– EPE – NEW NAME NEW DESIGN!

Practical Electronics

The UK's premier electronics and computing maker magazine

Audio Out
Analogue noise generator

Practically Speaking
Intro to SMD

Circuit Surgery
Problems with SPICE simulations

Micromite
Adding colour touchscreens

XOD ARDUINO

Visual programming for Arduino with XOD

Fascinating display system you can build

WIN! Microchip Curiosity PIC32MZ EF Dev Board 2.0 WIN!

Introduction to FPGAs with the low-cost iCEstick

Touchscreen and Micromite

Remote control for the ultra-low-distortion Preamplifier

PLUS!

Net Work – Cookies, data trails and security options
Max's Cool Beans – Best-ever flashing LEDs!
Techno Talk – A spot of nostalgia

Apr 2020 £4.99

With Over 250 Million Units in Stock, We've Got You Covered

Delivering You the World's Largest Inventory of Microchip Products

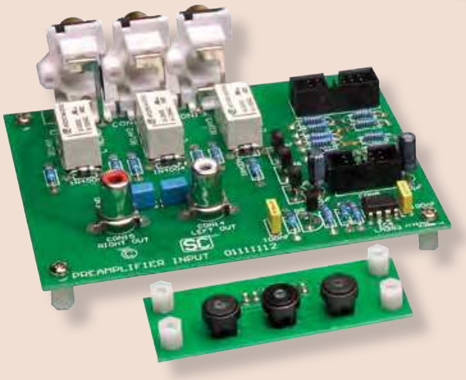
www.microchip.com/buy

MICROCHIP

Volume 49. No. 4
April 2020
ISSN 2632 573X

Practical Electronics

Contents



Projects and Circuits

Ultra-low-distortion Preamplifier with Tone Controls – Part 1 by John Clarke 14
Meet the 2020 Ultra-low-distortion Preamplifier! A superb design with tone controls and infrared remote volume control, input switching and muting.

iCEstick – Part 1 by Tim Blythman 26
Explore field programmable gate arrays (FPGAs) with the low-cost and compact iCEstick board and free IceStudio software.

Flip-dot Display by Tim Blythman 34
Build this remarkable dot matrix display which has digits/letters over 90mm high and is visible under a wide range of lighting conditions.

Series, Features and Columns

News 8
A roundup of the latest news from the world of electronics and technology

Techno Talk by Mark Nelson 10
A spot of nostalgia

Net Work by Alan Winstanley 12
The history of 'cookies', and the trail of digital data you leave in your wake when you surf the Internet, plus some of the options you have to boost privacy and security.

Circuit Surgery by Ian Bell 44
Problems with SPICE simulations

Practically Speaking by Mike Hibbett 49
Introduction to surface mount technology – Part 1

Make it with Micromite by Phil Boyce 52
Part 15: Having fun with a colour touchscreen

Audio Out by Jake Rothman 59
Noise about noise – a new slant on analogue noise with tilt control – Part 1

Visual programming with XOD by Julian Edgar 62
Fan Speed Controller

Max's Cool Beans by Max The Magnificent 66
Flashing LEDs and drooling engineers – Part 2

Regulars and Services

Subscribe to Practical Electronics and save money 4
Wireless for the Warrior 6
Reader services – Editorial and Advertising Departments 7
Editorial 7
The vulnerability of global supply chains... PE Mini-monitors
PE Teach-In 9 25
Practical Electronics – get your back issues here! 32
Exclusive Microchip reader offer 33
Win a Microchip Curiosity PIC32MZ EF Dev Board 2.0
Practical Electronics back issues CD-ROM – great 15-year deal! 43
PE Teach-In 8 65
Practical Electronics CD-ROMS for electronics 70
A superb range of CD-ROMs for hobbyists, students and engineers
Direct Book Service 73
Build your library of carefully chosen technical books
Practical Electronics PCB Service 76
PCBs for Practical Electronics projects
Teach-In bundle – what a bargain! 78
Classified ads and Advertiser index 79
Next month! – highlights of our next issue of Practical Electronics 80

Made in the UK.

Written in Britain, Australia, the US and Ireland.

Read everywhere.

The May 2020 issue of *Practical Electronics* will be published on Thursday, 2 April 2020 – see page 80.

WIRELESS FOR THE WARRIOR

by **LOUIS MEULSTEE**

THE DEFINITIVE TECHNICAL HISTORY OF RADIO COMMUNICATION EQUIPMENT IN THE BRITISH ARMY

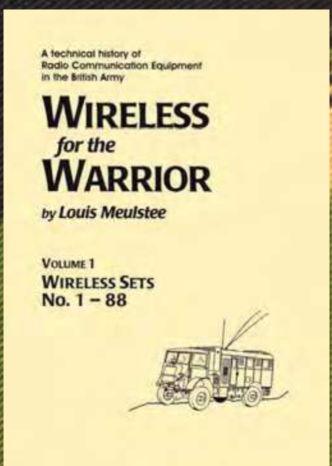
The *Wireless for the Warrior* books are a source of reference for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

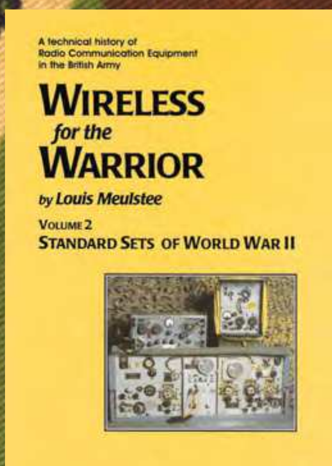
The books are very detailed and include circuit diagrams, technical specifications and alignment data, technical development history, complete station lists and vehicle fitting instructions.

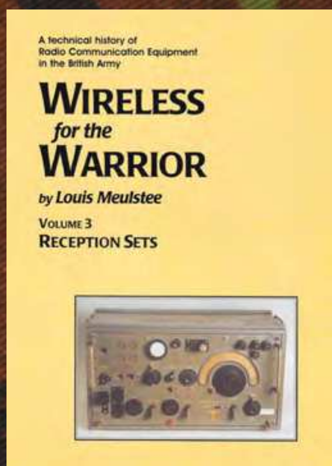
Volume 1 and *Volume 2* cover transmitters and transceivers used between 1932-1948. An era that starts with positive steps taken to formulate and develop a new series of wireless sets that offered great improvements over obsolete World War I pattern equipment. The other end of this timeframe saw the introduction of VHF FM and hermetically sealed equipment.

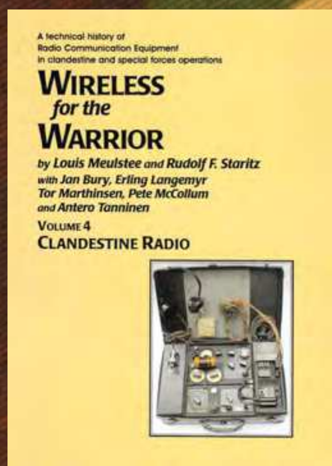
Volume 3 covers army receivers from 1932 to the late 1960s. The book not only describes receivers specifically designed for the British Army, but also the Royal Navy and RAF. Also covered: special receivers, direction finding receivers, Canadian and Australian Army receivers, commercial receivers adopted by the Army, and Army Welfare broadcast receivers.

Volume 4 covers clandestine, agent or 'spy' radio equipment, sets which were used by special forces, partisans, resistance, 'stay behind' organisations, Australian Coast Watchers and the diplomatic service. Plus, selected associated power sources, RDF and intercept receivers, bugs and radar beacons.









ORDER YOURS TODAY!

JUST CALL 01202 880299 OR VISIT www.epemag.com

Practical Electronics

Volume 49. No. 4
April 2020
ISSN 2632 573X

Editorial

Editorial offices

Practical Electronics
Electron Publishing Limited
1 Buckingham Road
Brighton
East Sussex BN1 3RA

Tel 01273 777619
Mob 07973 518682
Fax 01202 843233
Email pe@electronpublishing.com
Web www.epemag.com

Advertisement offices

Practical Electronics Adverts
1 Buckingham Road
Brighton
East Sussex BN1 3RA

Tel 01273 777619
Mob 07973 518682
Email pe@electronpublishing.com

Editor Matt Pulzer
General Manager Louisa Pulzer
Digital subscriptions Stewart Kearn **Tel** 01202 880299
Online Editor Alan Winstanley
Web Systems Kris Thain
Publisher Matt Pulzer

Print subscriptions

Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH
United Kingdom

Tel 01202 087631
Email pesubs@selectps.com

The vulnerability of global supply chains

You wouldn't think that a market in central China would have much to do with *PE*, but as the suspected ground zero for the current coronavirus health crisis and the subsequent knock-on effects in the rest of China I have been forced to understand in a practical, not just theoretical sense just how interconnected we are these days. Our PCBs are made in China, mainly through a trio of leading manufacturers. All of them have had to go through a period of shutdown and limited working, which resulted in an entirely understandable delay in our orders reaching us. Despite what must be very difficult working conditions, our suppliers have now delivered and we have an up-to-date inventory. Over the last few weeks some of you have written asking when PCBs will appear, and I'd like to thank you for your patience.

PE Mini-monitors

That's the good news, the not so good news is that Wavecor – the supplier of our LS3/5A-style Mini-monitor drivers – are, at the time of writing, still shut down, and not due to reopen until 1 March at the absolute earliest. I was about to place a large order for their drivers when they were forced to close, so unfortunately we will have to wait a month or two until we can complete the project. Disappointing, but I am confident we will get there.

This month

Now let's focus on this month's magazine. We have three superb projects that are about as different as it's possible to be. First up, a highly unusual display – the Flip-dot. A clever blend of electronics and electro-mechanical parts that's fun to watch – and listen to! For those of you who like to build unusual clocks – or any other kind of messaging system – this is a must-build project.

Next, a first for *PE*, as we start a two-part foray into using FPGAs. These ultra-flexible ICs are undoubtedly useful, but have a reputation for being a tough nut to crack without the benefit of professional tools (with professional pricing). Well, not anymore – a £20 development board and free software is all you need now, and we're here to get you up and running.

Last, we have a superb, high-fidelity preamplifier. It boasts not only advanced tone controls, but also IR remote control. There's quite a choice of configurations, so if you're in the market for this preamplifier then do ensure you read the options list at the end of this month's introductory article.

Of course that's not all. As always, our hardworking columnists have cooked up a *smorgasbord* of electronic delights for you to feast on… so read on!

Matt Pulzer
Publisher

NEWS

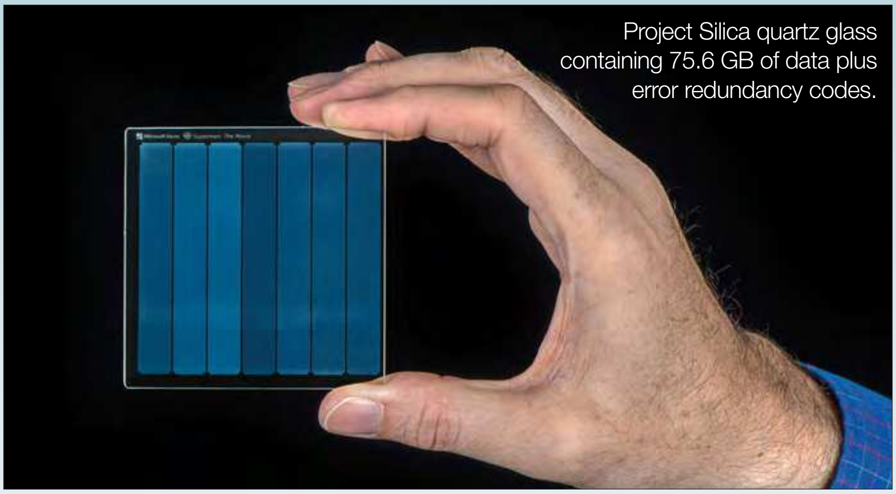
A roundup of the latest news from the world of electronics and technology

Project Silica

The demand for long-term data storage is reaching unprecedented levels. By 2023, Microsoft estimate that over 100 zettabytes of data will be stored in the cloud. Operating at such scales requires a fundamental rethinking of how we build large-scale storage systems, as well as the underlying storage technologies that underpin them.

Microsoft's *Project Silica* in collaboration with the University of Southampton Optoelectronic Research Centre, where researchers originally demonstrated how to store data in quartz glass with femtosecond lasers, is developing the first-ever storage technology designed and built from the media up, for the cloud.

An infrared laser encodes data in glass by creating layers of three-dimensional nanoscale gratings and deformations at various depths and angles. Machine learning algorithms read the data back by decoding images and patterns that are created as polarised light shines through the glass.


Project Silica quartz glass containing 75.6 GB of data plus error redundancy codes.

The lasers encode data in 'voxels' (**vo**lume-pi**xel**) – the three-dimensional equivalent of a pixel. Unlike other optical storage media that write data on the surface of something, *Project Silica* stores data within the glass itself. A 2-mm-thick piece of glass, for instance, can contain more than 100 layers of voxels.

Data is encoded in each voxel by changing the strength and orientation of laser pulses that physically deform the glass. It's somewhat like creating upside down icebergs at a nanoscale level, with different depths and sizes and grooves that make them unique.

The hard silica glass can withstand being boiled in hot water, baked in an oven, microwaved, flooded, scoured, demagnetised and other environmental threats that can destroy priceless historic archives or cultural treasures if things go wrong for more traditional digital storage media.

The Microsoft team, working with Warner Bros has successfully stored and retrieved the entire 1978 *Superman* movie on a piece of glass 75mm square by 2mm thick. It was the first proof-of-concept test for *Project Silica*.

What the zetta?

Yes, we all know what giga means, probably tera and even peta, but zetta? Well it's not the next prefix up from peta, that's exa. It's the one up from exa, so not just big, but unimaginably huge.

A zettabyte is one sextillion bytes, with unit symbol ZB. 1ZB = 1000^7 bytes = 10^{21}bytes = 1000000000000000000000 bytes = one billion terabytes.

Build a 100ZB stack of 1cm-thick 1TB SSDs, and you'll reach the moon… and back. That's a lot of data.

DNA?

Glass is not the only medium scientists are researching – they are looking to nature for inspiration. DNA is one attractive possibility because it is extremely dense (up to about 1 exabyte per cubic millimeter) and durable (with a half-life of over 500 years). Although there are challenges around high cost and very slow read and write times, in June 2019, scientists reported that all 16GB of Wikipedia had been encoded into synthetic DNA.

A spot of nostalgia

Techno Talk

Mark Nelson

Sorry to disappoint; there are no April Fool items to spot in this month's article. But you are guaranteed your normal diet of the weird and wonderful in the world of fairly practical electronics. Something old, something new and maybe even something blue. So let's crack on with it. Or as WC Fields used to say, 'Start the day with a smile – and get it over with'.

Let's begin with the first of two trips down memory lane in this month's sermon. Do you remember Tandy shops? Derided by some, Tandy often saved my bacon, mainly because being disorganised, I frequently ran out of components at awkward times, such as 5pm on a Saturday afternoon (not ideal if you have set Sunday aside for constructing an electronic project). Panic not; there was still just enough time to dash over to my nearby Tandy emporium and pick up a blister pack of a dozen 10kΩ resistors or a hank of solder. No matter that these cost me more than they ought to; paying over the odds avoided the frustration and disappointment of a wasted Sunday.

Eventually the convenience of nipping out to a local electronics shop came to an end, when Tandy shops closed in 1999, having been bought by Carphone Warehouse to be reborn either as cellphone shops or else as Tecno camera outlets. The trading name was revived subsequently by new owners and now operates online at **www.tandyonline.com**

Maplin, backwards and forwards

All was not lost when Tandy shops disappeared, as larger and slicker Maplin Electronics shops took up the slack in the retail vacuum created. Although highly successful for four decades, with 217 Maplin stores in mid-2017, the business suffered from online competition and on 25 June 2018, all Maplin stores ceased trading after failing to find a buyer for the business. The name was revived by new owners in January 2019, with a Maplin website brought back online. Amusingly, a rival operation opened with the reversed name of 'Nilpam', but currently the **https://nilpam.uk** website merely states 'We'll be back soon'.

Battery bonus

Tandy had an enterprising gimmick for regular customers in the form of their Tandy Battery Club. The sales staff would offer you a loyalty card that entitled you to one free AA, C, D or PP3 cell a month when you bought something in their shop (probably some more batteries, as a single AA, C or D cell was seldom of use on its own; whereas the Power Pack was viable of course). However, you get what you 'pay' for and the quality of these batteries was not very good; likewise the free flashlight torches that they sometimes gave away. I did wonder how many people had more than one Battery Club card on the go at once... No matter.

