PIC electronics for design and technology

Andy Cooper

ICON Coordinator, Nottingham Trent University

An overview of electronic development

It is easy to take the pace of technological change for granted. One hundred years ago I might have been producing this article on one of the first typewriters and writing about the opportunities for teaching electricity in schools. Today we have new materials and resources to inspire our youngsters. This article describes a development which could revolutionise the teaching of electronics.

While schools have been grappling with the National Curriculum and design and technology education has been fighting for its identity, the electronics industry has been experiencing an invisible revolution. The art of microelectronic miniaturisation has reached a level where the brain of the computer can be packaged in a single IC chip. These super chips have become known as PIC chips, a name coined by one leading chip manufacturer, Microchip Technology Ltd, which calls its range Peripheral Interface Controllers. PIC chips have been manufactured since the 1980s and are hidden in products such as TVs, videos, cars, washing machines, microwave ovens and mobile telephones.

PIC chips come in a variety of packages – the more you pay the more you get in terms of memory and input/output features. Recent models include an 8 pin IC (the size of a 555 timer) at a cost of approximately £1. The opportunities for school use begin to look exciting but before you rush out to buy a PIC chip, stop to consider the constraints. PIC chips may be small and cheap but they have limited memory. For example, the PIC 16C55 offers 512 words of ROM, 24 bytes of RAM and 20 input /output lines. Compared to a modern computer with memory sizes in the megabyte range it can be seen that the programming options are somewhat restricted. The only efficient way of making use of this memory is to use machine code programming.

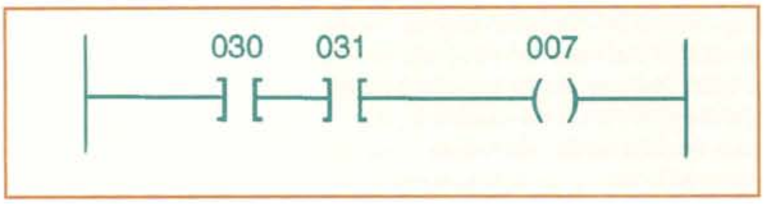
If you are one of the 99.99% of the population who have not been trained in machine code programming then don't despair – providing you have determination and a spare week or two you can learn. My five-day introduction culminated in making a buggy move in a rectangular path. However, I had forgotten most of the skills after a few days back in the classroom and reached the conclusion that machine code programming is totally inappropriate for the average pupil and should be left to the experts. If you have reached similar conclusions or are not prepared to devote the time to learning machine code then you might be interested in the developments taking place at the Nottingham Trent University.

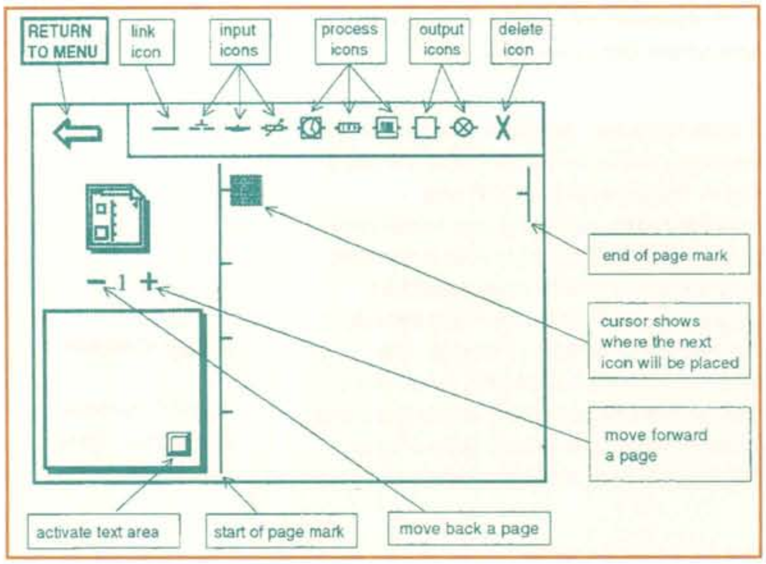
A team has been working on a new control language, ICON, for pupils at Key Stages 3 and 4. The concept for ICON was straightforward: develop a user friendly but versatile control language, write a compiler program which would convert an ICON program into machine code, then transfer the machine code into a PIC chip. The development of the ICON-PIC language is nearing completion and this article describes ICON in order to give you a taste of what PIC programming could be like.

