

# FLOWCODE6<sup>®</sup>

Simplifying Technology

Computer Science and Control 1



SE7761

**MATRIX**  
[www.matrixmultimedia.com](http://www.matrixmultimedia.com)

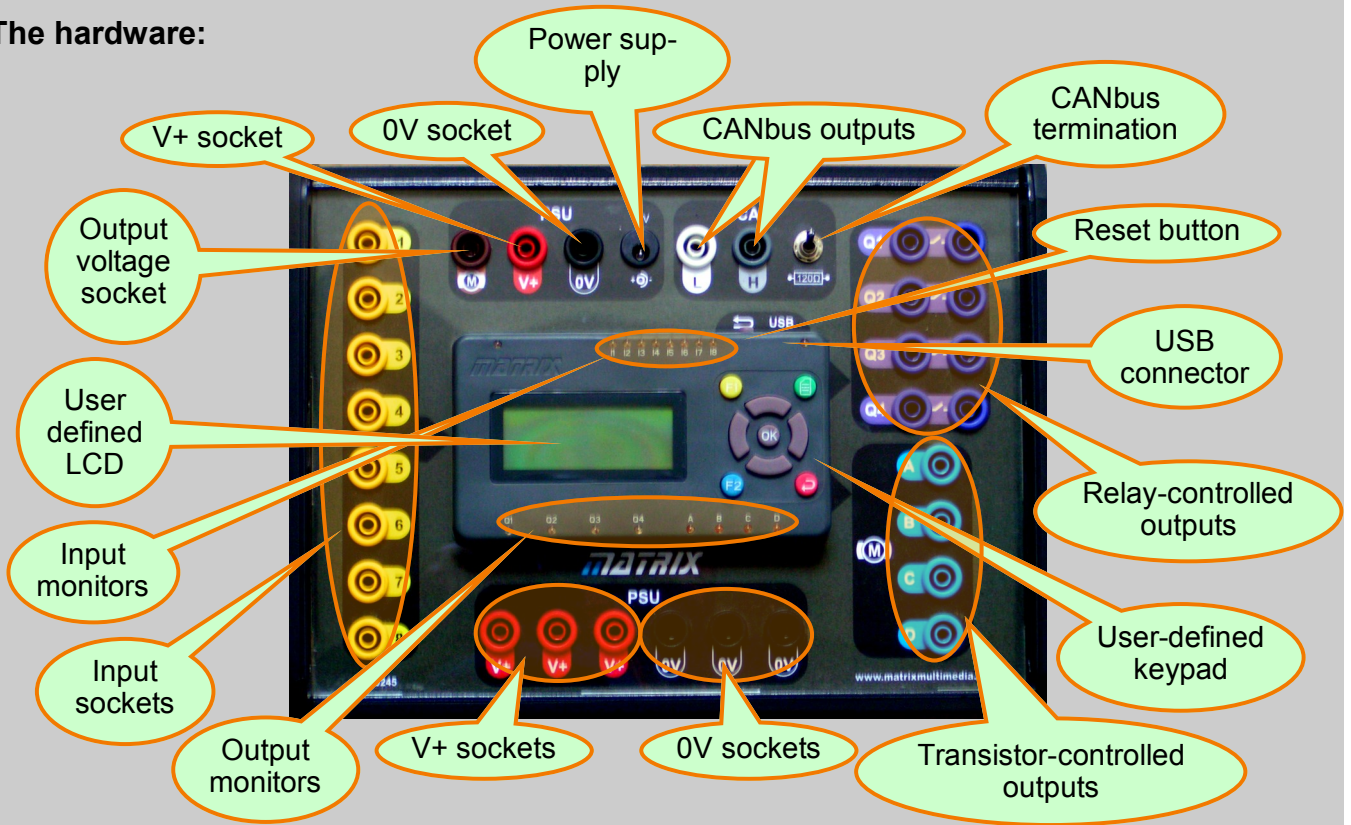
Copyright © 2013 Matrix Multimedia Limited

# Contents

# Computer Science and Control—1

Introduction	3
Worksheet 1 - Switch on the LED	5
Worksheet 2 - Make the LED flash	7
Worksheet 3 - Keep the LED lit for a short time	9
Worksheet 4 - A latch	11
Worksheet 5 - A lighting sequence	13
Worksheet 6 - The AND gate	15
Worksheet 7 - A logic system	17
Worksheet 8 - Anyone home?	19
Teacher Guide	22

The hardware:



The hardware:

- MIAC requires a supply voltage of 12V, delivered either via a 'V+' and '0V' socket, or via the power supply connector. It must be powered both when downloading a program and when running it.
- The USB connector is used to connect MIAC to a computer to allow
  - a program to be downloaded to the microcontroller inside the MIAC;
  - the MIAC to be used as a slave device to the PC.
- Once programming is completed, the Reset button is pressed to start the program.
- When MIAC is used as a standalone microcontroller, (i.e. not as a PC slave,) the USB cable can be disconnected.
- The input sockets can be used for analogue or digital signals.
- The state of these inputs is shown on the eight LEDs in the Input monitor.
- The four transistor-controlled output sockets deliver an output voltage determined by the power supply connected to the 'M' socket.
- The four sets of relay-controlled outputs act as switches to control devices in a separate circuit. They can deliver higher currents than the transistor outputs, but switch more slowly.
- The state of the outputs is shown on the eight LEDs in the Output monitor.
- The CANbus output sockets, and termination switch are used in automotive applications.

**Flowcode and MIAC:**

- When creating Flowcode 6 programs for MIAC:
  - open 'Flowcode 6';
  - select 'New Project' from the 'Startup Selection' screen
  - click on the 'PIC' tab when the 'Project Options' dialogue box opens;
  - click on the down arrow following the 'Family' label, and select 'Misc';
  - select 'MIAC' as the target and click on 'OK'.

- MIAC has eight identical inputs, labelled 1 to 8. Each is connected to different pins on the 18F4455 PIC microcontroller. These connections are given in the table opposite.

MIAC Input	Microcontroller pin
1	Port A bit 0
2	Port A bit 1
3	Port A bit 2
4	Port A bit 4
5	Port E bit 0
6	Port E bit 1
7	Port E bit 2
8	Port B bit 2

- Its four identical relay outputs, labelled Q1 to Q4, are connected to the microcontroller pins given in the table.

MIAC Output	Microcontroller pin
Q1	Port B bit 4
Q2	Port B bit 5
Q3	Port B bit 6
Q4	Port B bit 7

- The four identical transistor-controlled outputs, A to D, have connections given in the table.

MIAC Output	Microcontroller pin
A	Port C bit 2
B	Port C bit 0
C	Port C bit 1
D	Port C bit 6

**Using the worksheets:**

- The worksheets focus on situations based on machine-control systems. However, the program examples have a much broader applicability.
- The Teacher Guide at the end of the module details all the steps needed to build the hardware, create the programs and test the systems. Please refer to this when necessary.

# FLOWCODE 6

## Worksheet 1

### Switch on the LED

# Computer Science and Control—1

In today's world, we use microcontrollers more and more. These marvels of modern electronics are all-pervading, yet their use is so subtle that we often use them without realising it. Billions are sold every year!

We use them in devices such as:

- the security token device used in on-line banking ;
- mobile phones, to set up and oversee our calls;
- microwave ovens to control the display and operation;
- engine management systems in our cars;
- digital cameras to control the focussing and shutter control;
- television remote controls.

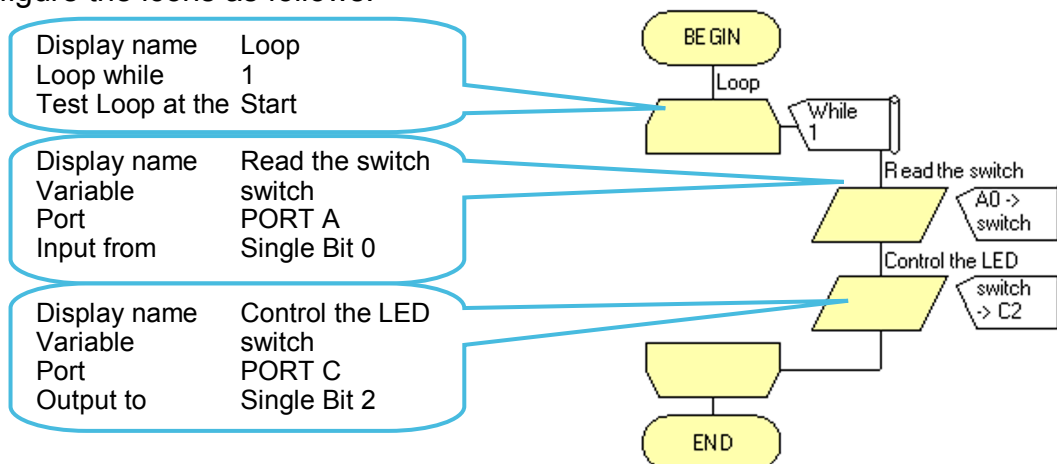
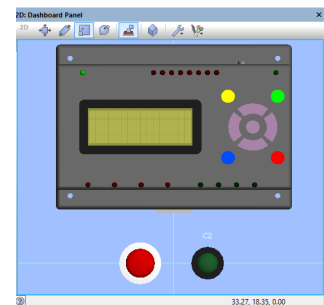


This worksheet introduces a very simple application of a microcontroller - to turn on a 'system active' LED when a switch is operated.

### Over to you:

#### The program:

- Build the Flowcode flowchart, adding a MIAC, a 'Switch (Push, Panel)' and a 'LED (5mm Panel)' to the dashboard panel. Arrange them as shown opposite.
- Connect the switch to PORT A bit 0, (MIAC Input 1) and the LED to Port C bit 2 (MIAC Transistor output A).
- Open 'Project Explorer' using the 'View' menu.
- Click on the down arrow to the left of the 'Variables' label, and select the 'Add new' option.
- Name the new variable called 'switch', of type 'Bool', and click on 'OK'.
- Configure the icons as follows:



- Save the Flowcode program.
- Simulate it to check that it works. Click on the push switch, and check that the LED lights.
- Now, power up the MIAC and compile the program to the chip built into MIAC.
- Build the circuit shown on the next page.