Talking of batteries brings me to an environmentally friendly means of recycling them, mooted by Julien Leclaire at the University of Lyon in France. For decades, scientists in Lyon have tried to find a use for atmospheric carbon dioxide and now the **Agchemigroup.eu** blog reports that the researchers at Lyon have discovered that CO_2 can be used to extract useful metals from discarded batteries. The breakthrough helps solve two major problems: reducing greenhouse gas emissions from industry and the waste of raw materials that could be recycled.

Leclaire's team has devised a process for extracting useful metals from recycled technology, such as smartphone batteries. Not only would this help solve the problem of toxic metals leeching into ground water at landfill sites, but it could also provide a practical source of the dwindling supplies of rare-earth elements used in high-tech electronic products, not to mention other raw materials. Says Leclaire, 'By simultaneously extracting metals and injecting CO_2, you add financial value to a process that is known to be cost-intensive.' You can read an abstract of the technology involved at: **https://go.nature.com/39PROuJ**

Sausages past and present

Now for our second look back in time – to 1979 and the BBC television programme *That's Life*. This featured Prince, a talking dog that loved 'talking' about his favourite food, sausages. There's a clip online of this clever creature at: **https://youtu.be/ajsCY8SjJ1Y**. Talking of vintage sausages, I noticed that one of my 12V 'wall wart' power supplies has a small sausage-like object incorporated in the output lead. Yes, I know that it's some kind of ferrite suppressor for stopping any RF hash generated by the switch-mode power supply (SMPS) entering whatever device the power supply is feeding, but why don't you see these suppression sleeves much nowadays?

According to the Internet (which must be right), the ferrite core acts as a one-turn common-mode choke, and can be effective in reducing conducted and/or radiated emissions from the cable, as well as suppressing high-frequency pick-up in the cable. Basically, ferrites can be thought of as high-frequency resistors, having little or no impedance at low frequencies or DC. Ferrite cores are most effective in providing attenuation of unwanted noise signals above 10MHz.

So why were they necessary in the past but not now? It could be that the SMPS of today does not generate as much hash as in days gone by. Or perhaps it's just a sign of cheapskate cost-cutting – sorry, intelligent value engineering in China's factories. Reader feedback on this conundrum would be very welcome. I note that the subject has come up on an online forum, with some amusing responses. For instance: 'I once bought a replacement charger for my Lenovo laptop from China. I found that the suppressor was a dummy. The Chinese manufacturer wasn't hampered by some silly-ass regulator.' Also: 'Modern cables tend to be shielded and so a lot of cables don't really need these.'

Really? The output cable from a SMPS is always made of normal unshielded wire in my experience. Maybe the suppression components are all placed inside the SMPS nowadays, but I have never seen any inside the wall warts that I've torn apart.

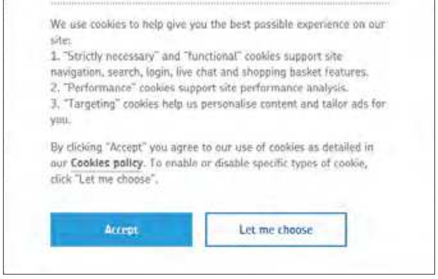
Net Work

Alan Winstanley

This month, *Net Work* looks at the history of 'cookies' and the trail of digital data you leave in your wake when you surf the Internet, plus some of the options you have to boost privacy and security.

The first ever web browser was NCSA Mosaic, which was a product of the University of Illinois' National Center for Supercomputing Applications. It was designed as a simple graphical means of rendering and sharing information over a network. Mosaic was developed by a very bright student (Marc Andreessen) and fellow programmer Eric Bina, who reportedly worked flat out on this university project to earn some pocket money. Released for free public download in 1993, the early version of Mosaic gained 1,000 users in a few weeks, but as the Internet started to mushroom, there were a million users of Mosaic worldwide by the following year.

According to a 2006 biography by Simone Payment (see *Marc Andreessen and Jim Clark: The Founders of Netscape* from the *Internet Career Biographies* series), NCSA hogged the limelight for their new 'Mosaic web browser', while Andreessen earned no recognition for its success. Andreessen would eventually join forces with Jim Clark, a wealthy and highly successful businessman who had decided to cut his ties with Silicon Graphics Inc (SGI), the graphics workstation manufacturer that he had founded. The result was an all-new, reworked version of NCSA Mosaic initially produced by Mosaic Communications Corporation, their new company formed in 1994. Their browser was initially named *Mozilla* after their dinosaur-like mascot (a portmanteau of Mosaic and the mythical monster Godzilla).



Website cookie opt-ins control the type of cookies dropped onto your system.

Mosaic changed their name to Netscape and Mozilla's successor, Netscape Navigator 1.0, was launched late in 1994 at a time when Internet users had little to choose from in web browsers. Over time, Netscape tried to monetise Navigator, but it eventually folded into today's Firefox web browser, which is downloadable for free from **Mozilla.org**

The curse of cookies

Those early foundations have left us with something that many users remain deeply suspicious of: cookies. In a paper published by France's Inria (see later), a young and highly talented Netscape developer named Lou Montulli, one of the first half dozen that formed the new Netscape team with Andreessen, is credited with an idea in 1994 that would enable websites to 'remember' their visitors, something that was a thorny problem for the emerging web industry at the time.

As web surfers will doubtless agree, cookies are both a blessing and a curse. These encrypted, innocuous-looking little text files are dropped onto a user's system to enable a website to recognise and interact with that user. They can be genuinely useful at times: a cookie helps a website to remember the contents of your shopping cart so you don't have to re-enter your choices again, for example. Cookies are also needed sometimes to make a website work properly, but they are also used to follow your journey across the web. By tracking which websites you visit, cookies can shape the adverts that appear when you visit other websites such as Facebook, eBay or media portals. Add-on browser extensions such as Ghostery show the true extent of trackers that larger websites might typically utilise. I discussed Ghostery in a 2013 column, and the problem of trackers has not gone away.

Advertising clicks are the corpuscles of the online ad industry and, as you would expect, the software and analytics that monitor the delivery and performance of ads, their click-though rates, cpm (cost per mille, or cost per thousand clicks) and cookie metrics have all been refined to a granular degree over the years. The use of cookie controls means that users are supposed to consent to receiving them when browsing. Website cookie opt-ins can be distracting, annoying and intrusive, and many everyday users simply click 'accept' and dismiss the opt-in without a second thought. If you 'reject' cookies, you may block personalised ads but you may still see generic adverts instead.

The rise of Cookieless Monsters

Some disreputable sites may harness cookies for more malicious purposes, possibly leading to the installation of spyware or malware scripts hosted by infected websites. Cookies can be deleted from popular web browsers via the usual settings menu, something that one third of us do within a month, industry sources say. Using 'Privacy' mode when surfing will block cookies and hide one's browsing history (but not much else). Software that helps clean up cookies includes CCleaner, now owned by Avast, from **www.ccleaner.com** or consider PrivaZer from **https://privazer.com/en** (not tested by the author). 'Personalities' or 'containers' can also be used when surfing to ring-fence your browsing session, which prevents a website from sniffing out other cookies stored on your system. Extensions or plugins designed for your browser can also help with cookie management.

Each web user is seen as a marketing opportunity, and every online marketer somehow wants to identify our system and by implication, profile the person using it. Our IP address, our browsing history, our location, date of birth, things we've bought, things we've seen but haven't bought yet, our interests – this personal usage data enables vendors to join the dots and target our profile with relevant advertising.

Even though cookies don't identify users *individually* and contain no personal data as such, the fact that savvy web users can defeat them so easily has created a problem for online marketers: how can a user's web-browsing session be linked to a device if it doesn't contain any cookies? One way is through

Panopticlick by EFF will reveal any browser fingerprinting vulnerabilities in your system.

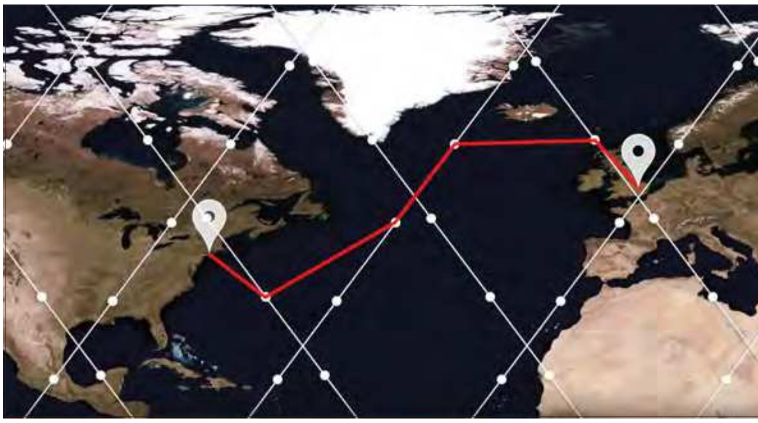
the use of non-consensual *browser fingerprinting*¸ which has given rise to the term 'cookieless monsters'. When visiting a web page, a wide range of seemingly benign data is exchanged between your browser and the website. Much of it is already collected by web server logs for use in statistics, such as the visitor's IP address and country of origin, the web browser type (called the 'user agent'), screen settings and the client's operating system. It's how websites know to render the mobile or desktop version of a page. Website operators know that such data is not always reliable as it can be spoofed. However, this ordinary-looking data (and more besides) that travels to and fro during your web-surfing session can be amalgamated to form a 'browser fingerprint' identifying your system at that moment in time.

How browsers leave fingerprints

Parameters that can be checked this way include the user agent, the screen resolution and colour depth, any browser extensions, add-ons or plugins installed, any fonts installed (derived from the use of Flash), the system language, WebGL (Javascripted web graphics) and other esoteric settings. Cybersecurity developers Seon (**https://seon.io**) claims up to 500 fingerprint parameters can be extracted and 'hashed' this way. These factors undoubtedly change over time (weeks/months) but if the marketers (or fraudsters) hit lucky, the fingerprint will be unique to your device at that moment in time.

In a paper published by Inria, the French Institute for Research in Computer Science and Automation, the techniques for browser fingerprinting were explored and they analysed nearly 10,000 fingerprints collected from about 2,000 browser sessions. Even though the browser fingerprint was likely to change over time, they discovered that they could track browsers for over 54 days, and 26% could be tracked after 100 days, all without using cookies, says the Inria paper (see **https://hal.inria.fr/hal-01652021/document**).

'Browser fingerprinting is both difficult to detect and extremely difficult to thwart,' say the digital privacy activists at Electronic Frontier Foundation (EFF). One way to test your browser for tracker vulnerabilities is at the Panopticlick website (**https://panopticlick.eff.org/**), a ten-year-old research project run by the EFF. It recognised that my own system was unique among the 215,000 users tested in the last six weeks.



Brilliant.org offers a fascinating and thought-provoking analysis on Youtube of the Starlink network.

To safeguard privacy, one way of helping defeat trackers is to enable the Do Not Track (DNT) option in your web browser privacy settings, but most websites fail to observe DNT anyway, says the EFF. The ultra-anonymous TOR browser from **www.torproject.org/** could be used, but it will be far too slow for everyday users. Mainstream web browsers such as Google's Chrome have lacked fingerprint protection, but the latest version (72.0) of the Firefox Quantum browser is a step in the right direction; it helps defeat fingerprinting by blocking third-party requests to companies that are known to participate in this form of system snooping. Mozilla has partnered with **Disconnect.me** which offers free and paid-for anti-tracking tools for mobile and desktop browsers. An optional Disconnect add-on shows graphically any blocked trackers but I found it blocked some Ebay functions as well. In addition, my browser has blocked more than 10,000 trackers a month thanks to Firefox's Enhanced Tracking Protection.

Firefox has developed into a fast and powerful web browser that is worth a look and the new browser fingerprint countermeasures are likely to be welcomed. It's perhaps ironic that the first mainstream browser to actively help defeat 'cookieless monsters' is derived indirectly from Netscape, which created cookies in the first place.

Other news

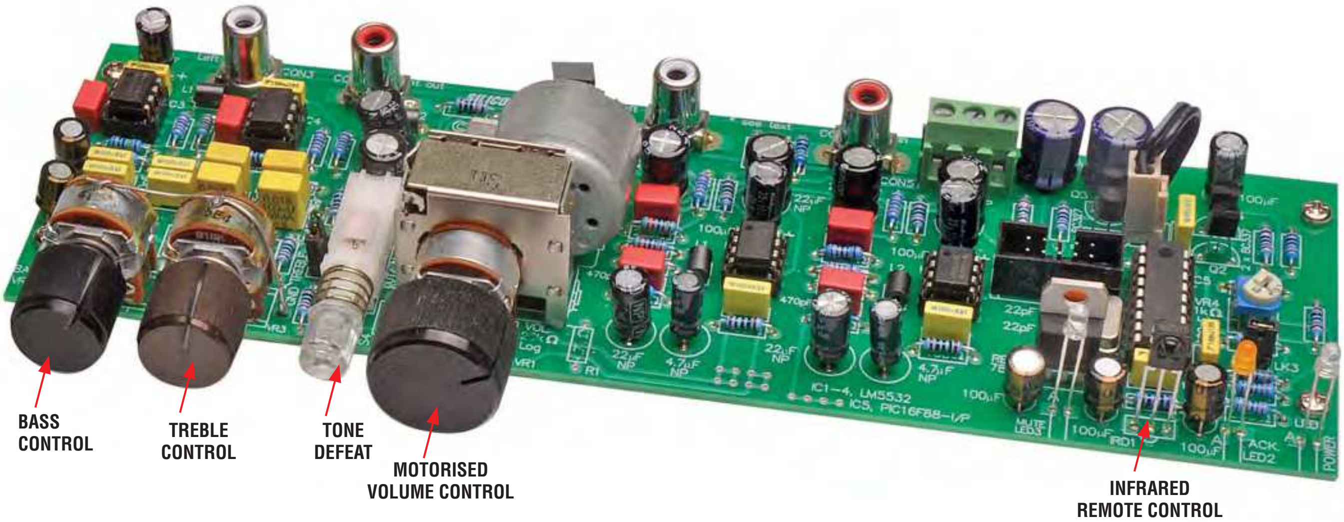
SpaceX launched its fourth crop of Starlink satellites at the end of January in its quest to offer low-cost Internet access around the globe. So far, so good, unless you're an astronomer. What could be the real reason for pumping billions of dollars into launching a constellation of cheap satellites offering broadband access? The website **Brilliant.org**, which offers online courses supporting STEM and engineering topics, offered a compelling insight into the technology as well as a thought-provoking critique of the business model behind Starlink. They also explain the trade-off between latency and area of coverage, highlighting some major commercial benefits that satellites flying in low-earth orbit may offer. A matrix of 12,000 Starlink satellites will circle the globe and inter-communicate using lasers, they think; you can see more in Brilliant's must-watch YouTube video at: **https://youtu.be/giQ8xEWjnBs**

In a move that has infuriated the US administration, Britain has opted to allow Huawei to play a very limited role in building the UK's 5G network. The British government is confident that any supposed risks can be managed and mitigated by the country's security services. The US embargo on Huawei and Britain's involvement with the UK's Huawei Cyber Security Evaluation Centre (HCSEC) was discussed in *Net Work*, August 2019.

Facebook will pay $550m to settle a class-action privacy lawsuit in the US covering its 'tagging' feature that used facial recognition to identify those appearing in photos, in breach of biometric privacy laws in the state of Illinois. Facebook's tagging function is now an opt-in feature. Meantime, London's Metropolitan Police is activating a network of overt live facial recognition (LFR) cameras in busy areas, linking to a database of wanted persons in an effort to apprehend villains or maybe locate missing persons. What could possibly go wrong?

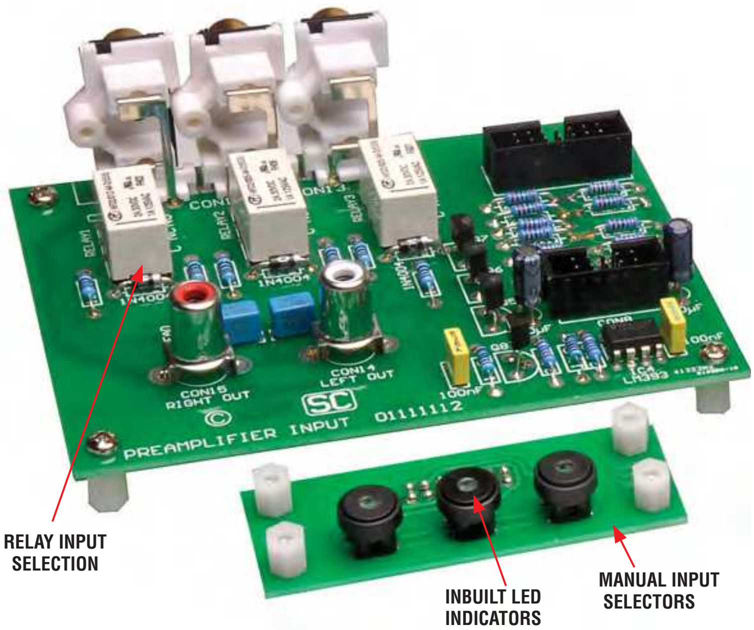
That's all from *Net Work* this month – see you next month!

