MATRIX | ROBOT ARM

Robot Arm Pendant Programming

MATRIX

CP9125

www.matrixtsl.com

Contents

Getting started

Before starting the worksheets, you need to understand some aspects of the hardware and software that you will be using.

Firstly, please read through the information in the Reference section - not to absorb it all but in order to know what is there and where it is so that later on you know where to find it.

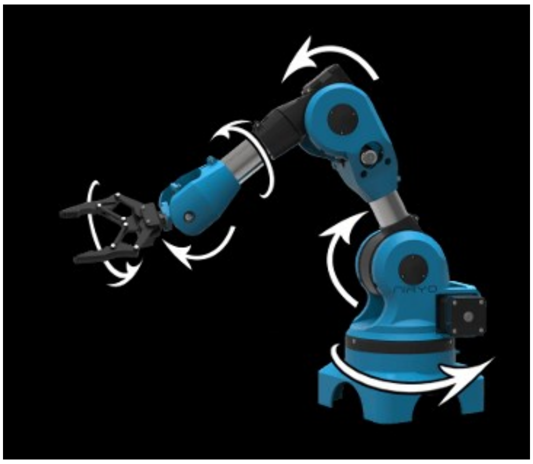
For the teacher, the Teachers' notes section outlines a number of teaching options with this Robot Arm. This section guides you through them, suggesting ways of using the product and giving you a view of the learning objectives and difficulties of each worksheet.

You need to decide exactly how you want to program the Robot Arm:

Once you are familiar with the hardware and software, then begin work on the worksheets provided. These are structured to build your learning, step-by-step.

Worksheet 1

Investigating the workspace



Robot arms can rotate around a number of axes. This determines their range of movement and capability.

Most have **base** rotation, **shoulder** rotation and **elbow** rotation. This, and the length of its limbs, determine the reach of the Robot Arm.

The Arm movement is confined to a three-dimensional space known as the 'workspace'.

Over to you:

- Read the following in the Reference section:
  - Understanding the Robot Arm
  - Connections and Circuit
  - PC USB driver and software set up
  - Basic Control software app

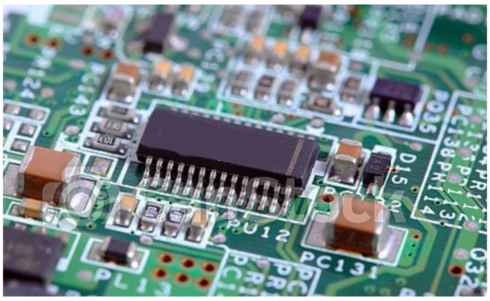
- On your PC, load the 'BasicControl.bat' software.
- Click on the 'Run' icon, ▶, situated at the top left. The 'Connected' box should turn green.
- Click on the 'Enable Motors / Home' button. The 'Enabled' box should turn green.
- Use the 'Base', 'Shoulder' and 'Elbow' slide bars to explore movement of the arm.
- Click on the 'Enable Motors / Home' button again. The arm returns to its 'home' position.
- Use the 'Set' buttons to specify angles for the 'Base', 'Shoulder' and 'Elbow' motors.
- Click on the 'Open' and 'Close' buttons for the gripper. Use the 'Set' button to specify the extent to which the gripper opens.
- Click on the 'Enable Motors / Home' button again.
- Use the 'Set' buttons to move the arm to the following position:

  'Base' = 90^0, 'Shoulder' = 55^0, 'Elbow' = 65^0.
- Open the gripper and place a counter between its jaws.

Challenge:

- Use the gripper to pick up a counter and return it to the arm's 'home' position.

Worksheet 2
Store it



It is important that robot arms can 'remember' where things are!
They contain electronic memory which enables them to store programs, positions, speeds etc.
This worksheet uses this built-in memory.

Over to you:

- Load the 'BasicControl.bat' software.
- Click on the 'Run' icon and the 'Enable Motors / Home' button. The 'Connected' and 'Enabled' boxes should turn green.
- In the 'Position Index', press on '+' to select position 1.
- Use the Base 'Set' button to specify a Base motor angle of 90^0.
- Click on 'Save' in the 'Position Index'.
- Select position 2 and save a Base motor angle of 90^0 and Shoulder angle of motor angle of 55^0.
- In the same way, for position 3, store 'Base' = 90^0, 'Shoulder' = 55^0, 'Elbow' = 65^0.
- Return the arm to its 'home' position.
- Select position 1 in the 'Position Index'.
  Click on the 'Goto' button and notice what happens.
- Do the same for positions 2 and 3.
- From the 'home' position, send the arm to position 3. Notice that rotation occurs around all three axes simultaneously.

- Drive the arm until the gripper sits on the green circle labelled '**U**' on the mat.
- Use 'Position 4' to store its co-ordinates.
- Test that this works.

So what?

- The Robot Arm can store up to thirty sets of co-ordinates in this way.

Challenge:

- Use 'Position 5' to store the co-ordinates of the blue circle labelled '**B**' .
- Test that this works.

Worksheet 3
Sort it

The Robot Arm's colour sensor allows it to distinguish between coloured objects and hence to sort and select them.

Traditionally a human task, sorting done by robot is faster, more reliable and accurate. Humans get tired and bored by repetitive tasks but robots love repetition.

Robotic sorting is used in a wide range of industries, e.g. pharmaceuticals, agriculture, waste recycling .... It is used to sort different types of grain, different waste materials, precious stones ...

Over to you:

- Load and run the 'BasicControl.bat' software, as before.
- Place a red counter on the colour sensor, manually.
- Click on the 'Sample' button in the 'Colour Sensor'.
- The coloured rectangle in the 'Colour Sensor' should turn red.
- Notice the number displayed above the coloured rectangle. This number is a measure of the red, green and blue (RGB) components in the light reflected from the counter.
- Now do the same for the green and blue counters.

So what?

- The Robot Arm software uses the RGB value to distinguish between red, green and blue objects and select the appropriate colour for the rectangle in the 'Colour Sensor'.

Worksheet 4

Drive it with the Pendant

The most common methods for programming a robot are:

- using a programming language;
- ‘teaching’ the robot on the factory floor.

When teaching the robot, we often make use of a ‘pendant’, connected to a PC or to the robot itself.

This worksheet uses a PC-based app with a simulated pendant.

The image shows a robot arm connected to a teaching pendant.

Over to you:

- Read through the information in the Reference section on the Pendant program.
- On your PC, close the Basic Control program and load the ‘PendantControl.bat’ software.
- Click on the ‘Run’ icon, ▶, situated at the top left. The ‘Connected’ box should turn green.
- In the ‘LIVE ARM CONTROL section, click on the ‘Enable Motors / Home’ button. The ‘Enabled’ box should turn green.
- Investigate the effect of clicking on the ‘Position’ and ‘Gripper’ buttons.
  Notice what happens to the ‘Coords’ and ‘Angles’ readings as you do so.
- Use these controls to move to and pick up a green counter and return it to the arm’s ‘home’ position.

So what?

- The ‘LIVE ARM CONTROL section offers sensitive control of the arm’s position, moving it by a fraction of a degree each click.


```
G00 X-25 Y20
G91 X-10
G02 X10 Y-10 I10 J0 F200 M3
G02 X-10 Y10 I0 J10;
G90 G00 M5 X0 Y0;
G00 X25 Y20
G91 X-10
G02 X10 Y-10 I10 J0 F200M3
G02 X-10 Y10 I0 J10;
G90 G00 M5 X0 Y0;
G00 X40 Y0
G02 X-40 Y0 I-40 J0 F200 M3
G90 G00 M5 X-55 Y0;
G02 X0 Y-55 I55 J0 F200 M3
G02 X-55 Y0 I0 J55
```

G code is a programming language used in manufacturing and engineering to control the movement and behaviour of machines.

G code stands for “Geometric Code”.
Its commands tell the device where and how fast to move.

It combines a set of standard commands with machine-specific commands - known as ‘M codes’.

Over to you:

- Read through the information in the Reference section on G code.
- On your PC, load the ‘PendantControl.bat’ software and click on the ‘Run’ icon, .
- The first activity focuses on the G code commands ’G0’ and ‘G1’, which move the arm to a position specified by the co-ordinates that follow the commands.
- Previously, when Robot Arm was controlled manually, the activities started by clicking the ‘Enable Motors / Home’ button. When using a G code program to control the arm, it should start with the command that does the same thing, command ‘G28’.
- In the left-hand panel of the Pendant control app, click on ’New’ button to start a new file.
- Some G code commands are listed in the right-hand panel. Click on the ‘G28 Motors On’ button. On the screen, the display should show ‘0001 G28’. Your program has started with the instruction on line 0001 to turn on the motors.
- Next, click on the button labelled ‘Insert After’ and then on the command button ‘G0 Rapid Move’. The software asks you for the co-ordinates of the destination for the arm.
  Insert ‘X = 213’, ‘Y = 194’ and ‘Z = 17’.
- In the ‘Execution’ section of the app, click on ‘Start’. Notice what happens.
- Next, click on the ’New’ button and do the same thing but this time using the command ’G1 Move’. As before, the software asks you for the destination co-ordinates.
  Insert ‘X = 213’, ‘Y = 194’ and ‘Z = 17’. This time, you are then asked for a speed value. Insert the number ‘10’.
- In the ‘Execution’ section of the app, click on ‘Start’. Notice what happens.