# FLOWCODE 6

## Worksheet 1

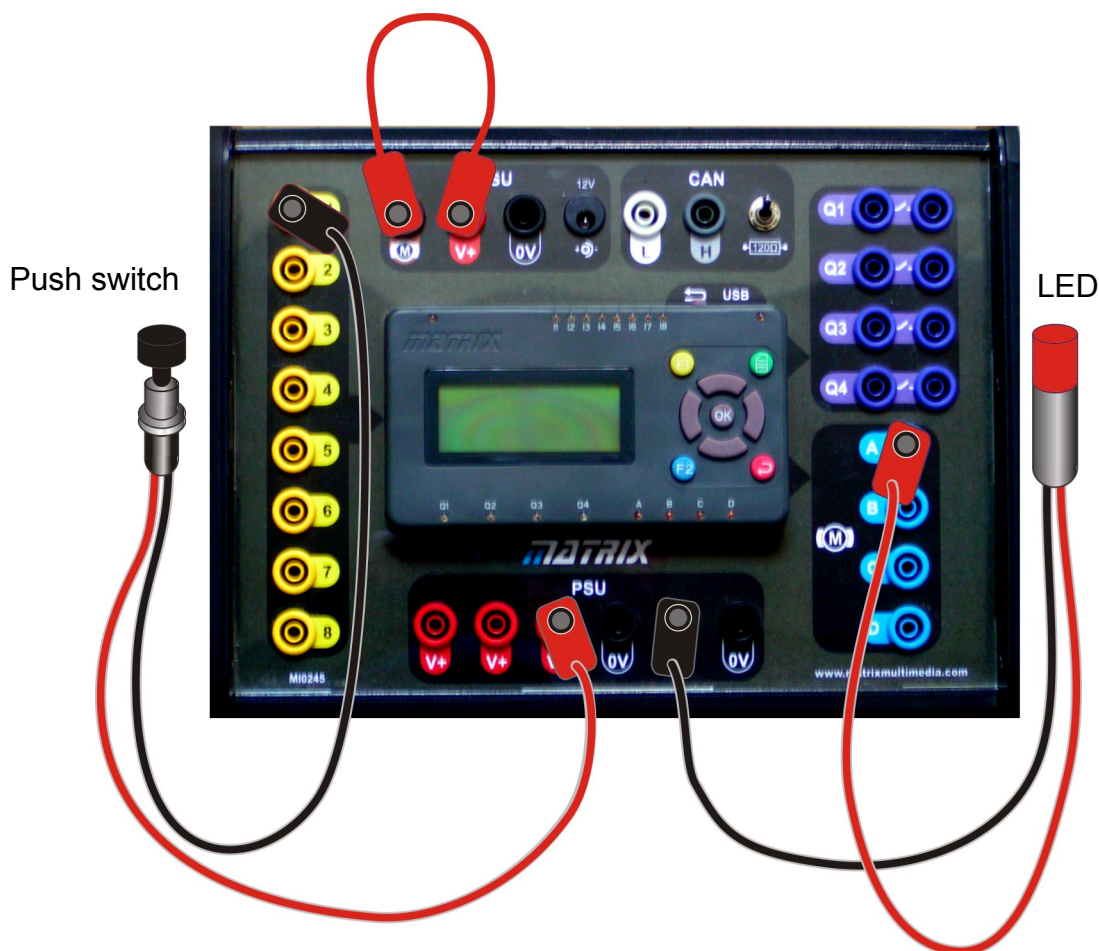
### Switch on the LED

# Computer Science and Control—1

#### Over to you:

#### The circuit:

- Build the circuit, shown below, on MIAC.
  - The LED is a polarised component, meaning that it must be connected the right way round (red wire to MIAC output, black wire to 0V).
  - The switch is not polarised, and so it does not matter which wire, red or black, goes to input 1.
- Make sure the MIAC is still powered from the 12V power supply, and disconnect the USB lead.
- Test the circuit by pressing the switch, and then releasing it. You should find that it switches the LED on and off.



#### What to do next:

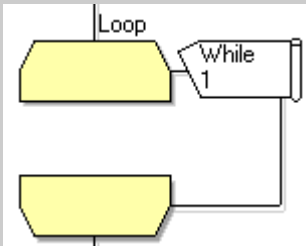
Modify the program so that the switch lights two LEDs, one connected to Port B bit 0 and the other to Port B bit 1.

# FLOWCODE 6

## Worksheet 2

Make the LED flash

# Computer Science and Control—1



The 'loop' function makes the program cycle through repeatedly.

It allows us to observe what happens more easily, since each step in the program lasts

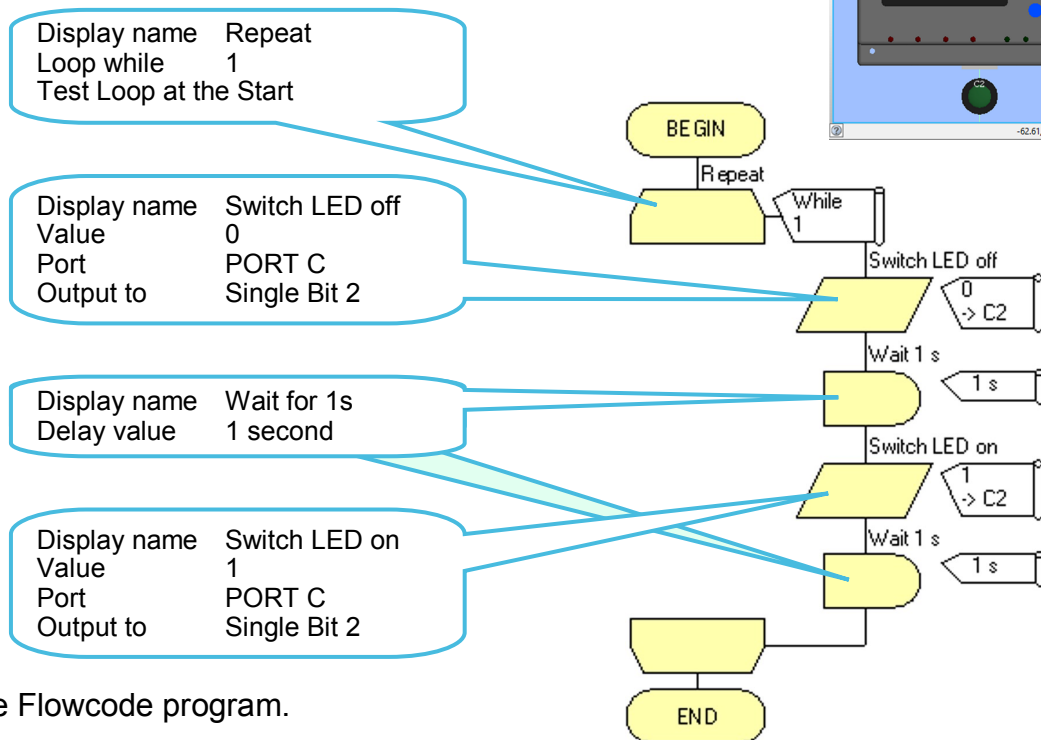
only microseconds.



### Over to you:

#### The program:

- Build the Flowcode flowchart, adding a 'LED (5mm Panel)' to the dashboard panel.
- Arrange them as shown opposite.
- Connect the LED to PORT C bit 2, (MIAC Transistor output A).
- Configure the icons as follows:



- Save the Flowcode program.
- Simulate it to check that it works - the LED should flash on and off every second.
- Now, power up the MIAC and compile the program to the chip built into MIAC.
- Build the circuit shown on the next page.

# FLOWCODE 6

## Worksheet 2

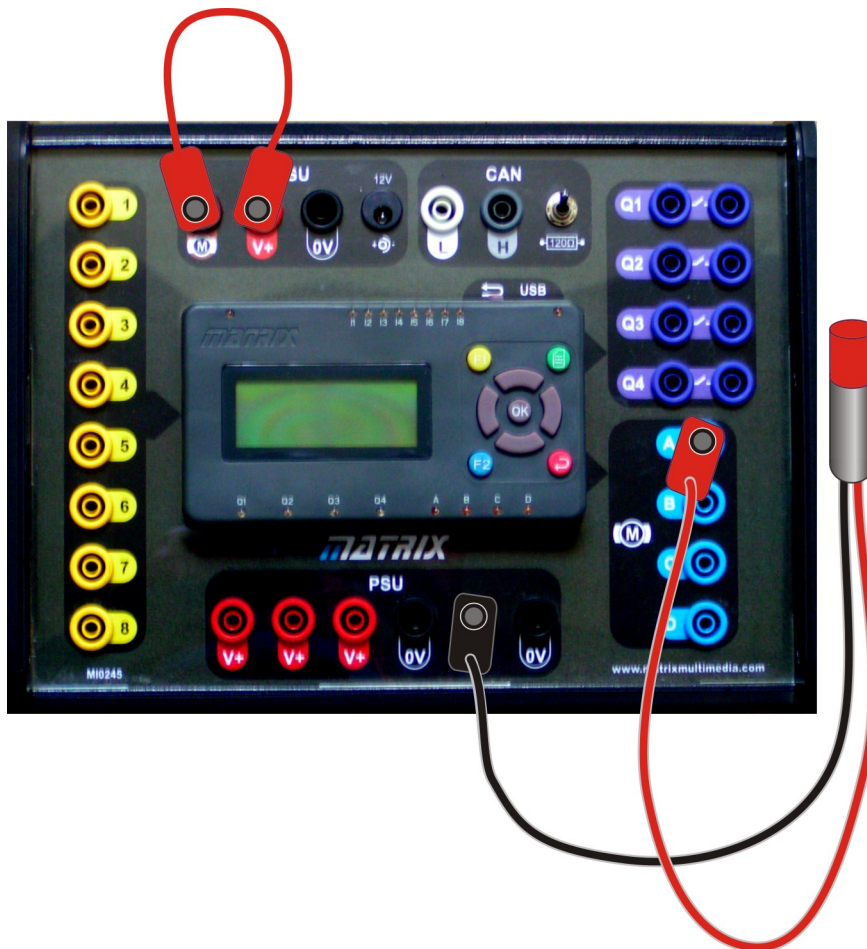
Make the LED flash

# Computer Science and Control—1

### Over to you:

#### The circuit:

- Build the circuit, shown below, on MIAC.
  - Remember that the LED is a polarised component, and must be connected the right way round.
- Make sure the MIAC is still powered from the 12V power supply, and disconnect the USB lead.
- Test the circuit.
  - The LED should flash on and off continuously
    - on for a second, then off for a second.



### What to do next:

Modify the program so that the LED flashes twice as fast.  
Modify it again so that the LED flashes only ten times.

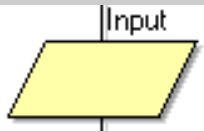


# FLOWCODE 6

## Worksheet 3

### Keep the LED lit for a short time

# Computer Science and Control—1



The microcontroller can respond to changes in its surroundings. To do this, it takes in information from sensors. In this case, it monitors the state of a switch, connected to Port A, bit 0, via an 'Input' function.

The information from the switch ('on' or 'off') is stored in a variable called 'switch'. (The Teacher Guide at the back of the module gives more information about using variables.)