The author can be reached at:
alan@epemag.net



Ultra-low-distortion Preamplifier with Tone Controls Part 1

We have published many fine low-distortion stereo preamplifiers. But often without tone controls – and we are regularly asked, 'how do I add tone controls?' Well, this design not only has tone controls but also has infrared remote volume control, input switching and muting. Meet the 2020 *Ultra-low-distortion Preamplifier!*



Features

- Very low noise and distortion
- Remote-controlled input selection and volume control with muting
- Manual volume control plus bass and treble cut/boost controls
- Tone control defeat switch bypasses bass and treble controls
- Minimal interaction between tone controls
- Can be used with just about any power amplifier
- Designed to be mounted in the front of a stereo amplifier chassis, but is also suitable for standalone use
- Three status LEDs
- Runs from ±15V DC

by John Clarke

Constructors – please read note in the Parts List before purchasing components.

We present a high-quality, low-distortion and low-noise stereo preamplifier that can be used with just about any power amplifier modules to form a stereo amplifier. It can also be used as a standalone preamplifier.

A low-cost infrared remote control is used to switch between three separate inputs, adjust the volume or temporarily mute the output.

It also includes manual volume, bass and treble controls and pushbuttons to select between the three stereo inputs.

LED indicators in the pushbuttons show which input is active. It also has power, acknowledge and mute status LEDs. All in all, it offers considerable advantages over previous models.

You could use it with the easy-to-build, low-cost SC200 amplifier modules (January-March 2018; Altronics kit Cat K5157). Or build it in a case and use it with an existing power amplifier. It's up to you.

It has a motorised potentiometer for volume control, so you can adjust the volume directly with a knob if you don't want to use the remote. It has an effectively infinite number of possible volume settings, unlike most digital volume controls, which can have quite large steps.

This *Preamplifier* provides wide-range bass and treble adjustment knobs to allow you to overcome deficiencies in your loudspeakers, compensate for the room response or just adjust the sound to be the way you like it.

While the performance is excellent when the tone controls are active, we have provided the option to bypass them using a push-on, push-off switch. Its integrated LED indicator shows when the tone controls are switched in or out.

This switch has three benefits. One, it's difficult to centre the tone controls precisely when you want the response to be flat, so the switch provides an easy way to achieve that. Two, it provides slightly better performance with the tone controls switched out. And three, it gives you an easy way to hear exactly what effect the tone controls are having, by toggling them on and off.

A PIC microcontroller is used to provide the remote control, muting and input selection functions.

Input selection is by way of a separate PCB interconnected to the main preamplifier using 10-way ribbon cable. If you don't need the input selector, you can build the project without it.

The micro remembers the last input selection, so it will go back to the same set of inputs even if it's switched off and on again.

Specifications *(2.2V RMS in/out, 20kHz bandwidth unless otherwise stated):*

Frequency response:	flat from 20Hz to 20kHz (see Fig.3), -1.25dB @ 100kHz
Bass adjustment range:	±15dB at 20Hz; ±13dB at 75Hz
Treble adjustment range:	±15dB at 20kHz; ±14dB at 10kHz
Input impedance:	22kΩ
Output impedance:	100Ω
THD+N:	<0.001%, 80kHz bandwidth;
	typically <0.0003%, 20kHz bandwidth (see Fig.1)
Signal-to-noise ratio:	-121dB with tone controls out; -114dB with tone controls in
Channel separation:	>80dB @ 1kHz; >67dB @ 10kHz (see Fig.4)
Input separation:	>98dB @ 1kHz; >80dB @ 10kHz
Maximum gain:	two times (6dB)
Signal handling:	up to 4V RMS input, 8V RMS output

Performance

This preamplifier has excellent performance. It uses low-distortion, low-noise op amps throughout, plus we have taken great care to specify very linear types of capacitor and to keep resistor values low, where their Johnson (thermal) noise contribution is likely to affect the signal.

Inevitably, the tone control circuitry adds some noise when it is switched in. But performance is still very good with the tone controls in, giving a THD+N figure of just 0.00054% at 1kHz and 0.0007% at 10kHz. By comparison, with the tone controls out, those figures become 0.00044% and 0.00048% respectively – see Fig.1.

Those measurements were made with a bandwidth of 20Hz-80kHz, which is necessary to measure distortion at higher frequencies accurately. But such a measurement includes a significant amount of ultrasonic noise (ie, in the 20-80kHz range). And Fig.1 shows that the distortion performance is dominated by noise.

So we also made measurements with a 20Hz-22kHz bandwidth, shown in blue on Fig.1, and this reveals that the true audible distortion and noise level is closer to 0.00025% – an astonishingly low figure.

Fig.3 shows the frequency response with the tone control at either extreme, and switched out (the blue curve). This demonstrates that when you're not using the tone controls, the frequency response is very flat. You can barely see the deviation on this plot; zooming in, we can see that the response is down only 0.2dB at 20Hz and less than 0.1dB at 20kHz.

Fig.4 shows the coupling between channels, which is typically less than –80dB, and the coupling between adjacent inputs, typically around –100dB. So isolation between channels and inputs is very good. The signal-to-noise ratio figure is especially good; over 120dB with a 2.2V RMS input signal (typical for CD/DVD/Blu-ray players), the tone controls switched out and the volume pot at unity gain.

In summary, you can be confident when using this *Preamplifier* that it will not negatively affect the audio signals passing through it, regardless of whether you are using the tone controls.

Capacitor and potentiometer selection

We mentioned earlier that we're using linear capacitor types where that's important, and also keeping resistance values low to minimise thermal noise.

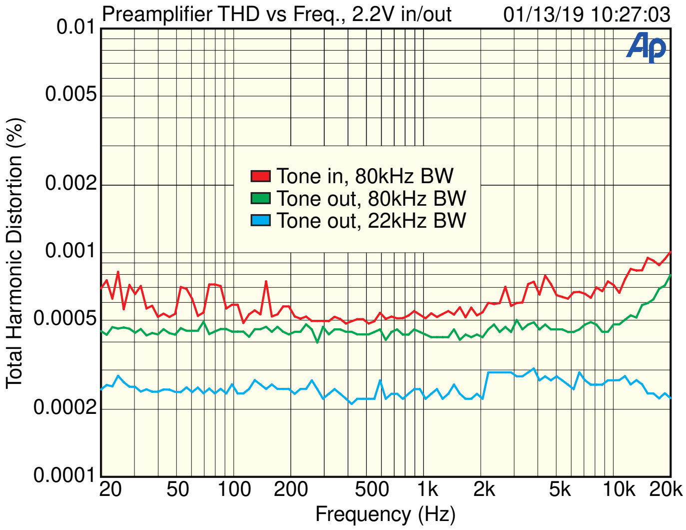
For capacitors between 10nF and 100nF, we have specified MKT polyester (plastic dielectric) types. While polyester is not quite as linear as polypropylene or polystyrene dielectrics, none of those capacitors are critical enough to cause a measurable increase in distortion, as demonstrated by our performance graphs.

But there are some capacitors with values below 1nF where the dielectric is important and this presents us with some difficulty, since MKT capacitors with values below 1nF are not particularly easy to get.

However, we've found them (see parts list) and that is what we have used in our prototype, with good result.

If you can get MKP (polypropylene) capacitors instead, those will certainly work well and we would encourage that. But we have also mentioned the possibility of using NP0 ceramics. We have tested these in the past and found that they are just as good as the best plastic dielectrics in situations where linearity is critical.

However, do be careful because many ceramic capacitors are not NP0 (also known as C0G) types, especially values above 100pF. Fig.5 shows a distortion plot for a simple low-pass filter comparing two capacitors of the same value, one polypropylene and one ceramic (not NP0/C0G). As you can see, the ceramic capacitor produces a lot more distortion. So make sure you use one of the types specified.



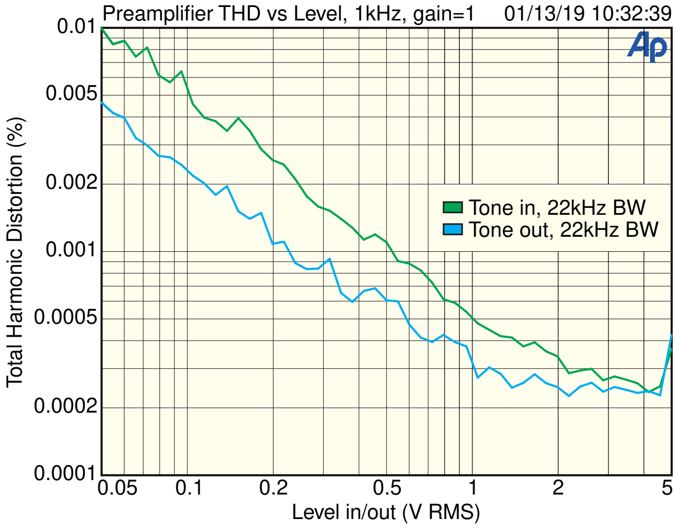


Fig.1: distortion across the entire range of audible frequencies is extremely low, whether the tone controls are active or not. There is a slight rise in distortion above 10kHz, but below that, the distortion is below the noise floor.

Fig.2: this shows the effect of noise; as you reduce the volume and thus the output signal level, the fixed circuit noise becomes larger in proportion and so total harmonic distortion goes up. However, even at very low volume levels, it's below 0.01% so it won't be noticeable.

Regarding resistance, you may find it a bit strange that we have specified a 5kΩ volume control potentiometer as values in the range of 10kΩ-100kΩ are more commonly used. But we have chosen 5kΩ because the thermal noise contribution of the volume control pot can be a major limiting factor in the performance of a low-distortion preamplifier and suitable motorised pots are available.

Op amps IC1a and IC2a buffer the signal from the source so that it does not have to drive the 5kΩ impedance; these op amps are more than capable of driving such a load without increased distortion.

If you can't get the 5kΩ motorised pot (available from Altronics; see parts list), you can use a 20kΩ pot instead; also a pretty standard value.

In that case, we have made provision for two 4.7kΩ shunt resistors to lower the impedance seen by the following stage, giving you most of the performance benefits of a 5kΩ pot. These have minimal effect on the pot curve, so it still works well as a volume control.

Fig.6 shows the difference in distortion with and without these shunts (the signal level is lower here than in the other figures, hence the higher base level). The performance with the proper 5kΩ pot is slightly better again.

Remote control

Pressing the Volume Up or Volume Down buttons on the infrared remote causes the motorised pot to rotate clockwise or anticlockwise. It takes about nine seconds for the pot to travel from one end to the other using these controls.

For finer adjustment, the Channel Up and Channel Down buttons on the remote can be used instead. These cause the pot shaft to rotate about one degree each time one of these buttons is briefly pressed. Holding one of these buttons down rotates the pot from one end to the other in about 28 seconds.

If any of these buttons is held down when the pot reaches an end stop, a clutch in the motor's gearbox begins to slip so that no damage is done to the motor.

The code also provides a convenient automatic muting feature. Press the Mute button on the remote and the volume control pot automatically rotates to its minimum position and the motor stops. Hit the button again and it returns to its original position. If you don't want the pot to return all the way to its original setting, you can simply increase the volume to your desired new level instead.

So how does the unit remember its original setting during muting? The microcontroller monitors the time it takes for the pot to reach its minimum setting and the minimum pot setting is detected when the load on the motor increases at the potentiometer end stop, as the clutch begins to slip. When the Mute button is pressed again, power is applied to the motor drive for the same amount of time, rotating it back to the original position.

The orange 'Ack' LED flashes whenever an infrared signal is being received from the remote, while the yellow Mute LED flashes while the muting operation is in progress and then remains on when the pot reaches its minimum setting.

Circuit description

Fig.7 shows the main preamplifier circuit – but for clarity, only the left channel components are included. The right channel is identical and the matching part designators are provided, in brackets. The following description refers to the left-channel part names.

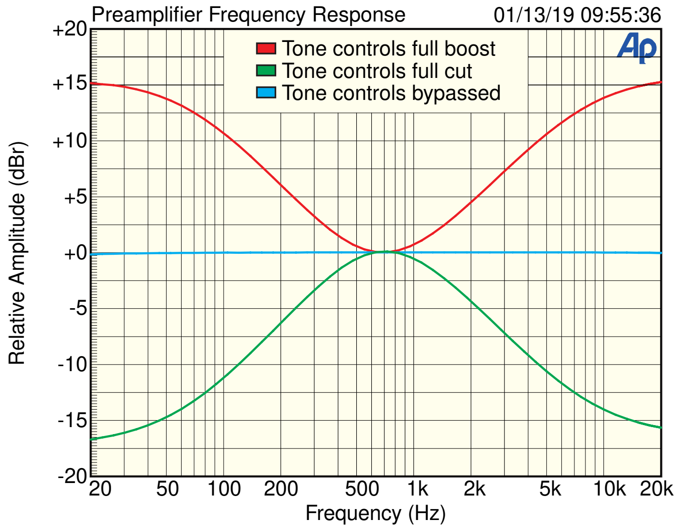
The audio signal from the *Input Switching* board is AC-coupled to the input of the first op amp (IC1a) via a 22µF non-polarised (NP) electrolytic capacitor and 100Ω resistor. A 22kΩ resistor to ground provides input DC biasing and sets the input impedance to around 22kΩ. The 100Ω resistor, ferrite bead and 470pF capacitor form a low-pass filter to attenuate radio frequency (RF) signals ahead of the op amp input.

IC1a operates as a voltage amplifier with a gain of two, due to the two 2.2kΩ feedback resistors. The 470pF capacitor combines with the feedback resistors to roll off the top-end frequency response, with a –3dB point at about 150kHz. This gives a flat response over the audio spectrum while eliminating the possibility of high-frequency instability or RF demodulation.

IC1a's pin 1 output is fed to the top of volume control potentiometer VR1a (5kΩ log) via a 22µF non-polarised capacitor. The signal on its wiper is then AC-coupled to the pin 5 non-inverting input of IC1b via a 4.7µF non-polarised capacitor.

This coupling arrangement prevents direct current from flowing through any part of the volume control potentiometer, VR1. Even a small direct current can cause noise when the volume is adjusted.

As mentioned earlier, the circuit was designed for a 5kΩ motorised volume control pot, as this results in good noise performance. However, if you



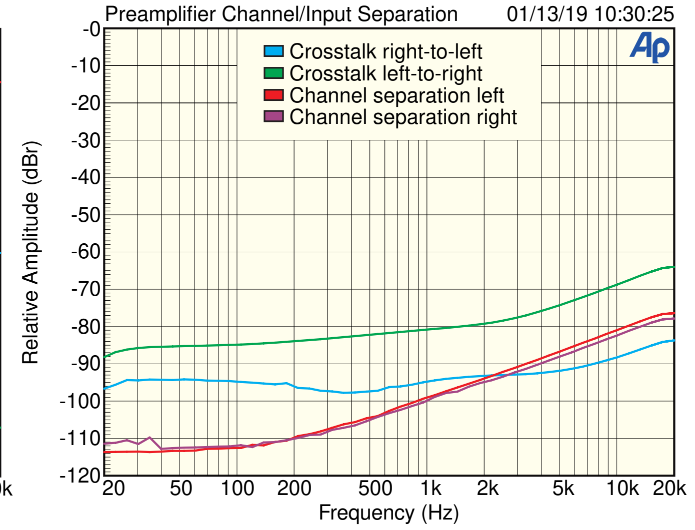


Fig.3: the blue line shows the *Preamplifier's* frequency response with the tone controls switched out; you can see that it's very flat, varying by only 0.2dB across the entire audible frequency range. The red and green curves demonstrate the range possible with bass and treble adjustments.

Fig.4: the crosstalk and separation figures are good. Crosstalk is how much of the left channel signal feeds into the right channel or vice versa. Channel separation is how much signal from input #1 couples into input #2 or vice versa.

can't get one you can use a more common 20kΩ potentiometer and fit resistors R1 and R2, so that the circuitry has a similar impedance, resulting in the same overall frequency response.

IC1b operates as a unity-gain buffer and provides a low-impedance output regardless of the volume control setting. Its pin 7 output is fed to the tone control section and also to switch S4a. When S4a is set to the 'tone out' position, the output from IC1b is coupled via the 22µF capacitor to output socket CON3, via a 100Ω resistor. Therefore, the tone controls are effectively out of circuit.

The 100Ω resistor isolates the op amp output from any capacitive loads that might be connected to ensure stability. This resistor and ferrite bead in series with the output also attenuate any RF noise which may have been picked up by the board.