Worksheet 5
Starting G code

So what?

- Although there are around 100 different G codes, the Pendant Control software will recognise only those listed in the right-hand panel of the image.
- The ‘G0 Rapid Move’ instruction is used to relocate the arm to a new position.
- The ‘G1 Move’ instruction is used when the arm is carrying out a task, like machining or cutting, between the start and end points.

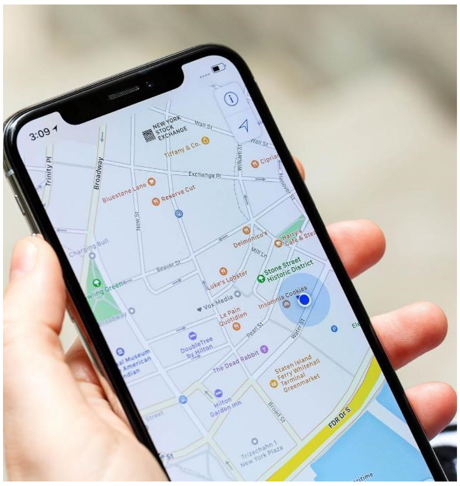
- Another useful command is ‘Pause’ G4. To configure it, follow it with the desired time delay, in milliseconds. For example, for a 2s delay, enter G4 2000. The software adds a ‘P’ automatically as part of the G code instruction.

Challenge:

- To compare the G0 and G1 commands, combine the instructions you just used.
  Write a program that:
  - moves the arm rapidly to the destination X= 0, Y = 200, Z = 15;
  - pauses for 2s;
  - returns to its home position (X=186, Y=0, Z=116) at a slow speed (speed value 10).
- Test it to see how G0 and G1 compare.

Worksheet 6

Absolute vs relative



The G code program usually specifies a destination for the Robot Arm.

It can do this in one of two ways:

- absolute positioning;
- relative positioning.

The first spells out precise co-ordinates of the target destination.

The second tells it how to get to the next destination from its current location, rather like a sat-nav app.

Over to you:

This worksheet contrasts the two types of positioning.

- Start Pendant Control and click on ‘New’ to start a new program.
- Create the following program:

Line	Instruction	Comment
0001	G28	Turn on motors (in home position)
0002	G90	Select absolute positioning;
0003	G1 X20 Y100 Z5 F25	Move to X = 20, Y = 200, Z = 10 at speed 25;
0004	G4 P5000	Delay / Pause for 5s (5000ms);
0005	G91	Select relative positioning;
0006	G1 X-20 Y10 Z-1 F25	Move to add -20 to X, 10 to Y and -1 to Z at speed 25;
0007	G4 P5000	Pause for 5s (5000ms);
0008	G90	Select absolute positioning;
0009	G28	Return to home position.

The quick way to do this:

- ‘Copy-and-paste’ the instruction column, (but not the line numbers or comments,) into Notepad.
- Save the file as ‘W6A’ or similar.
- In Pendant Control, click on ‘Open’ and select the file. The software opens it as a G code file.

Worksheet 6

Absolute vs relative

Over to you.....

- To see the effect of each command, click on ‘Step’ in the ‘Execution’ area. This causes only the line indicates by the cursor to execute. The cursor then moves on to the next line.
- Each time that the arm moves, notice the ‘Coords’ readings and compare them with the command just executed. Hence, contrast the effect of absolute and relative positioning.
- Click on the ‘Start’ button to run the whole program.
  Notice that the cursor now progresses through the whole program.

So what?

- The G code instructions can be strung together to define a series of movements and activities for the Robot Arm.

Challenge:

A segment of a program is shown below.
In the second column of the table, part of this program has been ‘decoded’.
Complete the table by adding explanations for the remaining part.

```
G28
G0 X140 Y0 Z81
M6 A255
G1 X180 Y20 Z50 F2000
M6 A70
G1 X140 Y0 Z81 F2000
G18
```

Instruction	Comment
G28	Turn on motors
G0 X140 Y0 Z81	Rapid move to X,Y,Z co-ordinates 140,0,81
M6 A255	Open gripper fully
G1 X180 Y20 Z50 F2000	Move to X,Y,Z co-ordinates 180,20,50 at speed value 2000
M6 A70	
G1 X140 Y0 Z81 F200	
G18	

Worksheet 7

Pick and place

ROBOT ARM

Pendant Programming

The image shows a robot arm moving a car windscreen into place.

This worksheet mirrors that task.

It uses both absolute and relative positioning to pick up a blue counter and move it to a new position.

Over to you:

- Start Pendant Control and click on ‘New’ to start a new program.
- Create the following program:

Line	Instruction	Comment
0001	G28	Turn on motors (in home position)
0002	G0 X214 Y277 Z150	Move rapidly to blue counter
0003	G91	Select relative positioning
0004	G1 X0 Y0 Z-130 F25	Lower arm slowly
0005	M6 A50	Close gripper
0006	G1 X0 Y0 Z130 F25	Raise arm slowly
0007	G90	Select absolute positioning
0008	G0 X0 Y200 Z150	Move rapidly to new position
0019	G91	Select relative positioning
0010	G1 X0 Y0 Z-130 F25	Lower arm slowly
0011	M6 A255	Open gripper
0012	G1 X0 Y0 Z130 F25	Raise arm slowly
0013	G28	Return to home position

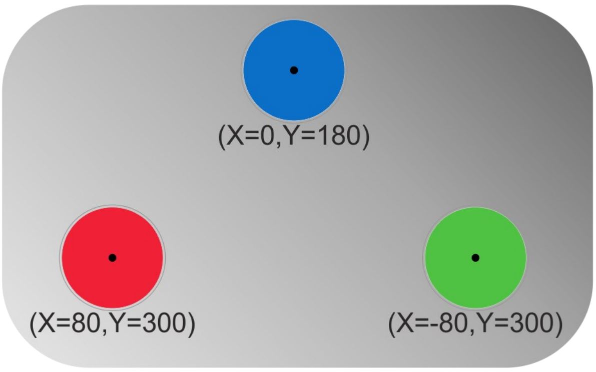
- To see (and check) what each command does, click on the ‘Step’ button in the ‘Execution’ area and move through the program one command at a time.
- When you are happy with it, click on the ‘Start’ button to run the whole program.

So what?

- The Pendant Control software defaults to absolute positioning mode. That is why there is no need to specify it at the start of the program.
- The first move, to the position of the blue counter is done using absolute positioning, hence the need to specify the absolute co-ordinates 214,277,150 of the blue counter.
- Raising and lowering the gripper is done more slowly, in relative mode, using command G1 at speed value 25.
- Line 0012 is still in relative positioning mode. The final command, G28, is automatically in absolute mode.

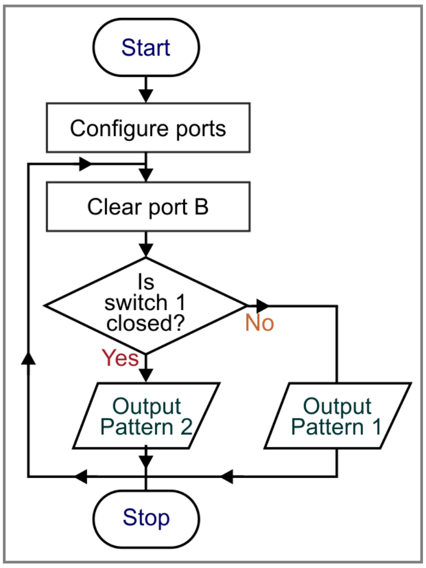
Challenge:

- Extend the program so that the arm picks up each counter in turn from its ‘home’ position, (**B**, **D** and **F**,) and repositions them. The diagram gives the target X,Y co-ordinates for their final positions. Having done that, the arm should return to its ‘home’ position.



Worksheet 8

Branching - unconditional



Some programs are just a series of activities that run one after the other, regardless. In others, the program can branch and take a route that depends on what is happening.

There are two types of branching:

- conditional branching - in some situations the program will take a branch, whereas under others, it does not;
- unconditional branching - the program always branches.

The Pendant Control app offers both types.

- The GOTO command is unconditional. Whenever, the program reaches a GOTO command, it branches.
- 'If...' commands are conditional. Branching takes place only if certain conditions apply, e.g. the object is green, but more of that later.

Over to you:

- In Pendant Control, open the program you created in worksheet 6, shown opposite:
- Click on 'Loop' in the 'Execution' section.
- Notice what happens. Watch the behaviour of the cursor!
- When you understand what is happening, click on the 'Stop' button and return the arm to its home position.

```
0001   G28
0002   G90
0003   G1 X20 Y100 Z5 F25
0004   G4 P5000
0005   G91
0006   G1 X-20 Y10 Z-1 F25
0007   G4 P5000
0008   G90
0009   G28
```

- Modify the program by inserting an extra command 'GOTO 0005' after line 0007.
- The program now looks like that opposite:
- Click on 'Start' in the 'Execution' section.
- Notice the difference in the behaviour of the program and the cursor.
- Once again, when you understand the result, click on the 'STOP' button and return the arm to its home position.

