Physical Computing/Computer Control KS1 to KS3

This document is an outline scheme of work that is designed to provide guidance to pr KS1 and KS2 teachers on the teaching of physical computing and computer control.

It has been mapped to the new programmes of study for Computing and Design & Technology.

What is physical computing?

In simple terms physical computing is the use of a computer to build physical systems that can interact with the real world. Many of the systems are designed to perform very specific sensing and control tasks. The control system built into a modern washing machine is a good example. The temperature and wash cycle can be set with a dial. The system can sense when the drum is full of water and the temperature of the water. It will automatically start and keep track of the wash cycle, the number of rinses and will stop the machine and unlock the door when the wash is finished.

View this Prezis for a quick overview.

Why should I teach physical computing?

Physical computing is included in the Computing programmes of study (POS) as an essential element of the subject. A stated aim of the Design and Technology POS is that all pupils should develop the creative, technical and practical expertise needed to participate successfully in an increasingly technological world. One of the most important developments in information technology in the next five to ten years will be the expanding 'Internet of things'. One definition of the Internet of things is a world where physical objects are seamlessly integrated into the information network, and where physical objects can become active participants in business, service and domestic processes. It is estimated that by 2020 there will be 50 billion devices connected to the internet.

The UK needs large numbers of people with the skills to design, manufacture, install and manage these devices. If this country is to be a global leader in the creation of the internet of things we must make the most of the opportunities presented within the Computing POS to educate our children from an early age. Children should leave primary school with an awareness that computers are involved in the control of a vast array of everyday devices as well as some very specific and highly technical applications. They should also be given the experience of controlling models through the connection of physical objects to a computer via an interface. Ideally, children should be introduced to control in Foundation Stage via programmable toys such as the Roamer, BeeBot or Big Track.

In primary schools an ideal context for teaching sensing and control can be provided by Design & Technology. There is a growing trend to combine Computing and D&T as part of a STEM (science, technology, engineering and maths) curriculum. Children should be challenged to design and make objects or models with inputs and outputs that are controlled by a computer. Through such projects the children will understand simple *input* > *process* > *output* control systems as well as the need for *feedback* to maintain a stable state (such as central heating temperature).

Some children will gain a quick grasp of control technology and will make rapid progress through the work so a considerable amount of extension material needs to be built into the teaching schemes. In the interests of a smooth transition from primary to secondary it is essential that primary schools have an understanding of the knowledge and skills that their children will need in the future. It is also important for secondary schools to know what they can expect of their new Year 7 children. Ideally primary schools will work in partnership with their secondary schools to develop their schemes of work for physical computing. Most secondary schools will have a specialist computing teacher and some may even be willing to lend hardware to their feeder primary schools.

What are the issues?

Physical computing in the form of sensing and control is not new of course. It has been taught in schools since the early 1980s. Control was included in the old ICT programme of study. Yet it is evident from Ofsted reports that control has not been well taught within a substantial proportion of UK primary and secondary schools. There are a number of reasons for this to do with the cost of resources, a paucity of teachers with the confidence to teach control, a perception that this is very technical and therefore a difficult topic to teach. Another issue has been the endless hardware and software upgrade cycle that renders expensive resources obsolete. Set against these genuine hurdles should be the rationale for teaching control outlined above and the fact that physical computing is one of the most engaging, creative and enjoyable things that children can do with a computer.

Miss Smith's Tale - view this Prezis for a summary of the issues.

Is this topic very technical?

The electrical components and all the wiring to connect them can be a little daunting for those without a technical background. In fact the circuitry required for computer control is really very simple although it helps if the children have already done some work with simple series circuits before they begin control.

As the components cannot be connected directly to a computer an **interface** is needed to enable the computer to interact with the components.

For sensing, a component is connected across two **input** terminals on an interface. The component acts like a switch and in many cases *is* a switch of some sort. When the component connects the two terminals on the interface it tells the computer that the input is ON.

For **output** the interface is acting as the battery (the voltage, typically 6V, is determined by the power supply used (see the manufacturer's instructions). The red output terminal can be thought of as the positive side of a battery and the black can be thought of as the negative side of a battery. When the computer sets an output to ON, the output terminals will produce a voltage that will power any component connected between the red and the black output terminals.

Slightly more complicated are motor control outputs. Not only can these be switched on and off but positive and negative can be swapped round to make the motor turn in a forward or reverse direction. Some control software also changes the power to the motor to make it speed up and slow down.

What resources are needed?

