SIEMENS
Ingenuity for life

TEACHERS GUIDE

INTRODUCTION

The Siemens Driverless Car Challenge aims to supplement robotics education in Computing, and Design & Technology lessons at KS3. Contextualising robotics as part of the future drive towards autonomous vehicle technology, this resource consists of three classroom activities used in conjunction with the micro:bit programming platform. In this guide you will find the necessary notes to deliver introductory robotics learning in a variety of contexts, with or without having vehicles of your own.

Additionally, in this collection of resources you will find support materials for our interactive challenge, Siemens Self-Driving Challenge (http://www.siemens.co.uk/education/en/students/interactives.htm) that can be used in conjunction with the micro:bit focussed activities or as a standalone activity. An additional Student Activity Sheet has been included to support this interactive resource.

Resource Contents:

- Teachers Guide with Curriculum notes
- Three classroom activity sheets for use with the micro:bit programming environment
- An introduction sheet for students, including an introduction to robot buggies and small sized map
- Large sized printable Auto City Map

LEARNING OBJECTIVES

- To investigate driverless car technology and its impact
- To apply computing knowledge to control motors using a micro:bit

DRIVERLESS CARS IN THE CLASSROOM

Robotics and wireless communication technologies are changing the world around us; from the smartphones to the Internet of Things, lots of everyday objects are becoming increasingly interconnected through ever growing networks of information infrastructure. This technological revolution has put driverless cars on the horizon, with the rate of current development suggesting that, within 20 years, most cars will be steered by on-board computers, with drivers instead becoming 'passengers' in autonomously driven vehicles.

In these activities we have scaled down the challenge of driverless cars to the programming of robot buggies, following routes around Auto City, our own driverless car-testing zone. Students will have the opportunity to unscramble navigation codes within the micro:bit programming environment, mimicking the way code will be used to control and steer vehicles. In the final activity, students will also use the radio function of micro:bit coding, simulating the way that real world data will be sent to actual driverless cars, with students taking on the role of intelligent traffic control.

SIEMENS
Ingenuity for life

TEACHERS GUIDE

SIEMEN'S DRIVERLESS CAR ACTIVITY SHEETS

The Siemens Driverless Car activity sheets each follow one route, A, B or C, around the fictional Auto City. Each sheet increases the programming complexity, whilst Route C incorporates radio and accelerometer inputs. The questions of each sheet also incorporate additional related topics as follows:

Route	Programming Skills	Question Topics
Route A	Simple linear route	Motors, Sensors & Communication
Route B	Complex linear route	ADAS technology
Route C	Radio control, accelerometer	Radio & RADAR

Each sheet requires access to the micro:bit Blocks programming environment, and Students will require use of Internet research to answer some questions. Students can work in groups or individually, depending on hardware availability, group size or ability.

The Auto City Map has been provided for testing each code, however you will require your own micro:bit robots. The *Student Introduction* sheet should provide adequate information for students to complete the Route A & B activities as a theoretical coding exercise - you can provide the answer sheets included here for students to bug-check their own codes.

You may wish to use the Auto-City map for your own challenges. You could use markers to set your own start and end points, or challenge students to programme the fastest route between two points. For additional extension ideas, see section *Extension Ideas*

Though the sheets are designed to be used sequentially as a mini-scheme of work, you may wish to pick and choose activities and use the resources flexibly. Route C is significantly more complex, as is it's programming, and you may wish to use this only with higher ability students. Alternatively, you can build towards this over two or three lessons with younger students. These lessons could form a superb introduction to students designing their own micro:bit devices, as by Route C, students will have learned many inbuilt functions of the micro:bit.

Should you wish to develop this into a longer scheme of work, you may wish to include our Siemens Self-driving Car Challenge, Siemens Self Driving Car, which would make an excellent introductory or starter activity to these resources. http://siemens.zincmediadev.com/selfdrivingchallenge

LESSON PLANNING EXAMPLE

How you use these resources will very much depend on your groups previous learning along with hardware availability, however we have included a lesson plan below incorporating routes A and B to show how these might be used in a KS3 Design & technology lesson of 21 students. The programming activities for routes A & B should last approximately 15-20 minutes each, this may last longer if students have not used micro:bit before. This lesson is for one hour - but can be split into two half hour lessons.

