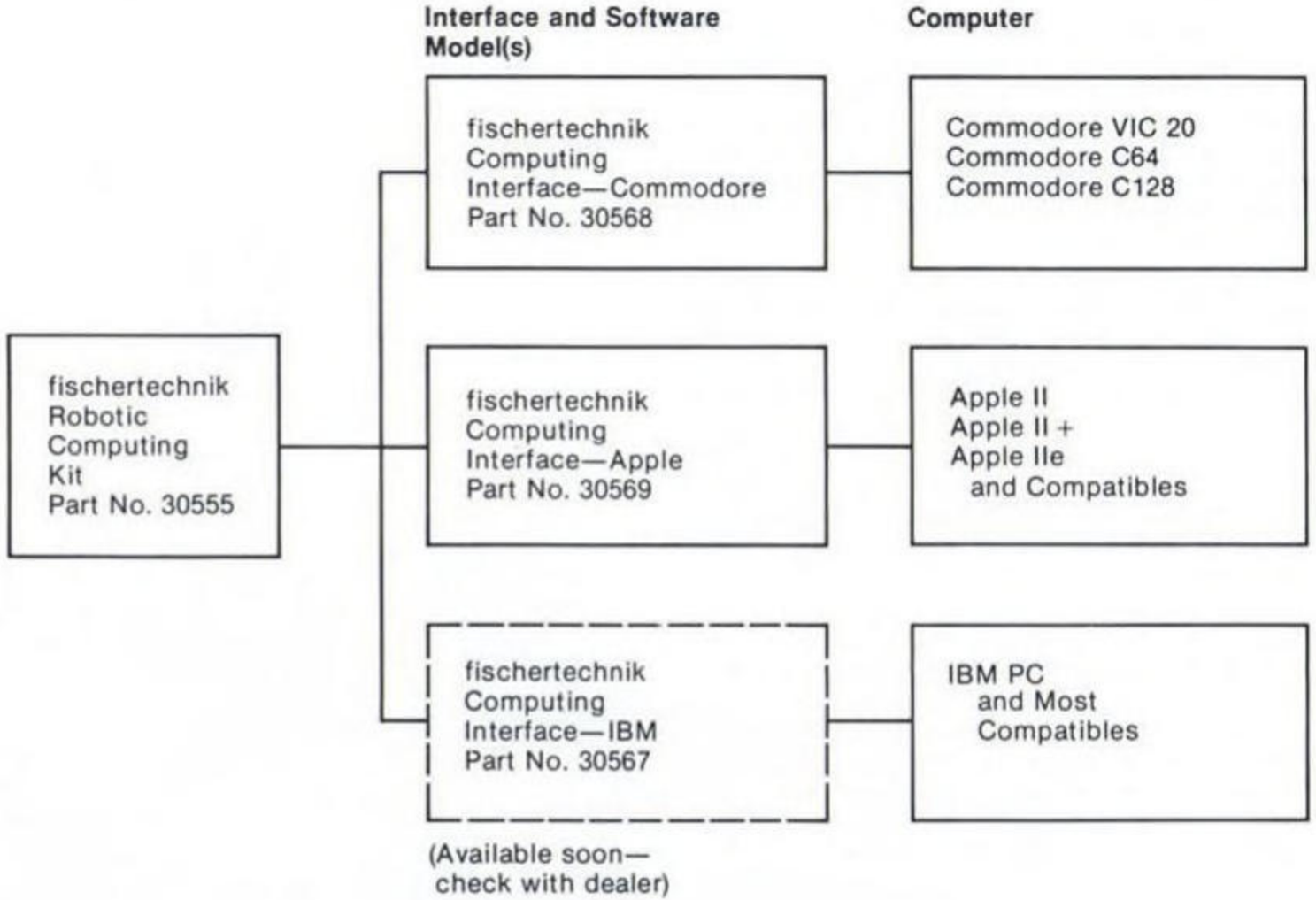
In addition to the fischertechnik Robotic Computing Kit you now have, the fischertechnik computing system also includes other model sets. These include:

The 3-Axis Training Robot—This model has been designed to imitate an actual industrial robot. With four motors, it offers the following functions: 1) rotation of the structure, 2) pivoting of the upper arm section, 3) pivoting of the lower arm section, and 4) opening/closing of the "hand."

The positioning system of this unit allows an operating accuracy of approximately 1mm across the operating range, which makes it suitable for industrial and educational uses as well as hobby activity.