The rationale for ICON

The starting point for the development in 1987 was the realisation that software programmed electronics could bring a number of advantages to design and technology education. Most of the process functions in a circuit design could be programmed into a single PIC chip, reducing the component count and assembly complexity and allowing more sophisticated designs. If you are concerned that electronic concepts such as potential division and component matching would be lost, bear in mind that input and output devices would still need to be interfaced to a PIC chip. A significant feature of software programmed electronics is that pupils would have greater opportunity for system design with less dependence on circuits designed by someone else.

The choice of a programming language for ICON was not immediately obvious. The majority of school control languages are based on LOGO and BASIC and although they are useful for developing mathematical concepts and teaching accepted programming techniques, their use for real-time processing is limited. The very nature of these languages restricts them to sequential events which is fine for sequential tasks such as programming traffic lights. However, tasks which involve





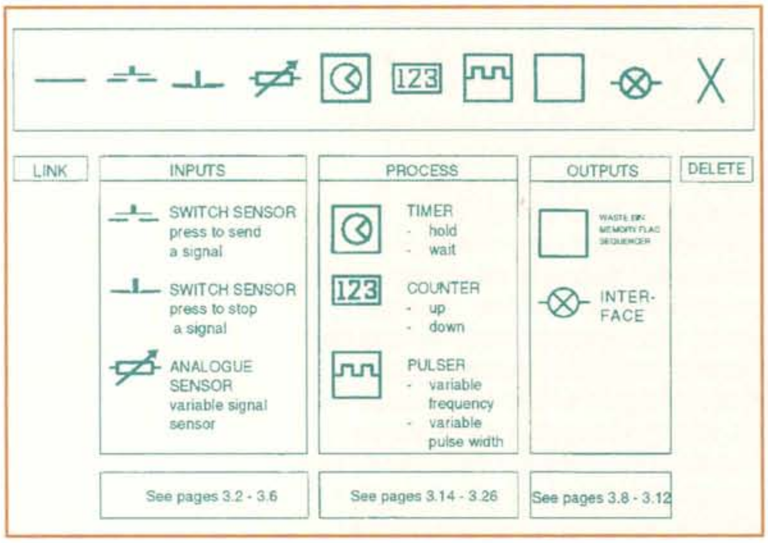

LINK
INPUTS
PROCESS
OUTPUTS
DELETE
SWITCH SENSOR press to send a signal
SWITCH SENSOR press to stop a signal
ANALOGUE SENSOR variable signal sensor
TIMER - hold - wait
COUNTER - up - down
PULSER - variable frequency - variable pulse width
WASTE BIN MEMORY FLAG SEQUENCER
INTER-FACE
See pages 3.2 - 3.6
See pages 3.14 - 3.26
See pages 3.8 - 3.12

Figure 1: Ladder logic program
Figure 2: ICON programming page
Figure 3: ICON functions

handling more than one event at a time, such as modelling a railway crossing control system with flashing lights and the simultaneous control of a barrier, require the programmer to develop multi-tasking routines. This introduces programming complications which do not arise with an electronic systems approach.

Investigations into industrial control systems revealed interesting parallels relating to control rationale. Industry's need for cost-effective and flexible control systems had led to the development of Programmable Logic Controllers (PLCs). These became increasingly important during the 1980s offering an intermediate route lying between dedicated electronic systems and the use of computers for the control of industrial machinery. They used a control language adapted from **Ladder Logic**, based on an electrician's circuit wiring convention.

An example of a simple ladder logic is program is shown in figure 1. The output (007) is energised when two switches (030 and 031) are simultaneously operated.

Separate control functions are represented by different rungs on the ladder but unlike programs such as LOGO or BASIC where one program line is completed before moving to the next, all the rungs are quickly scanned and any changes are implemented at the end of the scan. The system emulates parallel processing and requires a different approach to programming. Flow charts and the use of repeat loops are irrelevant, instead it is necessary to consider logic statements and appreciate that any changes will be acted on in real-time.

The Ladder Logic concept was seen to offer the advantages of an intuitive electronics systems approach to programming and became the focus for the ICON development.

Using ICON

ICON uses interconnected graphic symbols (icons) to represent logic statements. The user is presented with an on-screen programming page consisting of an ICON menu bar and left and right boundaries for the logic statement.

ICON programming involves selecting icons and assembling them to form a SENSE, PROCESS, OUTPUT statement working from left to right across the page. The available functions are summarised in figure 3.

Function icons are shown in the menu bar and are selected by using the mouse pointer. A selected icon appears at the cursor position and prompts appear for a label number and function values.

ICON programming

An example of a solution requiring a timer to come on for 30 seconds when a switch is operated is shown in figure 4.