EQUIPMENT

6 x Robot Buggies	1 x Auto City Map
6 x Micro:bits	21 x Introduction Sheets
6 x Micro USB Connectors	21 x Route A Sheet
6 x Available computers with internet access	21 x Route B Sheet

SIEMENS
Ingenuity for life

TEACHERS GUIDE

PREPARATION

- Ensure that all robots have fully charged batteries
- Check that your robots motors match the codes
- Prepare a large area for the map
- Group students into groups of three or four

SEQUENCE

- **5 minutes - Starter -** Students are to read the Route A activity sheet, and begin to answer the questions individually.
- **5 Minutes - Feedback Answers -** First ask students "What is a driverless car"? Take suggestions on the board, then go through the question answers.
- **5-10 Minutes - Introduce the Programming Activities -** Show students the map and introduce the robot buggies. Explain briefly how the microbit controls the motors. If your students have not used micro:bits before, show them how to load and use code on them. Frame the programming challenge as problem solving activity, a bit like putting a recipe in the right order.
- **30-35 mins - Practical Programming -** Students have 35 Minutes to make both codes and test them in their groups. If they have finished both sheets, they can answer the questions on the Route B activity sheet. Including a further demo if students need it. Students should print screen their finished codes.
- **5 Mins - Plenary -** Consider questions which evaluate what went well about the activity and what went wrong. Did the robots follow their code? How could we improve the robots to make them safer? Do they feel more confident about coding than they did at the beginning?
- **5 Mins - Pack Away -** check that all robots have been returned, are off and are complete.

DIFFERENTIATION

- Easier - Mark the start and end of the codes, so that students just have to put the middle parts in order or Give a complete code for Route A, and allow students to unscramble Route C instead.
- Harder - Students could program Route B from scratch without the scaffolding of the unscrambled parts.

EXTENSIONS

- Challenge students to write new codes between points of your choosing. Challenge Students to code Route C.

A NOTE ON USING YOUR OWN ROBOTS

The robots that we have used to design this activity are widely available to purchase micro:bit buggy kits, using two small, geared DC motors and large wheels with rubber tires, with a front bearing for turning (See diagram in the *Student Information Sheet*). The market is now quite wide and these activities are compatible with many other kit robots, which you can choose according to your own budget and needs.

Due to the variation in design however, you may have to adjust the motor power across the activities to ensure that your robots will follow the routes correctly using the same length of pause between instructions. You will notice that we have used the same motor power, of '310' for each manoeuvre; you will only have to adapt this number for your own robots specifications. You can do this using trial and error over Route A. You may find that each of your robots is slightly different - in which case we recommend that your write the correct motor speed on the robots themselves, and students can drop this number into their code.

In addition to this, on testing these activities we found a slight fluctuation in the power requirements of the robots as the batteries became more depleted over a days use, requiring adjustment of the power to 320. Should you be using your robots multiple times over a school day, we recommend recharging the batteries between classes if possible.

SIEMENS
Ingenuity for life

TEACHERS GUIDE

OTHER ROBOTS AND PROGRAMMING

For our robots, we used the Robo:Bit Plugin, which uses Pin 0 (PWM) and Pin 8 (Direction) for the left motor and Pin 1 (PWM) and Pin 12 (Direction) for the right motor. You may require a different plugin for your own robo buggies, and you may find recommendations from the manufacturers website or documentation. Most of these are in a similar format to the ones on your activity sheets.

CRUMBLE ROBOTS

Though these activities have been designed for Micro:Bit, you could easily adapt these codes for your Crumble robots. Crumble uses a similar block style coding language, and has built in motor outputs. Crumble based robo buggies will not be suitable for activities related to route C however, as it has no radio or accelerometer functions at time of writing.