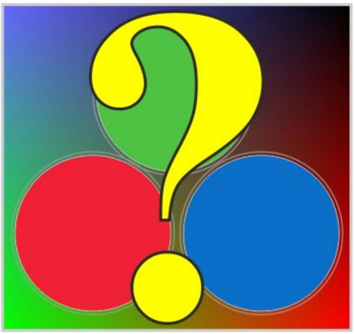
```
0001   G28
0002   G90
0003   G1 X20 Y100 Z5 F25
0004   G4 P5000
0005   G91
0006   G1 X-20 Y10 Z-1 F25
0007   G4 P5000
0008   GOTO 0006
0009   G90
0010   G28
```

So what?

- The 'Loop' instruction in the 'Execution' section of Pendant Control is really a form of the 'GOTO' command that sends the program back to the beginning each time it reaches the end.
- The 'STOP' button does what its name suggests but only on completion of the current instruction.

Worksheet 9

Branching - conditional



Colour sorting machinery is used widely in industry.

For example:

- in agriculture and food processing, grains, nuts, beans, tea and herbs are usually sorted mechanically;
- in recycling plants, a mechanical sorter can distinguish between different types of plastic.

Over to you:

- In Pendant Control, write a program that picks up a blue counter and places it on the colour sorter.
  One way to do this is shown opposite.
- Next, add a 2s pause at line 0010 and then the following code:

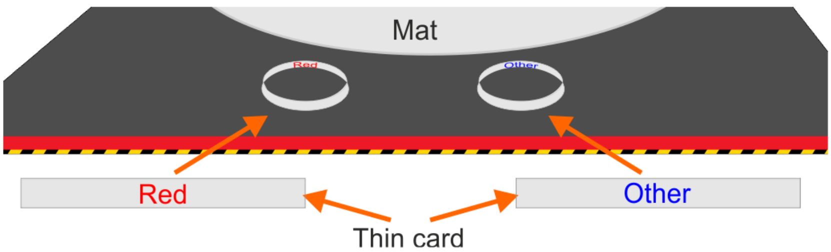
```
0011 If Col Red GOTO 0013
0012 GOTO 0016
0013 G1 X262 Y45 Z150 F100
0014 G0 X0 Y180 Z35
0015 M6 A255
0016 G4 P2000
0017 G28
0016 G28
```

```
0001 G28
0002 M6 A200
0003 G0 X214 Y270 Z150
0004 G1 X214 Y270 Z15 F100
0005 M6 A40
0006 G1 X214 Y270 Z150 F100
0007 G0 X262 Y45 Z150
0008 G1 X262 Y45 Z35 F100
0009 M6 A255
```

(The line numbers, shown in blue, are correct if you use the program shown in the box. If you have used your own program, then you may need to change them.)

Challenge:

- Make two collars, roughly circular, out of strips of card, 3cm by 25cm. Label them 'Red' and 'Other' and place them on the robot arm mat.
- Extend the above program so that the arm picks up each counter in turn, checks its colour and places it either in the 'Red' area or the 'Other' area.



Worksheet 10

Final challenges!

Robot technology pervades a wide range of industries.

Using robots has a number of benefits. They can:

- work with consistent accuracy for long periods of time without fatigue;
- work in environments too dangerous for humans;
- be re-programmed to perform a wide range of tasks;
- have low running costs.

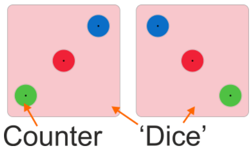
To round off this introduction, here are five challenges, given without any support!

They are not specified in detail but mirror the tasks that robots carry out in industry.

1. Move the blue counters from ‘A’ to ‘V’ and ‘B’ to ‘Y’, the green counters from ‘C’ to ‘U’ and ‘D’ to ‘X’ and the red counters from ‘E’ to ‘W’ and ‘F’ to ‘Z’ and then return them to different locations.

2. Move a red counter on the ‘70’ mark on the mat, a green counter to the ‘90’ mark and a red counter to the ‘110’ mark.

3. Cut two squares of card to represent the bodies of two dice. Place counters as shown in the diagram to represent the dots when the dice are showing two number ‘3’s.



4. Have the gripper trace out a circle in mid-air.

5. Imagine a stream of sugar cubes travelling along a conveyor belt. Your task is to make the robot arm pick up each from the end of the conveyor belt (imaginary) and stack them to make a pile of five sugar cubes on top of each other on the ‘90’ mark.
   (You act as the conveyor belt and position cubes, in turn, so that the arm picks them up. Take care not to damage them by squeezing too tightly with the gripper!)

Teachers' notes

Teacher's notes

About this course

Aims:

This document has two functions:

- as a datasheet on the Allcode Robot Arm;
- as a source for activities.

Target audience:

It is written for students aiming to be technicians in the field of robotics and automation.

At this level, we want technicians to:

- learn about robot arms and robot arm workspaces;
- understand the advantages of working with robots and understand their limitations;
- plan simple robot activities to prepare them for using real commercial robots in the workplace.

We do not expect that these students will be skilled in full programming.

Syllabuses:

The Robot Arm is suitable for use with a wide range of syllabuses including:

BTEC National

Unit 6 - microcontroller programming

BTEC Higher National

Unit 6 - Mechatronics

Unit 15 - Automation, robotics and PLCs

Unit 77 - Industrial robots

Scope:

- Robot construction and geometry
- Workspace planning
- Simple robot arm movement
- Robot arm pendant programming
- Robot arm G code programming

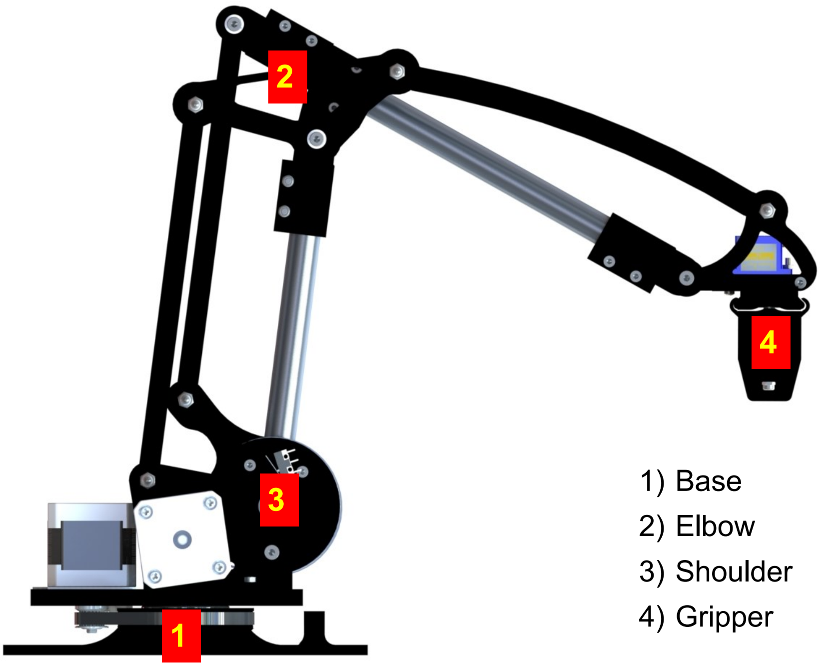
Worksheet	Notes for the Instructor	Time
1 Investigating the workspace	In this worksheet, students investigate the fundamentals of Robot Arm movement. They discover that: • there are three stepper motors that control the geometry of the arm; • moving the arm using the angle controls is really difficult. To get the arm to where you want it to be requires continuous adjustment of the motors. Students may find this frustrating but it teaches them how the arm moves. Students should appreciate the limitations that the arm geometry puts on the workspace and, as a result, the value of planning the workspace of a robot. They also learn that to get to an X, Y, Z location needs more than simple adjustment of the stepper angles. It also requires movement in a correct sequence or the two parts of the arm will interfere with each other. When this happens the arm judders. This is caused by the steppers jumping from one position to the next and will not harm the arm. Certain actions may trigger a 'Comms Failure' error message. If this happens, switch the power supply off and back on. It is a wise precaution to copy any programs onto Notepad as they develop them. Get them to keep an eye on the position of the plastic cover, as it can tend to wander over the mat. Using the scales printed on the mat then causes errors. Don't let the students spend too long on this worksheet. They will want to get on to programming tasks!	**60 mins**
2 Store it	The Robot Arm can store arm positions in internal EEPROM memory. In this worksheet, students are encouraged to use this facility. The Challenge is to store the position of one of the blue counters. This will be used later. Other tasks can also be set.	**60 mins**
3 Sort it	The Robot Arm can distinguish between colours, using a colour sensor. This enables it to sort objects by colour, a technique used widely in a range of industries. (This could be the focus of a student research project.) The 'Colour Sensor' section of the Basic Control app has a rectangle which copies the red, green or blue colour of the counter placed on the sensor, when the 'Sample' button is clicked, and also displays a RGB reading of the strength of the three colour components. The sensor works best in subdued lighting, surrounded by a screen, for example.	**45 mins**