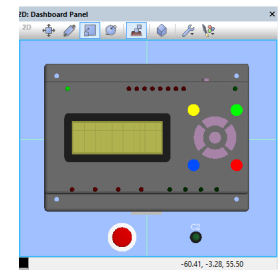
This worksheet shows how to set up another common application - a lamp that stays on for a short time when



### Over to you:

### The program:

- Build the flowchart.
- Add a 'Switch (Push, Panel)' and a 'LED (5mm Panel)' to the dashboard panel, and arrange them as shown opposite.
- Connect the switch to PORT A bit 0, and the LED to Port C bit 2.
- Use 'Project Explorer' to create a new 'Bool' variable called 'switch'.
- Configure the Loop icon as previously.
- Set up the other icons as follows:



Display name	Read the switch
Variable	switch
Port	PORT A
Input from	Single Bit 0

Display name	Is it pressed?
If	switch=1

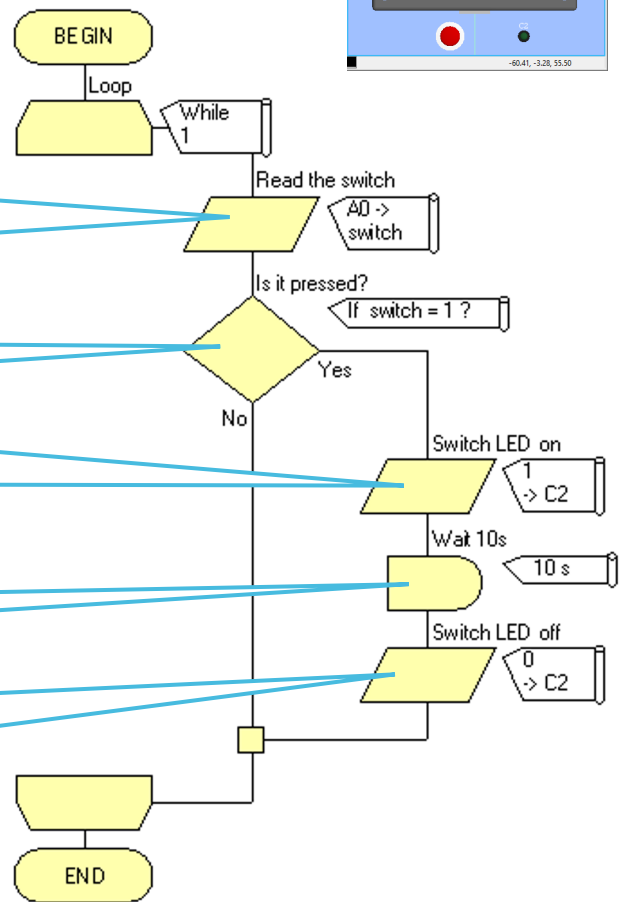
Display name	Switch LED on
Value	1
Port	PORT C
Output to	Single Bit 2

Display name	Wait 10 s
Delay value	10 seconds

Display name	Switch LED off
Value	0
Port	PORT C
Output to	Single Bit 2



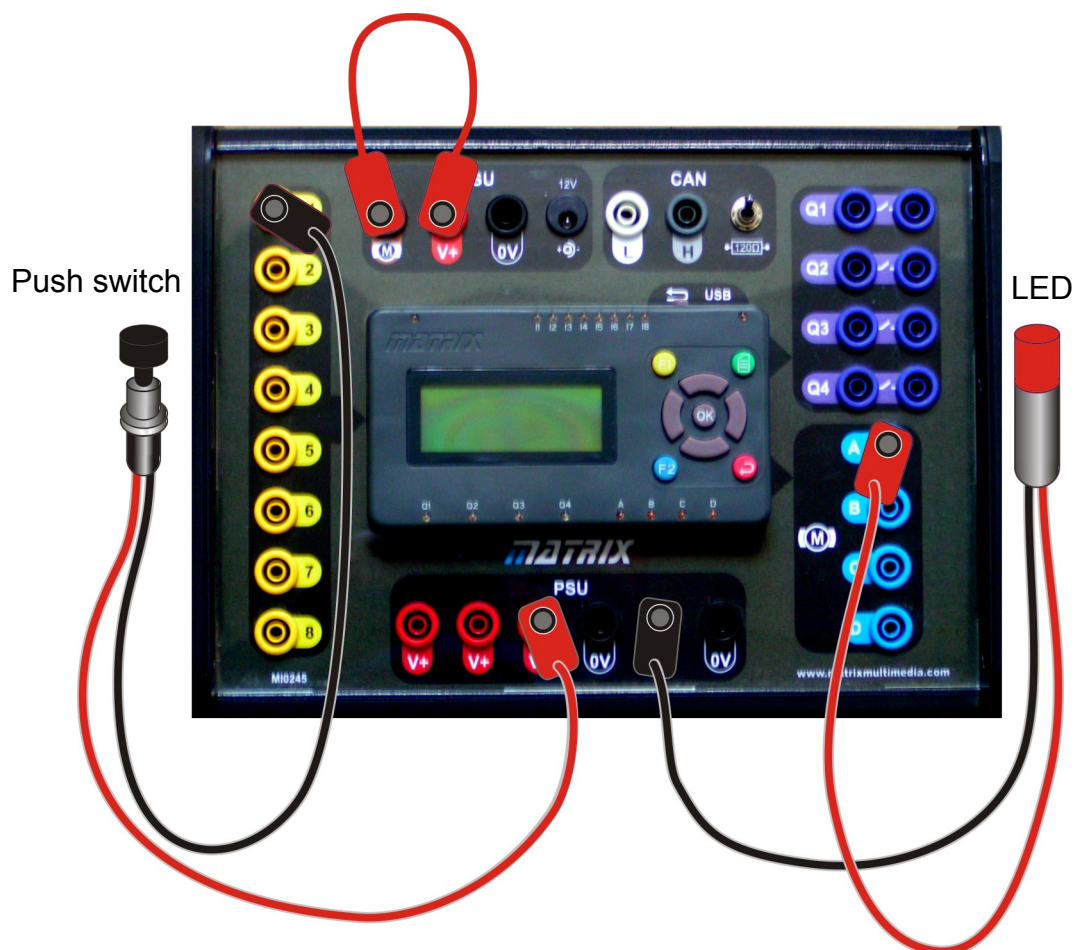
- Save the Flowcode program.
- Simulate it to check that it works.  
When you press the switch, the LED comes on for ten seconds and then goes off.
- Now, compile it to the chip, as before, build the circuit shown on the next page, and test it.

**FLOWCODE 6****Worksheet 3**

Keep the LED lit for a short time

**Computer Science  
and Control—1****Over to you:****The circuit:**

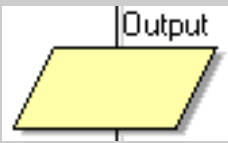
- Build the circuit, shown below, on MIAC.
  - It is identical to the one you built for worksheet 1.
- Test the circuit by pressing and releasing the switch.
  - The LED should light for ten seconds and then turn off.

**What to do next:**

Modify the program so that the LED stays on for 5 seconds when the switch is pressed.

**FLOWCODE 6****Worksheet 4**

## A latch

**Computer Science  
and Control—1**

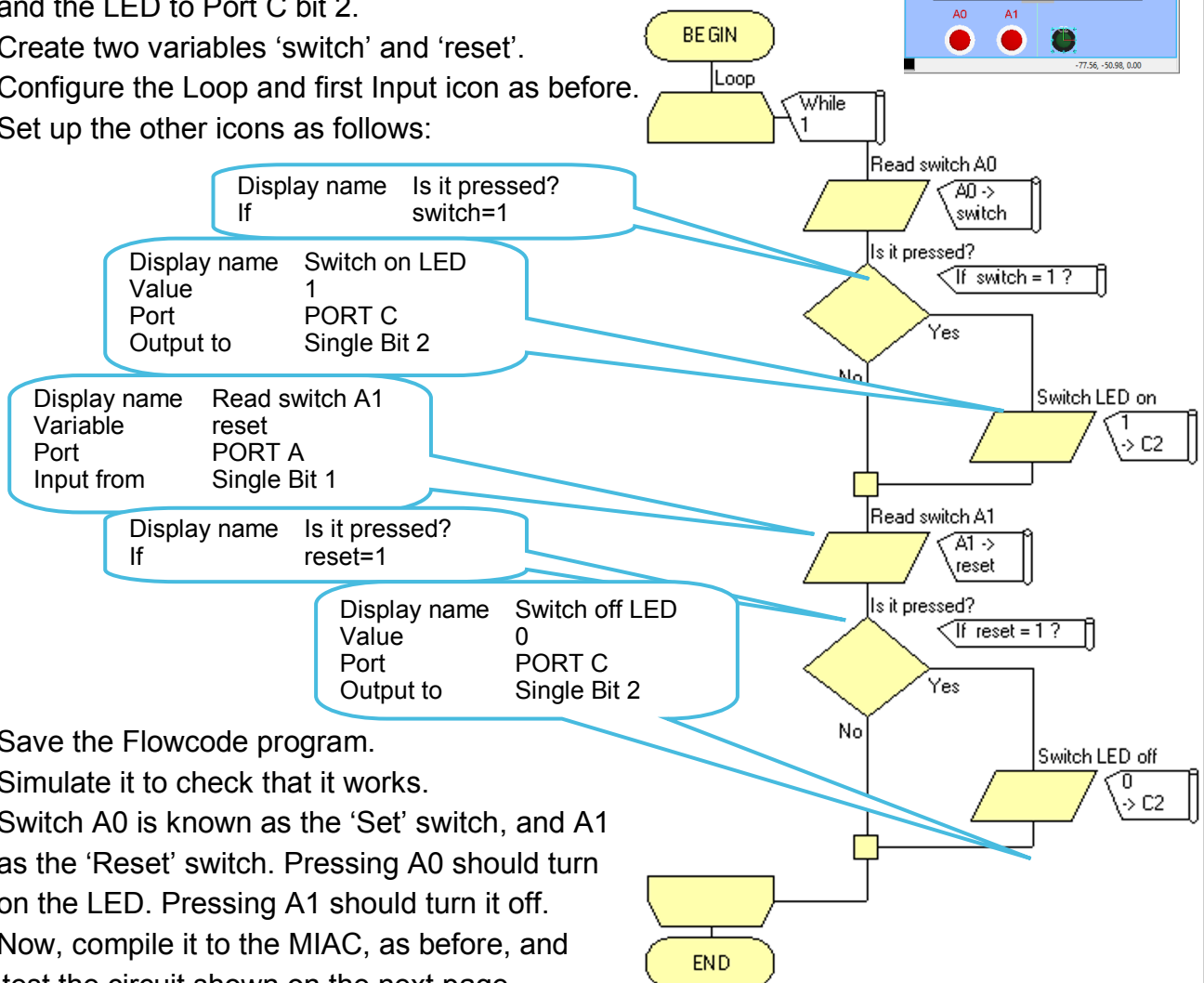
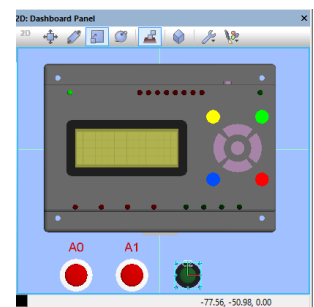
In this application, the microcontroller switches on a machine and keeps it switched on, until a second switch is pressed.

The program uses an 'Output' function to provide power to one of MIAC's output sockets.

This worksheet shows how to set up a latch.