Tone controls

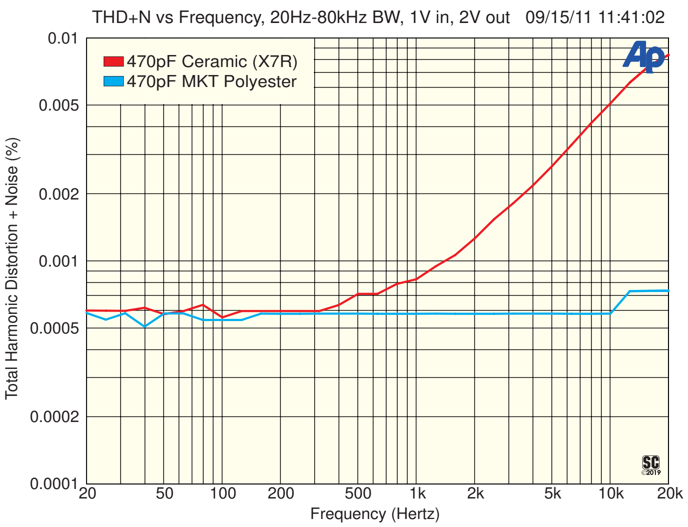
When S4a is in the 'tone in' position, output CON3 is instead driven from the tone control circuitry, so potentiometers VR2a and VR3a adjust the amount of bass and treble in the signal.

Op amp IC3a forms the active tone control in conjunction with VR2a and VR3a and associated resistors and capacitors. The bass and treble tone circuitry is a traditional Baxandall-style design. This is an inverting circuit, so it's inverted again by unity-gain buffer IC3b to restore the original signal phase.

When the wipers of potentiometers VR2a and VR3a are centred, the impedance between output pin 1 of IC3a and each wiper is equal to the impedance between the wiper and output pin 7 of IC1b. So in this condition, IC3a operates as a unity-gain inverting amplifier for all audio frequencies. Therefore, in this case, the tone controls have little effect on the signal – they just add a little noise.

Bass adjustment

The bass control (VR2a) provides cut (anti-clockwise) or boost (clockwise) to low frequencies. The impedance of each of the two 100nF capacitors for high-frequency signals is low and so they can bypass VR2a entirely.



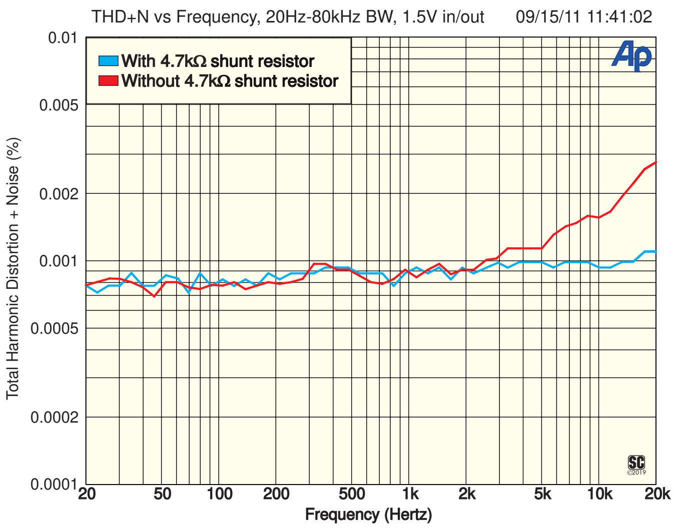
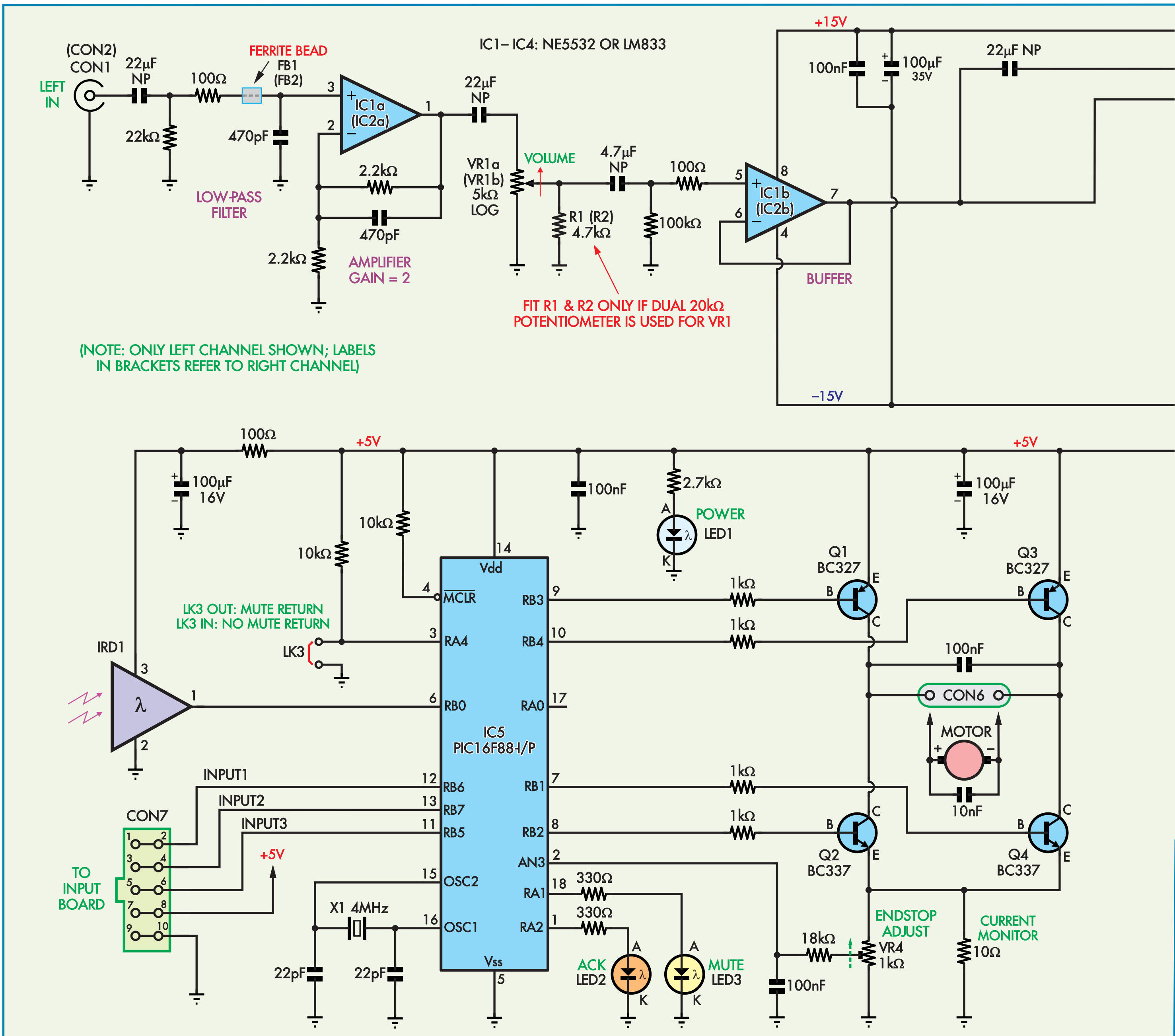


Fig.5: distortion versus frequency of a simple low-pass filter using either a 470pF MKT capacitor or a 470pF ceramic (non-NP0/C0G) capacitor. As you can see, distortion rises dramatically at higher frequencies with the ceramic capacitor due to its non-linearity and its lower impedance at higher frequencies, which causes it to shunt more of the signal and thus have a stronger effect.

Fig.6: if you must use a 20kΩ motorised potentiometer to build this *Preamplifier*, fitting the two extra 4.7kΩ resistors (R1 and R2) will keep high-frequency distortion low, by lowering the input impedance seen by the following buffer stage. This allows it to perform optimally and also lowers thermal noise.



Low-noise Preamplifier with tone controls and remote volume control

Any change in the position of VR2a's wiper will thus have little effect on high frequencies. For example, at 1kHz, the 100nF capacitors have an impedance of 1.6kΩ each. That is considerably lower than the 5kΩ value of the half of the potentiometer track that they are connected across when VR2a is centred and therefore the capacitors shunt much of the signal around VR2a.

But at 20Hz, the 100nF capacitors have an impedance of 80kΩ and so minimal current passes through them; almost all of it goes through VR2a. Therefore, VR2a has a significant effect on the amplitude of a 20Hz signal and so it provides much more boost or cut at lower frequencies.

When VR2a is rotated clockwise, the resistance from output pin 1 of IC3a to its wiper increases, while the resistance from the wiper to the input signal decreases, providing increased amplification. And when rotated anti-clockwise, the opposite occurs, decreasing amplification. Because the capacitors shunt a different amount of signal around the pot at different frequencies, this gain is also frequency-dependent.

The 1.8kΩ resistors set the maximum boost and cut range. They have been chosen to allow up to ±15dB adjustments at around 20Hz, dropping to around ±1dB at 1kHz. The measured frequency response with the controls at minimum, centred and at maximum is shown in Fig.3.

Treble adjustment

Treble control VR3a operates differently to VR2a. It is configured to have more effect on higher frequency signals. This is achieved by connecting capacitors in series with the pot channel, rather than across it.

At low frequencies, the 15nF capacitors have a high impedance, eg, 106kΩ at 100Hz. This is very high compared to the 10kΩ channel resistance and so most of the feedback signal at this frequency will flow through the bass network, which has a DC resistance of 13.6kΩ and therefore a much lower impedance. So VR3a will have little effect on the gain at low frequencies.

At high frequencies, the 15nF capacitors have a lower impedance, eg, around 1kΩ at 10kHz and so the treble controls are effectively brought into circuit, providing adjustable gain similarly to the circuitry surrounding VR2a. The 1kΩ resistors at each end of VR3a set the maximum boost or cut for

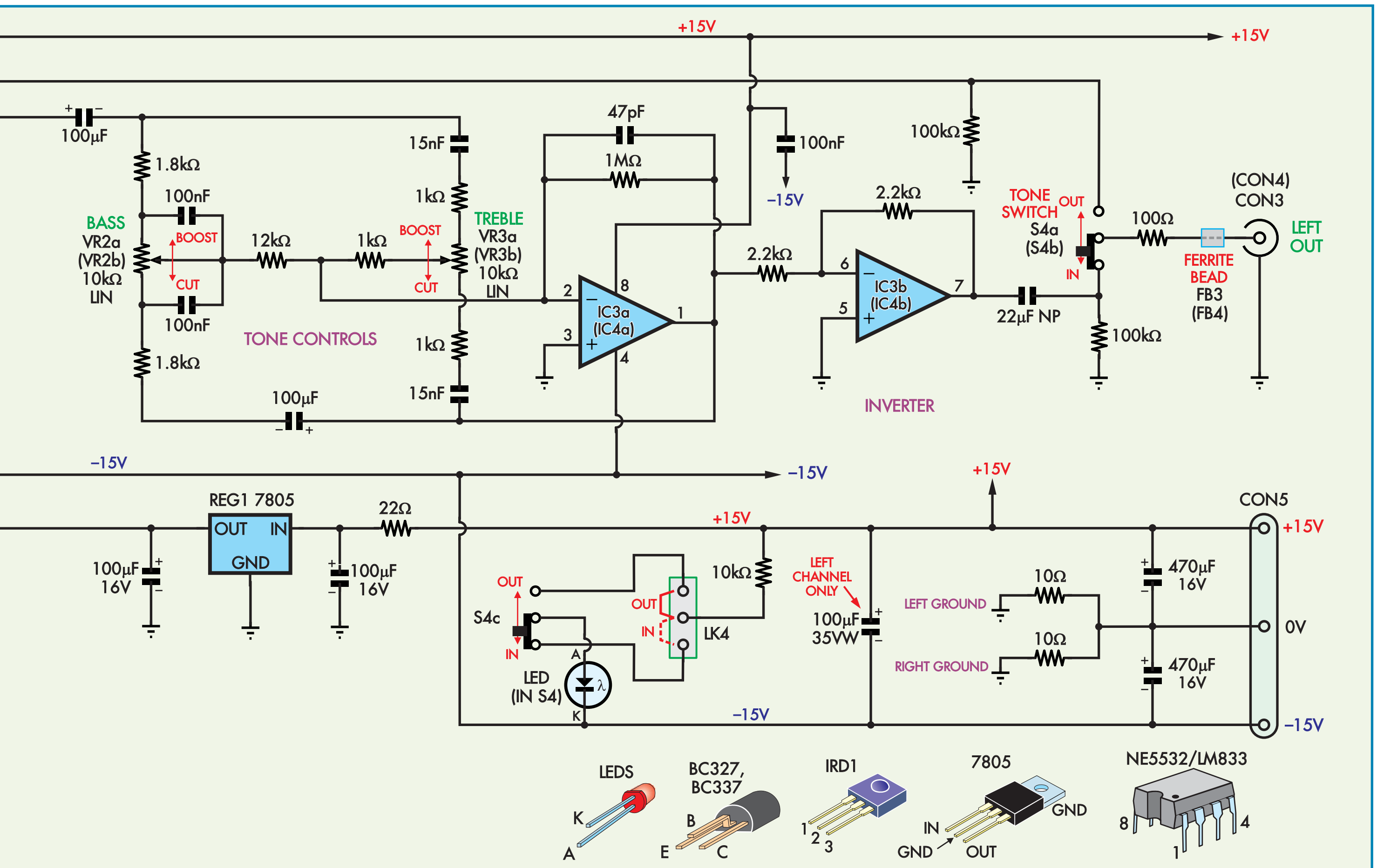


Fig.7: here's the circuit diagram for the main *Preamplifier* PCB, incorporating the volume and tone controls and tone switching (at the top) and the infrared remote volume control and input switching circuitry (at bottom). The analog signal path is built around dual low-noise op amps IC1-IC4 and motorised potentiometer VR1. The volume control and input selection circuitry is based on microcontroller IC5, motor driver transistors Q1-Q4 and infrared receiver IRD1.

high frequencies, up to around ±15dB, similar to the bass control. You can see this in Fig.3.

The 12kΩ and 1kΩ resistors between the bass and treble potentiometer wipers minimise the inevitable interaction between the two controls.

Note that while the treble potentiometer is isolated from direct current flow due to the 15nF capacitors in series, the bass potentiometer requires two extra 100µF capacitors. These do not affect the action of the bass control; they are just there to block direct current flow through VR2a. This is for the same reason that DC is blocked for VR1; to prevent noise during adjustments.

The 1MΩ feedback resistor between pins 1 and 2 of IC3a provides DC bias for the pin 2 input, while the 47pF capacitor prevents high-frequency oscillation of the op amp by reducing the gain at ultrasonic frequencies.

When S4a is set to the 'tone in' setting, the output from IC3b (reinverting IC3a's signal inversion) is then fed to the CON3 output as mentioned above. Another pole of the switch (S4c) controls the indicator LED that is contained within the switch. It is powered from the ±15V supplies via a 10kΩ resistor and therefore receives about 3mA.

Jumper link LK4 can be removed to prevent this LED from lighting, or moved into one position or the other to invert its function. In other words, LK4 selects whether the LED lights when the tone is in or out. Note that the 'tone out' position of S4 is when the switch is pressed in. In other words, it acts like a defeat switch.

Remote control circuitry

The Remote Control circuitry is also shown in Fig.7. Signals from the handheld remote are picked up by infrared receiver IRD1. This is a complete infrared detector and processor. It picks up the 38kHz pulsed infrared signal from the remote and amplifies it to a constant level. This is then fed to a 38kHz bandpass filter, after which it is demodulated to produce a serial data burst at its pin 1 output.

The resulting digital data then goes to the RB0 digital input (pin 6) of PIC16F88-I/P microcontroller IC5 for decoding. Depending on the button pressed on the remote, IC5 either drives the volume control motor (via an external transistor circuit) to change the volume, or sends one of its RB6, RB7 or RB5 output low to select a new input.

The input routing is controlled by the *Input Selector* board which is connected via CON7.

IC5 is programmed for a remote control which sends Philips RC5 codes. It supports three different sets of RC5 codes, normally referred to as TV, SAT1 or SAT2. You must also program the universal remote control with the correct number for one of these sets of code. We will explain how to do that next month. You also need to set IC5 to expect the correct set of codes; we will also describe that next month.

Driving the pot motor

IC5's RB1-RB4 outputs (pins 7-10) drive the bases of transistors Q1-Q4 via 1kΩ resistors. These transistors are arranged in an H-Bridge configuration and control the motor. The motor is off when the RB1-RB4 outputs are all high. In that state, RB3 and RB4 turn PNP transistors Q1 and Q3 off, while RB1 and RB2 turn NPN transistors Q2 and Q4 on.

As a result, both terminals of the motor are pulled low and so no current flows through it and it won't rotate.

The emitters of Q2 and Q4 both connect to ground via a common 10Ω resistor, which is used for motor-current sensing. The transistors operate in pairs so that the motor/potentiomenter can be driven in either direction, to increase or decrease the volume.