Worksheet	Notes for the Instructor	Time
4 Drive it with the Pendant	The first important point is that the Pendant program cannot run alongside the Basic Control app. The latter needs to be closed for the activities in this worksheet. In the 'LIVE ARM CONTROL' section of the Pendant screen, the 'X+', 'X-' etc. controls change the co-ordinates by 1mm for each click. Although this is the topic in a later worksheet, these controls act in relative mode, i.e. change the existing co-ordinates by 1mm, rather than absolute. The aim of this worksheet is to allow students to familiarise themselves with these controls on the Pendant program.	**30 mins**
5 Starting G code	Now that students are confident in using the arm and its controls, they start to build a basic G code program. This contrasts the behaviour of codes G0 and G1, the first causing rapid movement and the second a speed that is configurable. The task also introduces some of the controls used in creating programs. The challenge is to set up a program to move the arm between specified positions, using these commands and G4, the command to pause. Students need to recognise that the length of the pause is specified in milliseconds. Some may need help in converting between seconds and milliseconds.	**60 mins**
6 Absolute vs relative	As mentioned earlier, there are two ways to control the movement of the arm, using absolute mode and using relative mode. The first is set using the command G90 and the second using G91.The program created in this worksheet uses both in order to contrast them. It is worth spending time with the 'Notepad' application, present on most PCs, to create, copy, modify and transfer G code programs between the application and the Pendant program. This will save time and ensures that, should something unforeseen occur, a program will not be completely lost.	**60 mins**
7 Pick and place	This worksheet creates and examines step by step, a program to pick up and relocate a counter. This task is a common one for robotics in a range of industries. It uses a combination of absolute mode positioning, for fast actions, and relative mode positioning, for the more delicate manoeuvres. The challenge extends the scope of the program to create a triangle of counters, each at a specified position. This kind of exercise can be adapted to relocating different objects to different positions and matched to the abilities of the students in each group.	**60 mins**

Worksheet	Notes for the Instructor	Time
8 Branching - unconditional	The instructor could expand on the topic of branching, unconditional and conditional, as an introduction to this activity. The flowchart shown at the top of the worksheet gives a visual explanation of the two types. Adding the GOTO instruction to the previous program means that the robot arm continues to nudge its way along, never returning to its 'home' position until the 'STOP' button is activated.	**30 mins**
9 Branching - conditional	The 'If... GOTO...' is an example of the more general 'If... then ... else...' command that looks for a particular condition to divert the program flow. In real life, students come across this type of situation - "If it rains, then stay indoors, otherwise ('else...') go to the gym." In this activity, when the colour sensor detects a particular colour, it triggers the GOTO branch, changing the route followed by the program. If the sensor does not detect that colour, the GOTO is ignored and the 'else' aspect of the command occurs - the program continues on its original path. In the challenge, students expand this program into a sorting task, where the Robot Arm divides the counters into 'Red' and 'Other'. The paper collars have to be big enough to hold the four 'Others'. The program could simply drop the counters as opposed to lowering them gently. A further aspect could be added to see which student(s) write the program that completes the task in the shortest time.	**60 mins**
10 Final challenges	Here are five open-ended challenges. They are not precisely specified and are given without any support. Stacking dice could replace the stacking sugar cubes challenge if it is more appropriate (and less tempting to students!) The instructor could allocate different challenges to different groups. The tasks could be timed to add a further element of challenge.	

Reference

Understanding the Robot Arm



1) Base
2) Elbow
3) Shoulder
4) Gripper

There are four joints in the Robot Arm: Base, Shoulder, Elbow and Gripper. In robotics terms there are 'three degrees of freedom'. This relates to the fact that there are three axes in which the Robot Arm can move.

The mechanics of the Arm are designed to maximise the payload (the amount the Arm can lift). This is achieved by placing the heavy motors on the base platform and by using a system of levers and cogs to allow the Arm to move with great precision within its range of motion.

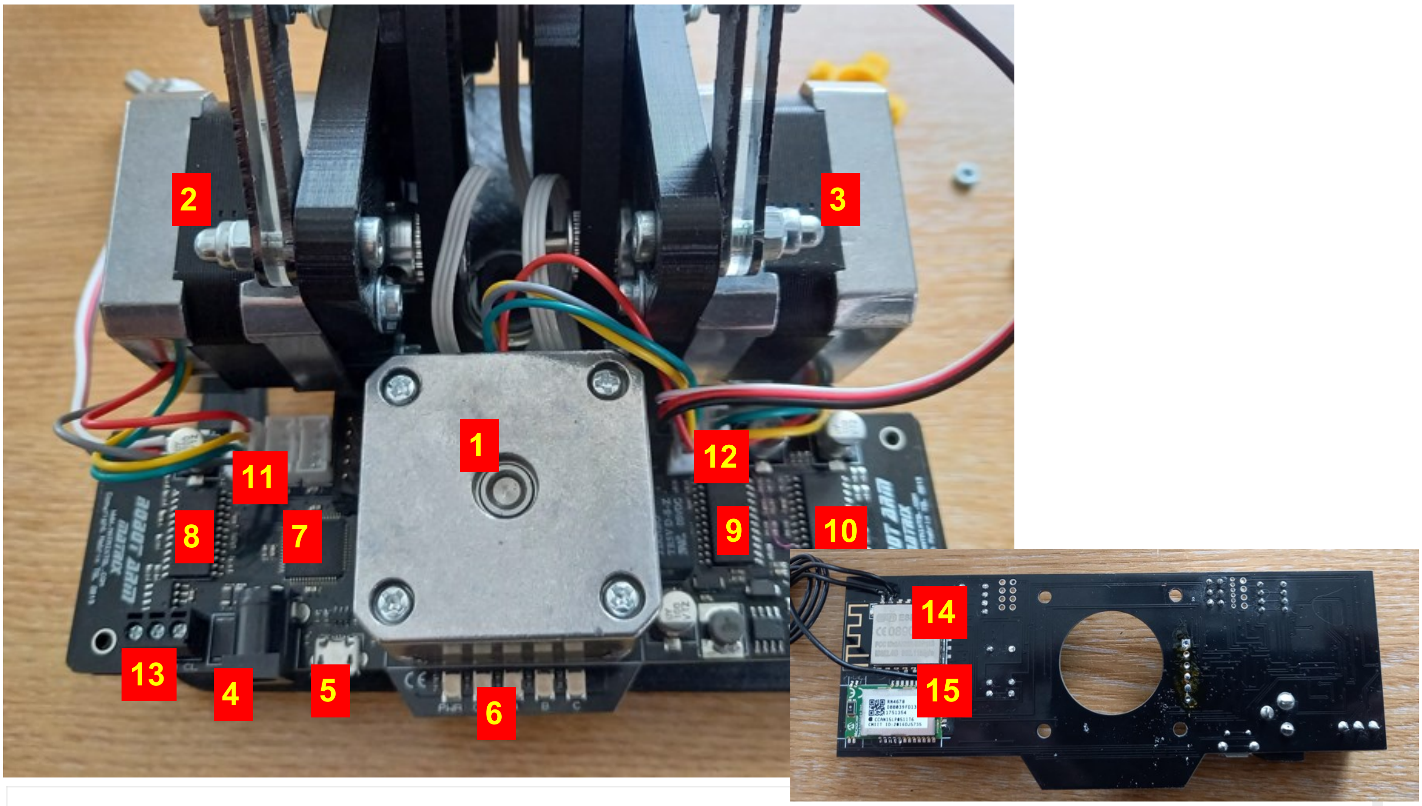
The fundamental actuator for each axis is a stepper motor. Each stepper motor is connected to a drive cog and an axis cog. The drive cogs have 20 teeth. The axis cogs have 102 teeth. This gives us an effective gearbox ratio of 5.1:1.

Each stepper motor takes a single step of 1.8^0 . The driver chips make use of 'microstepping' technology which gives 8 positions within each 1.8^0 increment. This gives an effective step accuracy of 0.044^0 per step.

The gripper consists of a small servo motor and a mechanical actuator. It is possible to set the position of the servo with 255 different positions between fully open and fully closed.

The two main levers are manufactured from aluminium tubing for light weight and strength. The bulk of the mechanics are manufactured from 3D printed plastic.

A single circuit board controls the Robot Arm.



1) Base stepper motor
2) Elbow stepper motor
3) Shoulder stepper motor
4) 2.1mm power jack connector (12V)
5) USB connector
6) LED status lights:
    - PWR: power present
    - COM: communications from host
    - A: Base movement
    - B: Shoulder movement
    - C: Elbow movement
7) Microcontroller
8) Elbow motor driver
9) Base motor driver
10) Shoulder motor driver
11) Left to right: Elbow stepper connector, no fit connector, colour sensor connector, stepper microswitches —used for detecting ‘home’ on all three axes, relay connections.
12) Left to right: Gripper servo connector, Base stepper connector, Shoulder stepper connector.
13) CAN bus terminals: Terminate, High, Low. (120R between T and CH for termination)
14) Wi-fi module
15) Bluetooth module

Understanding the Robot Arm

LED functions on startup

During startup, we have the following pattern after the scrolling animation. (Some steps are too fast to see unless there is a problem.)