**Over to you:****The program:**

- Build the flowchart.
- Add two switches ('Switch (Push, Panel)') and a 'LED (5mm Panel)' to the dashboard panel. Arrange them, and add labels as shown.
- Connect one switch to PORT A bit 0, the other to PORT A bit 1, and the LED to Port C bit 2.
- Create two variables 'switch' and 'reset'.
- Configure the Loop and first Input icon as before.
- Set up the other icons as follows:



- Save the Flowcode program.
- Simulate it to check that it works.  
Switch A0 is known as the 'Set' switch, and A1 as the 'Reset' switch. Pressing A0 should turn on the LED. Pressing A1 should turn it off.
- Now, compile it to the MIAC, as before, and test the circuit shown on the next page.

# FLOWCODE 6

## Worksheet 4

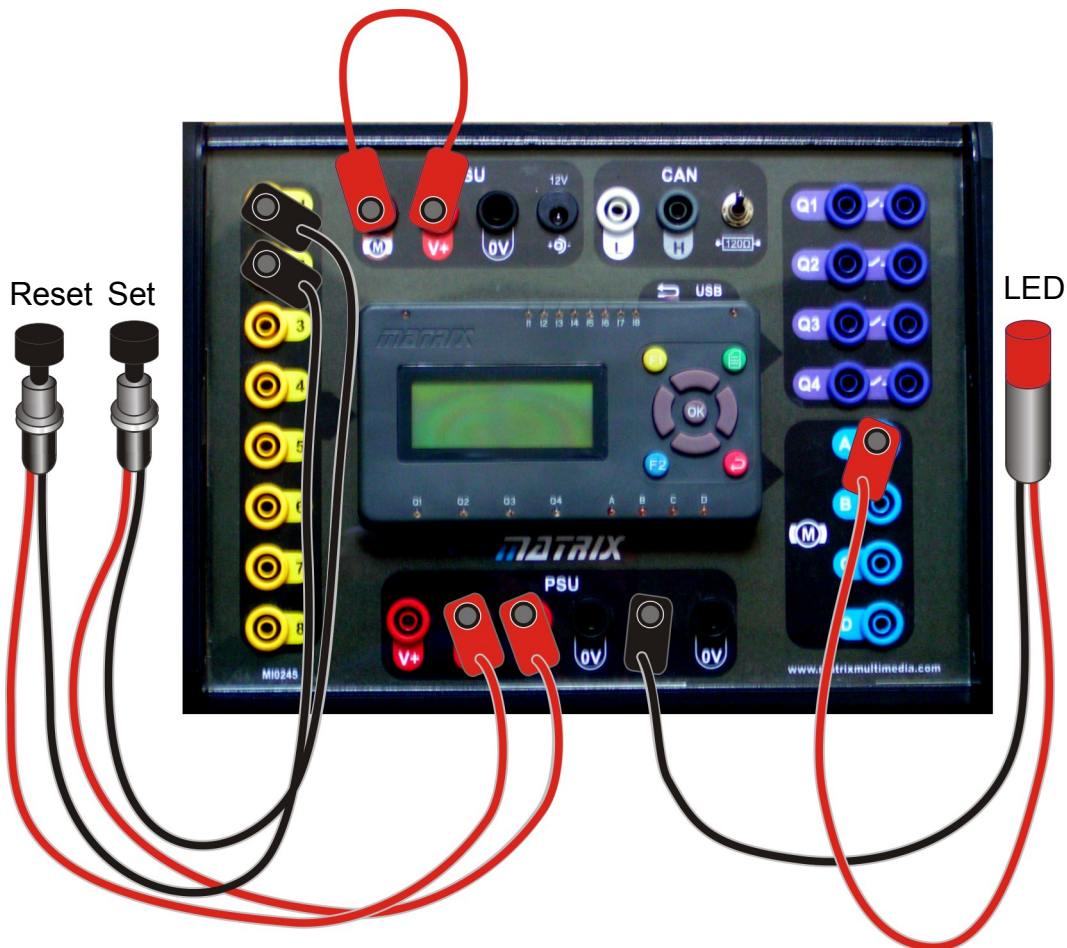
### A latch

# Computer Science and Control—1

### Over to you:

#### The circuit:

- Build the circuit, shown below, on MIAC.
  - The LED represents the machine under control - ensure that the LED is plugged in the right way round!
  - Make sure that you can distinguish between the two switches.
- Test it by pressing switch A0 (the 'Set' switch,) and then switch A1(the 'Reset' switch).
  - Does it matter how many times you press switch A0?
  - Does it matter how many times you press switch A1?



### What to do next:

Modify the program so that the LED is turned on *until* switch A0 is pressed, and then lights again when switch A1 is pressed.

# FLOWCODE 6

## Worksheet 5

### A lighting sequence

# Computer Science and Control—1

Control systems often create sequences of operation, whether controlling motors, valves etc.



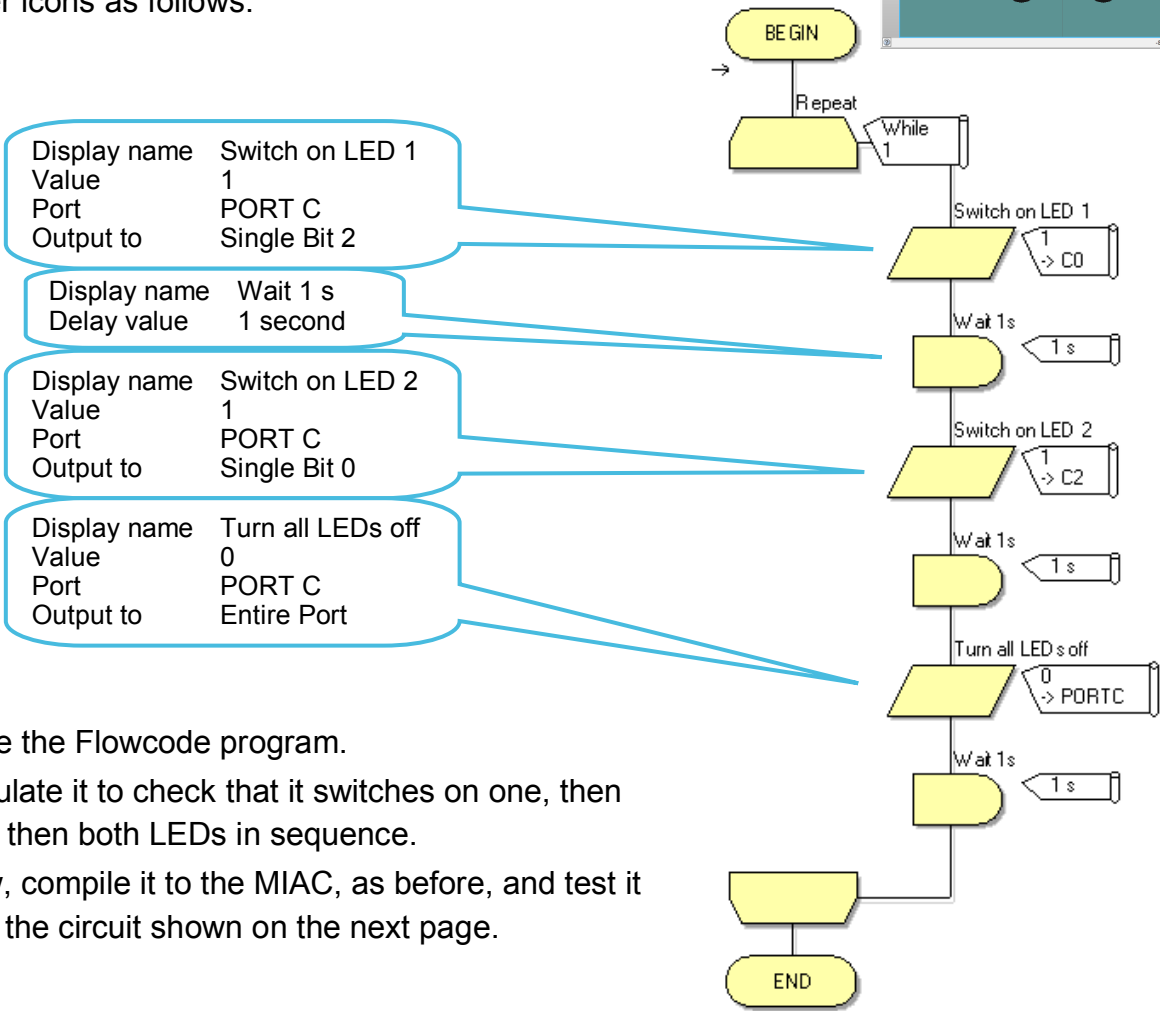
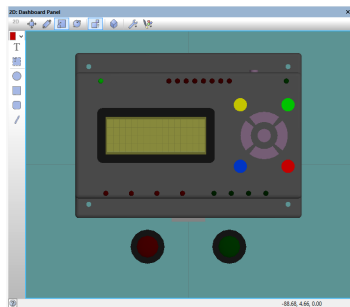
In this worksheet, the microcontroller creates a simple light sequence. The program uses delay functions to allow each stage to remain long enough to be seen.

The picture shows rail crossing warning lights in North America, though these are more sophisticated than the simple LEDs used here.

### Over to you:

### The program:

- Build the flowchart, adding two 'LED (5mm Panel)'s to the dashboard panel.
- Connect one LED to PORT C bit 2, (MIAC Transistor output A), and the other to PORT C bit 0, (Transistor output B).
- Configure the Loop icon as in previous programs, and set up the other icons as follows:



- Save the Flowcode program.
- Simulate it to check that it switches on one, then two, then both LEDs in sequence.
- Now, compile it to the MIAC, as before, and test it with the circuit shown on the next page.

# FLOWCODE 6

## Worksheet 5

A lighting sequence

# Computer Science and Control—1

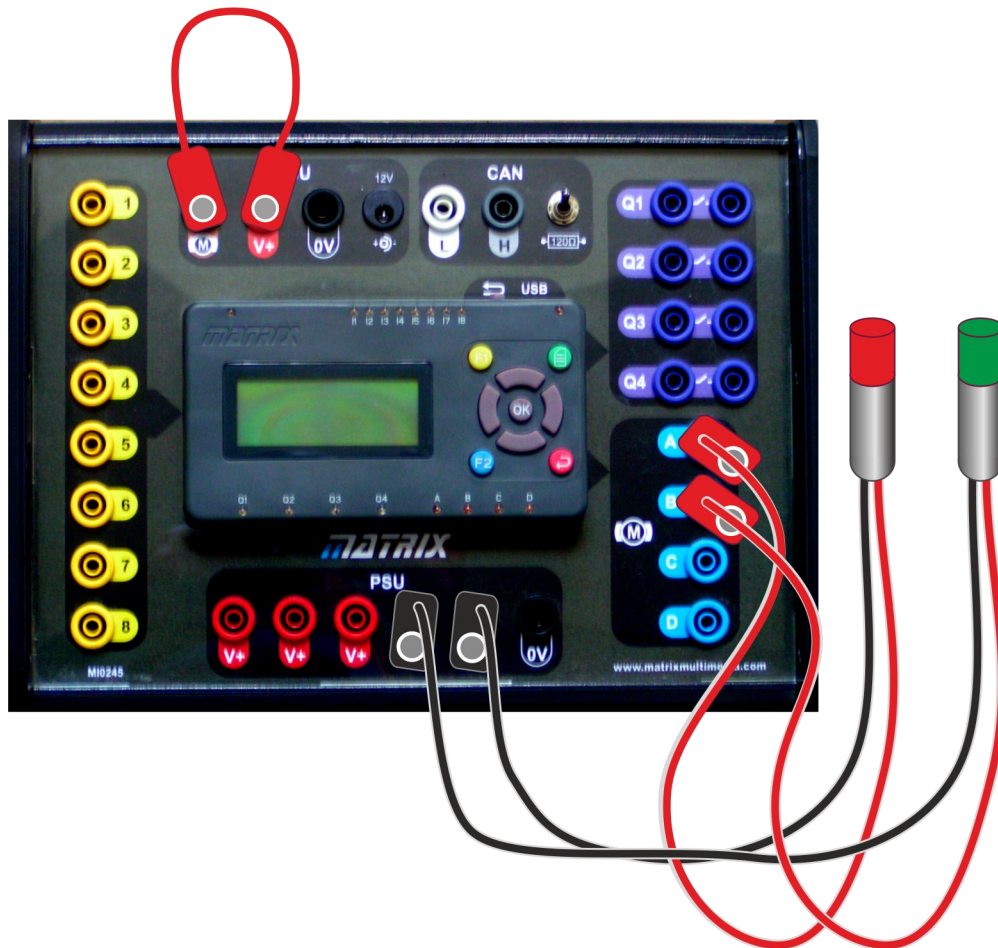
**Over to you:**

**The circuit:**

- Build the circuit, shown below, on MIAC.