The Plotter/Scanner—This is a far more accurate plotter than the one offered in the basic computing kit. In addition, it can be used to scan patterns or as an X-Y coordinated graphic panel. The positioning system used is similar to that of the training robot, and thanks to its direct drive of the read-write head, better accuracy is achieved.

Further kits are planned. Be sure to send in the registration card from your interface so that we may keep you fully informed of new projects when they become available. Interface units for other computers will also be made available, so soon your friends with brands of computers different from yours will be able to join you in the exciting world of fischertechnik computing.



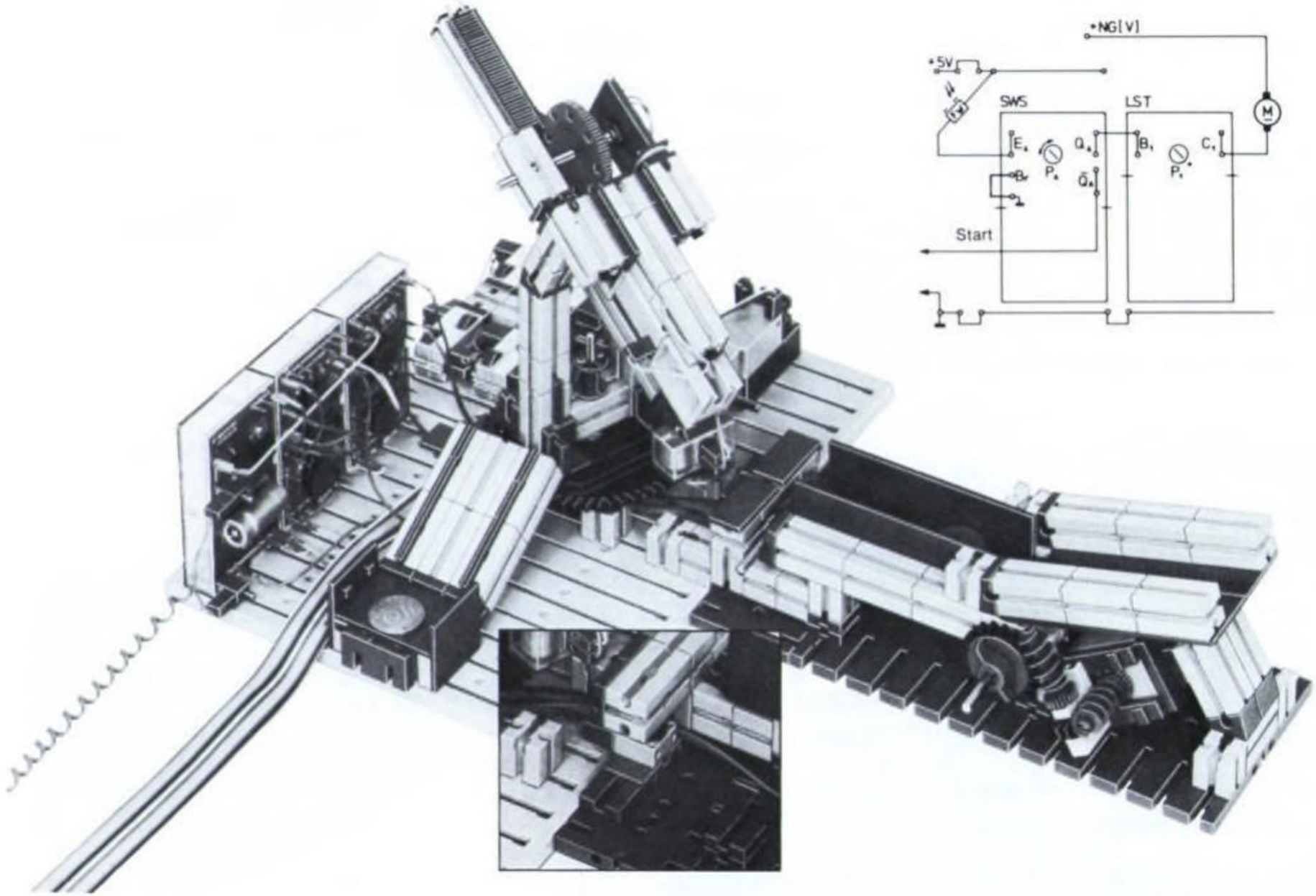
Another Robot (Pick & Place)

Here's another variation of our teachable robot from page 56. As you can see from the illustration on the next page, he's undergone some changes. Using parts from the Start 100 fischertechnik Building Set and from the Service Set, a conveyor belt and a chute have been added. The belt is driven by an additional motor taken from the Motors and Gears Set.