1) C Flashing – USB Comms Starting Up
2) B Flashing – CAN Comms Starting Up
3) C + B Flashing – Bluetooth Comms Starting Up
4) A Flashing – WiFi Comms Starting Up
5) A + C Flashing – Loading Stored Calibration Settings

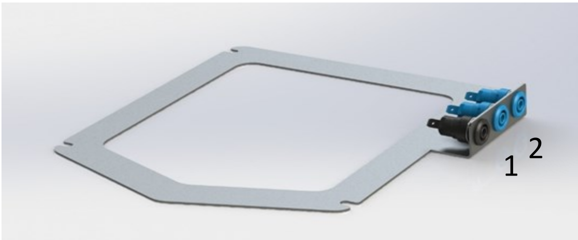
Connections on the expansion port

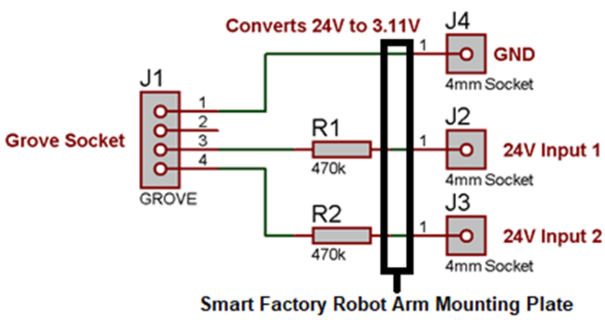
One of the connectors allows the Robot Arm to interface to other programmable devices and to use these inputs as a trigger for various actions.

The connector is a standard Grove connector.

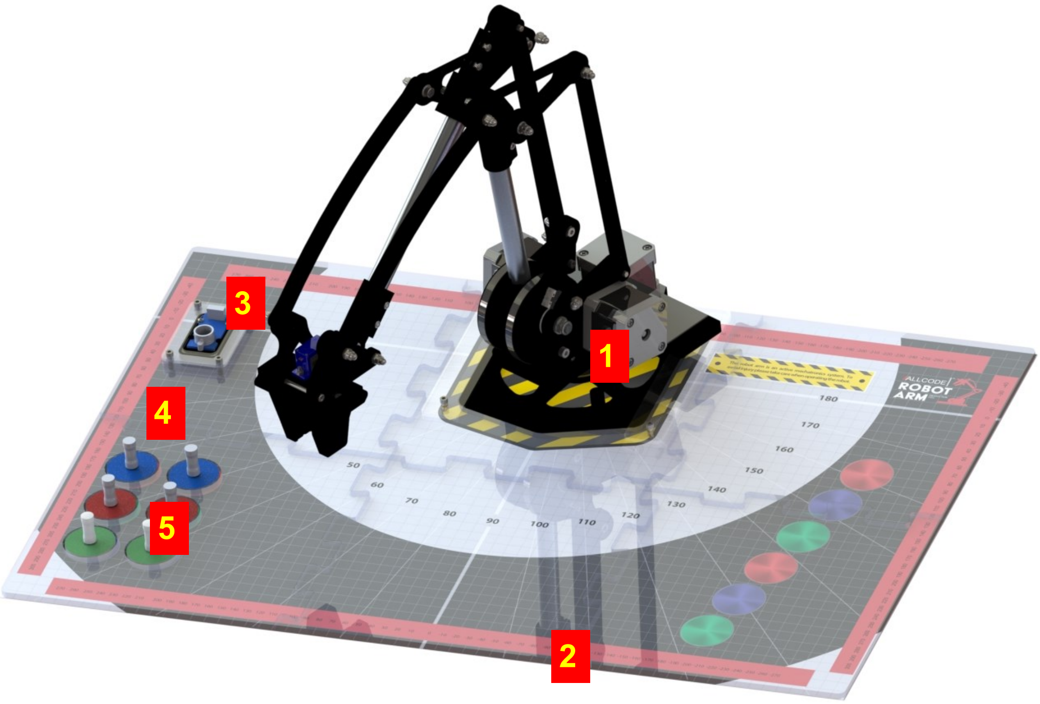
The Grove Connector Pinout is as follows:

1: GND
2:VCC
3: D0 or Input 1
4: D1 or Input 2





Understanding the Robot Arm



What's in the package?

The Robot Arm workspace can be seen above. It consists of:

1) Robot Arm. Supplied fully assembled and tested. Simply plug power supply in to start.
2) Plastic mat cover: 3mm clear acrylic to provide a location template for arm, counters and sensor
3) Colour sensor: returns Red, Green and Blue readings for any object placed on top of it.
4) Counters with spindle that can be picked and placed.
5) Colour work mat placed underneath the acrylic cover.
6) Power supply and micro USB lead.

Assembly and installation instructions

1) A 12V power supply with multiple country mains adaptor plugs is supplied with the arm. Make sure the power supply is configured for 12V output using the small key supplied with the power supply. (See image above right)
2) Insert the power supply into the Robot Arm power jack.
   Use A USB to micro USB cable to connect the Arm to the PC.
3) Install the Robot Arm driver by following the instructions on PC driver installation below.
4) Download the Robot Arm software from our web site. Run the 'Basic Control' software.
   See the page below on using the 'Basic Control' software and make sure that all parts of the Robot Arm function.

PC USB driver and software set up

ROBOT ARM

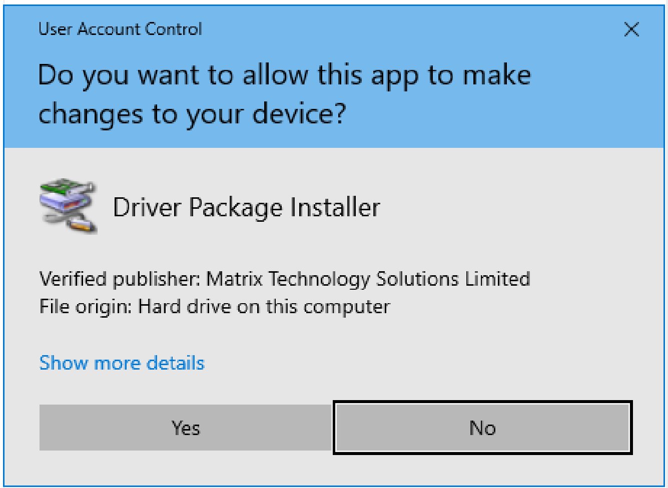
Pendant Programming

Before using the Robot Arm with a PC you will need to install USB driver.

This can be downloaded from www.matrixtsl.com along with the basic software applications for using the Robot Arm.

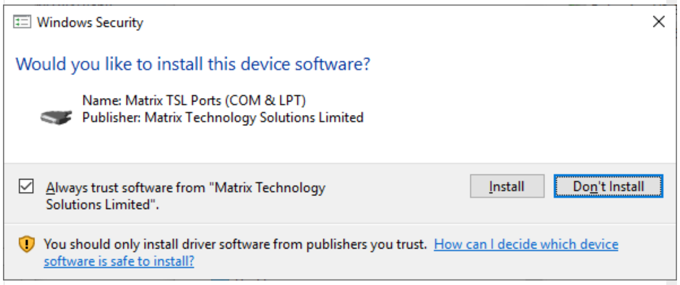
To install the driver you may need administrator permissions.

Run the 'Robot ArmV2 Installation' for your machine.

