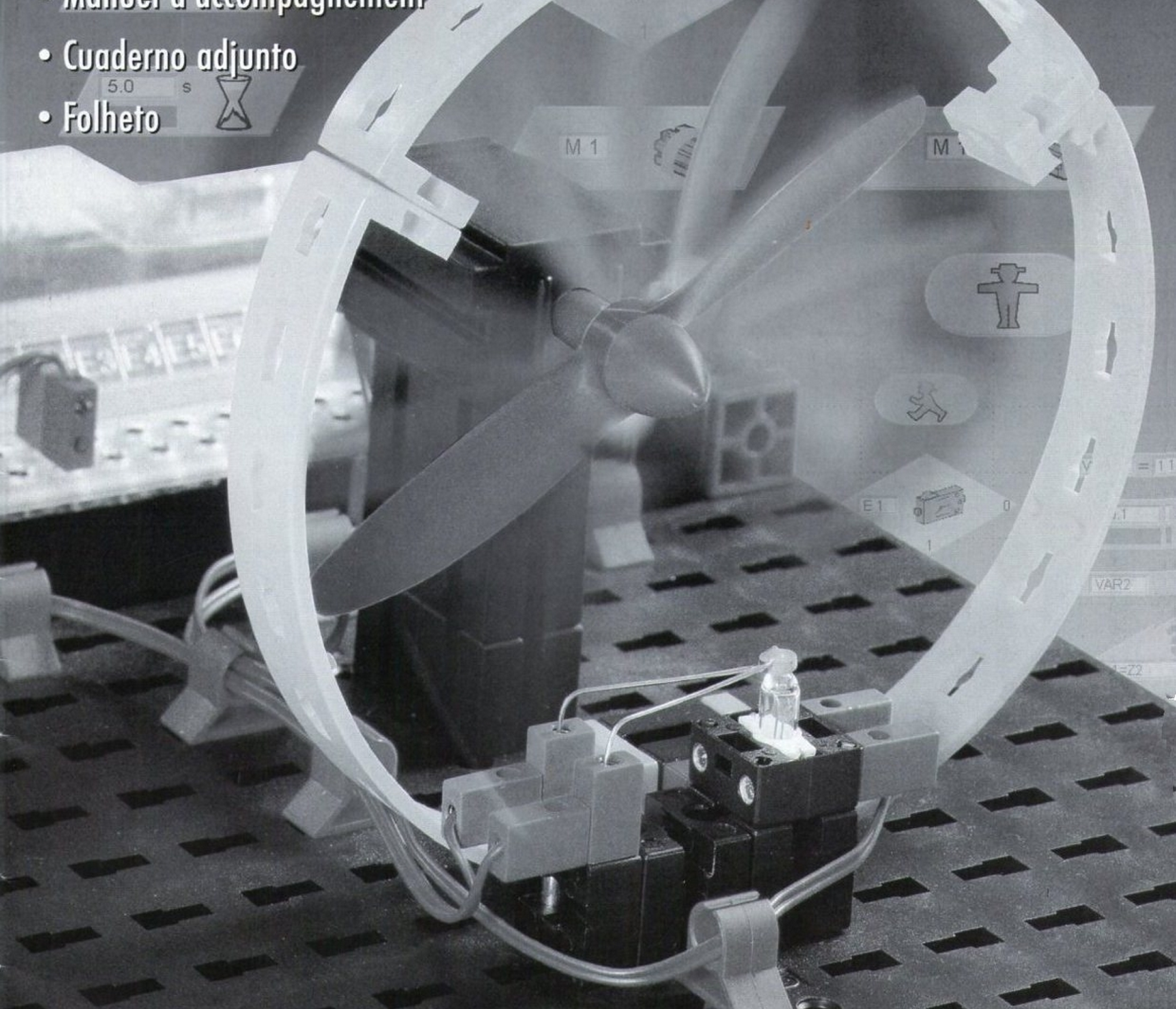
Computing Starter

- Begleitheft
- Activity booklet
- Manuel d'accompagnement
- Cuaderno adjunto
- Folheto



fischertechnik®

GB+USA

Table of Contents

GB+USA

1. Introduction

Greetings PC and fischer fan. Welcome to our computing world. At fischertechnik, computing signifies programming and controlling models using a PC. The Computer Starter construction kit provides you with an optimum way for getting started with this. You can set up eight different models, from a hand dryer to a car park barrier to a welding robot with the help the instructions in a very short time. You connect the models with the PC via the Intelligent Interface part no. 30402. Then you can program the models quickly and easily using the graphical programming software LLWin 3.0.

This Activity Booklet is designed to help you find your way around in the computing world very quickly. It first explains how you can get started and then what additional steps to take. You can also find programming tasks for the construction kit models here. Of course, tips for finding the correct solutions are also included. There are precise descriptions of how you can program the models using the LLWin 3.0 software. You'll discover how much fun it is. Okay, let's get started!

2. Getting Started

What is contained in the construction kit?

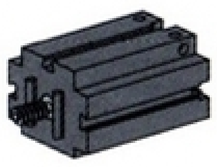
Everything is contained in the "**Computing Starter Pack**" that you need. Only a connection to a power supply (i.e., electricity) is required.

2.1 The Construction Kit

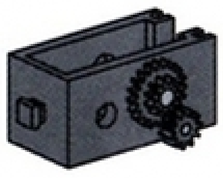
First, you can find numerous fischertechnik building blocks, motor, lamps and sensors as well as color instructions for building eight different models. Let's look at them first.

After you have unpacked all building blocks, you must assemble a few components before you can start (e.g., cable and plug). The exact ones are described in the assembly instructions under "Assembly Help and Notes." Take care of that first.

The most important construction components are:



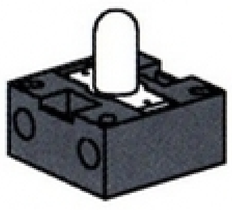
Motor: This motor drives the fischertechnik models. It is powered with 9 volt DC (direct voltage). The maximum power is approx. 1.1 watt with 7000 rpm (revolutions per minute).