To drive the motor clockwise, RB2 goes low and turns off transistor Q2, while RB3 goes low and turns on Q1. When that happens, the left-hand terminal of the motor is pulled to +5V via Q1, while the right-hand terminal is pulled low via Q4. As a result, current flows through Q1, through the motor and then via Q4 and the 10Ω resistor to ground.

Conversely, to turn the motor in the other direction, Q1 and Q4 are switched off and Q2 and Q3 are switched on. As a result, the right-hand motor terminal is now pulled to +5V via Q3, while the left-hand terminal is pulled low via Q2.

Regardless of the direction of rotation, current flows through the 10Ω shared emitter resistor and so the voltage across it varies with the current drawn. Typically, the motor draws about 40mA when driving the potentiometer but this rises to over 50mA when the clutch is slipping. As a result, there is about 0.4-0.5V drop across the 10Ω resistor.

This is ideal because the motor is rated at 4.5V and the result of subtracting the resistor voltage from the 5V supply is that it provides the correct motor voltage.

Current sensing and muting

Once the potentiometer has reached full travel in either direction, a clutch in the motor's gearbox begins to slip. This prevents the motor from stalling and possibly overheating if the button on the remote continues to be held down. The clutch mechanism also allows the user to rotate the pot shaft manually.

As mentioned earlier, when you press the mute button on the remote control, the volume control is rotated fully anti-clockwise. Microcontroller IC5 detects when the wiper reaches its end stop by detecting the increase in the motor current when the limit is reached and the clutch slips. That's done by taking a sample portion of the voltage across the 10Ω resistor using trimpot VR4.

The voltage at VR4's wiper is filtered using an 18kΩ resistor and a 100nF capacitor to remove the motor commutator hash and is applied to lC5's analogue AN3 input (pin 2). IC3 then measures the voltage on AN3 to a resolution of 10 bits, or about 5mV ($5V \div 2^{10}$).

Provided this input is below 200mV, the PIC microcontroller allows the motor to run. However, as soon as the voltage rises above this 200mV limit, the motor is stopped. When the motor is running normally, the current through it is about 40mA, which produces 0.4V across the 10Ω resistor. VR4 attenuates this voltage and is adjusted so that the voltage at AN3 is slightly below the 200mV limit.

Note that the AN3 input is monitored only during the muting operation. At other times, when the volume is being set by the Up or Down buttons on the remote, the clutch in the motor's gearbox assembly slips when the potentiometer reaches its clockwise or anticlockwise limits.

As described above, pressing Mute on the remote again *after* muting returns the volume control to its original setting, by driving it clockwise for the same amount of time that it was driven anti-clockwise to reach its end stop.

This mute return feature in the software is enabled by leaving shorting link LK3 open. This allows the RA4 input (pin 3) to be pulled to 5V by a 10kΩ resistor. Installing the jumper shunt at LK3 will pull RA4 to ground, disabling the mute return feature.

Status LEDs

LEDs1-3 indicate the status of the circuit. The blue Power LED (LED1) lights whenever power is applied to the circuit. The other two LEDs, Acknowledge (LED2) and Mute (LED3) light when their respective RA2 and RA1 outputs are driven high (ie, to +5V). LED2 indicates that an infrared command was received and LED3 lights when the mute function is active.

Pins 15 and 16 of IC5 connect to the oscillator which drive 4MHz crystal X1, providing the microcontroller system clock. This oscillator runs when the circuit is first powered up for about 1.5 seconds. It also runs whenever an infrared signal is received at RB0 or when a button on the front panel switch board is pressed and then for a further 1.5 seconds after the signal ceases.

The oscillator then shuts down and the processor goes into sleep mode, as long as a muting operation is not in process. This ensures that no noise is radiated into the sensitive audio circuitry when the remote control circuit is not being used.

A 10nF capacitor connected directly across the motor terminals also prevents commutator hash from being transmitted along the supply leads, while further filtering is provided by a 100nF capacitor located at the motor output terminals on the PCB. This reduces the amount of noise that gets into the preamplifier signals when the volume pot motor is being driven.

Input selection

Digital outputs RB6, RB7 and RB5 of IC5 (pins 11-13) control the relays on the *Input Selector* board. These outputs go low when the 1, 2 or 3 buttons on the remote are pressed respectively; they are high-impedance (set as inputs) the rest of the time. As shown, RB6, RB7 and RB5 are connected to pins 1-6 of 10-way header socket CON7; each output is connected to two pins in parallel.

Pins 7 and 8 of CON7 are wired to the +5V rail while pins 9 and 10 go to ground. CON7 is connected to a matching header socket on the *Input Selector* board via an IDC cable. This provides both the control signals and the supply rails to power this module.

The *Input Selector* circuit is shown in Fig.8. It uses three 5V DPDT relays (RLY1- RLY3) to select one of three stereo inputs: Input 1, Input 2 or Input 3. The relays are driven by PNP transistors Q5-Q7, depending on the signals from the IC5 microcontroller in the *Remote Control* circuit (and fed through from CON7 to CON8).

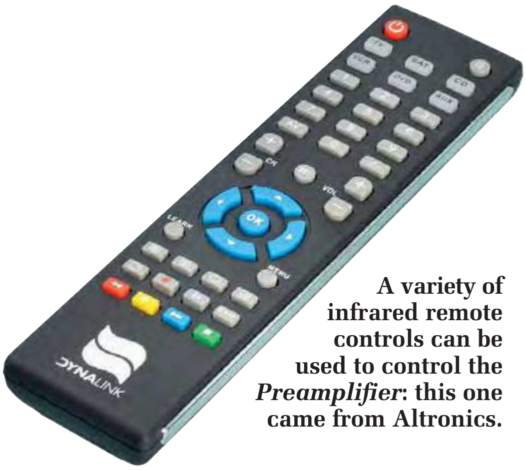
One relay is used per stereo input so that the audio signal only has to pass through one relay. As shown, the incoming stereo line-level inputs are connected to the NO (normally open) contacts of each relay. When a relay turns on, its common (C) contacts connect to its NO contacts and the stereo signals are fed through to the left and right outputs via 100Ω resistors and ferrite beads.

The resistors isolate the outputs from the audio cable capacitance, while the beads and their associated 470pF capacitors filter any RF signals that may be present.

When button 1 is pressed on the remote, pins 1 and 2 on CON8 are pulled low (by output RB6 of IC5 in the *Remote Control* circuit). This pulls the base of transistor Q5 low via a 2.2kΩ resistor and so Q5 turns on and switches on RLY1 to select input 1 (CON11). Similarly, RLY2 and RLY3 are switched on via Q6 and Q7 respectively when buttons 2 and 3 are pressed on the remote.

Only one relay can be on at any time. Pressing an input button (either on the remote or the switch board) switches the currently activated relay off before the newly selected relay turns on. If the input button corresponds to the currently selected input, then no change takes place. The last input selected is restored at power up.

Fig.9 shows the circuitry for the separate front panel Pushbutton Switch Board. This consists of three momentary contact pushbuttons with integral blue LEDs (LEDs1-3) plus a 14-way header socket (CON10) which is connected to CON9 via an IDC cable.



A variety of infrared remote controls can be used to control the *Preamplifier*: this one came from Altronics.

One side of each switch is connected to ground, while the other connections to S1-S3 are respectively connected back to the RB6, RB7 and RB5 digital I/Os of IC5 in the *Remote Control* circuit.

When a switch is pressed, it pulls its corresponding pin low and this wakes the microcontroller up, which then turns on the corresponding relay and promptly goes back to sleep again. The anodes of LEDs1-3 are connected to +5V, while their cathodes are respectively connected to the RB6, RB7 and RB5 I/Os of IC5 (pins 11-13) via 2.2kΩ current-limiting resistors.

As a result, when one of these pins goes low to select a new input, it lights the corresponding switch LED as well. This occurs whether the input was selected using the remote control or pressing a switch button. The cathodes of the other LEDs are held high via 2.2kΩ pull-up resistors to the +5V rail and are off.

Note that the pins which are used to sense when buttons are pressed and drive the switch LEDs are the same pins which are used to drive the transistors which drive the relay coils. So if you press the button corresponding to the input which is already selected, that line is configured as an output but it's already low (at ground potential), so pressing the button has no effect.

If you press one of the other buttons, as mentioned earlier, that pin on IC5 has been configured as an input and there are 2.2kΩ pull-up resistors on the *Input Selector* board. So pulling that line to ground will bring that line low, signalling to the microcontroller that you wish to switch inputs, which will then switch off the relay selecting the currently active input.

Preventing switch conflicts

Comparator IC4 and NPN transistor Q8 prevent more than one relay from switching on if two or more input switches are pressed simultaneously. This circuit also ensures that the currently activated relay is switched off if a different input button is pressed, before the newly selected relay is switched on.

IC4 is an LM393 which is wired so that its non-inverting input (pin 3) monitors the three switch lines via 100kΩ resistors.

These resistors function as a simple DAC (digital-to-analogue converter). If one switch line is low, the voltage on pin 3 of IC1 is 3.3V; if two are low (eg, if two switches are pressed simultaneously), pin 3 is at 1.67V; and if all three lines are low, pin 3 is at 0V.

This pin 3 voltage is compared to a 2.5V reference on IC1's inverting input (pin 2), formed by a resistive divider across the 5V supply. So its pin 1 output is high only when one switch line is low and this turns on Q8 which connects the bottom of the relay coils to ground. This allows the selected relay to turn on.

However, if two or more switch lines are low, lC4's output will be low and so Q8 and all the relays turn off. Similarly, if one switch line is already low and another input is selected (pulling its line low), IC4's output will briefly go low to switch off all the relays before going high again (ie, when the micro changes the state of its RB5-RB7 outputs) to allow the new relay to turn on.

Power supply

The *Preamplifier* is powered from ±15V rails. These are typically derived either from two separate 15V windings on the main power transformer, or a small secondary 15-0-15 transformer and rectifier.

These 15V rails are bypassed on the preamp board by 470µF capacitors. There are other capacitors connected across the supply rails at various points of the circuit which provide local bypassing for the op amps on the PCB.

We use both 100nF capacitors and 100µF capacitors to ensure low impedance at a range of frequencies. The capacitors connected across the full 30V supply are rated at 35V or more.

The 5V supply for microcontroller IC5 is derived from the +15V rail via a 22Ω dropping resistor and 5V linear regulator REG1. The 22Ω resistor reduces the dissipation in REG1 and provides some additional filtering, in combination with REG1's 100µF input capacitor. The power LED (LED1) lights up when 5V is present and its current is set by a 2.7kΩ series resistor.

We published a suitable regulated supply design in the May 2016 issue: the *4-Output Universal Voltage Regulator*. It has adjustable outputs which can be set for ±15V, plus 5V and 3.3V outputs that could be used to power other circuitry in your preamplifier/amplifier. Its PCB (coded 18105151) is available from the *PE PCB Service*.

Construction

Fig.10 shows the assembly details for the main *Preamplifier* module. It is built on a PCB coded 01111119, which measures 216 × 66mm.

Begin by installing the resistors (use your DMM to check the values), followed by the four ferrite beads. Each bead is installed by feeding a resistor lead off-cut through it and then bending the leads to fit through their holes in the PCB. Push each bead all the way down so that it sits flush against the PCB before soldering its leads.

Following this, install the IC sockets for the five ICs. Make sure that each socket is seated flush against the PCB and that it is oriented correctly, as shown in Fig.10. Note that IC5 faces in the opposite direction to the op amp ICs (IC1-IC4). It's best to solder two diagonally opposite pins of a socket first and then check that it sits flush with the board before soldering the remaining pins.

The MKT and ceramic capacitors can now go in, followed by the electrolytic capacitors (regular and non-polarised). The electrolytic capacitors must be oriented with the correct polarity; ie, with the longer lead through the pad marked with a '+' symbol. The 100µF capacitors that are marked on the overlay and PCB with 35V must be rated at 35V or higher.

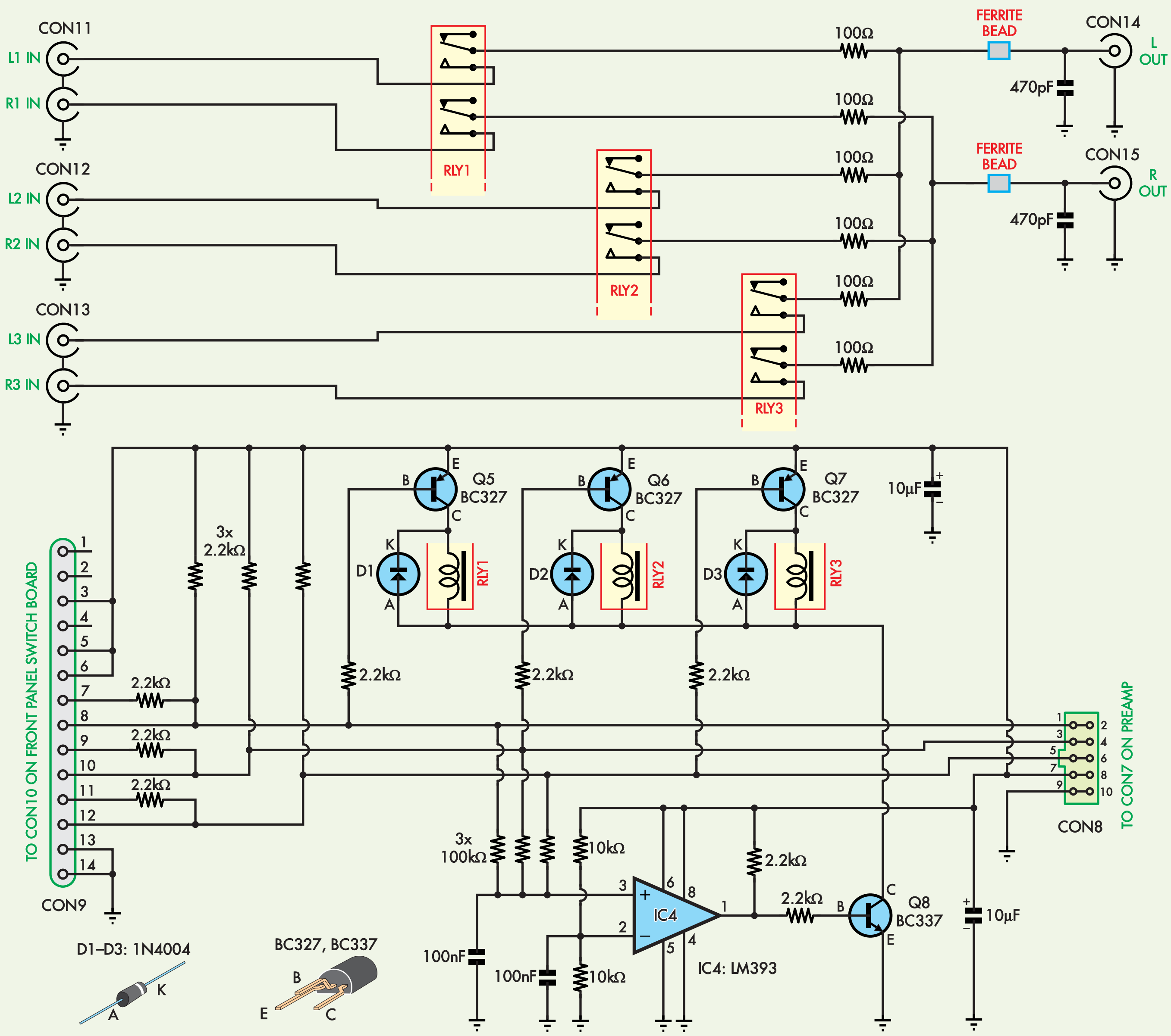
If you use ceramic 470pF or 47pF capacitors, make sure they are the specified NP0 (or the equivalent C0G) type. Using other types of ceramic capacitors in these positions will degrade the distortion performance.

The next step is to install the four transistors (Q1-Q4) in the remote control section. You need to ensure you use the correct type at each location. Q1 and Q3 are both BC327s, while Q2 and Q4 are BC337s.

The PC stake (near VR3), 2-way SIL pin header for LK3 and 3-way SIL header for LK4 can now be installed, followed by polarised pin header CON6 and box header CON7. Crystal X1, trimpot VR4, the 3-way screw terminal block (CON5) and the four vertical RCA sockets (CON1-CON4) can then be fitted.

Ensure the terminal block wire entry holes face the nearest edge of the PCB. Use white RCA sockets for the left channel input and output positions and red ones for the right channel positions.

Switch S4 can be mounted now. Take care that all the pins are straight before attempting to insert them into the PCB. Press the switch fully down onto the PCB before soldering each pin. Also fit REG1, taking care to orient this correctly.