For a solution, that will allow the children to write control programs in Scratch, a control workstation based on a Raspberry Pi

computer and gPiO interface is the only option available. A special version of Scratch. called **Scratch GPIO**, has been created for school control projects.

Some control interfaces and software allow the use of analogue inputs. These will measure an input voltage which will depend on the state of the input device. For example, if a thermistor (temperature sensor) is connected to the interface, the higher the temperature the higher the voltage sent to the computer. With the software it may be possible to change this data into a temperature value for data logging or to give an output only if a certain temperature value has been reached. The PicoBoard is a device that, when connected to the Raspberry Pi, enables the use of analogue inputs.

This Raspberry Pi / gPiO / Scratch GPIO combination affords a number of major advantages,

Firstly it is cheap (about £200) and it will be sustainable. You will not be forced to pay for upgrades on a cyclical basis. Wires can be connected to the gPiO using quick fit solderless 4mm spring or screw in connectors. Standard low cost electrical components can be purchased from electrical component suppliers. Components can be purchased already wired up but teaching children how to make circuits from generic components is a valuable way of enriching the DT/STEM curriculum.

Secondly, Scratch and Python can be used as the programming languages. Teachers and children are already familiar with Scratch and Scratch allows the programmer to model and test programs before connecting the model.

Thirdly, and most importantly, the children can develop their simple control routines to build whole systems that consider and cater for the needs of the user, gather and analyse data and provide feedback to the user based on the analysis produced. This important level of sophistication cannot achieved through standard school control applications.

What follows is an illustration of a scheme of work for computer control. The aim is to help teachers to plan activities that teach age appropriate skills in a progressive way. The illustration is intended for primary schools. The inclusion of Year 7 and Year 8 topics is to enable teachers to plan for progression and to include challenging extension activities for their gifted and talented children.

	KS 2 (Yr 4/5) **Traffic light**		
	Children learn: The role of an interface in physical computing. How to connect a gPiO interface to a Raspberry Pi computer. How to connect components to output terminals.	**Computing POS**: • design and write programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output; generate appropriate inputs and predicted outputs to test programs	**D&T POS:** • pupils should work in a range of relevant contexts [for example, the home, school, leisure, culture, enterprise, industry and the wider environment]. • understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers and motors] • apply their understanding of computing to program, monitor and control their products.
	Task To control a single traffic light to make the lights turn on and off in the correct sequence with the appropriate delay between changes.		
	Resources - Raspberry Pi / gPiO / Scratch GPIO combination.	**Inputs** - this task does not require inputs	**Outputs** - three coloured LEDs (with protecting resistors - FlowGo gives 6v output voltage so 'protecting resistors' are normally required to prevent burning out the LEDs).

	KS 2 (Yr 5/6) **Pelican crossing**		
	Children learn: The role of an interface in physical computing. How to connect an interface to a computer. How to connect switches and sensors to input terminals and components to output terminals. How to make a control system interactive through the inclusion of an input.	**Computing POS:** • design and write programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output; generate appropriate inputs and predicted outputs to test programs • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	**D&T POS:** Pupils should work in a range of relevant contexts [for example, the home, school, leisure, culture, enterprise, industry and the wider environment]. • understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers and motors] • apply their understanding of computing to program, monitor and control their products.
	Task To control a pelican crossing - this is similar to the traffic light project. It requires the use of a switch as an input and a sound output. For extension work the children can include cross/don't cross lights for the pedestrians.		
	The Raspberry Pi / gPiO workstation is used to control a model of the crossing - one set of lights is enough. Scratch GPIO can be used to write and test the control program.	**Inputs** - a simple push to make switch (this can be homemade).	**Outputs** - three coloured LEDs for traffic lights - two LEDs for pedestrian lights. Buzzer for sound.

	KS 2 (Yr 5/6) **Lighthouse**		
	Children learn: About the need for automated control systems. How to use an LDR as a light sensor. The important construct of selection in programming - if something happens the result is one thing but if something else happens then the result is another thing.	**Computing POS:** • design and write programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output; generate appropriate inputs and predicted outputs to test programs • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	**D&T POS:** Pupils should work in a range of relevant contexts [for example, the home, school, leisure, culture, enterprise, industry and the wider environment]. • understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers and motors] • apply their understanding of computing to program, monitor and control their products.
	Task To control the main lighthouse beam to make it flash with a specified signal, continuously, when it is dark and turn it off when light. To control a fog horn that should sound with a specified sound signal when it is foggy. Fog sensors use a technique called backscatter to measure the amount of water particles i e fog in the air. These are expensive components so a simple switch can be used to represent the fog sensor to test the program - switch on = foggy. The main beam and fog horn must match a prescribed sequence e.g. on for two seconds off for 5 seconds.		
	Resources - Raspberry Pi / gPiO / Scratch GPIO combination. Scratch can also be used for on screen simulation.	**Inputs** - light sensor (LDR) and fog sensor (simple switch to represent a fog sensor)	**Outputs** - filament lamp or white LED for the main light beam. 6V buzzer to represent the fog horn.