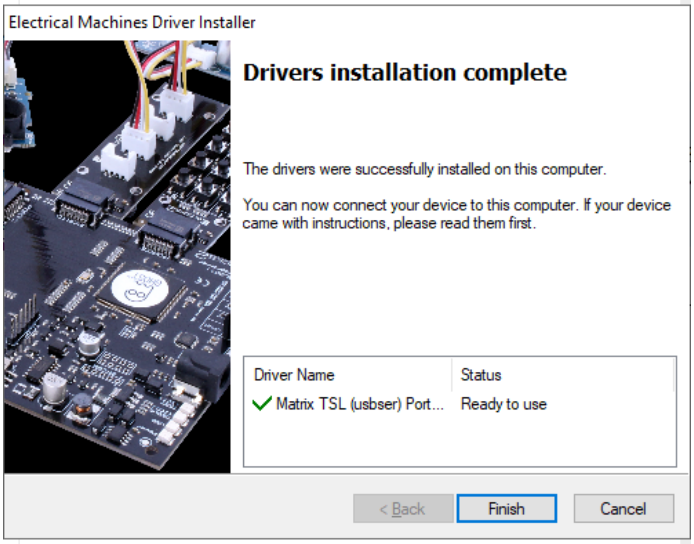


Click **Yes**

Click **Next**



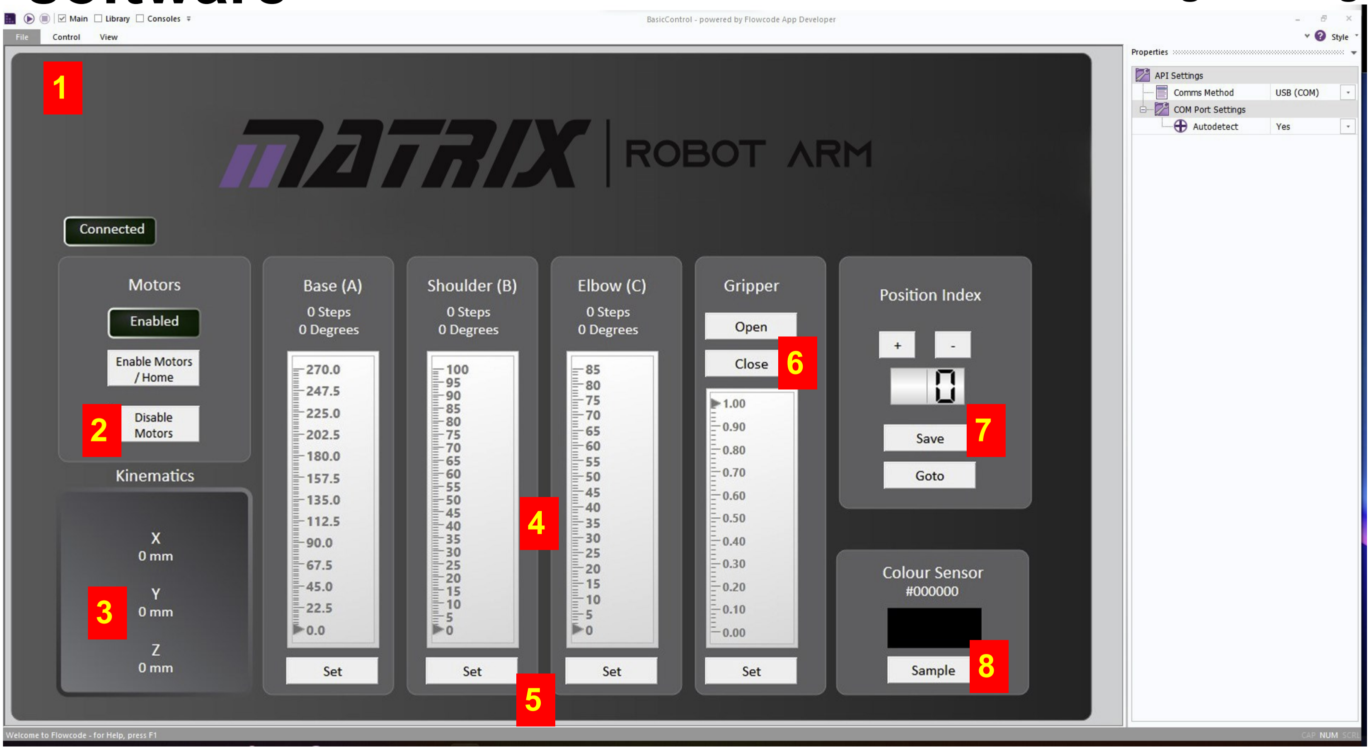
Click **Install**



Click **Finish**.

The USB driver is now installed and ready to be used.

Basic control software



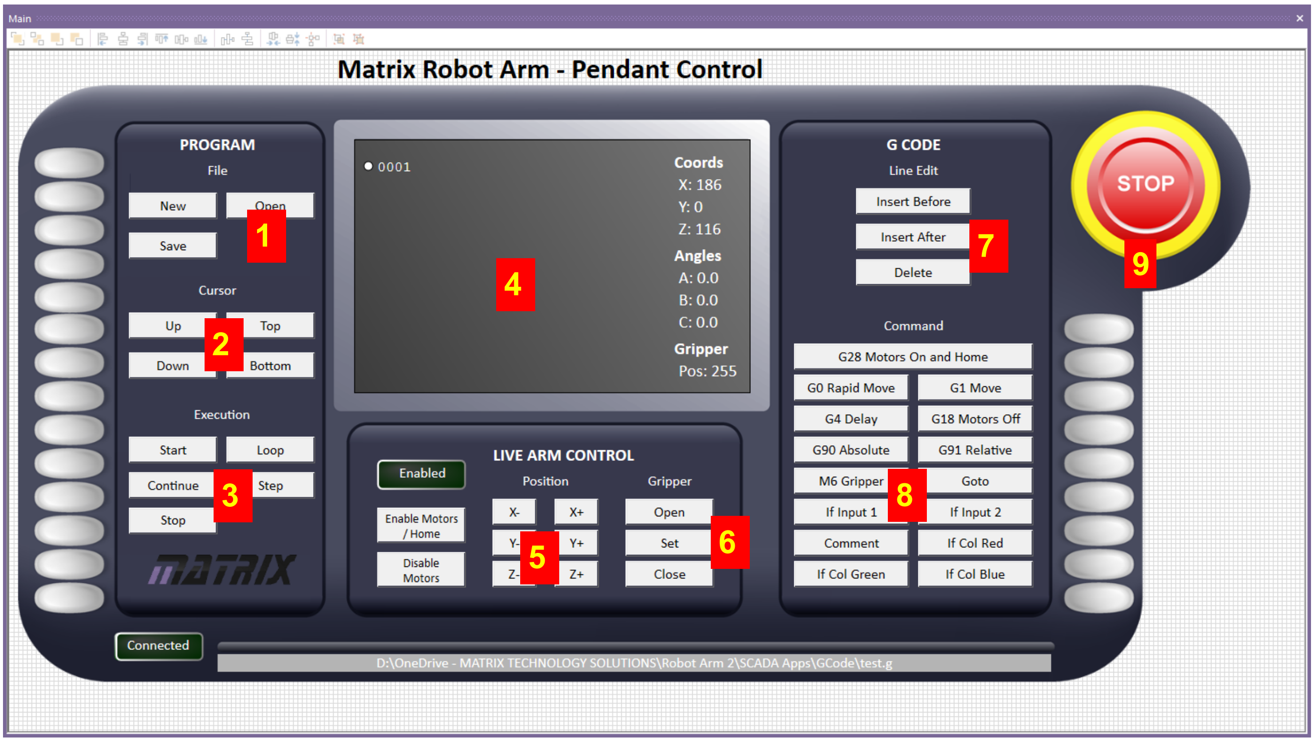
This software is supplied with the Robot Arm package and can be downloaded from our web site. To run the program, simply run 'BasicControl.exe' and click on the PLAY button.

The controls on the software are as follows:

1. PLAY/STOP controls.
2. Enable/Disable motors - 'Home' switch sends arm to the home position.
3. Co-ordinates of current arm position.
4. Sliders for adjusting Base, Shoulder, Elbow and Gripper motors.
5. 'Set' controls for specifying Base, Shoulder, Elbow and Gripper positions.
8. Gripper servo motor controls.
9. Position control - allows users to store up to 30 different positions.
10. Colour sensor controls and display.

When using Wi-Fi, you need to enter the IP address of the arm (see Wi-Fi set up page), the Network Interface (1 to 4: choose the one that gives non-zero results) and the port you set.

Pendant control software



The Pendant program is designed to replicate pendant style robot programming tools used in industry and give students experience of programming the system using this style of programming.

The controls on the software are as follows:

1. Program load and save controls.
2. Cursor controls to move through the program.
3. Program execution controls.
4. Display ‘screen’ for current G code program.
5. X, Y, Z controls for ‘live’ arm operation.
6. Gripper controls.
7. G code line editing controls.
8. G code commands recognized by Pendant Control.
9. ‘STOP’ control - stops current program at the end of the current instruction.

Note that it is possible to save a program, edit it with Notepad, and then re-load it and execute it.

When using Wi-Fi, you need to enter the IP address of the arm (see Wi-Fi set up page), the Network Interface (1 to 4: choose the one that gives non-zero results) and the port you set.

G code reference

You can use the Pendant program either in 'LIVE ARM CONTROL' mode or using a G code program. G codes are used extensively in manufacturing engineering to describe how robots and machines move and act. The Pendant program makes use of a variant of this G code. This can be written directly into the Pendant program or using a simple text editor and then loaded into the Pendant program.