Gear unit: A gear unit is installed on the motor, which revs down the rpm. The rev down is composed of the toothed gear with the output shaft at a ratio of 64.8:1.

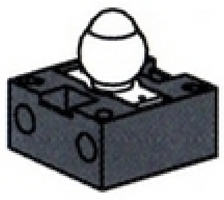
Lamps:

Two different lamps are available in the construction kit.



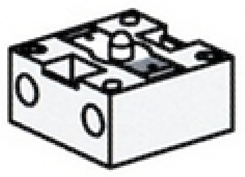
Bulb lamp:

This is a normal incandescent bulb for 9 V DC and power consumption of approx. 0.1 A (ampere).



Lens tip lamp:

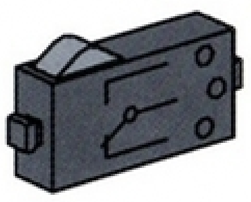
A lens is built into this lamp, which collimates the light. It looks very similar to the bulb lamp. Be careful that you do not mix them up. You need the lens tip lamp to construct a light barrier.



Phototransistor:

This is a "brightness sensor". It is a sensor that reacts to brightness. It is the counterpart to the lens tip lamp for a light barrier. When it is very bright, i.e., when a lamp shines on the transistor, it provides current. If the light beam is interrupted, the transistor does not provide any current.

Caution:
When you connect the phototransistor to the power supply, make certain that you connect the correct poles: red = plus.



Pushbutton: The pushbutton is also called the contact sensor. When you press the red button, a switch is activated, and current flows between the contacts 1 (middle contact) and 3. At the same time, the connections between 1 and 2 are interrupted. In this way, you can use the pushbutton in two different ways:

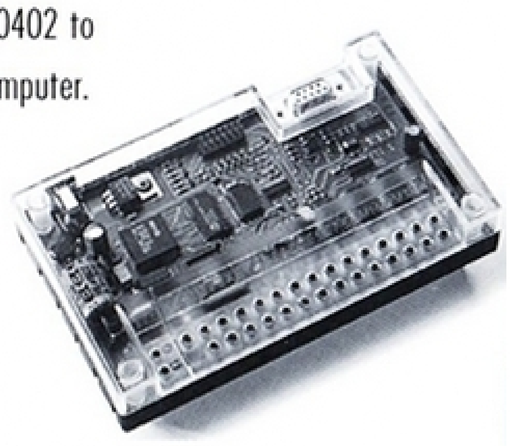
As a "connector", contacts 1 and 3 are connected.
Button pushed = current; button not pushed = no current.
As an "opener", contacts 1 and 2 are connected.
Button pushed = no current; button not pushed = current.

NTC Resistance: This component is a thermal sensor, with which you can measure temperature. The resistance at 20 °C is 1.5 kΩ (kilo-ohm). **NTC** means **N**egative **T**emperature **C**oefficient. This simply means that the resistance value sinks with increasing temperature.

You can send the information that sensors provide (e.g., light/dark, pressed/not pressed, temperature values) to the PC via the interface. Then – with the help of the software – you can program a motor, so that it opens a door as soon as the light barrier is interrupted, for example.

2.2 Interface and Software

Before you start to build models and write programs, you must install the control software LLWin 3.0 on your PC.



Then connect the Intelligent Interface part no. 30402 to an unused serial port (COM1–COM4) of your computer. Chapters 1 and 2 of the LLWin handbook describe how to do this in detail. Follow the instructions there exactly, and you should not have any problems to get the software and interface running. The CD with the software is "hidden" in the LLWin 3.0 handbook. You need a fischertechnik power supply unit with 9 V DC and amperage of 1000 mA for the interface (e.g., Energy Set part no. 30182 or Accu Set part no. 34969).

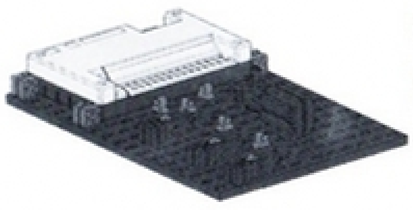
Because many of you already have these fischertechnik components, no power supply is included in the "Starter Pack" package.

Here's wishing you success installing and connecting the software and interface. Then you can continue.

2.3 First Programming Steps

After the hardware and software are running properly, you can finally start programming. Use the LLWin 3.0 handbook for this too. No better introduction to programming exists than in Chapters 3 and 4. Consequently, refer to those chapters now and work through them.

You will notice that "Testing the First Control Program" in Chapter 3.5 refers to the "Motor Control" model in the Computing Starter construction kit.

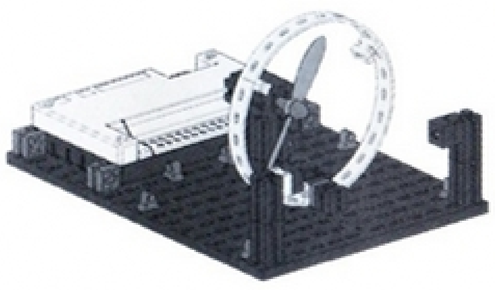


Construct this model following the instructions and then test your first program. Have fun!

3. Programming Tasks – Part I

After you have read through Chapters 3 and 4 of the LLWin 3.0 handbook, you can then program a few models of the Computing Starter construction kit. Let's start right now. Every time you have completed a model and wired it, check using the Check Interface diagnosis tool whether all outputs and inputs are connected correctly and whether the sensors, motors and lamps function properly.

3.1 Hand Dryer (see assembly instructions page 6)



New hand dryers were installed next to the sink in the rest rooms of your school. They are equipped with a light barrier via which a blower is switched on and off. First construct the model.

Task 1:

Program the hand dryer, so that as soon as the light barrier is interrupted, the blower is switched on and then off after five seconds. Create a new project for this in LLWin 3.0.

Programming tips:

- First switch on the lamp at M2 for the light barrier (OUTPUT block).
- Then wait a second, so that the phototransistor has time to react to the light (WAIT block).
- Then query the phototransistor E1 (INPUT block). If the value equals 1 (light barrier not interrupted), the phototransistor should be queried in a loop continually. If the value is 0 (light barrier interrupted), switch the motor M1 on (OUTPUT block) and then off again after five seconds (WAIT and OUTPUT blocks). Then the phototransistor should be queried again.