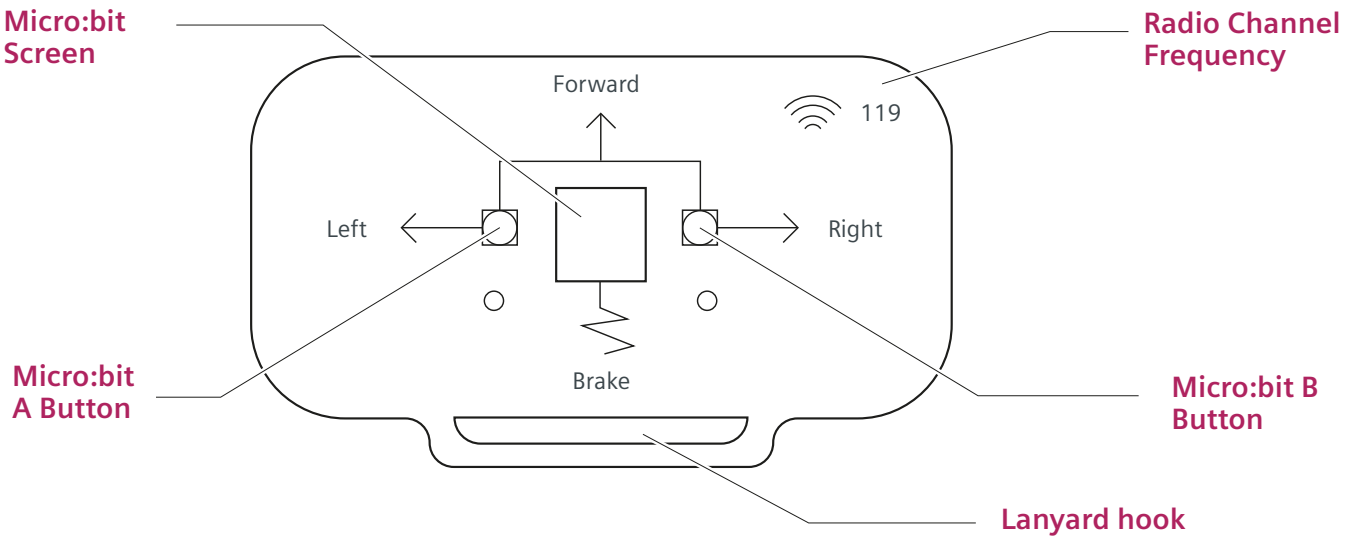
ROUTE C AND REMOTE CONTROL

Route C and its activities contain more complex coding and open questions. This activity should really take a whole lesson. Students may wish to experiment with designing a handheld controller to contain the controller micro:bit as an extension (a great opportunity for ergonomics and anthropometrics), or you can just use a second micro:bit with a battery pack as a rudimentary handheld. Be certain to ensure different group channels are used, and that both controller and buggy are set to the same number. The limited controls will be a fun challenge for students' driving skills, and students should experiment with motor speed for optimum controllability.

We have included a .dxf file, compatible with 2D design for you to laser cut your own controller. A labelled diagram is included below, as well as a table of control functions.

CONTROLLER INSTRUCTIONS:

A Button	Rotate left
B Button	Rotate right
A + B Button	Drive forwards
Shake	Brake



SIEMENS
Ingenuity for life

TEACHERS GUIDE

EXTENSION ACTIVITIES

The scope for this topic is ever evolving, and is only limited by your own imagination and hardware.
You could extend these topics by introducing sensors, using the micro:bits on-board accelerometer, or compass.
In addition, you could add sonar, bump switches or line following to your robots, adapting the map accordingly.

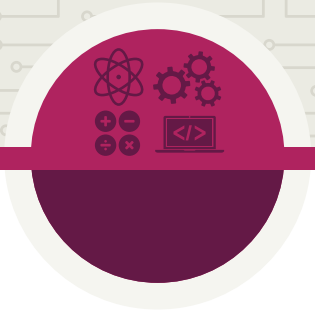
The addition of sensors opens up many curriculum relevant lines of questioning about driverless car design; a suitable activity would be to equip each robot in a class set with a different type of sensor, and then get students to evaluate their effectiveness in the context of passenger safety. Interesting discussions can be had, for example, over bump switches vs sonar; bump switches may be more reliable at stopping a buggy, but would it be too late to prevent injury?

Should you wish to turn these resources into a scheme of work, you could consider additional design challenges: these could include designing a handheld controller for your remote control micro:bit, to packaging designs for the robot kit.

Students could also program more competitively. You could ask students to join two points via the quickest route, and time students attempts.