The commands used in the Pendant program are:

G0 - Rapid Move

A movement to an XYZ location at the pre-defined speed of 8000.
Example
G0 X140 Y0 Z10

G1 - Controlled Move

A movement to an XYZ location at a specified speed ranging from 0 (Slow) to 10000 (Fast).
Example: G1 X140 Y0 Z10 F2000

G4 - Delay

A delay specified in milliseconds.
Example: G4 P2000

G18 - Motors Off

Turns off all the motors and leaves the arm in its current X, Y position.
Example: G18

G28 - Motors On and Home

Enables all motors and returns the arm to its 'home' position.
Example: G28

M6 - Gripper

Controls the gripper servo motor - ranges from 0 (Closed) to 255 (Open).
Example: M6 A128

G90 - Absolute

The co-ordinates given in later commands specify the actual destination of the arm
Example: G90
G0 X140 Y0 Z10

G91 - Relative

The co-ordinates given in later commands specify increments in X, Y, Z from the current position of the arm
Example: G91
G1 X10 Y0 Z0 F20

Comment

Comments help the user to understand the program. The ';' tells the software that what follows is a comment and not a command.
Example: ;Move arm to first destination

Goto

Goes to a line number **#** in the program
Example: GOTO 2

If input 1

Goes to a line number **#** in the program if input 1 is high.
Example: IF INPUT1 GOTO 4

If input 2

Goes to a line number **#** in the program if input 2 is high.
Example: IF INPUT2 GOTO 20

If Col Red / If Col Green / If Col Blue

When the colour sensor detects a red / green / blue object, it triggers the 'GOTO' command that follows.
In the following example, sensing red sends the program to line 20.
Example: IF RED GOTO 20

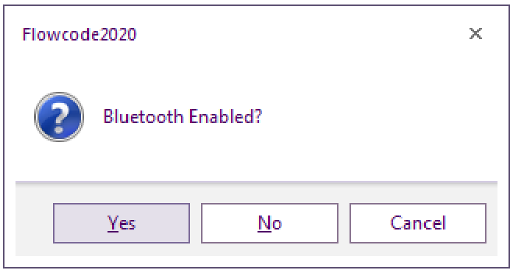
Bluetooth setup on the arm

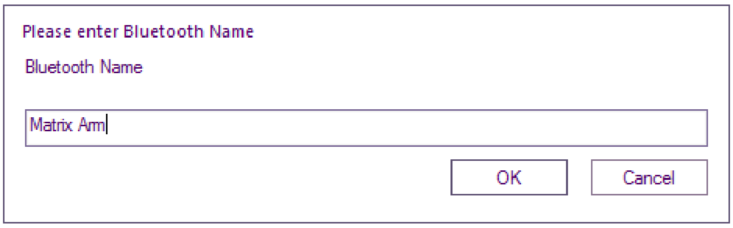
The WIFI and Bluetooth communications are setup using the Configure software.

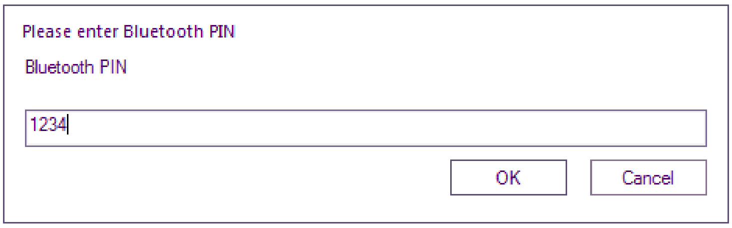
Configuring Bluetooth

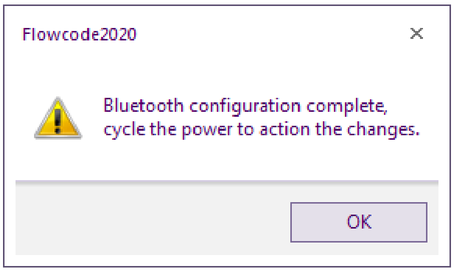
To configure the Bluetooth connection first connect to the robot arm using a USB cable.

1. Run the CONFIGURE.EXE application
2. With communications set to USB click the Play button at the top left of the screen to start the program running.
3. The communications icon should turn to a Green Tick to indicate comms to the arm have been established.
4. Click the Configure Bluetooth button. You will be asked to enable the Bluetooth device, then for a name for the arm you are connected to, then for a PIN.
5. Once you have configured the Bluetooth connection you will need to switch the robot arm off and back on to allow it to setup the Bluetooth module as required.









Any details that you have setup using the configure software will be retained in none volatile memory so you should only have to re-run the configure software if you want to edit the settings.

You will be able to run a number of arms in the same room.

Once you have done this you will need to pair the arm with your computer and set the COM port.

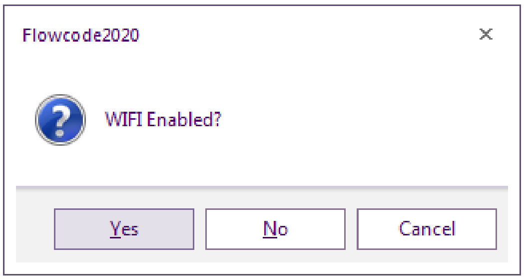
To operate any of the Apps using Bluetooth you will need to set the Comms to Bluetooth and set the Com port.

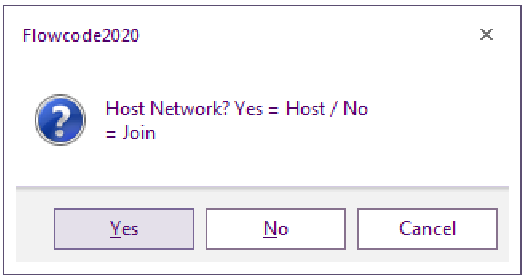
Wi-Fi setup on the arm

To configure the WIFI connection first connect to the robot arm using a USB cable.

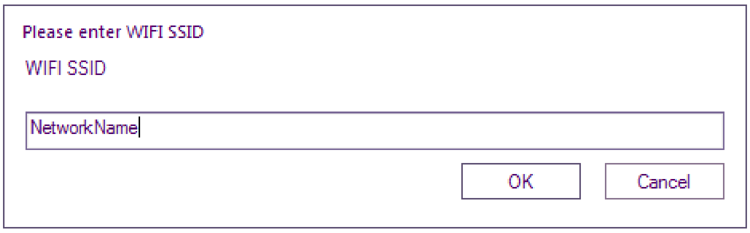
1. Run the CONFIGURE.EXE application
2. Click the Play button at the top left of the screen to start the program running.
3. The communications icon should turn to a Green Tick to indicate comms to the arm have been established.
4. Click on the Configure WiFi button. You will be taken through a number of dialogues as shows on the right.
5. Click the stop button at the top left of the screen.
6. Once you have configured the WIFI connection you will need to switch the robot arm off and back on to allow it to setup the WIFI module as required.
7. Once the robot arm has been switched back on click the Play button at the top left of the screen
8. Click the WIFI Get IP button to collect the IP address assigned to the robot arm by the network

Any details that you have setup using the configure software will be retained in none volatile storage so you should only have to re-run the configure software if you want to edit the settings.
If you want the robot arm to be connected from the internet then you need to enable port forwarding on your router to forward any external requests on the specified port onto the local IP address of the arm. You must then share the external IP address of the router as well as the port.

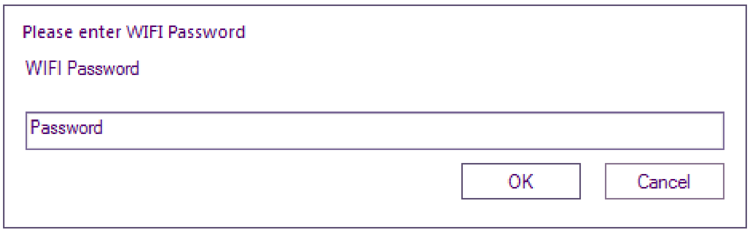




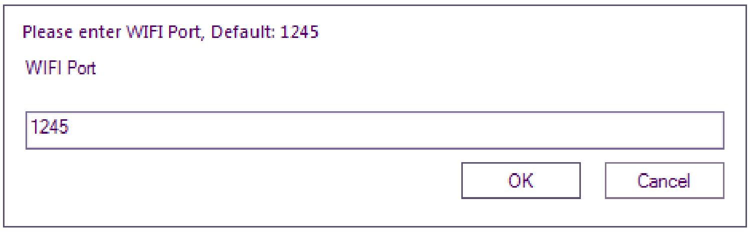
Select NO



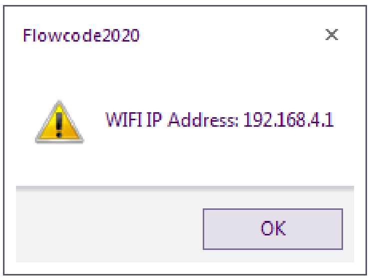
Enter the Wireless network same



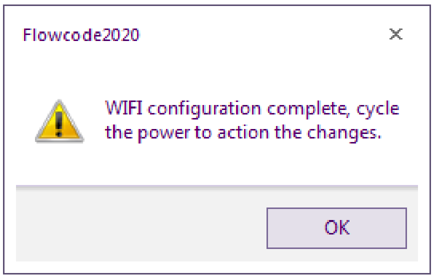
Enter the Wi-Fi password



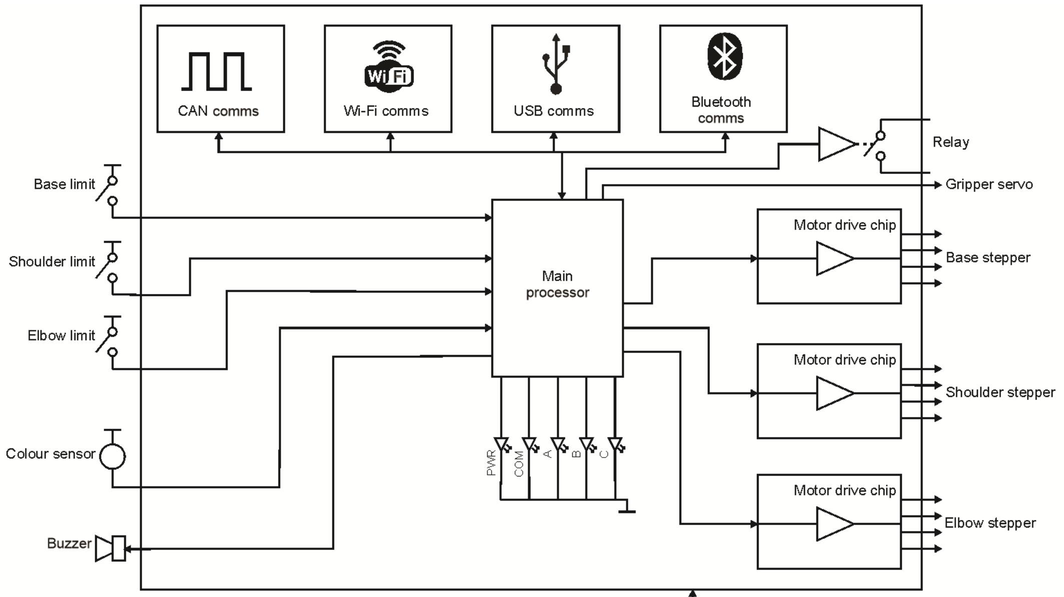
Use default 1245



This is the address set by your router
You will need to enter this into the App.