Start the project with `Run – Start` and check whether the program functions properly. If yes, then you are on the right path to becoming a professional LLWin programmer.

If it does not function properly, try to find out where the problem is:

- You can check in `Run – Init` whether all blocks are connected with lines and whether the entries in the function blocks are correct. If a line or an entry is missing for a block, the block appears violet.

- When the program runs in online mode (`Run – Start`), you can trace the program process using the red-marked building blocks. Then you can also detect quickly where the error crept in (e.g., if the process does not continue at the INPUT block, perhaps because E2 is queried instead of E1).

- If you still cannot get your program running, you can load the ready-to-use example project Hand dryer 1.mdl from the "Computing Starter" subdirectory of your LLWin directory and compare it with your flowchart. Then at the latest you will see where the error is.

After you have passed this first hurdle, we want to change the task a bit.

Task 2:

The principle, who is always looking for ways to save energy, is not happy that the hand dryer continues to run for a certain time even when hands are already dry. He asks you to modify the program, so that the blower switches off immediately when the hands are removed from it. No problem for you, isn't that right?

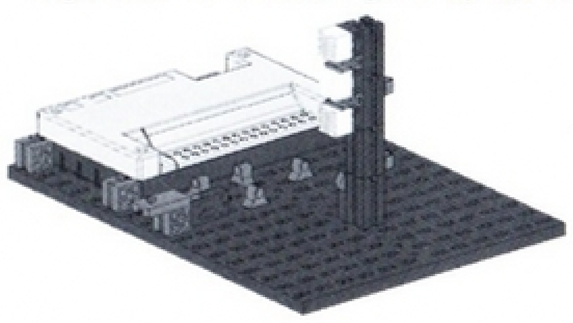
Programming Tips:

- You query the phototransistor E1 again (INPUT block). If the value equals 0, switch the motor M1 on (OUTPUT block). If the value is 1, switch the motor M1 off (OUTPUT block), and so on.

- A ready-to-use project also exists for this task for an emergency situation, Hand dryer 2.mdl.

3.2 Traffic Light (see assembly instructions page 8)

A traffic light was installed in front of your home. Because the installation engineer from the traffic light company is in a hurry, you offer to program the traffic light control for him. The man explains to you how the control should function, but first you should assemble the model.



GB+USA

Task 1:

The traffic light should normally be green. If a pedestrian presses the pushbutton E1, the light should switch to yellow three seconds later and then to red four seconds after that. The red phase should last 10 seconds, the subsequent red-yellow phase lasts three seconds, and then it should turn green again.

Programming Tips:

- The different lamps belong to the following interface outputs: green = M3; yellow = M2; red = M1. Simply switch the lamps on and off one after another (OUTPUT block), so that the required switching sequence is achieved.
- Use the INPUT block to query the pushbutton E1.
- Ready-to-use: Traffic light 1.mdl

Task 2:

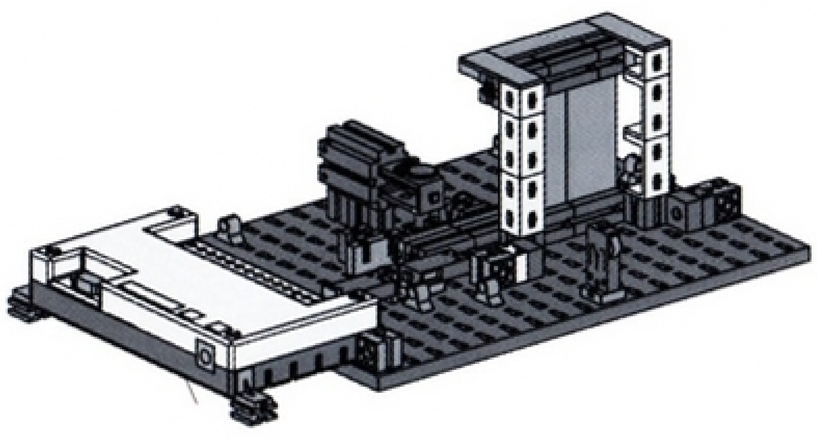
The installation engineer calls the traffic light company the next day. He forgot to tell you that a switch E2 is in the switchbox on the sidewalk, which should switch the light to a blinking yellow as soon as it is activated. You assure the installation engineer that you will integrate this function quickly into the program.

Programming Tips:

- Query with another INPUT block between the pushbuttons E1 and E2. If E2 is pressed, you branch the flowchart to a blinking light. If E1 is pressed, the light control runs as in task 1.
- You program the blinking light by switching lamp M2 on and off again 0.5 seconds. Write a "blinking light" subprogram for this. You learned in Chapter 4 of the LLWin handbook how to do this.
- Consider carefully at which point of the flowchart you query E2, so that the switch to a blinking light is not only possible at the start of the program, but instead during every green phase.
- The ready-to-use project is called Traffic light 2.mdl by the way. But don't look at it right away. Try your hand at it first.

3.3 Sliding Door (see assembly instructions page 10)

The supermarket in which you have a part-time job stocking the shelves has received a new entrance door. The control software must be written for this now. The store manager knows that you are an expert in programming and asks you to handle it. But first build the model.



Task 1:

When pushbutton E3 is pressed, the door should open and then close again after five seconds.

Programming Tips:

- First close the door. Then it is in the starting position. Let the motor M1 run to the left for this until the limit switch E1 becomes "0" (INPUT block).
- Query pushbutton E2 (INPUT block). If it is pressed, the door opens. Run the motor M1 to the right for this until the limit switch E2 changes to "1".
- Close the door again (loop back to the beginning) after five seconds (WAIT block).
- Ready-to-use project: Sliding door 1.mdl.

Task 2:

The door control works well. However, when the first customer gets a leg stuck in the door because he tried to pass through just at the moment when it closed, you decide to improve the program. The door has a light barrier, which should prevent the door from closing when somebody is passing through. You want to expand the program, so that:

1. The door is only closed when the light barrier is not interrupted.
2. The door opens again when the light barrier is interrupted during closing.
3. The door also opens without the button being pressed as soon as the light barrier is interrupted.