Ultra-low-noise Preamplifier Input Selector

Fig.8: the circuitry of the optional module used for input switching. One of DPDT relays RLY1-RLY3 is energised at any given time, feeding one of the input pairs (CON11-CON13) through to CON14/CON15, which are wired to inputs CON1 and CON3 on the main *Preamplifier* board. IC4 and Q8 ensure that only one relay can be energised at a time, so the signal sources are not shorted to each other.

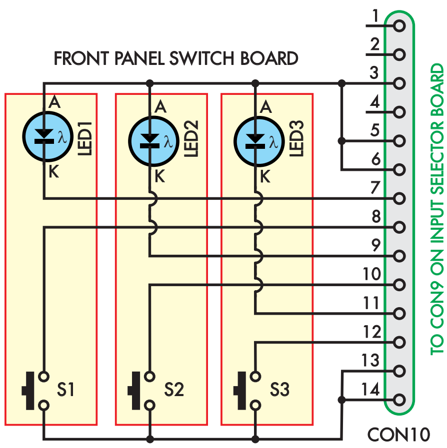


Fig.9: the circuitry on the front panel pushbutton switch board. LEDs 1-3 are actually inside the pushbutton switches and light when the corresponding input is selected

Mounting the pots

Before mounting the potentiometers, the shafts should be cut to length. The length depends upon the knobs and the type of box that the preamplifier is to be mounted into. The thickness of the front panel will have an impact on the required shaft length.

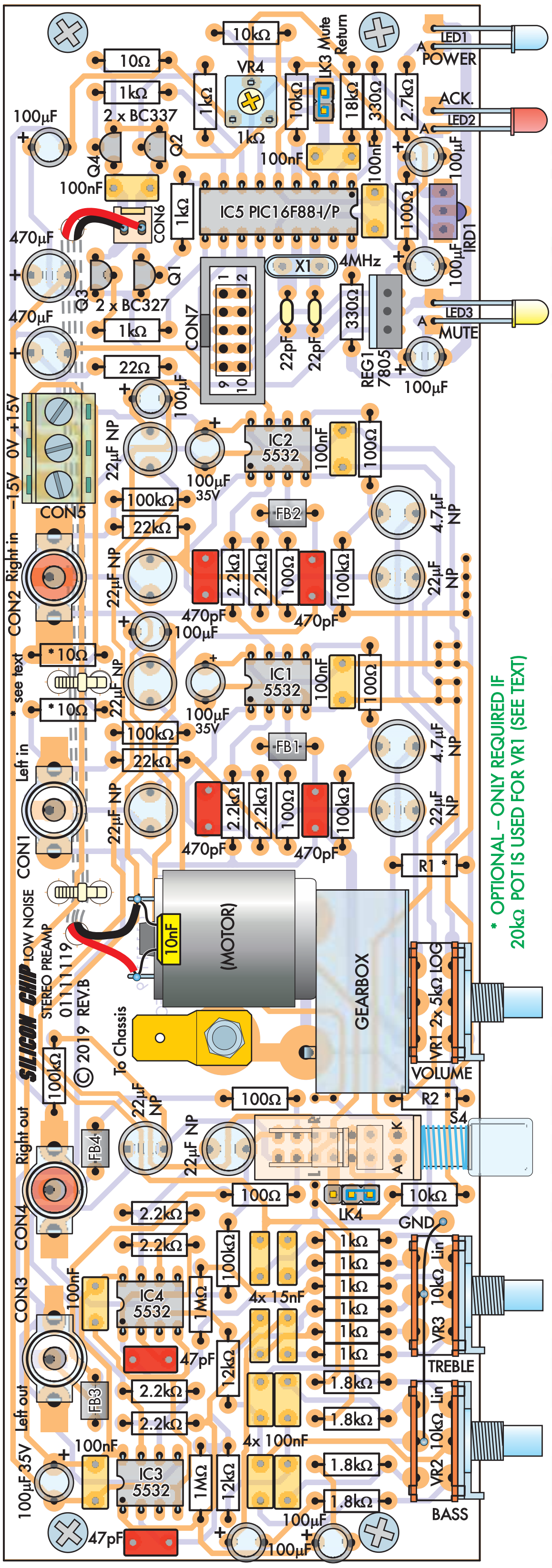
Make sure the motorised pot (VR1) is seated correctly against the PCB before soldering its leads. Once the pot fits correctly, solder two diagonally opposite pot terminals and check that everything is correct before soldering the rest. The two gearbox cover lugs can then be soldered.

That done, connect the figure-8 wire to the motor terminals along with the 10nF capacitor that also connects to these terminals.

These leads pass through a hole in the board immediately behind the motor. They are then secured to the underside of the PCB using cable ties and then brought up to the top side of the PCB just behind CON6.

Strip the wire ends and crimp them to the header pins. The wire from the positive motor terminal (marked with a red dot) should connect to the CON6 pin that is closer to IC5. Insert the pins into the 2-way shell and plug it into the CON6 header.

Before fitting VR2 and VR3, scrape off some of the coating on the top of the pot body using a file so that they can be soldered to. **Don't breathe in the resulting dust.** VR2 and VR3 must be seated correctly before being soldered to the board. They are then earthed using 0.7mm-



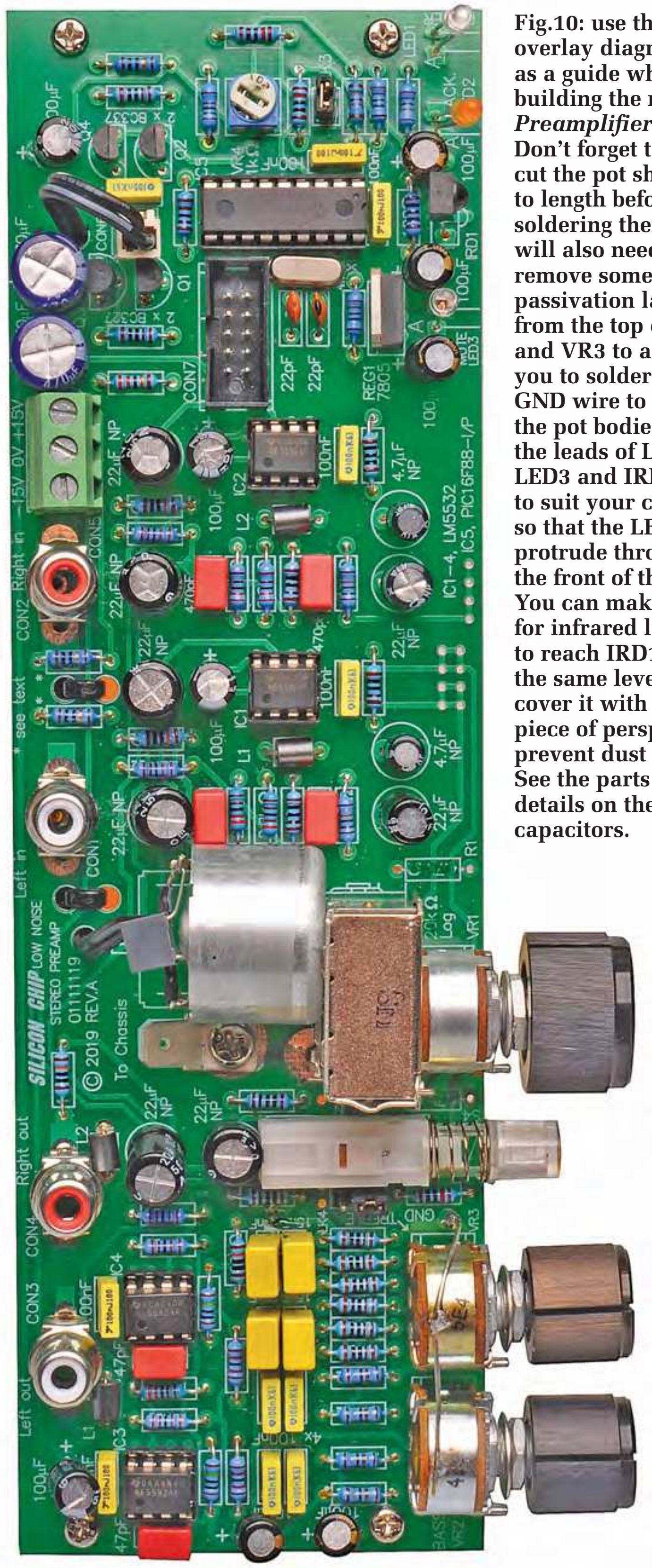


Fig.10: use this PCB overlay diagram as a guide when building the main *Preamplifier* board. Don't forget to cut the pot shafts to length before soldering them. You will also need to remove some of the passivation layer from the top of VR2 and VR3 to allow you to solder the GND wire to earth the pot bodies. Bend the leads of LED1-LED3 and IRD1 to suit your case, so that the LEDs protrude through the front of the case. You can make a hole for infrared light to reach IRD1 at the same level and cover it with a small piece of perspex to prevent dust ingress. See the parts list for details on the red capacitors.

diameter tinned copper wire soldered to the GND PCB stake and the top metal shield on both pots. Make sure that you apply sufficient heat for the solder to form a good joint.

Mounting the LEDs and IRD1

We mounted the infrared receiver IRD1 with its lens about 18mm above the PCB. Similarly, the LEDs were mounted with the base of the LED body 18mm above the PCB. This will allow sufficient length for the LED leads to be bent forward, to line up with the potentiometer shafts, and then poke forward through the front panel of the amplifier. When bending the LED leads, the longer (anode) leads must go into the PCB pads marked 'A'. IRD1 should be fitted with its hemispherical lens facing towards the front of the board.

The assembly can now be completed by installing the spade connector to the left of the motorised pot, secured with an M4 screw, shake-proof washer and nut.

Leave the ICs out of their sockets for now. They are installed later, and only after the power supply checks have been completed.

Conclusion

Next month, we'll describe the *Input Selector* module and *Switch Board* assemblies and detail the test procedure. We'll have more details on the power supply arrangement and describe setting up the remote control.

Parts list – 2020 Ultra-low-distortion Preamplifier with Tone Controls

Main module

1 double-sided PCB, code 01111119, 216 x 66mm
1 universal remote control [Altronics A1012 or similar]
1 dual-gang 5kΩ log motorised potentiometer (VR1) [Altronics R1998] (a 20kΩ log pot can be substituted)
2 dual-gang 10kΩ linear 16mm potentiometers (VR2,VR3) [Altronics R2296]
1 1kΩ mini horizontal trimpot (VR4)
3 knobs to suit VR1-VR3
1 4PDT push-on, push-off switch (S4) [Altronics S1451]
4 8-pin DIL IC sockets (for IC1-IC4)
1 18-pin DIL IC socket (for IC5)
4 ferrite beads (FB1-FB4) [Altronics L5250A, Jaycar LF-1250]
1 4MHz crystal (X1)
2 vertical PCB-mount RCA sockets, white (CON1,CON3) [Altronics P0131]
2 vertical PCB-mount RCA sockets, red (CON2,CON4) [Altronics P0132]
1 3-way PCB-mount terminal block, 5.08mm pitch (CON5)
1 2-way vertical polarised header, 2.54mm pitch (CON6) [Altronics P5492, Jaycar HM-3412]
1 2-way polarised header plug (for CON6) [Jaycar HM-3402, Altronics P5472 and P5470A]
1 10-pin PCB-mount IDC vertical box header (CON7) [Altronics P5010, Jaycar PP-1100]
1 2-way SIL pin header (LK3)
1 3-way SIL pin header (LK4)
2 jumper shunts (LK3,LK4)
1 6.35mm chassis-mount single spade connector
4 12mm long M3 tapped nylon spacers
1 M4 x 10mm panhead machine screw
1 M4 hex nut
1 M4 star washer
4 M3 x 6mm panhead machine screws
2 100mm cable ties
1 150mm length of light-duty figure-8 hookup wire
1 50mm length of 0.7mm diameter tinned copper wire
1 PC stake

Semiconductors
4 NE5532AP or LM833P dual op amps (IC1-IC4)
1 PIC16F88-I/P microcontroller programmed with 0111111A.hex (IC5)
1 infrared receiver module (IRD1) [Altronics Z1611A, Jaycar ZD1952]
1 7805CV 5V regulator (REG1)
2 BC327 PNP transistors (Q1,Q3)
2 BC337 NPN transistors (Q2,Q4)
1 3mm blue LED (LED1)
1 3mm orange/amber LED (LED2)
1 3mm yellow LED (LED3)

Interconnecting cables

1 350mm length of 14-way IDC cable
1 250mm length of 10-way IDC cable
2 10-pin IDC line sockets [Altronics P5310]
2 14-pin IDC line sockets [Altronics P5314]

This is a rewarding but complicated project with several options. It is spread over three issues and constructors are strongly urged to read the panel opposite before purchasing components.

Capacitors
2 470µF 16V PC electrolytic
3 100µF 35V PC electrolytic
8 100µF 16V PC electrolytic
8 22µF small non-polarised electrolytic
2 4.7µF small non-polarised electrolytic
11 100nF MKT polyester
4 15nF MKT polyester
1 10nF MKT polyester
4 470pF MKT polyester, MKP polypropylene or NP0 ceramic [eg, element14 1005988]
2 47pF MKT polyester, MKP polypropylene or NP0 ceramic [eg, element14 1519289]
2 22pF ceramic

Resistors (all 0.25W, 1% metal film)

2 1MΩ	6 100kΩ	2 22kΩ	1 18kΩ	2 12kΩ
3 10kΩ	1 2.7kΩ	8 2.2kΩ	4 1.8kΩ	10 1kΩ
2 330Ω	7 100Ω	1 22Ω	3 10Ω	

Input Switching module

1 PCB, code 01111112, 109.5 x94.5mm
3 DPDT 5V relays, PCB-mount (RLY1-RLY3) [Altronics S4147]
3 PCB-mount vertical stacked dual RCA sockets (CON11-CON13) [Altronics P0212]
1 vertical PCB-mount RCA socket, white (CON14) [Altronics P0131]
1 vertical PCB-mount RCA socket, red (CON15) [Altronics P0132]
1 10-pin PCB-mount IDC vertical box header (CON8) [Altronics P5010, Jaycar PP1100]
1 14-pin PCB-mount IDC vertical box header (CON9) [Altronics P5014]
2 ferrite beads [Altronics L5250A, Jaycar LF1250]
4 12mm long M3 tapped nylon spacers
4 M3 x 6mm panhead machine screws

Semiconductors
1 LM393P comparator (IC4)
3 BC327 PNP transistors (Q5-Q7)
1 BC337 NPN transistor (Q8)
3 1N4004 diodes (D1-D3)

Capacitors
2 10µF 16V electrolytic
2 100nF MKT polyester
2 470pF MKT polyester, MKP polypropylene or NP0 ceramic [eg, element14 1005988]

Resistors (all 0.25W, 1% metal film)

3 100kΩ	2 10kΩ	11 2.2kΩ	6 100Ω

Front Panel Pushbutton module

1 PCB, code 01111113, 66 x 24.5m
1 14-pin PCB-mount IDC vertical box header (CON10) [Altronics P5014
3 PCB-mount pushbutton switches with blue LEDs (S1-S3) [Altronics S1173, Jaycar SP0622]
4 6.3mm long M3 tapped nylon spacer
4 M3 x 6mm panhead machine screws

Decisions, decisions, decisions – read this before purchasing parts!

This is a great project – which is why we are running it! However, the author is Australian, and not surprisingly has referenced parts easy to get locally for him. Some of the components must be absolutely the right parts type (eg, the capacitors – see text) and others, especially the hardware (switches, pushbuttons and motorised pot) are board mounted, which means they must not only be functionally correct, but also match the PCB layout dimensions.

For these reasons we recommend purchasers buy some parts/sections (as kits) from Altronics in Australia. Altronics are good value – BUT, you must remember that you have to pay postage, VAT, (possibly) import duty and a delivery import handling charge. Please factor this in before making any purchasing decisions. (At the time of writing £1 = AU$1.95.)

Costs

Our estimates of the additional costs are the following. Roughly £20 for TNT Express delivery to the UK; VAT at 20% is applied to both goods and postage; and duty costs nothing for goods under £135 in value, but 2.5% over £135. Also, many couriers charge a 'handling' fee for dealing with VAT/duty, which is typically £10. These figures are likely accurate, but we cannot guarantee them. More details here: **www.gov.uk/goods-sent-from-abroad/tax-and-duty**

Build options

We are presenting this project in three parts and strongly recommend you read all three articles before ordering components, PCBs or kits. This is important because you have a choice of building a preamplifier with three or six inputs and you need to decide which of three power supply options to use.

Part 1 (this article) covers the heart of the project – the *Preamplifier* board. If you want to keep things simple then buy the whole kit from Altronics (part number K5171). If you are absolutely certain you can source the parts yourself and want to buy the PCB then email us and we will get one in for you – but we will not be keeping it as a standard stock item.

Part 2 will cover the three-input switcher, remote control and power supply sections. Altronics do not sell a kit for the three-input option, but when you order kit K5171 you can also order all the important input/remote control parts and we will sell you the two boards needed (01111112, 01111113).