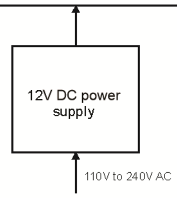
Connections and circuit



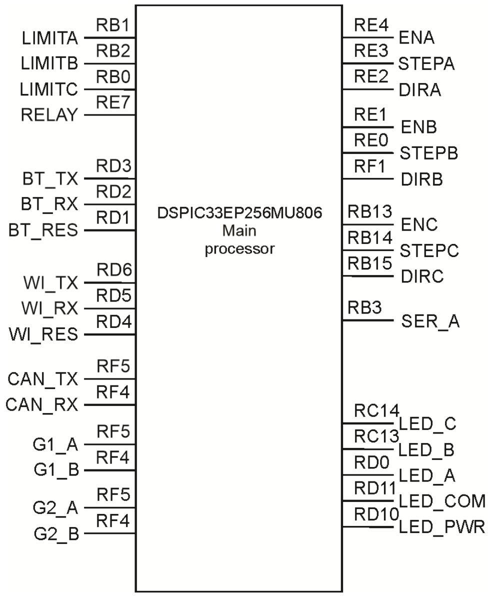
The Bluetooth module is a RN4678.

The Wi-fi module is a ESP12F.

The CAN bus driver is a MCP2551.



To reprogram the microcontroller at a low level, you need to understand which pins of the microcontroller connect to the Robot Arm. You get this data from the diagram.



The microcontroller used is a DSPIC33EP256MU806 dsPIC.

The motor driver outputs feed A3967 driver chips, which convert ‘Enable’, ‘Step’, and ‘Direction’ into pulses for stepper motors. These chips make use of 8:1 microstepping to provide steps of 0.044^{0} per step.

The gripper is controlled by a small servo motor. The output on SER_A has as 20ms pulse for this purpose. The RELAY output is buffered by a transistor and it activates the on-board relay with RELAY dictating whether the output from the relay is +12V or 0V. This is for expansion with a vacuum sucker.

The colour sensor, G1_A and G1_B, and buzzer, G2_A and G2_B, are Grove devices.

The CAN screw terminals are: CH (high), CL (low) and T for termination.

T has a 120 ohm resistor connected to ground and can be used for termination if needed

The power connector is 2.1mm power jack (centre positive).

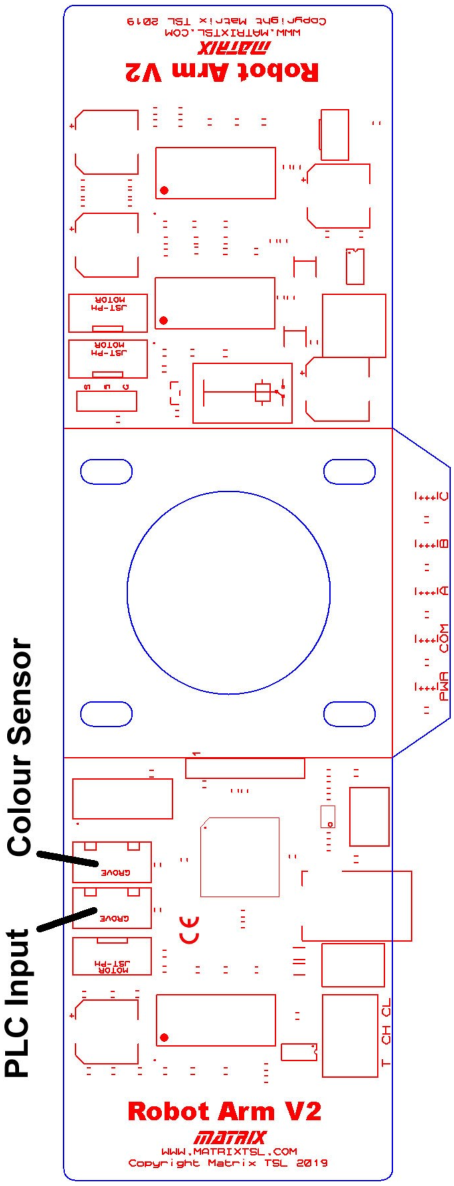
Datasheets on all these items are available on the internet.

To understand how to transfer your program to the microcontroller, please read the section on ‘Reinstalling the API’ in the Reference section.

This shows you how to send a hex program to the controller.

Layout changed

Connections and circuit



Layout changed

Calibration

There are several steps in the calibration:

Entering Calibration

Run the 'Open Configure' program.

Click the 'Go' button and the Communications light should turn green

Click the 'Set Kinematics Lengths' button and enter the lengths you record during calibration

Click the 'Enable Motors / 'Home' button and the arm should go to the 90^0 position on the mat at around coordinates 0, 300, 5.

Calibrating Base Angle

If the centre of the gripper is off the 90 line, then work out the angle difference.

Click the 'Set Motor Base Angle' button and enter the angle difference.

A little goes a long way - try increments of 0.2.

The arm should move to show the new position.

Once you have centred the arm on the line use the 'Motor Enable' and 'Home' button to check that the value is stored and the setting is correct.

(Note - from the initial start angle you cannot go back more then **-2.0**^{0}. You may need to back off the base motor limit switch if you cannot go back far enough.)

Calibrating Shoulder and Elbow

Now that the gripper is centred over the X0 axis line, it's time to calibrate the shoulder and elbow.

Click the 'Y-/Y+' and 'Z-/Z+' buttons until the centre of the gripper is over the Y300 line and 5mm from the surface of the jigsaw plastic mat. A 5mm spacer can be used to aid accuracy

Once the gripper is in position, click the 'Store Calibration' button to calculate and store the calibration.

Click the 'Position 2' button

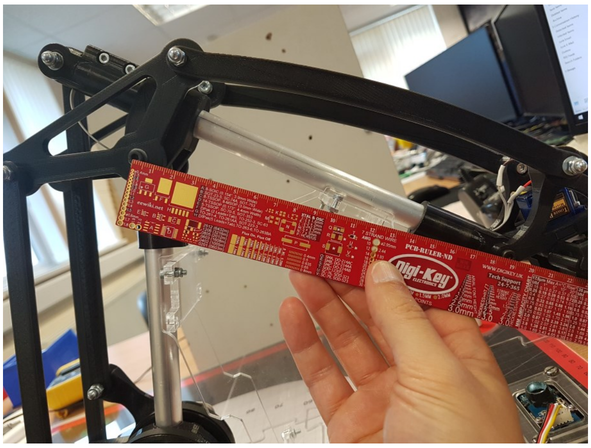
Using a ruler measure, the offset from the central line - left of the line is positive, right of the line is negative.

Click the 'Set Drift' button and enter the offset in mm.

Click the 'Position 2' button again to check that we are now on the central line.

Close the Configure application

Measure and record the length of the shoulder - centre of main pivot to centre of bolt. **The default is 176.0mm**



Measure and record the length of the elbow - centre of bolt to centre of bolt. **The default is 189.0mm**

Layout changed

Support and fault-finding

1) If you find any faults, then it will help us and yourselves to understand a little more about the nature of the faults.

2) On 'Any App', you select VIEW….CONSOLE.
This brings up the Flowcode console for the arm.
Select the Robot Arm tab.
This shows the messages passed between the API and the PC and effectively gives the status of the commands being executed.

Communications between the arm and the Wi-fi or Bluetooth system sometimes presents issues.
The API has various pauses in the code to allow for wireless communications.

On power up, the arm looks for communications systems. In this phase of power up, the LEDs take on a different meaning:

PWR: Power
COM: when flashing this shows the arm is in Bootloader mode.
- A: Most significant bit
- B: Middle bit
- C: Least significant bit

Where flashing:
- 1 = USB
- 2 = CAN
- 3 = Looking for Bluetooth connection
- 4 = Wi-Fi

Change log

14 09 23	First release
24 07 24	new software update - many changes.