SIEMENS
Ingenuity for life

TEACHERS GUIDE



CURRICULUM LINKS

These objectives are applicable to and incorporate aspects of the following areas of the National Curriculum:

KS3 DESIGN & TECHNOLOGY

Evaluate

- Analyse the work of past and present professionals and others to develop and broaden their understanding
- Investigate new and emerging technologies
- Understand developments in design and technology, its impact on individuals, society and the environment, and the responsibilities of designers, engineers and technologists

TECHNICAL KNOWLEDGE

- Understand how more advanced electrical and electronic systems can be powered and used in their products [for example, circuits with heat, light, sound and movement as inputs and outputs]
- Apply computing and use electronics to embed intelligence in products that respond to inputs [for example, sensors], and control outputs [for example, actuators], using programmable components [for example, microcontrollers].

KS3 COMPUTING

Subject Content

- Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions
- Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems

SCOTLAND CURRICULUM FOR EXCELLENCE

Computing Science

- I understand that sequences of instructions are used to control computing technology. (TCH 0-14a)
- I can experiment with and identify uses of a range of computing technology in the world around me. (TCH 0-14b)
- I can develop a sequence of instructions and run them using programmable devices or equivalent (TCH 0-15a)

SIEMENS
Ingenuity for life

TEACHERS GUIDE

TROUBLESHOOTING

- My robo buggies are not following the routes
  - Your robots may have a different specification from those used to design the resource. Adjust the motor power in the micro:bit program until it follows it the routes correctly. This may take some experimentation.
- My robo buggies do not go in a straight line
  - Education kits often use cheap motors with poor tolerances. This can mean that your motors may be 'unmatched' and so respond differently to the same amount of power. Try to find good pairs amongst your classroom kits. You can do this by 'characterising' the motors – measuring their rpm at different power settings, and choosing like pairs.
  - Check that your robots are equally balanced, as more weight on one side can cause your robo buggy to drift.
  - Check that your driving surface is smooth. Carpet causes drift along the weft of the material.
  - In extreme cases you can adjust the power supply to each motor in the program, compensating for the lack of balance.
  - If you really are struggling to tune your robots, use them off of the map – students will still see that their robots are following a 'route' compared with their smaller map in the introduction hand-out.
- How do you load programs onto a microbit?
  - Once you have completed your program, press the download button on the micro:bit website. The program should download to your downloads folder. Plug in your micro:bit, and drag and drop the file to the micro:bit icon on your drives menu, just as you might drag a file to a USB. If the micro:bit is in the robo buggy, ensure that it is switched off, as it could run away before you have unplugged it!
- I'm a technology teacher and am unfamiliar with programming – where do I start?
  - Programs are just like writing recipes – they are a list of instructions. The programs in these activities are relatively simple. You are really just giving instructions to two motors, controlling speed, direction and duration of command.
  - If in doubt, speak to your colleagues in Computing – they will (hopefully!) be happy to help guide you as they will use the very similar Scratch programming environment.
  - Don't rush into this as a classroom topic. Ensure that you are fully familiar with how the robo buggies work. Perhaps trial your teaching in a lunchtime club to find the common misconceptions that students may have about robotics.
- My school cannot afford robo buggies or robots, but I don't want my students to miss out.
  - You can still use these activities without robo buggies. The Micro:bit programming environment works without a micro:bit and is free. Get students to print screen their codes for you to mark (instead of testing) or get students to mark their codes themselves!
- I cannot find the 'motor' block in the microbit programming environment
  - You must install the advanced plugin, 'Bit Bot'. Choose the 'Advanced' function, search and install
- Our computers are offline
  - Do not fear - the micro:bit code editor works offline, as long as it is in your cache.
- Why doesn't the robot stop when I program it to 'pause'?
  - Pauses are lengths of time between instructions, not temporary stops. Use Pauses to indicate how long you would like an instruction to be carried out for before the next instruction.

USEFUL LINKS

Micro:bit Code editor	makecode.microbit.org/
Siemens Education	www.siemens.co.uk/education/en/
Siemens Driverless Cars	https://www.siemens.com/innovation/en/home/pictures-of-the-future/digitalization-and-software/autonomous-systems-selfdriving-vehicles.html