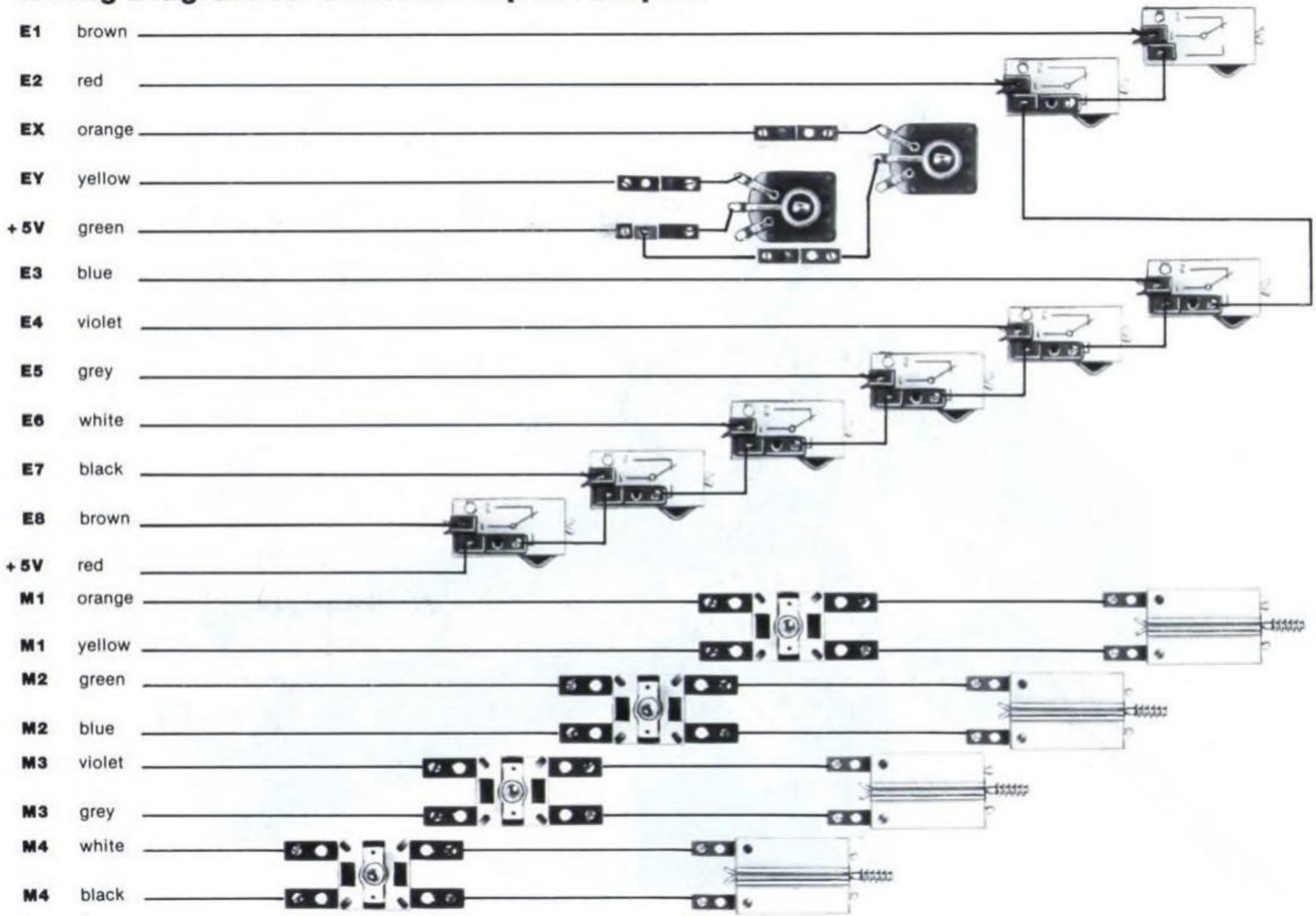
If we were to drop a magnetic metal token on the belt, we would expect it to just ride to the end and drop off. But not this time...we've added a photo-sensor which detects the presence of the coin. It is built with circuits from the Electronics Set, as shown in illustrations 29-3 and 30-7 of the instruction book for that set. The ground wire of this circuit is connected to the ground connector of the interface, and the output of the electronic circuit sends a signal to the interface that activates the robot.