The LEDs are powered from separate MIAC transistor outputs, A and B.

- Notice the black wires that make up the 0V connection to the LEDs!
  - Make sure that the LEDs are plugged in the right way round!
- Test the circuit by pressing the Reset button and watching the sequence that follows.



**What to do next:**

Modify the program so that a switch must be pressed to make the sequence start.

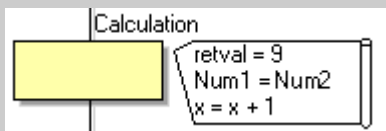
# FLOWCODE 6

## Worksheet 6

### The AND gate

# Computer Science and Control—1

Microcontrollers can take in information from sensors and make decisions based on it.



Part of the process can involve using calculation functions, which perform arithmetic and, as in this case, logic operations.

Several separate calculations can be included in the same icon, as the illustration shows.

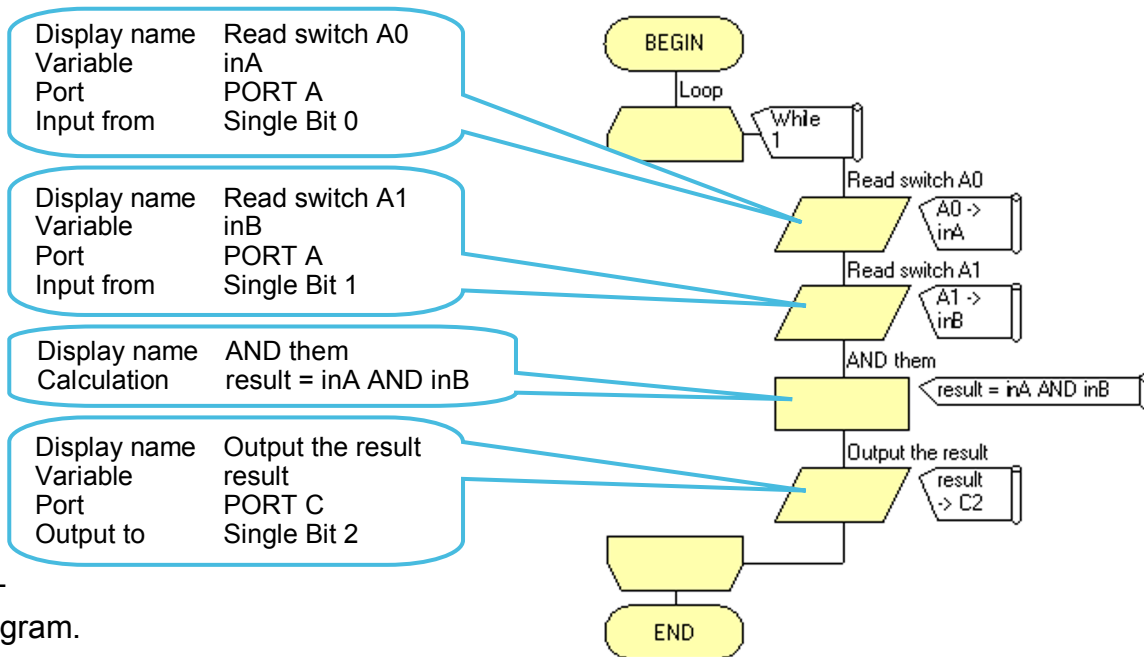
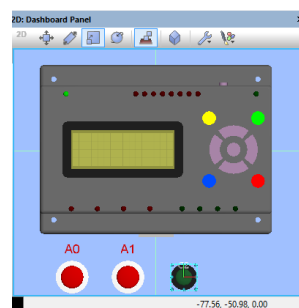
Many industrial machines have safety guards which must be in place before the machine can operate. In this worksheet, the machine operates only if the on / off switch is



### Over to you:

#### The program:

- Build the flowchart, and add two 'Switch (Push, Panel)'s and a 'LED (5mm Panel)' to the dashboard panel.
- Connect one switch to PORT A bit 0, the second to PORT A bit 1, and the LED to Port C bit 2.
- Create three new variables 'inA', 'inB' and 'result'.
- Configure the Loop icon as in earlier programs, and set up the other icons as follows:



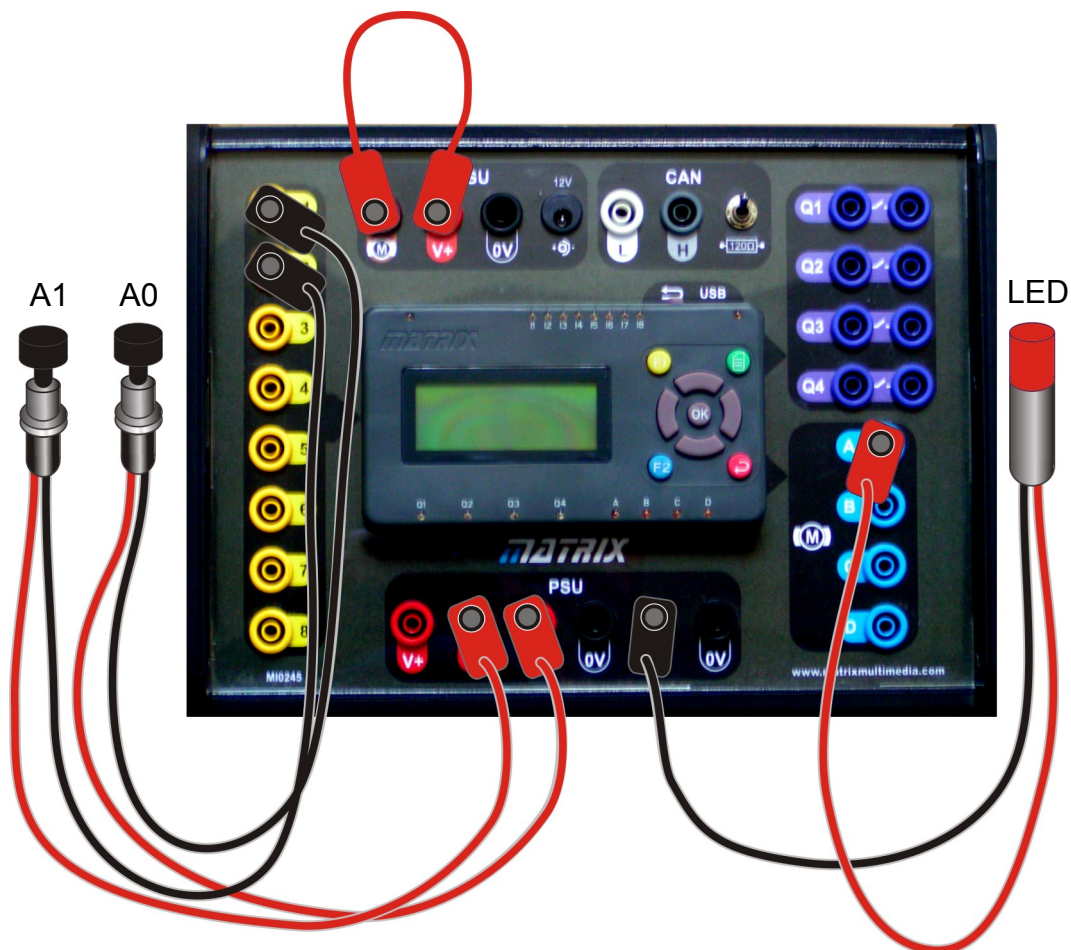
- Save the Flow-code program.
- Simulate it to check that it works. To make this easier, before simulating, click on each switch in turn, and change the 'Simulation / Operation' property to 'Latching'. The LED should light only when both switches are pressed.
- Now, compile it to the chip inside the MIAC.
- Build the circuit shown on the next page, and test it.

**FLOWCODE 6****Worksheet 6**

## The AND gate

**Computer Science  
and Control—1****Over to you:****The circuit:**

- Build the circuit, shown below, on MIAC.
  - Switch A0 represents the on / off switch, and A1 the safety guard on the machine. It is 'on' only when the safety guard is in place.
  - The LED represents the machine being controlled. Make sure that the LED is plugged in the right way round!
- Test the circuit by pressing one and then both switches in various combinations. The LED should light only when both switches are pressed.

**What to do next:**

Modify the program to set up an OR gate, where the LED lights if either switch A0 OR A1 OR both is/are pressed. This arrangement could be used to operate a machine from one of two locations.



# FLOWCODE 6

## Worksheet 7

A logic system

# Computer Science and Control—1

The previous worksheet showed how to set up a straightforward logic function, the AND gate. Microcontrollers can do far more powerful tasks.