You have set yourself a difficult task! But actually, it's not all that difficult. Maybe you need to think it over a bit. But don't look at the finished project yet. A bit of mind-training never hurt anybody!

Programming Tips:

- First switch – exactly as for the hand dryer – the lamp for the light barrier on and wait one second before the process continues.
- Query the phototransistor at every place where it is necessary and open the door when the phototransistor has the value of 0.
- Ready-to-use project: Sliding door 2.mdl

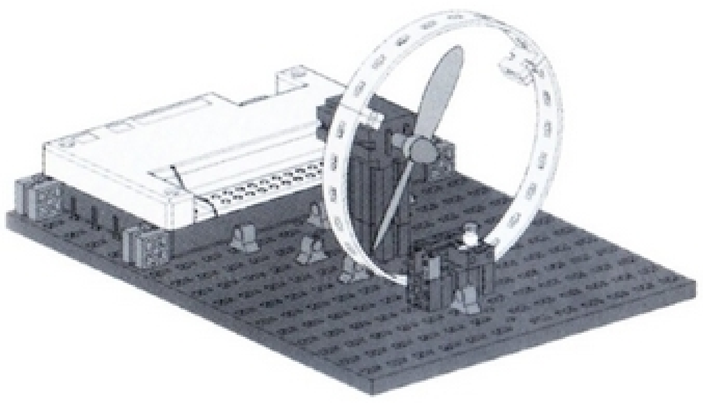
You did it! Your boss is proud of you. The door now operates flawlessly and safely. The boss doubles your hourly wage.

GB+USA

4. Programming Tasks – Part 2

Before you start with the second part of the programming tasks, you should take another look at the LLWin 3.0 handbook. Work through Chapters 5 and 6 carefully. The programming tasks are slowly becoming more demanding. We use variables, more parallel processes and the analog inputs EX and EY of the Intelligent Interface. But if you read these two chapters of the LLWin 3.0 handbook carefully, you will find it easier to handle these tasks later. Additionally, you will know all blocks, which are available in LLWin 3.0.

4.1 Temperature Control (see assembly instructions page 13)



A new air-conditioning system was installed in your home. Of course, you immediately asked the installation engineer how the temperature control operates. He readily explained to you that a temperature sensor constantly measures the room temperature. As soon as a threshold value is exceeded, the air conditioner switches on. If the temperature drops below the threshold value, the air conditioner switches off again and the heating switches on. Now you want to try to program such a control circuit using the "Temperature Control" model. First build the model.

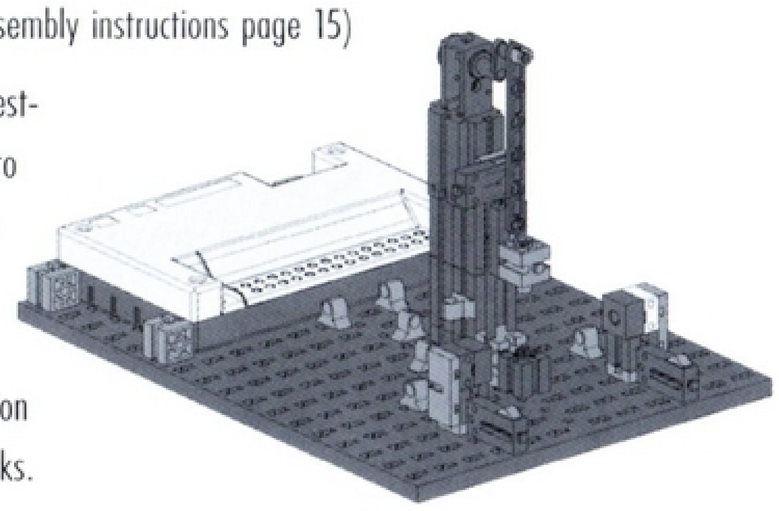
Task:

The heating is triggered by lens tip lamp M2. The "blower" serves as a "cooling unit" at output M1. We use the NTC resistance at input EX for temperature measurement. Program the model, so that the heating switches off and the blower switches on above a specific temperature. This should cool until the lower threshold value is reached. Then the blower should be switched off and the heating on.
You should set the two threshold values on the TERMINAL block using the parameters EA and EB. The value for the respectively current temperature should be shown in Display 1 of the TERMINAL.

Programming Tips:

- Query the analog value EX constantly in a separate process using the ASSIGNMENT block, and store the value in the variable VAR1.
- Note that the resistance value of the NTC resistance decreases with increasing temperature. Consequently, the upper temperature threshold value EA is the smallest value of the variable VAR1. The blower should switch on at this threshold value. The lower temperature threshold value EB is the largest value of the variable VAR1. The heating should switch on at this threshold value.
- Query constantly using the two COMPARE blocks whether VAR1 is below the smallest or above the largest value, and switch the blower or heater on accordingly.
- You can find out which value EX has at room temperature easily using the Check Interface diagnosis tool. Switch the lamp M2 on, and see how the value decreases. Then switch the blower on (so that it blows in the direction of the thermal sensor) and observe how the value increases. You select the threshold values EA and EB correspondingly. The values can fluctuate somewhat with the different NTC resistances.
- Ready-to-use project: Temperature control.mdl

4.2 Stamping Press (see assembly instructions page 15)



The workshop next door has invested in a highly modern machine to stamp metal parts. The machine has already been installed, but unfortunately the programmer, who is supposed to start operation of it, will only arrive in two weeks. Because the workshop urgently needs the machine, the owner asks you whether you could not get the machine running. Because you already have a lot of experience programming, you promise to have it running by tomorrow.

First assemble the stamping press model using the instructions.

Task:

The machine should press a part with four strokes in one work cycle. It may only start after the operator has pressed both pushbuttons E3 and E4 (two-hand operation) and the light barrier is not interrupted at the same time.
If the light barrier is interrupted during a work cycle, the machine stops. A warning signal sounds. The number of strokes can be set on the TERMINAL using the parameter EA. The number of parts to be processed should be shown in Display 1 on the TERMINAL.

Programming Tips:

You really got yourself into something now. But don't worry; it's not that difficult using LLWin 3.0.