We mention all this simply to make you aware that your projects are not limited to the parts that come with the fischertechnik Robotic Computing Kit... the entire fischertechnik system is available to allow you to add both mechanical and electronic elements of all kinds, so that your models can expand to the extent of your imagination. All fischertechnik parts and components are compatible, so your choices are unlimited.

```
•500 SYS INIT
 510 REM
 520 REM FISCHERTECHNIK COMPUTING
 530 REM
 540 REM PICK & PLACE
 550 REM
 560 REM COPYRIGHT (C) ARTUR FISCHER FORSCHUNG 1984
 570 REM
 580 REM ASSIGNMENTS FOR THE INTERFACE
 590 REM
 600 REM INPUT
 610 REM E1=PHOTO SENSITIVE CELL
 620 REM E2=START
 630 REM EX=POSITION OF ARM
 640 REM EY=ANGLE POSITION
 650 REM
 700 REM OUTPUT
 710 REM M1=TURN TABLE DRIVE
 720 REM M2=ARM DRIVE
 730 REM M3=MAGNET
 740 REM M4=CONVEYOR-BELT
 750 REM
 800 REM FUNCTION DESCRIPTION :
 810 REM AFTER START KEY WAS DEPRESSED,
 820 REM THE CONVEYOR-BELT TRANSPORTS THE COIN TO THE ROBOT
 830 REM WHICH PLACES IT TO
 840 REM THE SLIDE.
 900 REM TURNING POSITIONS (EMPIRICAL VALUES)
 910 DIM TURN(2),ARM(2)
 920 LET TURN(1)=113
 930 LET TURN(2)=35
 950 LET ARM(1)= 120
 960 LET ARM(2)= 30
•1000 PRINT CHR$(147)
 1010 PRINT"FISCHERTECHNIK"
 1020 PRINT"COMPUTING"
 1030 PRINT
 1040 PRINT"PICK & PLACE"
 1050 PRINT
 1060 PRINT"PLEASE PRESS START KEY"
 1070 REM START
•1080 IF USR(E2)=0 THEN GOTO 1080
•1090 SYS M4,CW
•1100 IF USR(E1)=0 THEN GOTO 1100
•1110 SYS M4,OFF
 1120 LET NOMINAL=TURN(1)
 1130 GOSUB 1300
 1140 LET NOMINAL=ARM(1)
 1150 GOSUB 1400
 1160 REM MAGNET ON
 1170 FOR I=0TO100
•1180 SYS M3,CW
 1190 NEXT I
 1200 LET NOMINAL=ARM(2)
 1210 GOSUB 1400
 1220 LET NOMINAL=TURN(2)
 1230 GOSUB 1300
 1240 LET NOMINAL=ARM(1)
 1250 GOSUB 1400
•1260 SYS M3,OFF :REM MAGNET OFF
 1270 LET NOMINAL=ARM(2)
 1280 GOSUB 1400
 1290 GOTO 1060
 1300 REM POSTIONAL CONTROL ROUTINE FOR TURNING
•1310 LET D=USR(EY)-NOMINAL
•1320 IF D>0 THEN SYS M1,CW
•1330 IF D<0 THEN SYS M1,CCW
 1340 IF D=0 THEN RETURN
 1350 LET D=ABS(D)
 1360 IF D>3 THEN GOTO 1310
•1370 SYS M1,OFF
 1380 GOTO 1310
 1390 RETURN
 1400 REM POSITIONAL CONTROL ROUTINE FOR ARM
•1410 LET D=USR(EX)-NOMINAL
•1420 IF D>0 THEN SYS M2,CW
•1430 IF D<0 THEN SYS M2,CCW
 1440 IF D=0 THEN RETURN
 1450 LET D=ABS(D)
 1460 IF D>20 THEN GOTO 1400
•1470 SYS M2,OFF
 1480 GOTO 1400
 1490 RETURN
```


+NG[V]
+5V
SWS
LST
E+
P+
Q+
B+
Q̄+
B1
P1
C1
M
Start

Wiring Diagram for Interface Inputs/Outputs



Art. No. 39499 Litho in U.S.A. 389 LA