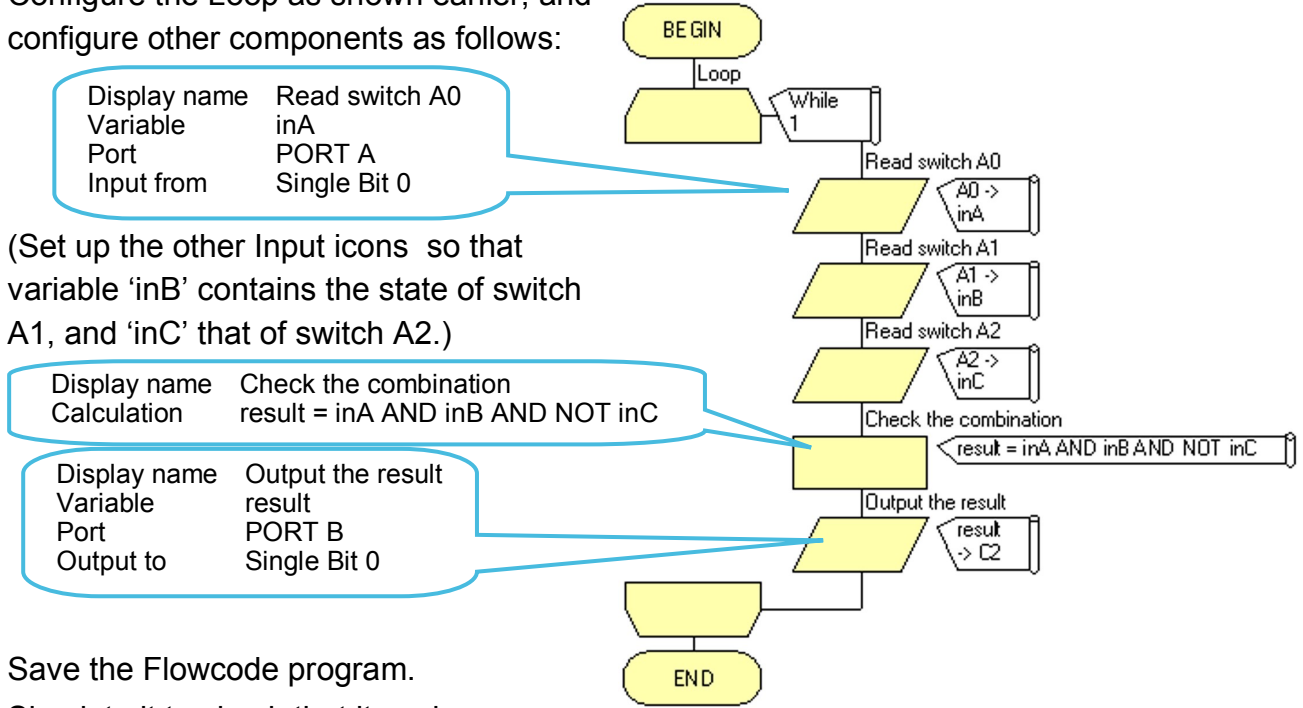
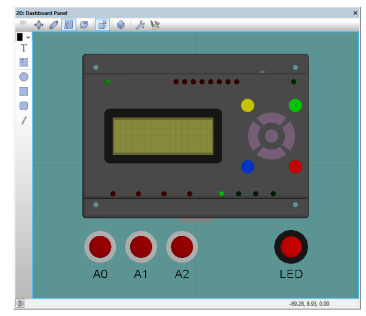
This time, the car seat-belt warning sign will light when the ignition switch is turned on, someone is sitting on the passenger seat (and the pressure switch inside it,) but has **not** fastened the



### Over to you:

#### The program:

- The aim is to light the LED only when switches A0 AND A1 are pressed AND switch A2 is NOT pressed.
- Build the flowchart, and add three switches ('Switch (Push, Panel)') and a 'LED (5mm Panel)' to the dashboard panel.
- Connect one switch to PORT A bit 0, the second to PORT A bit 1, the third to PORT A bit 2 and the LED to Port C bit 2. Add labels to identify them.
- Create four byte variables 'inA', 'inB', 'inC' and 'result'.
- Configure the Loop as shown earlier, and configure other components as follows:



- Save the Flowcode program.
- Simulate it to check that it works.
- Now, compile it to the chip.
- Build the circuit shown on the next page to test the program.



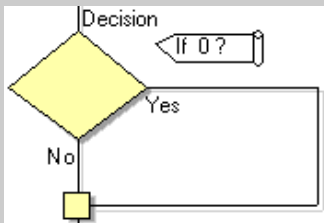
# FLOWCODE 6

## Worksheet 8

Anyone home?

# Computer Science and Control—1

Home security systems often have the ability to close curtains when it gets dark, as if someone is at home - another decision that a microcontroller can make, using information from a light-sensing unit.



The program uses a decision-making element to decide the outcome. In its simplest form, as used here, it makes a Yes/No decision - Is it dark, or not?

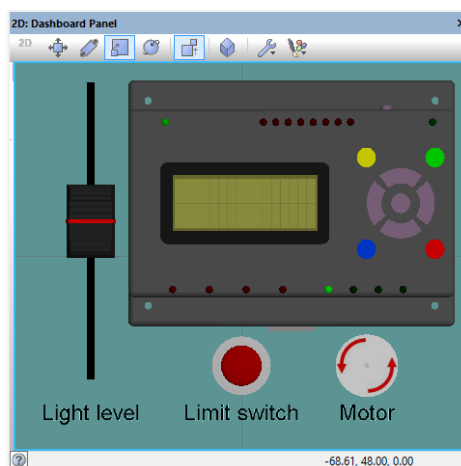


It makes this decision based on the signal from the light-sensing unit, contained in a variable called 'light'.

### Over to you:

#### The program:

- The motor is linked to the curtains and closes them at night. The motor stops when the curtains hit a limit switch. The aim of the program is to turn on the motor only when the light-dependent resistor (LDR) is in darkness, and to turn it off when the 'limit switch' is closed.
- Build the flowchart shown on the following page.
- Add a 'Potentiometer (Slider)', from the 'Inputs' toolbox, a 'Motor (Half Bridge)', from the 'Mechatronics' toolbox and a 'Switch (Push, Panel)' from 'Inputs' toolbox, to the dashboard panel.
- Arrange them, and add labels as shown below.



- The potentiometer represents the light-sensing unit. During simulation, moving the slider upwards represents more and more light shining on the LDR.
- Connect it to channel An 0.
- Connect the motor to Port C bit 2, (MIAC Transistor output A).

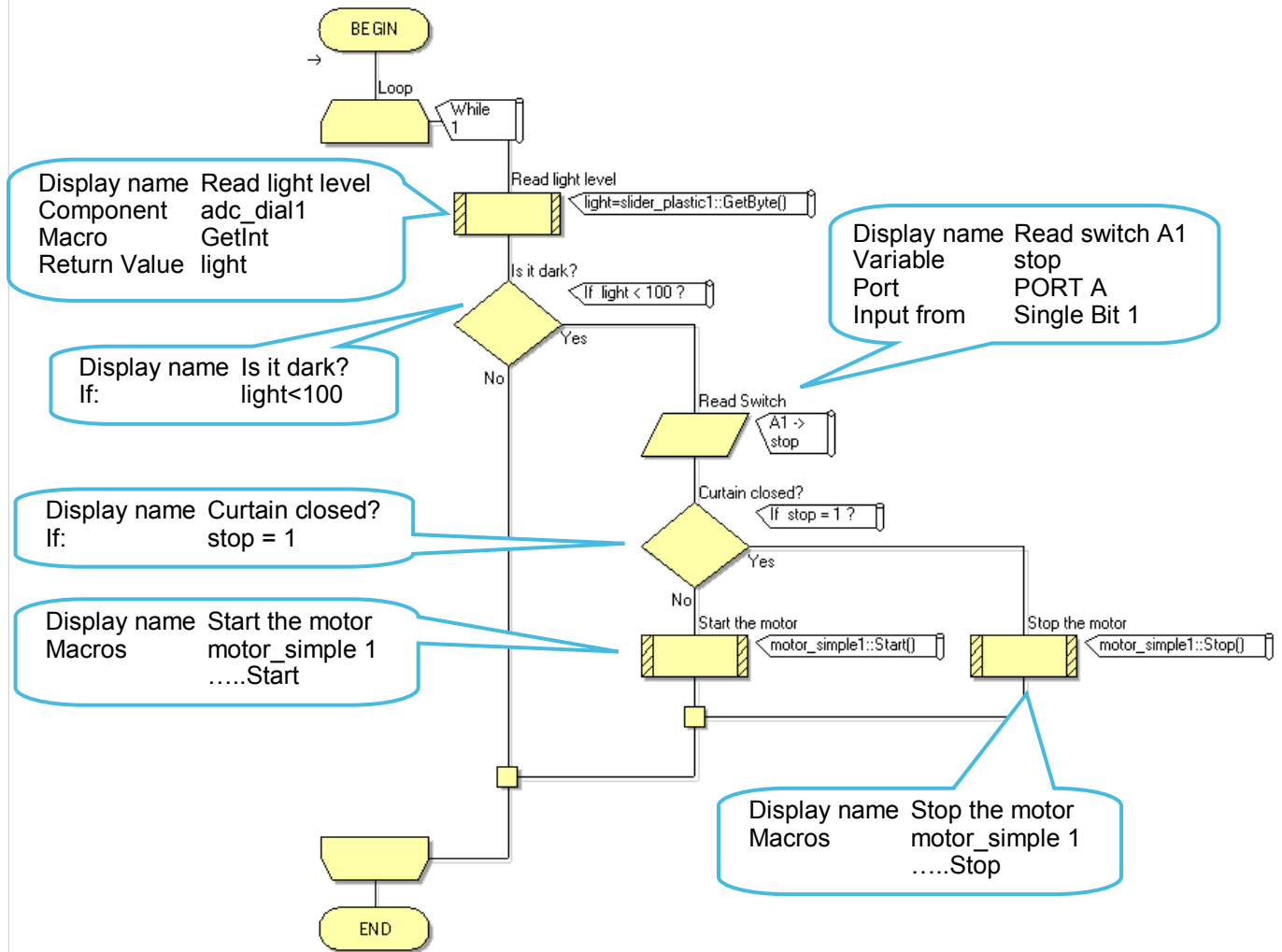
# FLOWCODE 6

## Worksheet 8

Anyone home?