The switch symbol represents an external switch connected to input one of the interface. The output symbol represents an external device connected to output two of the interface. The timer is handled by the ICON software.

ICON multi-tasking programs

Programs which involve two or more independent functions are programmed with the functions appearing on separate pages. The following program shows a solution for a model fairground which involves three rides. Each ride is operated by pressing a switch which activates a timer to hold the ride on for a set time (see figure 5).

The rides can be operated independently, in any order, without any apparent pauses and using program structures which are logically straightforward. A further advantage is that different pupils could easily program their own ride on their own page within a single program to provide a group display. It would be complex to program this situation using a sequential language because of the need to add multi-tasking routines.

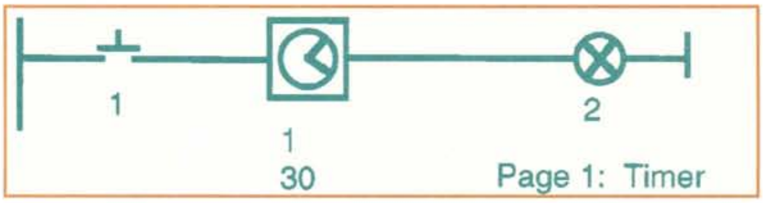
ICON advanced features

The basic menu functions can be modified in a number of ways to increase the versatility of the programming. The use of a backslash after an icon number inverts the output of the function. AND functions are obtained by connecting icons in series and OR functions by connecting icons in parallel. Output states can be fed back as software switches and a sequencer function can be triggered by manual or software events.

Some of these features are considered in an automatic car park barrier example (which uses a single barrier. Interface inputs and outputs are assigned as shown in figure 6.

Raising the barrier

The barrier rises when the ticket button (1) is pressed providing the park has spaces. If the car park has spaces then switch 4*\ which feeds back the inverse output state from the counter up mode will be true. The barrier up switch 2 is normally closed when the barrier is down. So when the ticket button is pressed all the switches are true and output 1 is activated. The state of output 1 is fed back as the software switch 1* and this latches the output on. The barrier rises until the barrier up switch 2 is activated.