- First switch the lamp for the light barrier on. Then run the machine to its starting position (the tappet activates pushbutton E1 at its upper reverse point).
- Then query using the COMPARE block whether E2, E3 and E4 equal "1" (formula: E2 AND E3 AND E4; also refer to the Chapter 6 in the LLWin handbook).
- If this is the case, switch the motor M1 on and let the tappet carry out four strokes, but only if the light barrier is not interrupted!
- Add a variable VAR1 each time when the pushbutton E1 is pressed, and then query whether the number of strokes set at EA is reached via the COMPARE block. Switch off the motor after each stroke, and check whether the light barrier was interrupted. If yes, output a warning signal

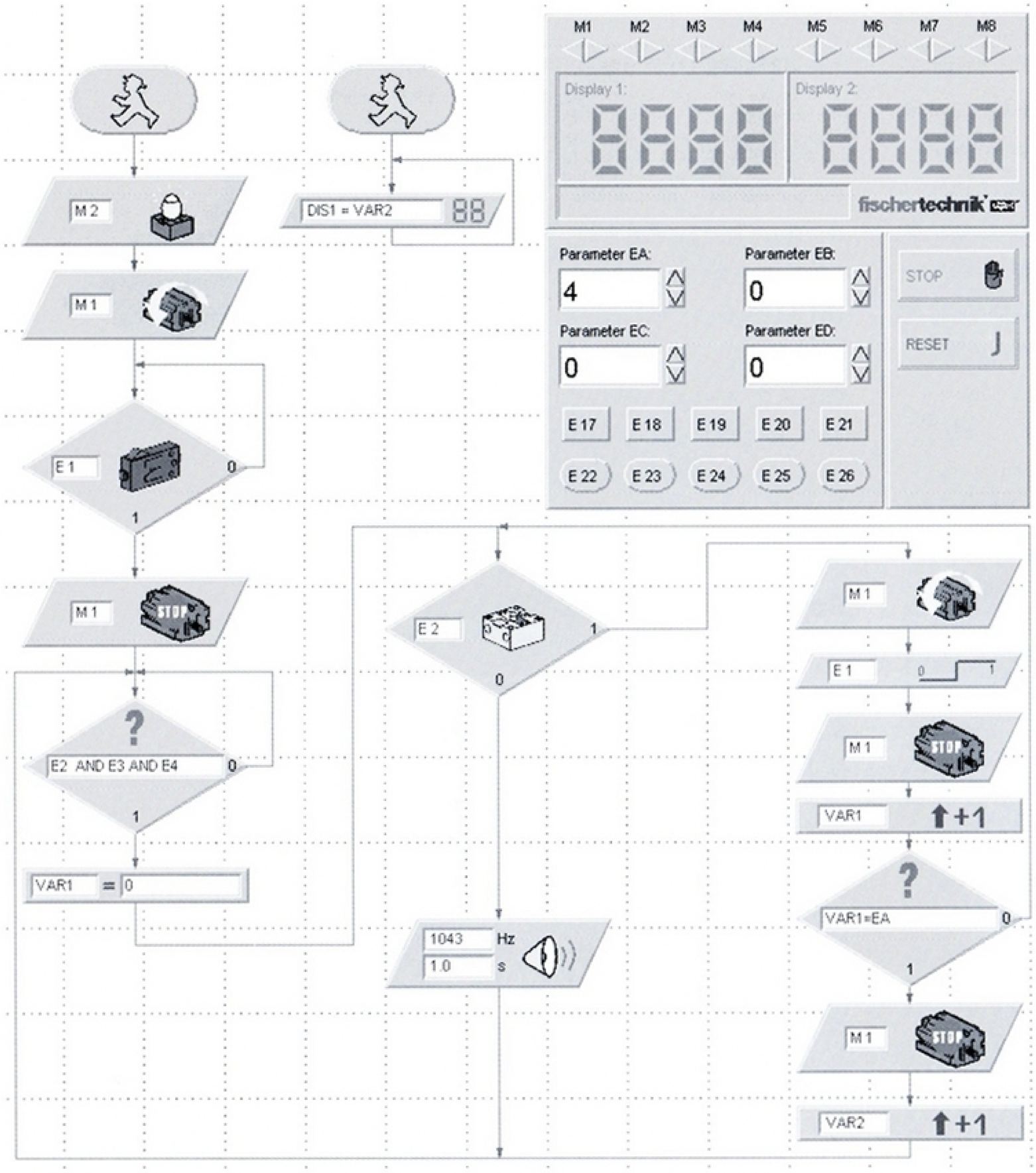
GB+USA

via the BEEP block and stop the work cycle.

- Then the next work cycle follows. After each successfully completed work cycle, add one to the variable VAR2. Show the value of the variables in Display 1. It corresponds to the number of finished parts.
- Ready-to-use project: Stamping press.mdl

Task 1:

The barrier should open when the pushbutton E3 is pressed. If the barrier is open, the light is green. The light turns red and the barrier closes again only after the light barrier has been passed.



Programming Tips:

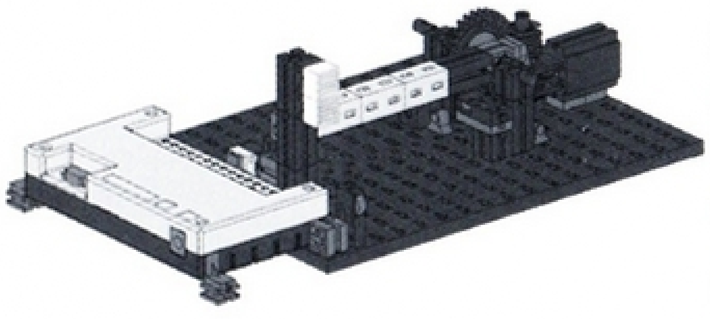
- Write the subprograms "Open" and "Close" for opening and closing the barrier.
- Switch the lamp on for the light barrier first in the flowchart and then switch the light to red.
- Close the barrier. It is then in its initial position.
- After you press pushbutton E3 (EDGE block), the barrier opens. Then the light turns green.
- If the light barrier is interrupted and then closed again (1-0 and 0-1 EDGE at E4), the light should turn red and close the barrier.
- Then the flowchart starts from the beginning again.