# Computer Science and Control—1

- Configure the Loop icon as usual, and set up the other components as follows:



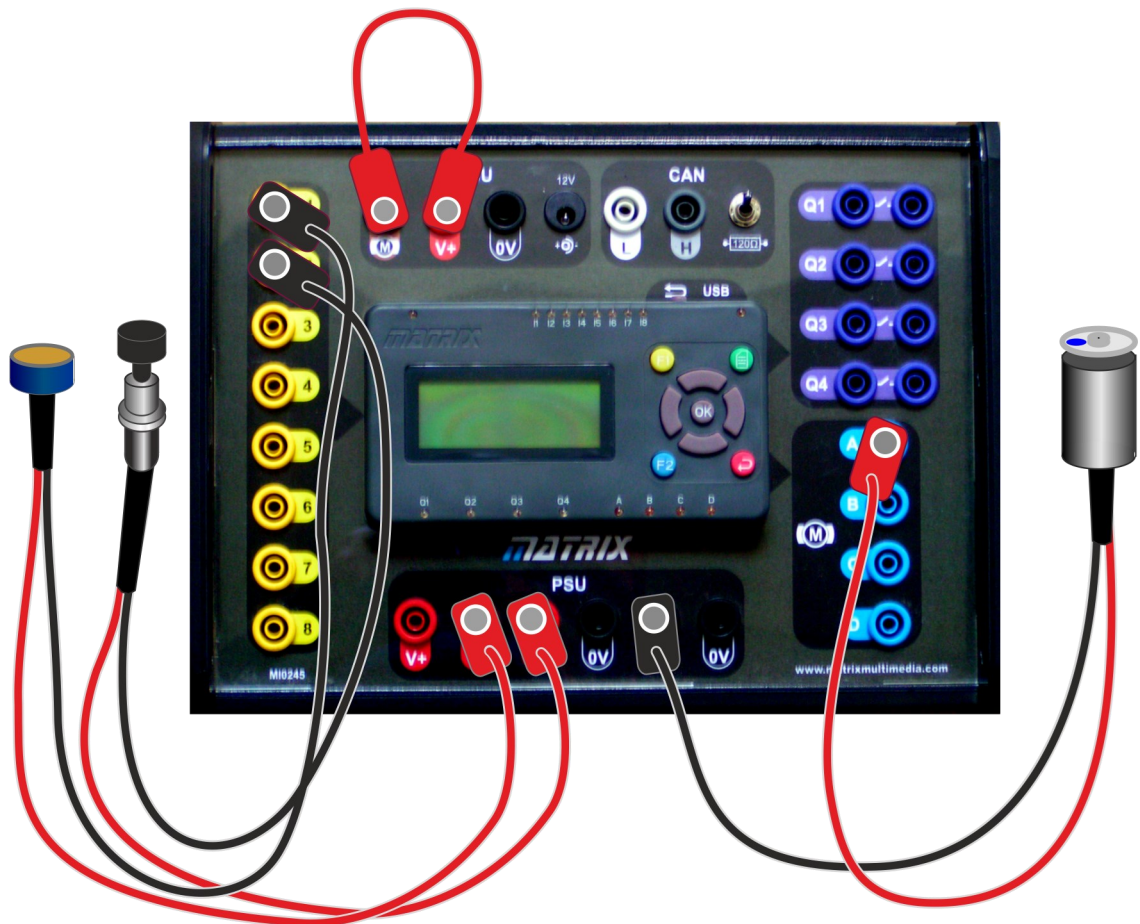
- The 'Component macros' are mini-programs, within Flowcode, created to make the job of controlling hardware easier.
- Save the Flowcode program, and simulate it to check that it works:
  - Move the slide on the potentiometer until the motor starts - the green LED on the MI-AC lights.
  - Then click on the switch to stop it.
- Now, compile it to the chip.
- Build the circuit shown on the next page.

**FLOWCODE 6****Worksheet 8**

Anyone home?

**Computer Science  
and Control—1****Over to you:****The circuit:**

- Build the circuit, shown below, on MIAC.
  - The LDR, (light-dependent resistor,) can be connected either way round. One wire is plugged into MIAC Input 1, and the other into V+.
  - The switch can be connected either way round. One wire is plugged in to MIAC Input 2, and the other to V+.
  - Connect the motor with the red lead connected to MIAC Output A, and the black lead to 0V.
- Test the circuit by shading the LDR with your hand. When dark enough, the microcontroller will switch on the motor. Then hold the switch on, to simulate the curtains reaching the end of the track and hitting the limit switch. The motor should stop.

**What to do next:**

When you simulate the program, under one condition it is not possible to stop the motor. When does that happen?  
How can you correct this problem?

**About this course****Introduction**

The course is essentially a practical one. The MIAC hardware makes it simple and quick to program and construct microcontroller circuits.

**Aim**

The course introduces the graphical programming language Flowcode 6, and its use in programming PIC microcontrollers.

**Prior Knowledge**

None needed.

**Learning Objectives**

On successful completion of this course, you will be able to:

- run the Flowcode 6 application;
- select a target microcontroller for it;
- create a Flowcode flowchart by adding and configuring the following icons:
  - loop;
  - input;
  - output;
  - decision;
  - delay;
  - calculation;
  - switch;
  - connection points;
  - component macros.
- create and use variables within the Flowcode program;
- open the Dashboard Panel and add and configure components on it;
- use on-screen simulation to debug the Flowcode program
- save the Flowcode flowchart;
- compile the Flowcode program into machine code;
- link MIAC to a computer and transfer the program;
- test a Flowcode program using hardware connected to MIAC
- use the MIAC in slave mode to the PC.

**What the student will need:**

To complete the course, the student will need the following equipment:

- 1xMIAC
- 1xPSU
- 1xUSB Programming Cable
- 2xLEDs
- 2xSwitches
- 1xDC Motor
- 2x5mm to 5mm cable

**Time:**

It will take students between four to six hours to complete the eight worksheets.

**Taking it further:**

Once you have completed these worksheets, you will want to experiment with your own programs. The 'Step-by-step' guide, on the pages that follow, provides a basis for these.

Use the examples in worksheets to help when adding hardware to MIAC. These cover a large number of hardware configurations. They show circuits with:

- a single LED;
- a switch and one LED;
- two switches and one LED;
- three switches and one LED;
- two LEDs;

**Further information:**

This course contains all the information you need to gain a firm foundation in the use of Flowcode 6 with the MIAC.

Once you have mastered this introduction to Flowcode 6, you are urged to look at the extensive volume of information and examples contained in the Matrix wiki, found at

[www.matrixsl.com/wiki](http://www.matrixsl.com/wiki)

to expand your knowledge and experience.

**Step-by-step Guide to Flowcode 6 flowcharts:****1. Open Flowcode 6:**

When the Flowcode application starts, you are presented with four options:

- New project;
- Open a template;
- Launch Flowcode Help;
- Open an existing Flowcode project.

Click on 'New project'.

**2. Select the target microcontroller:**

The 'Project Options' screen opens.

Click on the 'PIC' button.

In the drop-down menu next to 'Family:', click on the 'Misc' option, and then on the 'MIAC'. Then click on 'OK'

**3. Add the icons:**

A new flowchart appears.

If the 'System Panel' appears on the screen, delete it by clicking on the 'X' in the top right-hand corner, or by opening the 'View' menu and de-selecting 'System Panel'.

Down the left-hand edge of the workspace is the strip of possible icons. Click and drag the ones you want to make up the flowchart .

**4. Configure each icon:**

Double-click on each icon in turn, and use information like that given in the worksheets to complete the configuration dialogue box.

Click on 'OK' after completing each.

**5. Open the Dashboard Panel:**

Open the 'View' menu, and click on the 'Dashboard Panel' option.

**6. Add the components:**

Items such as switches, and the 'Potentiometer (Slider)' are found in the 'Inputs' toolbox.

The LEDs are found in the 'Outputs' toolbox, switches in the 'Inputs' toolbox.

MIAC itself is found in the 'MIAC Module' toolbox, and the motor in 'Mechantronics'.

Alternatively, use the 'Search' facility to locate the component you want.

Then click on the down arrow just to the left of its name, and select the "Add to dashboard panel' option.



## Step-by-step Guide to the Worksheets - continued...:

**7. Configure properties of components:**

Move the cursor over the component, and right-click the mouse.

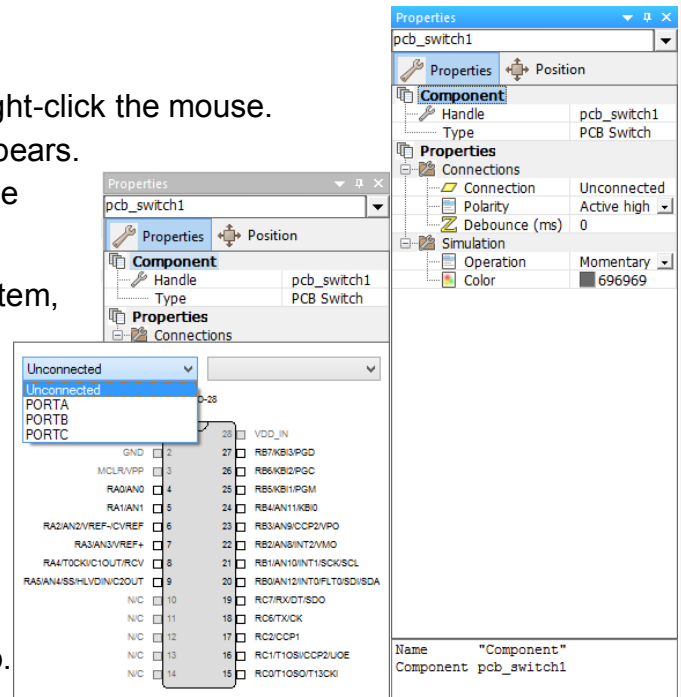
The 'Properties' panel for the component appears.

The important one for these worksheets is the 'Connections' property, shown on the right.

Click on the text alongside the 'Connection' item, (here "Unconnected" ) and a diagram of the chosen microcontroller chip appears.


Select the pin to connect the component by:

- selecting the port and bit from the two drop-down lists;
- or
- clicking on the pin on the image of the chip.

**8. Simulate the program:**

You can test whether your Flowcode program works by simulating it 'on-screen'.

To simulate a flowchart:

- select the 'Run' option  from the 'Debug' menu;
- or
- click the 'Run' button on the main toolbar (or press **F5**).



Flowcode will go into simulation mode and will start to execute the program in the flowchart. A red rounded rectangle indicates the next icon to be executed.

Simulations can be paused or stopped by selecting either the 'Pause' or 'Stop' options from the 'Debug' menu or selecting them from the simulation section of the main toolbar.


Alternatively, you can simulate the flowchart step by step, using the 'Step Into' function by pressing **F8**. You can also 'Step Over' icons by pressing **Shift+F8**.

All these options can be accessed from the 'Debug' menu or by clicking the buttons on the main toolbar.

**Step-by-step Guide to the Worksheets - continued:****9. Save the flowchart:**

Flowcharts must be saved before they can be downloaded to MIAC.

To save the flowchart:

- select either the 'Save' or 'Save As' option from the 'File' menu;
- **or**
- click the  button on the main toolbar.

**10. Connect MIAC to the PC:**

Connect the USB socket on MIAC to the USB port of a computer. This allows the program to be transferred from the computer to the PIC microcontroller inside MIAC.

**11. Compile the flowchart and transfer it to the chip:**

The next step is to compile the Flowcode program, (convert it into machine code,) and then transfer it to the microcontroller inside MIAC.

To do so:

- select the 'Compile to Chip'  option from the 'Build' menu;
- **or**
- click the 'Compile to Chip' icon on the toolbar.

**12. Test the program using the hardware:**

Press the 'Reset' switch on MIAC.

Now use the attached switches and LEDs etc. to confirm that the program does what you want it to do.

**Additional information:****Variables**

In programming, a variable is a named location in memory, used to store information. The information may change as the program runs, but the program can always find it, by searching for the name of the variable.

Flowcode allows a number of different types of variable, resulting in efficient storage of different types of information. They include:


- Boolean variables - store just a single bit - either a single '0' or a single '1'. They take up very little room in memory, but store little information, usually the state of a digital sensor such as a switch.
- Byte variables - store a 'byte' (8 bits,) and so can store more information, numbers from 0 to 255.
- Integer variables - store integers (whole numbers.) These can include both positive and negative numbers, or allow only positive numbers, and are 16 bits long.

You can set the variable type when you create the variable. The easiest way to do this is to use the Project Explorer. You can select this from the 'View' menu on main toolbar.