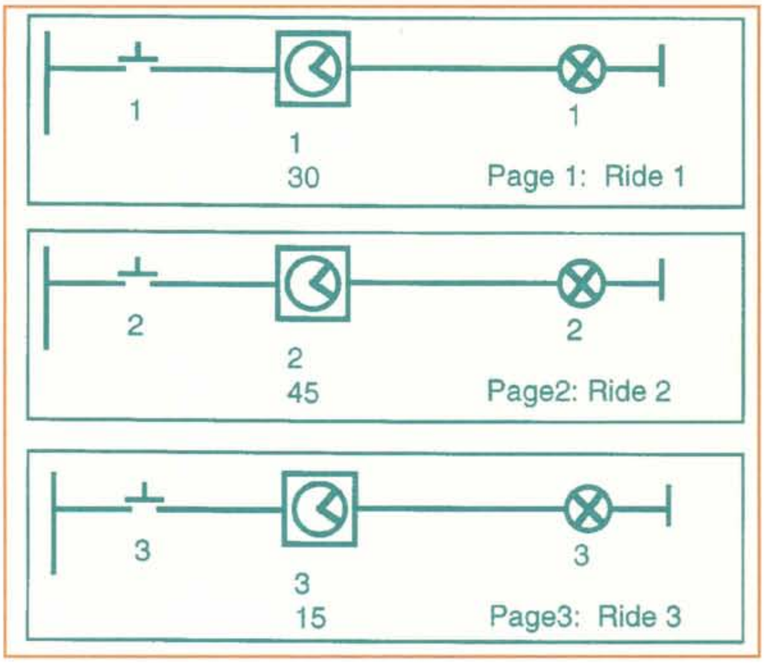
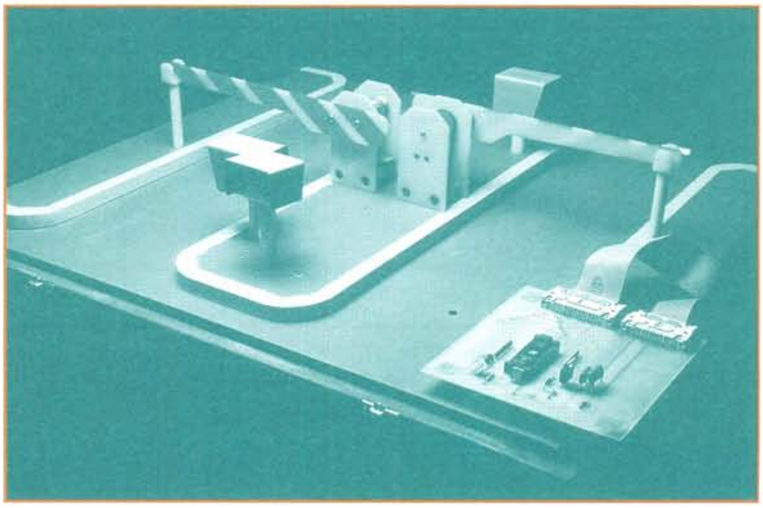
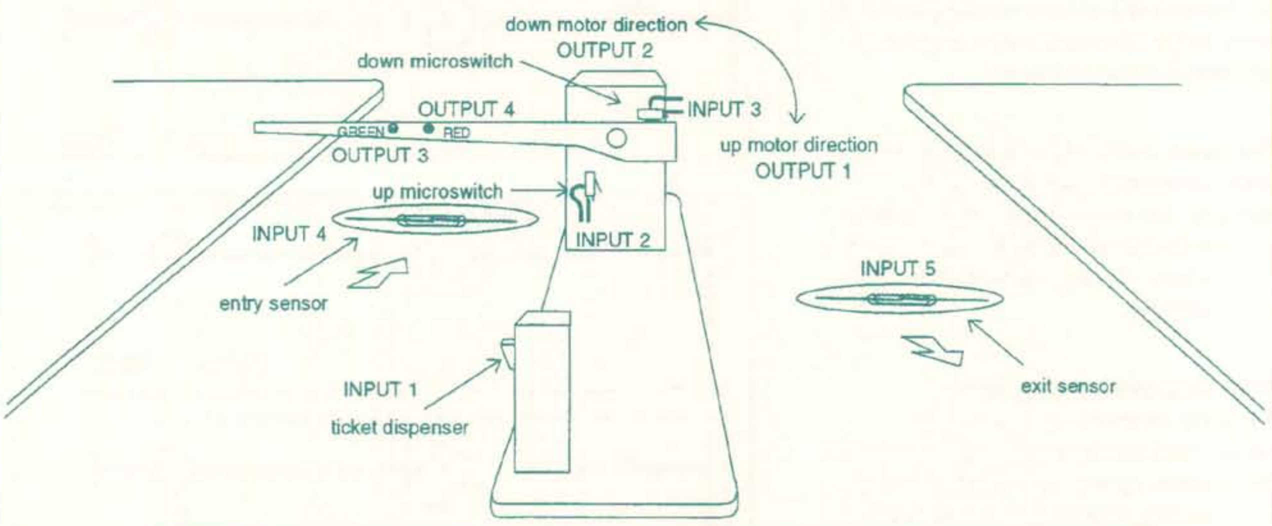


Figure 4: Timer program
Figure 5: ICON solution for three fairground rides

Car park barrier with a PIC chip





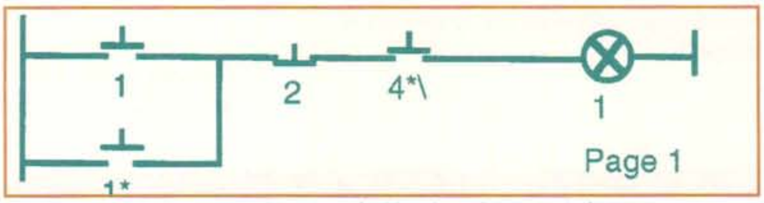
Inputs	
1	Ticket button
2	Barrier - up switch
3	Barrier - down switch
4	Car-in switch
5	Car-out switch

Outputs	
1	Barrier - motor up
2	Barrier - motor down
3	Spaces light
4	No spaces light
5	