Part 2 also includes options for power supplies. If you are building this preamplifier into an existing power amplifier then you may not need an additional power supply – check your amp for ±15V rails.

If you are also building a power amplifier then it is worth considering kit K5168 from Altronics, or you can buy its PCB (code 01109111) from the *PE PCB Service* and source your own parts. Assembling this power supply is covered in *Part 2*. It will supply both a power amplifier and pre-amplifier.

On the other hand, if you are building a standalone preamplifier then there are two options. You could build a simple low-power universal regulator, which is also covered in *Part 2*. The parts are easy to source and we can sell you the board (code 18103111). Last, you could use the *4-Output Universal Voltage Regulator* from our May 2016 issue. Again, the parts are easy to source and we can sell you the PCB (code 18105151).

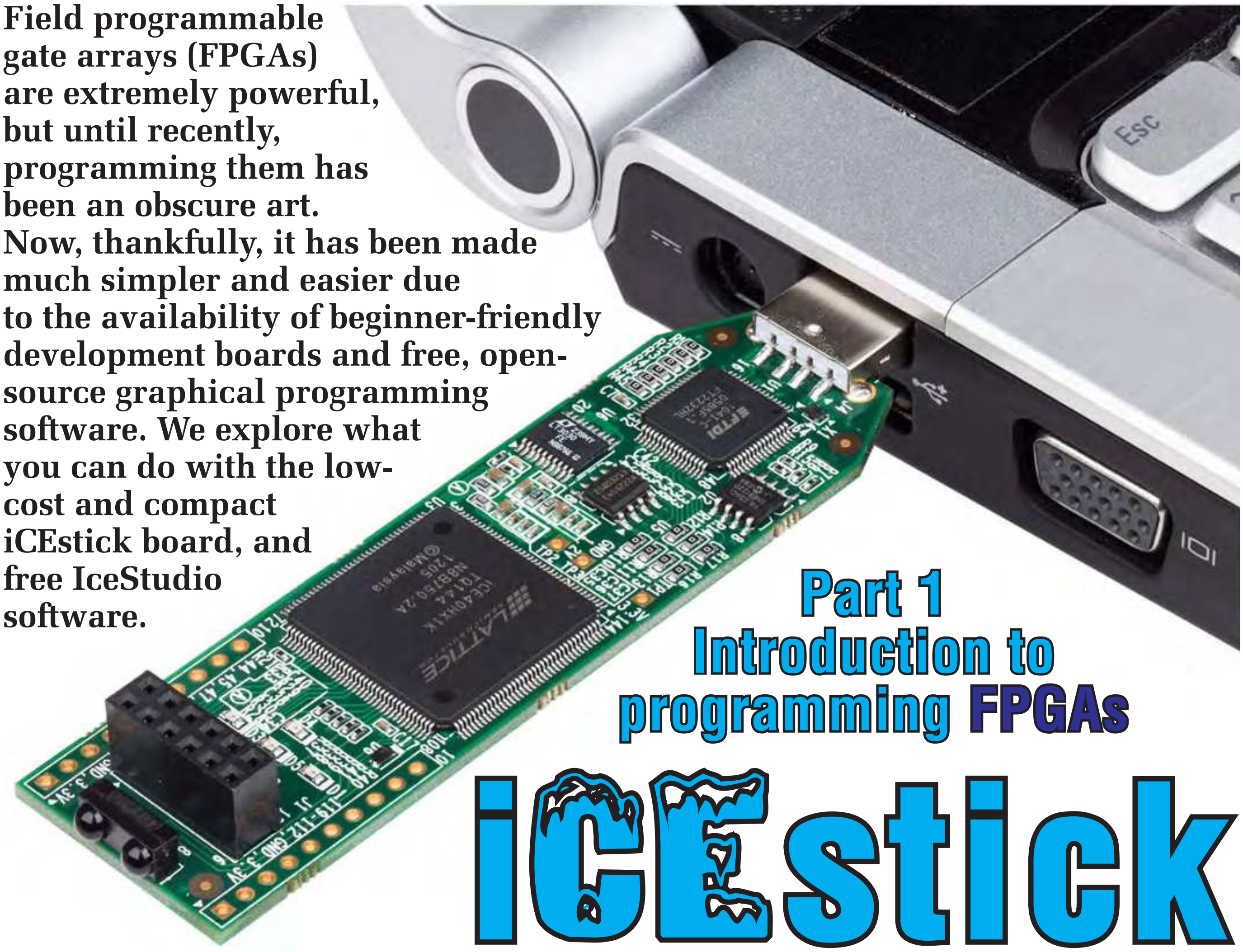
Part 3 covers an expanded input selector system. For many constructors, the three-input option in *Part 2* is plenty, but if you want to have a more flexible system then this option will give you six inputs – all remotely controlled. This is Altronics kit K5172.

PIC programming

Do note that in the Altronics preamplifier kit (K5171) the software programmed into the PIC is correct for the six-input switcher. If you are building the three-input version then you need to reprogram the PIC. Both versions of software will be available from the *PE* website (April 2020 for the three-input version, and June for the six-input version).

We don't normally need to go into such detail over purchasing components, but this project does merit it. Good luck with your builds and enjoy this great preamplifier.

Field programmable gate arrays (FPGAs) are extremely powerful, but until recently, programming them has been an obscure art. Now, thankfully, it has been made much simpler and easier due to the availability of beginner-friendly development boards and free, open-source graphical programming software. We explore what you can do with the low-cost and compact iCEstick board, and free IceStudio software.



Part 1
Introduction to programming FPGAs

iCEstick

For a long time, programming and developing FPGAs has been difficult, especially for the hobbyist who doesn't have access to the often expensive tools that are needed.

On top of this, understanding the language that is used to describe a design can be a challenge, as is getting one's head around the ways FPGAs work differently to microcontrollers.

The iCEstick development board from Lattice Semiconductor (a major FPGA IC manufacturer) is a compact unit which plugs into a USB port.

Thus the board and programming hardware are one and the same, requiring only the extra components for a particular application to be added on.

Even this is not always necessary, as the board sports five LEDs which can be controlled by I/O pins, plus an onboard infrared transceiver.

The code for the iCEstick can be generated using Lattice iCEcube development software, available with a free licence. The Diamond programmer software is then used to program the iCEstick with the resulting file.

We also tried an open-source alternative called IceStudio. It has a graphical interface, allowing logic blocks to be dragged and dropped, then connected by virtual wires to create a representation of the circuit to be synthesised. It is a complete IDE, allowing design, building and uploading to occur.

For users who are comfortable with how logic gates and other basic elements like flip-flops work, this is an ideal way to bridge the gap of understanding between having an idea in one's mind and turning it into a functioning circuit.

IceStudio also allows 'code blocks' containing Verilog code to be created, so those who are familiar with Verilog are not limited by the included graphical symbols.

Verilog is a bit like the C language, as used to program an Arduino, but is designed to produce logic block structures rather than machine code.

by Tim Blythman

What is an FPGA?

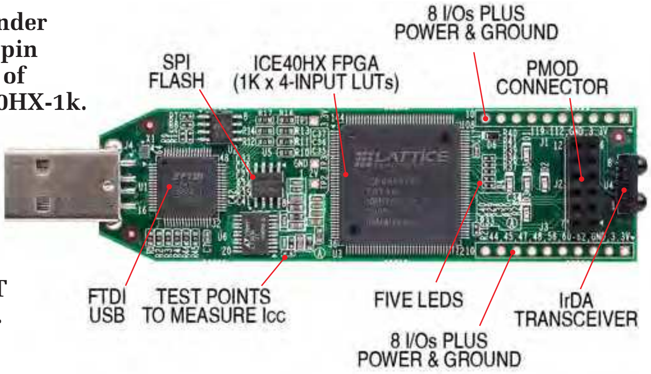
FPGA stands for 'field programmable gate array', and this means that it consists of logic gates, flip-flops and other 'glue' logic which can be (re-)configured to perform different functions.

While this is an over-simplification, you can think of an FPGA as an IC containing thousands of 4000B/74HC/74LS chips connected via crossbars; in effect allowing you to change how the inputs and outputs of those devices are connected, to form virtually any function. And since they are all inside the same chip, very high speeds are possible; up to 500-1000MHz in some parts.

The advantage that this arrangement has over a microcontroller is that everything happens at the same time in an FPGA.

Rather than having to wait for things to process in a sequence, determined by the list of instructions which form the program, everything happens practically instantly in an FPGA.

This makes them ideal for tasks where many different calculations can be made in parallel. While some



The iCEstick, slightly under life size. The huge (144-pin TQFP) IC in the middle of the iCEstick is the iCE40HX-1k. To the right of it are the various I/O headers and five user LEDs. To the left are the flash and EEPROM ICs, an FTDI 2232H dual UART and a 12MHz oscillator.

Table1: iCEstick physical pin to I/O pin mapping

Pin	Function	Pin	Function
21	12MHz Osc.	90	PMOD 9
8	UART TX	91	PMOD 10
9	UART RX	95	LED5 (GREEN)
78	PMOD 1	96	LED4 (RED)
79	PMOD 2	97	LED3 (RED)
80	PMOD 3	98	LED2 (RED)
81	PMOD 4	99	LED1 (RED)
87	PMOD 7	105	IR TX
88	PMOD 8	106	IR RX

microcontroller processors do have multiple cores, allowing several instructions to be executed simultaneously, in an FPGA, practically everything happens simultaneously.

So it's a bit like having a processor with thousands (or even millions) of cores; even though each of those cores may have fairly limited capabilities, the overall result is a much more powerful and capable device.

A good example of a digital task which is quite easy to do with an FPGA but virtually impossible with a regular microcontroller, as demonstrated by the Arduino MKR Vidor 4000, is the generation of an HDMI digital video signal.

The FPGA can produce the HMDI data (which is typically clocked at hundreds of megahertz) far quicker than any microcontroller could possibly manage.

And it can do this while performing whatever other tasks are required simultaneously, without any concerns that the different tasks may interfere with the time-critical video generation process.

Rather than software code (eg, BASIC, C, assembly language), the FPGA configuration is described in a hardware description language (HDL). There are two main HDLs in widespread use: Verilog and VHDL.

We will mostly be dealing with Verilog, which borrows some of its syntax from the C language; but due to the nature of FPGAs, it has some important and significant differences.

The HDL is synthesised into a 'bitstream' (basically, a blob of binary data), which is what is actually loaded into the FPGA chip to configure it. In the case of the iCE40HX-1k FPGA on the iCEstick, this is up to 34kB in size. The bitstream is roughly the equivalent of machine code to a microcontroller or microprocessor.

There is a lot more to this process than this simple description suggests, and much of how FPGAs and FPGA development tools work has been hidden by the manufacturers until the advent of the open source tools we are now using.

ICE40HX chip and iCEstick board capabilities

While touted as having a USB thumb drive form factor, it actually measures 95 × 25mm. But when you consider that a large portion of this board is taken up by the sizeable FPGA chip, its size seems reasonable.

This IC is a Lattice iCE40HX-1k FPGA which comes in a 144-lead TQFP package. While not all the input/output pins are broken out (the chip has 96 I/O pins in total), an ample number are available.

The iCE40HX-1k contains 1280 flip-flops, 1280 lookup tables, 160 programmable logic blocks and 16 RAM blocks. Each RAM block holds four kilobits (512 bytes), for a total of 8 kilobytes.

For comparison, its larger sibling, the iCE40HX-8k, can emulate a 32-bit RISC processor, but this is a bit beyond the iCE40HX-1k's capabilities.

The core of the chip runs at 1.2V, but external I/O on the iCEstick is 3.3V. There are four I/O banks on the iCE40HX-1k which can (in a different implementation) be set to other I/O voltages.

Also on the iCEstick board are several other components for communications and programming. The second-largest IC, nearest the USB plug, provides the USB interface.

This is an FTDI 2232H dual UART with USB 2.0 Hi-Speed. Typically, one of the UARTs is used in SPI mode for programming, and the second UART is available for communication with the bitstream that is 'running' on the FPGA.

The two 8-pin SOIC devices are a Flash IC and an EEPROM IC. The Flash IC is 32Mbit and is used to store the configuration bitstream in a non-volatile fashion. The FPGA is configured using internal RAM, the contents of which is lost on power-down, so it must be loaded from the Flash chip each time power is applied.

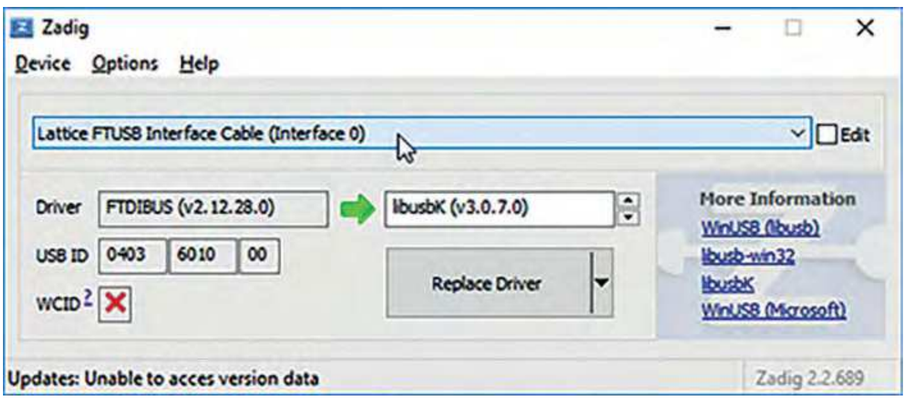
While the FPGA has the facility to load its configuration from its own internal non-volatile configuration memory, this memory can only be programmed once, so a reprogrammable Flash chip is used until a design is finalised.

The EEPROM is simply used to hold the configuration for the FTDI 2232H and the remaining IC is an LT3030 dual low-dropout linear regulator.

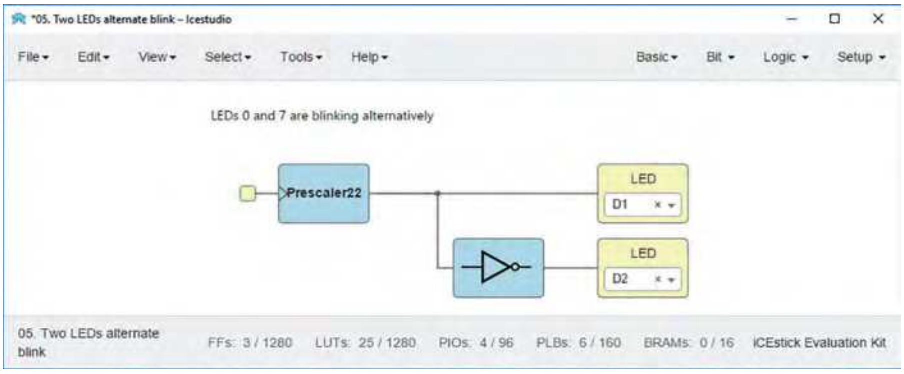
There is also a 12MHz clock source on the iCEstick. This clock source is necessary for all but the most basic logic designs.

The iCE40HX-1k also features a PLL, so designs are not limited to 12MHz, as higher frequencies can be generated by the PLL from the 12MHz source.

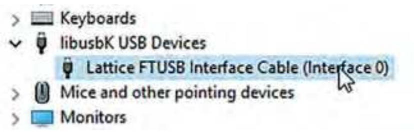
On the far side of the FPGA are the I/O breakout headers. Five LEDs (marked D1-D5) are arranged in a diamond pattern, flanked by two 0.1-inch pitch 10-pin breakouts. Each of these provides eight I/O pins plus ground and 3.3V power.



Screen1: ensure that the correct device is selected in the Zadig application, and that libusbK is selected before clicking 'Replace' and closing the window. If you do change the wrong driver, you can uninstall it via device manager.



Screen2: IceStudio's 'Two LEDs alternate blink' example (which they incorrectly refer to as 'alternative'). The small yellow box at left represents the 12MHz crystal clock on the iCEstick. It is followed by a 22-stage binary divider, effectively dividing the 12MHz clock by a factor of 4,194,304 (ie, by 2^{22}, to around 3Hz). Digital pins D1 and D2 are connected to two LEDs on the iCEstick board, and are driven with square waves derived from the 3Hz clock, one directly, and one via a NOT gate so that it is on while the other is off.



Screen3: if you need to remove IceStudio's drivers to allow the Diamond Programmer to work with the iCEstick then find this entry in Device Manager, right-click it and choose Uninstall Device. Unplug and replug the iCEstick and Windows should reinstall the default drivers.

The 6×2 female header block matches Digilent's PMOD interface, and provides eight more I/Os, plus ground and power.

Finally, at the end of the board opposite the USB connector is an IR transceiver chip, which is connected to another two of the FPGA's I/O pins. This gives 24 unallocated I/O pins available for use, plus at least ten dedicated to I/O functions on the board itself.