# Teacher Guide

## Scheme of work

# Computer Science and Control—1

Work sheet	Notes for the Instructor	Timing
	<p><b>Introduction:</b></p> <p>The diagram will be particularly useful for students starting this module allowing them to identify parts of the MIAC. In particular, they should note the position of the reset button, the USB connector and the various power sockets.</p> <p>The Introduction then gives an overview of MIAC and the internal connections to microcontroller pins.</p>	
1	<p>The task seems trivial - a switch connected in series with the LED could do the job. However, in many systems, such as in modern vehicles, this type of switching is carried out in software, not in hardware.</p> <p>Although the task is straightforward, it gives the student experience of:</p> <ul style="list-style-type: none"> <li>• 'drag-and-drop' to construct the flowchart;</li> <li>• configuring icons;</li> <li>• running the simulation within 'Flowcode';</li> <li>• transferring the program to hardware.</li> </ul> <p>Consequently, an inexperienced student may take longer than expected.</p> <p>'Flowcode' often offers several solutions to the same task. Here, the state of the switch ('off' = 0, 'on' = 1,) is stored in the 'switch' variable, and is then sent to the pin controlling the LED. The same result could be achieved by using component macros (  ) to switch the LED on and off.</p> <p>To create the 'switch' variable, open 'Project Explorer', click on the down arrow to the left of the 'Variables' label, and select the 'Add new' option. You can then name the new variable, give it an initial value and add a description. The next task is to select a variable type. The reason for doing so is ultimately to reserve sufficient space in memory to accommodate the value stored in the variable. In principal, then, it is best to choose the type that most efficiently fits the data you intend to store. For the purposes of this module, unless told otherwise, it is fine to accept the default type, byte variable.</p> <p>Very few problems are likely. The simulation might be made easier by turning the switch action from 'momentary' to 'latching'. Do this by right-clicking on the switch and selecting 'Properties'. Then click on the down arrow at the right-hand end of the 'Operation' row and selecting 'Latching'.</p> <p>If the program does not simulate correctly, then check that the student:</p> <ul style="list-style-type: none"> <li>• loaded the three components to the dashboard panel successfully;</li> <li>• connected them to the correct pins of the microcontroller, (switch to Port A, bit 0 and LED to Port C, bit 2);</li> <li>• configured the icons correctly.</li> </ul> <p>(If the Dashboard is not visible, use the 'View/Dashboard Panel' menu selection.)</p> <p>If the simulation works, but the hardware doesn't, then make sure that:</p> <ul style="list-style-type: none"> <li>• the USB lead is connected between MIAC and the PC;</li> <li>• a wire connects the 'V+' and 'M' sockets;</li> <li>• the LED is plugged in the right way round - black wire to 0V.</li> </ul> <p>The next step is to download the program again, watching the 'Compiler Messages' screen. If there is still a problem, use the program provided in the 'Solutions' folder.</p> <p>The 'What to do next' section invites student to modify the program to switch on two LEDs. For this, a second LED is connected to Transistor Output B (Port C, bit 0.)</p>	20 - 30 mins

# Teacher Guide

## Scheme of work

# Computer Science and Control—1

Work sheet	Notes for the Instructor	Timing
2	<p>This worksheet extends the task of the first, by making the LEDs flash on and off. This time, no switch is involved, making the hardware even easier to construct. The program uses a loop to make the LED switch on and off repeatedly. (Once again, the task could have been accomplished using component macros to switch the LED.)</p> <p>If the simulation does not work, review the checks listed for the last worksheet. The only new aspect is the use of loops. Check the configuration, by double-clicking on the icon. Similarly, for hardware problems, use the issues listed there.</p> <p>In the 'What to do next' section, the frequency of flashing is governed by the duration of the two delays. To double the frequency, as asked, the delays are halved.</p> <p>Taking this matter still further, the first delay decides how long the LED is off (i.e. the 'space') while the second controls how long the LED is on (i.e. the 'mark'). Adjusting the <i>relative</i> size of the delays controls the 'mark-to-space' ratio for the signal .</p> <p>To make the LED flash only ten times requires reconfiguration of the loop, to use a loop count set to ten.</p>	20 - 30 mins
3	<p>This worksheet uses the same hardware, set up in the same way, as in worksheet 1. The program introduces the Decision icon, which tests whether the switch has been pressed. As a result, the program will follow one of two paths - the 'Yes' (condition is true,) or 'No' (condition is false,) route. It is worthwhile stepping through the program with the students, looking at what happens when the switch is or is not pressed.</p> <p>If the program does not simulate correctly, check the structure of the program - that the 'Yes' and 'No' loops are the right way round, (as one option in configuring the Decision icons is to reverse them.) Then look at the advice given on the previous page, for worksheet 1. If there are any other problems either with the program or the hardware , consult the list of issues given for that worksheet on the previous page.</p> <p>The extension mentioned in the 'What to do next' section simply requires adjustment of the duration of the delay.</p>	
4	<p>In electronics, a latch is a subsystem which has two stable states. It is triggered into one (the 'on' state,) by a switch called the 'set' switch. The subsystem remains 'on' until it is triggered into the other state (the 'off' state) by the 'reset' switch.</p> <p>As before, the state of one switch, (the 'set' switch) is stored in the variable called 'switch'. A second variable ('reset',) is created to store the state of the other switch, the 'reset' switch. It can be created using the 'Project Explorer', as described earlier.</p> <p>The program uses Decision icons again, to test whether each switch is pressed. (For programming aficionados, the condition could be written simply as 'If switch' and 'If reset' meaning 'If switch / reset is <i>true</i> (i.e. =1.)') Once again, step through the program, looking at what happens when the switches are pressed or are not.</p> <p>If the program does not simulate correctly, check that the 'Yes' and 'No' loops are the right way round. Then look at the advice given for worksheet 1.</p> <p>The hardware is slightly more complicated, because of the second switch. Make sure that the signal from it is connected to Input 2. Then look at the other issues listed on the previous page.</p> <p>The task in the 'What to do next' section should be tackled by making straightforward changes to the structure of the program.</p>	

# Teacher Guide

## Scheme of work

# Computer Science and Control—1

Work sheet	Notes for the Instructor	Timing
5	<p>This worksheet looks at a common task for microcontrollers - controlling a number of devices, motors, heaters, solenoids, lamps etc, in sequence. This program can readily be modified to cover a wide range of similar applications.</p> <p>The program is linear - no branching. It progresses through a series of instructions lighting each LED in turn for a set time. It may not be obvious to students that the sequence turns on other LEDs while not turning off those already lit, until the program loops back to the beginning. They could be encouraged to experiment with the output icons, sending various decimal numbers to 'Entire Port' rather than 'Single Bit'. For example, sending the number '2' (=010<sub>2</sub> in binary,) will light only the second LED, while turning the first and third LEDs off. Similarly, sending number '6' (=110<sub>2</sub>,) lights the second and third LEDs, but not the first.</p> <p>It is unlikely that any new issues arise either in the program, or the hardware, so use the previous checks if there are problems.</p> <p>To tackle the task given in 'What to do next', students should remind themselves of what they did in the program in worksheet 3.</p>	20 - 30 mins
6	<p>This worksheet introduces an important area in control - logic systems . In these, the microcontroller makes decisions based on the information it receives, from switches in this case. The 'AND' function requires that switch A AND switch B are pressed. The 'OR' function waits until either switch A OR switch B (or both) is/are pressed.</p> <p>The student first creates variables 'InA', InB' and 'result', using 'Project Explorer' as before. The program examines each switch in turn, storing its on/off state as 1/0 in the appropriate variable. The logic operation is carried out inside the Calculation icon, generating a 1/0 value in the variable 'result'. This is displayed on Port B, bit 0.</p> <p>The program involves no branching and so is straightforward. If it does not simulate correctly, first check the calculation inside the Calculation icon, and then look back at the checks suggested earlier.</p> <p>The hardware arrangement is the same as in worksheet 4, and so the same checks apply if there are hardware problems.</p> <p>The modification suggested in the 'What to do next' section requires a very small adjustment to the calculation configured in the Calculation icon.</p>	25 - 40 mins
7	<p>This worksheet extends the area introduced in the last one, logic systems. The last logic functions, like AND and OR. This one shows how to handle more complex logic operations.</p> <p>The program is again a linear one. A third 'switch' is added to the system, to represent the seat-belt, and so a third variable, InC, must be created to store its state. An input icon reads the state of this third switch and stores it in InC. After that, all the hard work is carried out inside the Calculation icon, with the result processed as before. As before, if it does not simulate correctly, first check the configuration of the Calculation icon.</p> <p>The prototype board now looks a bit crowded, with the addition of the third switch and resistor, and it is easy to make mistakes. The usual checks apply, and should highlight any problems. The third switch provides a signal for Port A, bit 2.</p> <p>The modification described in the 'What to do next' section is straightforward to implement. The microcontroller is already gathering information about the states of the three switches from the variables InA, InB and InC. What it does with that is controlled by the calculation inside the Calculation icon. This modification amounts to a change in that calculation.</p>	25 - 40 mins

# Teacher Guide

## Scheme of work

# Computer Science and Control—1

Work sheet	Notes for the Instructor	Timing
8	<p>Now a departure! This worksheet involves handling an analogue signal.</p> <ul style="list-style-type: none"> <li>• Digital signals have only two possible values - off and on, or equivalents.</li> <li>• Analogue signals, on the other hand, vary over a wide range of values.</li> </ul> <p>This may be a good time for the teacher to elaborate on the differences between analogue and digital quantities, and to talk about analogue-to-digital conversion.</p> <p>Analogue signals are created from a huge variety of devices and subsystems. To simplify the matter, Flowcode handles analogue inputs through simple devices like potentiometers. Adjusting these creates analogue signals which 'Flowcode' uses to mimic the output of devices like the light sensor. As the light gets brighter, the output increases.</p> <p>The hard work is done inside the PIC chip, using a 'Component Macro', a section of program code written especially for one of the hardware components in 'Flowcode'. The 'GetByte' macro generates a numerical output representing the brightness of the light falling on the LDR. This output is stored in the integer variable 'light'.</p> <p>The rest of the program follows the line taken in previous exercises - using a 'Decision' icon to determine which branch the program should follow.</p> <p>If there is a problem with simulation, the teacher could disable the Component Macro, (by right-clicking on it and selecting the 'Disable Icon' option,) and add a Calculation icon to set the value of 'light' at 600, and later 50 to test whether the rest of the program works. If it does, then the problem lies in the configuration of the Component Macro. This could be checked against the solution provided.</p> <p>If there is a problem running the hardware, having checked the issues outlined earlier in this guide, check that the LDR and switch are plugged into V+ terminals and the correct inputs to MIAC ('0' for the LDR, and '1' for the switch).</p> <p>With the program in the form given, it is possible during simulation to move the Potentiometer slide down to make the motor start, then move it back up. The switch can now not be used to stop the motor. The 'cure' involves an additional step such as adding another 'Stop' component macro to the 'No' path from the 'Is it dark?' decision icon. In practice, this situation is very unlikely to occur with the hardware.</p>	20 - 30 mins