SIEMENS
Ingenuity for life

TEACHERS GUIDE

ANSWERS TO ACTIVITIES

Siemens Driverless Car Challenge: Route A

Student Activity 1: Auto City Route A

Activity 2: Answers

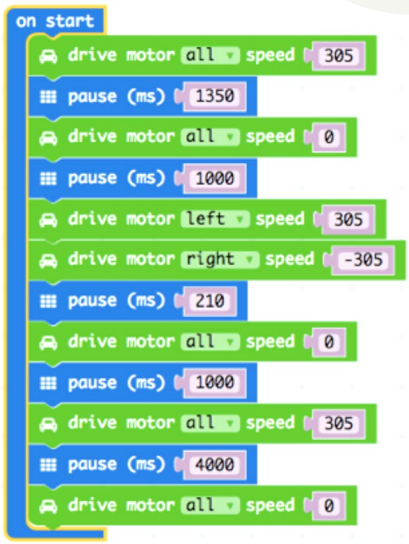
a) wifi, radio, satellites

b) traffic, roadworks

c) sensors

d)

Buggy Travel	Left Motor	Right Motor
Forward	Forward	Forward
Turn Left	Backward	Forward
Turn Right	Forward	Backward



Siemens Driverless Car Challenge: Route B

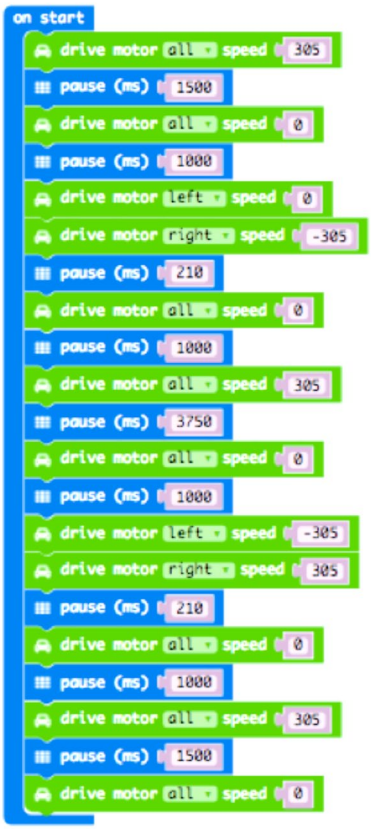
Student Activity 1: Auto City Route B

Activity 2: Answers

a) Advanced Driver Asssistance Systems are digital systems that help drivers, increasing safety.

b)

ADAS System	How it makes driving safer
GPS Navigation	Guides the driver without the distraction of maps, monitors speed and guides away from traffic.
Parking Sensor	Prevents collision, allows drivers to park safely
Rain Sensor	Automatically adjusts windscreen wipers and brake systems to compensate for rainy conditions. Improves visibility and braking.
Automatic braking	Helps drivers to brake and avoid collisions



Siemens Driverless Car Challenge: Route B

Student Activity 1: Auto City Route B

Students should be assessed on whether their code allows them to adequately navigate the route. This should be assessed on accurate transcription of the code and driving skill.

SIEMENS
Ingenuity for life

TEACHERS GUIDE

Activity 2: Answers

a) Radar Detection and Ranging

b) Radar detects at a distance - by the time a car is touching an obstacle it is too late to prevent collision or injury

c) Accept any reasonable answer including the following:

- Increased safety through ADAS technology
- Better management of traffic
- More relaxing travelling experience
- Greater accessibility for the disabled and older people
- Lower emissions from more efficient journeys
- Reduced accidents from human error

Table of Robo Buggy Speeds and Service Notes

Use this table as a central log to note down Robo Buggy motor speeds and any service notes. This will help when maintaining the buggies, for example in re-matching motors.

Robo Buggy	Speed	Notes on Character
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		

SIEMENS
Ingenuity for life

TEACHERS GUIDE

APPENDIX

PURCHASING YOUR OWN ROBO BUGGIES

This resource was created using Robo:Bit Mk2 Buggy for the BBC Micro:Bit available from 4tronix. Should you wish to purchase a classroom set for use with these resources, we would recommend a set of six to eight buggies for a class of twenty-four students, allowing for three to four students per group, or up to twelve robots for working in pairs, should budgets allow.

It is worthwhile taking the time to match the motors of your robots. The best approach to this is to mark each motor with a serial number or letter. Test each pair of motors together in an excel spread sheet, driving the test robot along a straight line. Rate each pairing in a spreadsheet or table, and choose the most well-matched pairings for your class set. It is time consuming though worth the effort - you could even turn this into a classroom activity!

Note that motors will differ in overall efficiency, acceleration and stopping time. An alternative is to get students to compensate for mismatched motors in their codes.