Task 2:

The car park should be reserved for prominent guests on opening day. They receive a secret code with three digits for this. They must press three of the five inputs E22-E26 on the TERMINAL block in the correct order. The barrier may only open after that. Program this function in a "Code" subprogram. A message should appear on the Terminal stating whether the code was correct or wrong. The code should be: E26 – E22 – E24.

Everything hopefully worked, because your neighbor will be extremely happy if his machine is running before the programmer shows up. As you know, he needs the machine urgently.

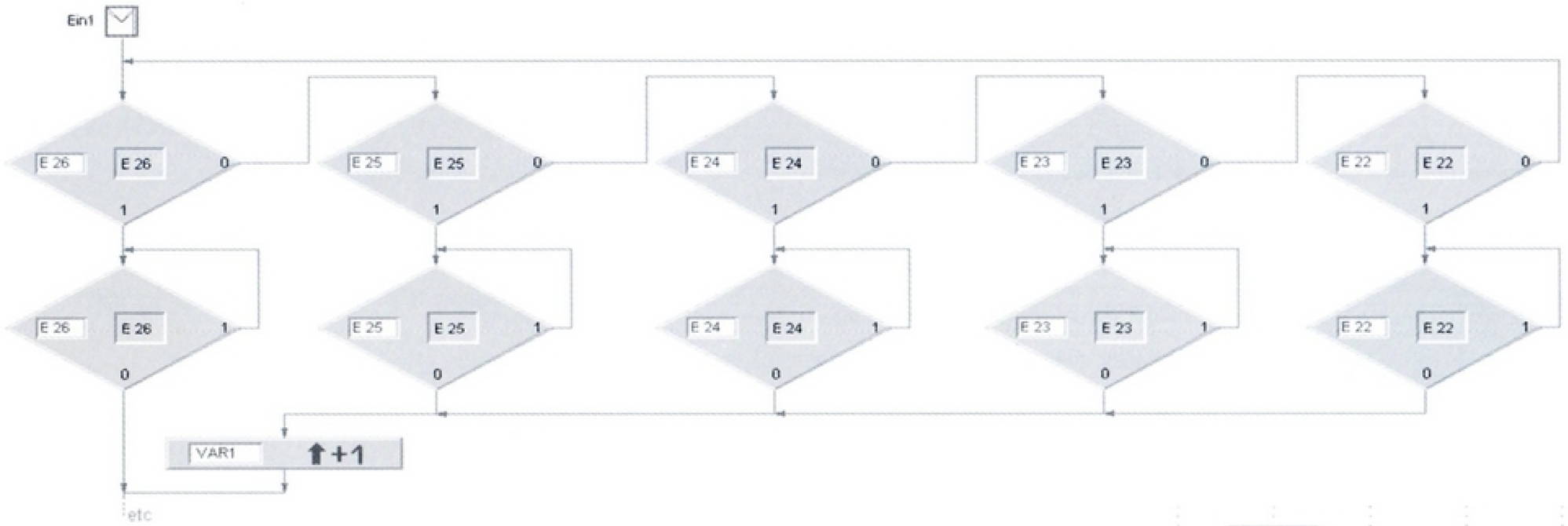
4.3 Car Park Barrier (see assembly instructions page 18)



The new car park should be opened in the city next Saturday. The barrier for the entrance was installed today. Because it is well known in the meantime that you are the best programmer in the city, you were asked right away to handle the programming. Of course, you are proud to have been asked and get to work immediately. Assemble the model.

Programming Tips:

- You can set up the "Code" subprogram in very different ways. The easiest method is to query the three inputs E26, E22 and E24 sequentially using the EDGE block. If the three pushbuttons are pressed, the message "Code correct" appears, and you exit the subprogram. However, the wrong pushbutton can be pressed in between using this solution without the program noticing it.
- If you want to make the code entry secure, so that really only three pushbuttons can be pressed, and if one of them is wrong, the message "Code wrong" appears, then programming is a bit more complex. You use the INPUT block and query all five inputs E26-E22 as follows:



GB+USA

- E26 should be pressed first and let go again. If the correct pushbutton is pressed, the flowchart continues downward. If an incorrect pushbutton is pressed (E25-E22), a variable VAR1 is added and the flowchart is only continued then.

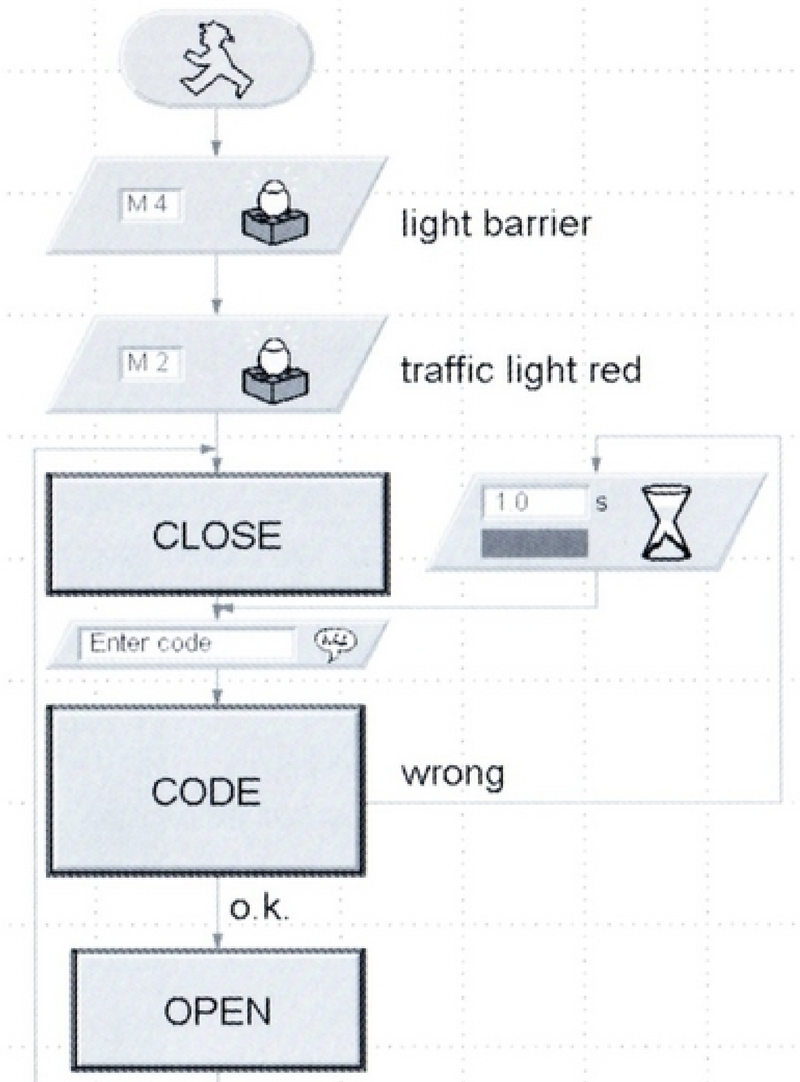
- You need this sequence a total of three times below each other. The correct pushbutton is always on the left, and the wrong one on the right. Every time an incorrect pushbutton is pressed, the variable VAR1 is increased by one.