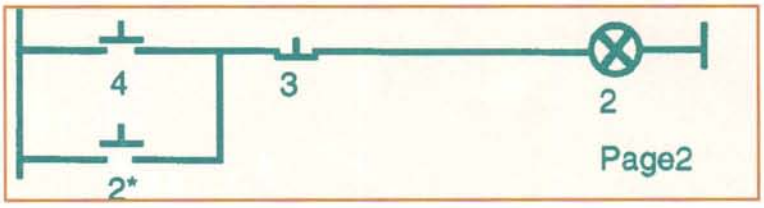
Figure 6: Barrier inputs and outputs

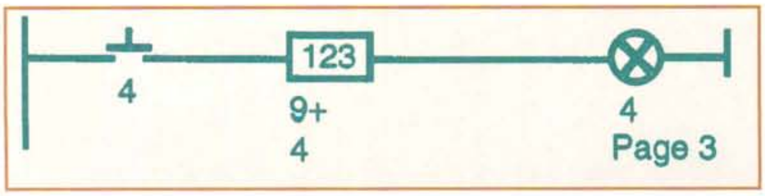
Lowering the barrier

The barrier lowers when the car-in switch 4 is activated. When the barrier is up the barrier down switch 3 is normally closed so that when switch 4 is activated the output is activated. This state is fed back as the software switch 2* which latches the output on. The output remains on until the barrier down switch is activated.



\ inverse mode * output fed back as input switch





Counting cars in

The counter increments each time the car-in switch 4 is activated. When it reaches 4 (the programmed set value) the output latches on turning on the park full light. The state of output 4 is also fed back as software switch 4* and inverted, which prevents the barrier from allowing more cars to enter.

Counting cars out

Counter 9 is used again but in down mode so that each time a car leaves the count decrements by 1. The down mode gives an output when the count is less than the set value and is used to indicate space availability.

How complicated is ICON programming?

If the automatic car park example seems complicated take heart from the fact that it is a challenging control problem worthy of a good GCSE grade. Try programming the car park system in your favourite control language and compare the solutions with ICON. The logical concepts and level of difficulty remain essentially the same

Figure 7: Raising the barrier
Figure 8: Lowering the barrier
Figure 9: Counting cars in

whatever programming language is used to solve the problem. But this does not mean that all languages are equally easy to use for problem solving. The importance of a language to its use can be illustrated by Roman numerals. They were fine for the communicating dates but try using them for adding and subtracting! We believe ICON has some positive advantages for control and creative design. It offers real time processing with a visual statement of the logic paths. It also removes the need for multi-tasking routines allowing students to focus on more challenging problems.

ICON is also supported by an extensive manual with numerous worked examples to aid understanding and creative application of the ideas. On the downside ICON does not support traditional structured programming procedures and only allows limited mathematical processing.

ICON – PIC programming

The full creative potential of ICON will be unleashed when it is used to program PIC chips. It will then be possible for pupils to develop products with an embedded computer at pocket money prices. Imagine a possible design and make assignment for Year 1 secondary pupils – ***the kitchen timer***. Using an 8 pin PIC with a prepared circuit board pupils would have access to 6 input/output lines. Possible control options could include combinations of the following:

The control circuit could be packaged in a variety of ways using different materials, fasteners and finishes.

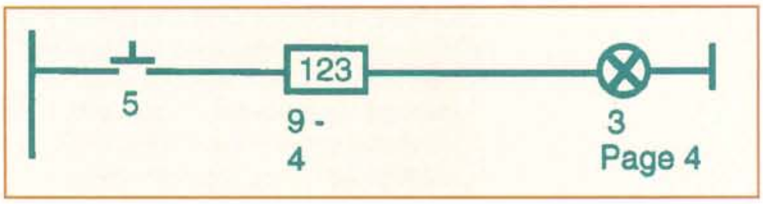


Figure 10: Counting cars out

If such a challenge were introduced at year 1, consider the possibilities for later years; environmental control systems, security devices, lighting displays, automata, robot animals. By the time pupils reach Key Stage 4 they could be working with designs currently only accessible to professional programmers. ICON-PIC electronics could open up a whole new world of opportunities for our youngsters and inspire our future technologists to help shape the next 100 years.

References

Cooper, A, Kicks, M (1991) *ICON, Computer control for the next generation.* TICST, Nottingham Polytechnic

Cooper, A, Kicks, M, Ghee, W (1996) *ICON, Computer control . Technology Education Group,* The Nottingham Trent University

Cooper, A, Kicks, M, Ghee, W (1995) *ICON: a computer control program for education, using parallel programming procedures* WCCE 95

Cooper, A, Kicks, M, Ghee, W (1996) *The potential for using PIC chips in school control projects* IDATER 96 Loughborough University

Department for Education (1995) *Design and Technology in the National Curriculum*

Kissell, T (1986) *Understanding and Using Programmable Controllers.* Prentice-Hall International Editions

Table 1

Inputs	Process	Outputs
Press switch	single time and continuous output until reset	Buzzer
Press switches	single time and one minute warning	Lamp
Variable resistor	single time and light display count down	LEDs
Reset switch	single time and pulse output	Motor
	different switched times	
	variable times	
	novelty movement indicator	