	KS 2 (Yr 5/6) **Fairground ride**		
	Children learn: About motor control - forward and reverse. About motor control - increasing and decreasing power for fine speed control.	**Computing POS:** • design and write programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output; generate appropriate inputs and predicted outputs to test programs • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	**D&T POS:** Pupils should work in a range of relevant contexts [for example, the home, school, leisure, culture, enterprise, industry and the wider environment]. • understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers and motors] • apply their understanding of computing to program, monitor and control their products.
	Task To control a rotating fairground ride to teach the range of functions provided by an electric motor (on/off, forward/reverse, slow/fast). There must be a switch to start the ride. The ride must operate in different modes to provide a very gentle experience suitable for young children, a faster ride for adults and a thrilling ride for teenagers and thrill seekers. For extension work the children can add additional features to their fairground ride such as flashing lights, music and possibly a recorded announcement such as ‘roll up roll up for the most amazing fairground ride in the world’.		
	The Raspberry Pi and gPiO interface can handle all of these control inputs and outputs. By creating alternative sequences that change the outputs in different ways, the children can program different experiences for the passengers on the ride. The Raspberry Pi can be used to add audio enhancements such as music and spoken announcements.	**Inputs** - at least three inputs will be needed to enable the operator to select the different ride modes. input 1 (slow speed), input 2 (higher speed) and input 3 (fast with lots of changes in direction) for thrill seekers.	**Output** - a single motor connected to a motor output on the interface. Other outputs could be used for flashing LEDs or for a buzzer if the children want to add extra features to their ride.

KS 2 (Yr 6)
Greenhouse

Children learn:	**Computing POS:**	**D&T POS:**
How to record analogue data. How to compare input data values against a default value. The principles of selection and 'feedback' in control systems. How to use selection and feedback in a control system.	• design and write programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output; generate appropriate inputs and predicted outputs to test programs • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	Pupils should work in a range of relevant contexts [for example, the home, school, leisure, culture, enterprise, industry and the wider environment]. • understand and use electrical systems in their products [for example, series circuits incorporating switches, bulbs, buzzers and motors] • apply their understanding of computing to program, monitor and control their products.

Task

To control the temperature of a greenhouse to keep it at a default value (the ideal temperature for growing plants).
This task introduces the important concept of **feedback** in control systems. Feedback occurs when an output has a direct effect on the input that controls it. A thermostat is a good example. A heater (the output) is switched on if the temperature measured by a temperature sensor (the input) falls below the default value. It is switched off if the temperature rises above the default value. It is switched on if the temperature falls below the default value.

This task illustrates a simple example of **feedback** and **selection** - **if** the temperature is above 20°C turn off the heater **else** turn it on.
A temperature sensor will continuously feed back temperature data to the system which will respond accordingly. Many control systems make use of feedback in this way to maintain a stable state. The household central heating thermostat is an obvious one. There are numerous examples in the human body that can be discussed - this is the science of *homeostasis*.

To be able to sense a range of values such as temperature, light intensity, sound level etc. your interface must have an analogue input. When the sensor is connected, the voltage of the input will change in proportion to quantity being measured.

A PicoBoard can be used to connect analogue components (such as a thermistor - in this case rated at 4.7KΩ - about £1 from Maplin) to the Raspberry Pi.

Alternatively a Kemo Temperature Switch can be connected to the input terminals and set to swithc the input on when a specific temperature has been reached.

Resources - Raspberry Pi / gPiO / Scratch GPIO /Picoboard combination.	**Input** - a temperature sensor connected via a Picoboard (this will need calibrating) or a temperature switch connected to a digital	**Outputs** - For safety use a filament lamp to represent a heater (this is unlikely to generate enough heat to cause the 'heater' to turn off - it

	For the previous tasks, selection in the children's programs have been based on a binary input (ON or OFF). For analogue input, a quantity or value (in this case the temperature) a **PicoBoard** ,is needed. The add on board costs about £35 and s connected to the Raspberry Pi via a USB port. A range of electronic sensors can be connected to one of the four inputs and a range of data values are fed into the Raspberry Pi and read by Scratch.	input.	is possible to buy a safe, 240v mains relay - operated by an output from the gPiO so that a mains fan heater can be controlled.