- At the end, you query with the COMPARE block whether VAR1=0. The correct code was entered only in such a case.

- You can also output a message and generate different sounds. Then you exit the subprogram via two different SUBOUT blocks:

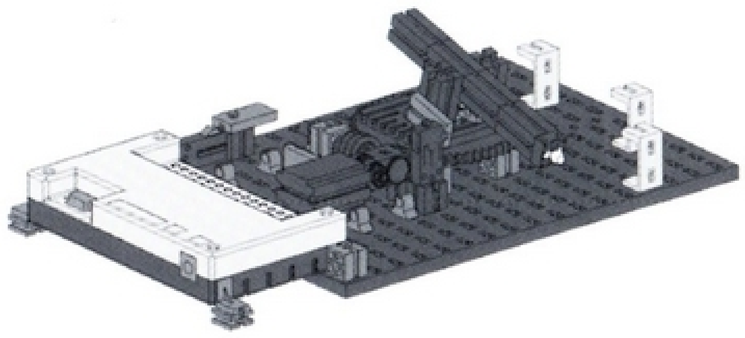
If the code was correct, you open the barrier. If it was wrong, a new code must be entered. ▶

This programming is rather demanding, but try it anyway. You will see that it functions. And if not? Don't worry; we have also stored the ready-to-run project here under: Car park barrier 2.mdl.



GB+USA

4.4 Welding Robot (see assembly instructions page 21)



The workshop already mentioned has now also purchased a welding robot. Because the owner was so happy about the way you programmed his stamping press recently, he also asked you to get his welding robot running. First assemble the model according to the instructions.

Task 1:

The robot should attach the lid on a metal housing at three different positions with a welding point. The welding electrodes are simulated by a lens tip lamp, and the three positions on metal housing by yellow blocks.

The robot should move to three different positions one after another and weld at each one. Then it should return to its starting position and start again from the beginning.

Because the robot also has to operate independent of the PC in download mode, it should be possible to trigger a reset via the E8 pushbutton, i.e., restart the program from the beginning after pressing and releasing the pushbutton.

Programming Tips:

- First move the robot to its starting position (press M1 left until E1).
- Move then to position 1. Use the POSITION block for this, and count the impulses at Input E2. You must experiment to see how many counter impulses are required until the position is reached. The best is to use the parameter EA as the result in the POSITION block. Then you can continually change the value on the TERMINAL.
- Write a subprogram "Weld" for the welding procedure. Switch on the lens tip lamp (M2) in this subprogram and then switch it off again immediately. Repeat this procedure a few times by programming a loop and counting a variable VAR1 until the value is 20, for example. Then exit the subprogram. Don't forget to reset the variable to zero first (ASSIGNMENT block).
- Do the same with positions 2 and 3. Use the parameters EB and EC as results in the Position blocks.
- Of course, you use the RESET block for the reset, which you place on the work sheet without a connection to the other blocks (formula entered in the dialog box: E8).
- Test the program in online mode until you have set all positions on the TERMINAL correctly. Then load it onto the interface (`Run – Download`), and run it in download mode. It functions there exactly as in online mode. However, you cannot change the terminal parameters anymore, because the interface no longer has a connection to the computer (i.e., you can change them on the screen, but the robot will not react to the changes anymore).
- Don't forget that to return to online mode after a download, you must interrupt the power supply to the Intelligent Interface for approx. three seconds.
- You can find the ready-to-run project here under Welding robot 1.mdl.

The program for the welding robot is not especially large. But imagine that the robot arm would move up and down as well as forward and back in addition to in a circle. If the robot would move to every position with a separate POSITION block as in the example project, and if the robot moved not only to three, but to more positions, the program would become very large. When programs become too large, they no longer function in download mode at some point. Additionally, it is not very elegant to call the "Weld" subprogram three times. Consequently, we are going to design the program completely differently in the way that real pros do.

Task 2:

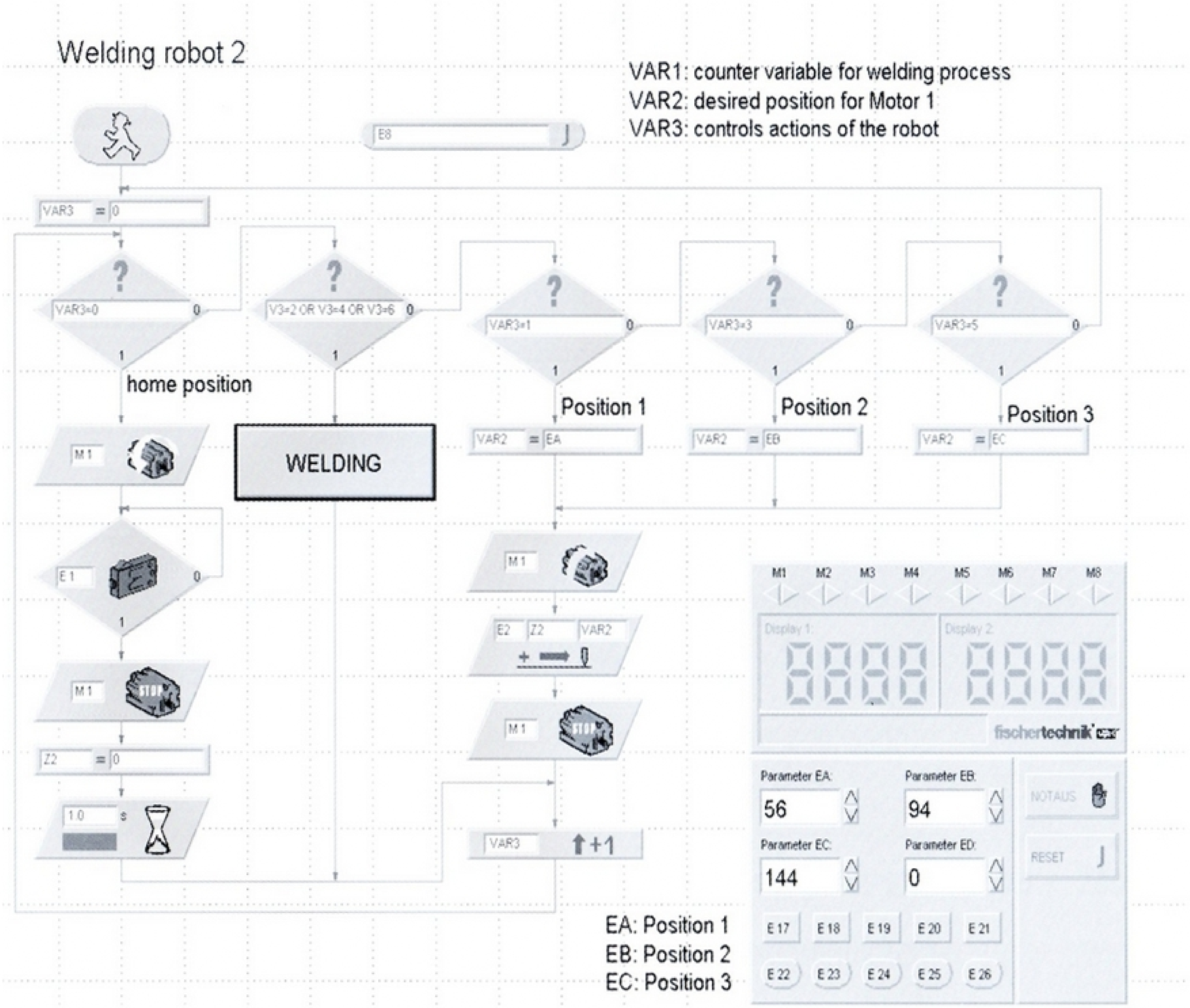
Design the program, so that the fewest number of blocks are required and the "Weld" subprogram only appears once in the flowchart.

Programming Tips:

- Before you get started, it can be helpful to read through Chapter 10.4 "Saving Function Blocks – More Efficient Programming in LLWin". Then you will have a good idea of the solution. Our program will even become simpler.
- You control the actions of the robot using the variable VAR3. The variable is raised by one after each action. The robot should do the following in order:

Value of VAR3	Action
0	Move to starting position
1	Move to position 1
2	Weld
3	Move to position 2
4	Weld
5	Move to position 3
6	Weld

- Query using the COMPARE block which value the variable VAR3 has. The robot moves to the home position at 0, to position 1 at 1, and welds at 2, 4 and 6, etc. The flowchart then looks like this:



- Each action only occurs once in the program. It is carried out when the variable VAR3 has the value that belongs to this action.
- The variable VAR2 is assigned to the respective final position of the motor M1 at the positions 1, 2 and 3. As a result, the process MOTOR CW - POSITION - MOTOR OFF only occurs once.
- The ready-to-run project is called Welding robot 2.mdl

Admittedly, this state-oriented programming appears more complicated at first than simply placing the blocks one after another as in the Welding robot 1.mdl project. Maybe it is a little exaggerated to program this simple robot like this. On the other hand, the more the robot can do and the more positions it can move to, the more meaningful it becomes. Consequently, it is helpful in any case to use it for a simple model. Then you will not have any problems later with large industry robots in which three motors move simultaneously to one position.

5. What Next?

You can certainly think up additional tasks for the models of the Computer Starter construction kit with using a little imagination and write programs for them. For example, the welding robot can weld at a fourth position, or it can move to the existing positions in a different order. If you have other construction parts, you can expand the traffic light at an intersection to design comprehensive traffic light control, for example. You can also program a melody using the Beep block, which sounds when the sliding door is opened. Simply use your imagination, because lots of possibilities exist. If the inputs and outputs of the Intelligent Interface no longer suffice, simply get a hold of Extension Module part no. 16554.

This Extension Module is connected directly to the interface. Then you have four additional outputs and eight digital inputs at your disposal. All function blocks, menu commands, the toolbar and keyboard combinations are described in Chapters 7 to 9 of the LLWin 3.0 handbook. These chapters are very useful as a reference. Chapter 10 contains valuable tips and tricks for programming with LLWin 3.0. Reading it would certainly be worthwhile.

fischertechnik also offers other Computing construction kits. Mobile Robots are movable robots, which you can program, so that they avoid obstacles or do not fall from a table. Using the Pneumatic Robots construction kit, you can control pneumatically driven machines using electromagnetic valves via the interface. You can build robots that grip with three axes using the Industry Robots construction kit, which can grip, move and stack objects in a three-dimensional room. This means building and programming fun without end. And the best is that something new is always added.

GB+USA

Computing Starter

- Begleitheft
- Activity booklet
- Manuel d'accompagnement
- Cuaderno adjunto
- Folheto

fischerwerke
Artur Fischer
GmbH & Co. KG
Weinhalde 14-18,
D-72178 Waldachtal
Telefon 0 74 43/12-43 69
Telefax 0 74 43/12-45 91
http://www.fischertechnik.de
email: info@fischertechnik.de

Printed in Germany

fischertechnik®