	KS 2/3 (Yr 7) **Greenhouse control**		
	Children learn: How to record analogue data How to compare input data values against a default value How to use feedback in a control system	**Computing POS**: • design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts • use sequence, selection, and repetition in programs; work with variables and various forms of input and output • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	**D&T POS:** • identify and solve their own design problems and understand how to reformulate problems given to them • develop and communicate design ideas using annotated sketches, detailed plans, 3-D and mathematical modelling, digital presentations and computer-based tools • 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].
	Task There can be a number of different aspects to greenhouse control. Children complete one then for extension they can add additional features to their greenhouse project. 1. Turn on lights when dark 2. Open windows when too hot 3. Turn on a heater when too cold 5. Turn on a fan when too hot 6. Turn on an irrigation system when too dry.		
	Resources - Raspberry Pi / gPiO / Scratch GPIO / Picoboard combination. Scratch can also be used for on screen simulation. 6v motor driven actuator arm (these can be used to open and close ventilation. Old computer cooling fans can be obtained to use as fans for control projects.	**Inputs** 1. Light sensor (LDR) 2. Temperature sensor (thermistor) 3. Actuator arm	**Outputs** 1. LEDs 2. Motor (200:1 reduction gearbox - forward and reverse) 3. Actuator arm 3. Filament lamp to represent heater.

	KS 2/3 (6/7) **Level crossing barrier**		
	Children learn: Fine motor control in forward and reverse direction. The importance of computer control to remove human error from hazardous control systems. The need for fail safe systems and other safety features.	**Computing POS:** design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; design and develop modular programs that use procedures or functions undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability	**D&T POS:** • identify and solve their own design problems and understand how to reformulate problems given to them • develop and communicate design ideas using annotated sketches, detailed plans, 3-D and mathematical modelling, digital presentations and computer-based tools • 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].
	Task The barrier must respond to an train. This must cause a light sequence to execute - flashing amber to red, in addition an alarm sounds and the barrier drops. Once train passes, lights and alarm are switched off and the barrier is raised. This project introduces fine electric motor control to raise and lowere the barrier.		

	Resources - Raspberry Pi / gPiO / Scratch GPIO combination. Scratch (for on screen simulation). 6v motor with (200:1 gearbox). LDR and LED to build a simple light gate. In their D&T lessons the children can construct a model to control using the Raspberry Pi and GPIO. The program can be written in Scratch GPIO or to extend the gifted children, teach them how to write control programs in Python.	**Inputs** - two 'train sensors' - could be push switches, pressure pads, LDR based light gates etc.	**Outputs** - three LEDs, one amber two red. Buzzer as alarm. Motor (with 200:1 reduction gearbox) is operated in forward and reverse to lift and lower the barrier.

KS 3 (Yr 7/8)
Car park barrier

Children learn:	**Computing POS:**	**D&T POS:**
How to develop a simple control project into a more complete system that takes full account of the human factors required by the system. How to use variables in control systems How to count and respond to a numerical data	design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; design and develop modular programs that use procedures or functions undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability	• identify and solve their own design problems and understand how to reformulate problems given to them • develop and communicate design ideas using annotated sketches, detailed plans, 3-D and mathematical modelling, digital presentations and computer-based tools • 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.

Task - The barrier must respond to an approaching car with an audible or text instruction such as ‘press button for ticket’ - or ‘car park full - please wait’.
When the button is pressed, this will both raise the barrier and increment a car counting variable.
The value of the variable is checked against the number of parking spaces available - if the number is reached a FULL sign is lit. If not, a SPACES sign is lit.
This task builds on the knowledge and skills gained from the Level Crossing Barrier task.

For extension the children could build the exit barrier which responds to the fee being paid (this could be simulated with a simple push switch - as the barrier is raised and the car passes through the car counting variable must be decremented by one.

	Barriers and signs can be constructed as a D&T project. This task will introduce children to the use of variables to keep count of the number of cars and to respond accordingly. Text messages can be displayed on the monitor to keep drivers informed. Scratch (for on screen simulation) or Scratch GPIO with a gPiO interface for the control of a physical model.	**Inputs** - a push switch for the IN barrier and another for the EXIT barrier. LDR or pressure pad - to detect when the car has passed through the barrier.	**Outputs** - two LEDs to light the FULL and SPACES signs. Computer screen - the number of spaces available and additional information/instructions can be displayed for the driver's information. **Outputs** - two motors (200:1 reduction gearbox) - on for the IN barrier and one for the EXIT barrier.