Other FPGA boards

After acquiring the iCEstick, we looked around for other FPGA development boards; in particular, those supported by IceStudio. Many of these are open-sourced hardware designs that are being promoted on crowd-funded websites.

In general, we found that most of them were more expensive than the iCEstick. A few were cheaper, but also required a separate programmer. So for this reason, and because the iCEstick is easy to buy, we decided to stick with it.

The fact that two different software packages can be used to program it is also a plus.

Software for the iCEstick

In the following discussion of the software options, we will only give very basic examples. If you want something more involved (and useful), see our *iCEstick VGA Terminal* project, which we will cover next month.

There, we'll delve much deeper into what can be done with the iCEstick and IceStudio.

IceStudio software

The open-source IceStudio software is a free download. We found it straightforward to use, and had a working project uploaded to the board in minutes. There are example projects available which appear quite basic, but they are all great building blocks.

The version we tried was just over 100MB, although you also need to download some other required software packages, such as the 'toolchain'.

Installing IceStudio

IceStudio can be downloaded from its Github page at **https://github.com/FPGAwars/icestudio**

Like many open source tools, it's available for Windows, Linux and macOS. We used the v0.4.0 release. Although this release number indicates it is still in beta, we found the software to be mature and didn't run into many bugs.

Behind the scenes, it uses the open-source IceStorm project to synthesise the bitstream alongside some configuration files, but you don't need to concern yourself with these details while using IceStudio.

In this regard, it is similar to Arduino, which uses the open source gcc compiler and the AVRDUDE programming tool to provide most of its functions, with inbuilt board configuration files. This means the user does not have to worry about the minute details of the specific hardware used.

Installing IceStudio was quite straightforward. About halfway down the Github page (link above), there is an installation guide, with brief, simple instructions for Linux, Windows and macOS, with links to the downloads.

We installed on Windows 10, so some of the steps below may not apply to Linux or macOS; in particular, the driver switching step is probably not needed on these other operating systems.

The installer does not automatically install the required toolchain – you will be prompted to install it when the program first runs. No further input is required apart from confirming that installation should proceed.

IceStudio also includes a bitstream programmer, but this does not work with the default device driver for the iCEstick under Windows. Again, a simple tool allows the appropriate driver to be installed and uninstalled (which is necessary if you wish to also use Lattice's iCEcube software).

The driver switcher uses the Zadig driver utility. IceStudio gives you some prompts which explain how to use Zadig, then opens the program, allows you to make the changes, and then prompts you to unplug and replug the iCEstick.

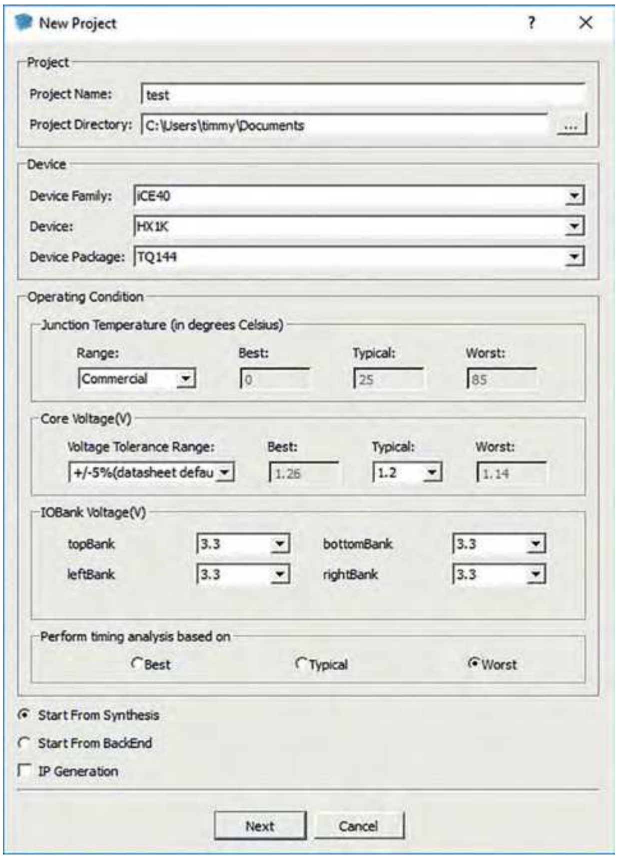
This is all fairly seamless, and it's comforting that the program is upfront about what changes you are making. The Zadig utility also has the option of changing other drivers, so great care should be taken that you don't inadvertently change the wrong driver.

We also noticed that, very occasionally, Windows would reload the old driver (perhaps when the iCEstick was plugged into a different USB port). In that case, it is merely necessary to rerun the driver switcher routine.

Setup

Once the installer has finished, start IceStudio. You will be prompted to install the toolchain, which requires the Python scripting language to be installed, plus a few other packages. If you are not prompted, check the Tools → Toolchain menu, and click Update if you are unsure. We found that this proceeded without any problems, although you need Internet access to download these extra packages.

You will then be prompted to update the drivers. This is only possible if you have an iCEstick connected.



Screen4: this screen grab shows the iCEcube2 new project settings to suit the iCEstick. The project name and location can be set to suit your system, but the device properties are critical for correct operation.

If you don't have an iCEstick, skip this step. Again, there is no harm in checking the drivers if you are not prompted.

Now click Tools → Drivers → Enable. IceStudio will indicate a few steps that will occur. Click OK to proceed. Note the message about using USB 2.0 ports. We ran into problems using the iCEstick on a USB 3.0 port, but were able to use a USB 2.0 hub to 'downgrade' our connection to USB 2.0 and it worked after that.

When the Zadig Driver Utility opens (Windows may ask for permission for the program to make changes), take great care to change the correct drivers. Zadig has facilities for many drivers, but we only want to change those for the iCEstick.

Ensure that 'Lattice FTUSB Interface Cable (Interface 0)' is selected in the dropdown and check that the item to the right of the green arrow is 'libusbK' (in our case, version 3.0.7.0), then click 'replace driver' (see Screen1).

IceStudio will now prompt you to unplug and replug the iCEstick. Do this to ensure that the drivers are loaded correctly.

The final step is to select the development board. This is done from the Select → Board menu; the iCEstick is found under the HX1K subheading. Selecting the correct board means that friendly names are available for the various I/O pins. For example, a pin named 'D1' can be selected, which maps directly to LED1 on the iCEstick.

This completes the setup. There are examples available under the File → Examples menu. Many of these appear to be written for other boards, but are simple enough to adapt for the iCEstick. The only real differences appear to be the I/O pin mappings, which are blanked on conversion.

We also suggest enabling the FPGA resources view, by clicking View → FPGA resources, and ensuring this item is ticked. The bottom bar of the window will now show the resource usage, which is empty at this stage. This will let you keep track of how 'full' your FPGA is.

Using IceStudio

A good place to start is the example available under the following menu: File → Examples → Basic → Two LEDs alternate blink.

Upon opening this, you will be prompted that it is designed for a different board; simply click 'convert'. As mentioned above, conversion involves removing any I/O pins associated with the old board. To complete the conversion, click on the LED dropdown boxes, and select D1 and D2 (see Screen2).

The next step is to compile the project into a bitstream. Click Tools → Build or press Ctrl-B. After a few seconds, a message will pop up which should say: 'Build done'. Finally, click Tools → Upload to send it to the iCEstick. The LEDs will all light up dimly during the upload stage, and if the upload is successful, two of the LEDs should be alternately flashing.

If you have trouble with the upload, check the drivers using the Enable Driver option or try a different USB port.

We recommend looking at the examples to see what can be done with IceStudio. The four menu items at top right are various items that can be dropped into the editor to create your project.

Included in these (under Basic) is a 'Code' option. This allows blocks containing Verilog code to be included. For those familiar with Verilog, the blocks are effectively the same as Verilog modules. Such a code block can even be exported and used in another project.

You can build just about any set of logic using Verilog, including adders, accumulators, multipliers, dividers, multiplexers, memories, register files and so on.

The various gates and other blocks can be joined by wires. To create a wire, move the mouse to an output pin of a block until the pointer becomes a black cross. Click, and drag the wire to the input of another block and release.

We found the wires to be one of the fiddliest parts of IceStudio. They can only be dragged from output to input, and often end up in awkward places. They can be dragged to neaten the layout or removed by hovering over the wire, and then finding the small red 'x' and clicking on it.

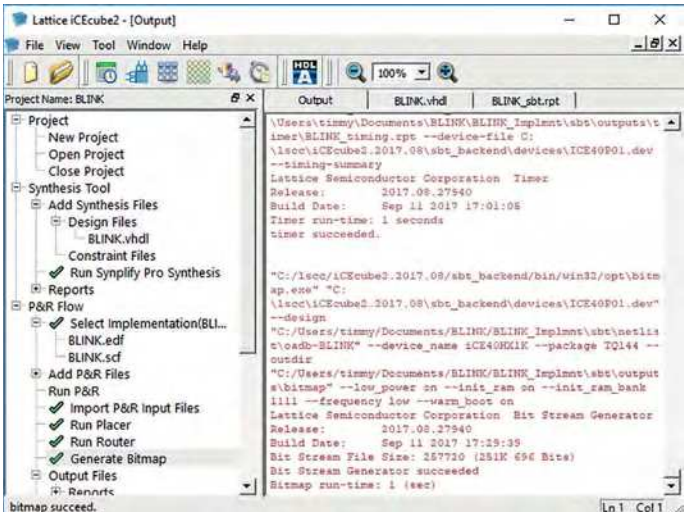
The software has all the usual editing facilities such as copy, paste and undo, and they all work rather well once you get used to it. You can press and hold the right mouse button to pan around the window, and the scroll wheel on the mouse allows zooming in and out.

A full user guide is available at: **https://icestudio.readthedocs.io/en/latest/**

Note that if you have used IceStudio to enable its driver, you will need to disable it to allow the Diamond Programmer to use its driver. The Tools → Drivers → Disable menu is a bit cryptic about this.

What you need to do is open Device Manager, find the libusbK driver entry, right click on it and uninstall it (Screen3). Then unplug and replug the iCEstick, and Windows will reinstall the default drivers.

This isn't necessary on macOS or



Screen5: the iCEcube window after our project has been converted into a bitstream.

Despite all the red text, everything completed without errors.

The Pin Constraints Editor is the icon below the left of the Window menu.

Linux, as the same drivers are used for both software packages.

iCEcube2

iCEcube2 is proprietary software, and while you can freely download it and install it, a license key is needed to run it. This is all available at no cost, but you will need to create an account on the Lattice website to receive a license key.

We found the process of setting up an account and requesting a key a bit slow, but it worked, and we got our key in the end.

The key is tied to a specific Ethernet MAC address, meaning you will need multiple licenses if you want to run the software on multiple computers. The iCEcube2 version we downloaded was around 750MB, and a separate download of the 'Diamond' programmer application is needed too.

There are versions of iCEcube2 available for Windows and Linux; download from: **http://bit.ly/pe-apr20-ice1**

The separate programmer software is found at: **http://bit.ly/pe-apr20-ice2**

Ensure that you have the license file for iCEcube2. There is a link on the information page for iCEcube2 detailing how to receive the license file via email. Although the email notes that the license file should be placed in the \license directory of our install, there did not appear to be such a directory.

Our install of iCEcube2 has the path C:\lscc\iCEcube2.2017.08, so we placed a copy of the license in both the lscc and iCEcube2.2017.08 directories, and everything seemed to work, although it did sometimes complain that the license file was missing.

We struggled to find simple examples that would work for the iCEstick under iCEcube2, and certainly didn't find any on Lattice's website. In the end, we found a basic 'blink' example at: **http://bit.ly/pe-apr20-ice3** but even this missed one or two steps, so we had to modify it.

ICEcube2 uses VHDL, so if you prefer VHDL over Verilog, this may be an option, although VHDL is generally stricter and more verbose than Verilog (Editor's note: in my opinion, Verilog is superior, although they both have roughly the same capabilities).

To use iCEcube2, first create a new project, and fill in the details as shown in the screen grab (Screen4), to match the hardware of the iCEstick. Click OK, and the 'add files' dialog box opens; click 'Finish', as files can be added from within the project.

You can download the **BLINK.vhdl** file from the April 2020 page of the *PE* website. Copy this file to within the project folder, then add it to the project by right-clicking on 'Add Synthesis Files'. Select the file and then press the '>>' button to add it to the project.

The 'Run Synplify Pro Synthesis' button is the first step in turning the project into a bitstream. Double-click this, and check that there are no errors. We got an error message about the license file, but it worked anyway.

Next click 'Import P&R Input Files'. You should see a pattern of working through several steps along the left-hand side of the project window, with the green triangles turning into ticks as the steps are completed by double-clicking on them (see Screen5).

After the P&R (place and route) files have been imported, the pins need to be assigned. This is done with the Pin Constraints Editor, selected from the row of icons below the menus. In our version, it is the fourth icon, which looks like a blue square with pins coming out of it.

The physical pin to I/O pin mapping is shown in Table 1. LED1 and LED2 should be set to pins 95 and 96 (or any of the other LED pins from the table), and 'clk' (the clock signal input) should be set to pin 21. Save the project to register the new pin assignments.

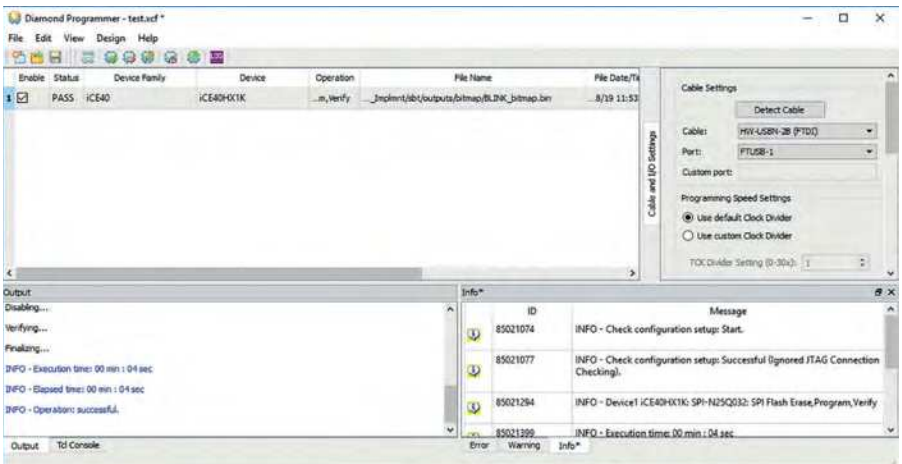
Finally, in turn, double click 'Run Placer', 'Run Router' and 'Generate Bitmap'. The generated bitmap is the file that will be loaded onto the iCEstick. It can be found buried within the project folder; eg, \BLINK_Implmnt\sbt\outputs\bitmap\BLINK_bitmap.bin

Diamond Programmer

Now we use the Diamond Programmer application to load the bitstream onto the iCEstick. Open Diamond Programmer, select 'Create a new blank project' and click OK.

Under 'Cable Settings' to the right, click 'Detect Cable'; the selected cable should include FTDI in its name. We found we had to set the port to FTUSB-1 (see Screen6).

In the main window, set the device family to iCE40 and the device to iCE-



Screen6: the Diamond Programmer window. Check the Device, Device family and Cable settings to ensure they are correct. The Device Properties icon is immediately below the Help menu item, while the Program button is the one with the large green arrow.

40HX1K. Under File Name, browse to the bitmap file created by iCEcube2 and select it.

Click the 'Device Properties' icon (a chip with a small yellow pencil) and set that as shown in the screen grab (Screen7). Finally, you click the 'Program' button to transfer the bitstream to the iCEstick.

If all is well, your Output Window at bottom left should look like our screenshot, and you should see two LEDs flashing alternately on the iCEstick.

Conclusion

We devoted more space to describing the iCEcube2 and Diamond Programmer software than IceStudio because it requires more work to achieve the same result.

We found that IceStudio was a real pleasure to use and would highly recommend it to anyone who has not worked with FPGAs before. We found a couple of small glitches, including occasional crashes even on quite small projects. So save your work often.

We also found IceStudio became quite sluggish on larger projects, taking some time to zoom and move around. We imagine that as we become more proficient with Verilog, that our IceStudio projects will consist of nothing more than a single large code block, which should not present the same performance issues as lots of smaller blocks.

IceStudio also appears to have the benefit of being written specifically for development boards such as iCEstick.

If you are a professional developer, especially someone looking to build an FPGA into an end product, the flexibility and complexity of iCEcube2 will be warranted. Just choosing a different Flash IC to that fitted or another small hardware change different to the iCEstick would probably be enough to hamper IceStudio in these cases.

With iCEcube2, once we had our project set up, everything worked quite well; similarly, Diamond Programmer worked quite well, although the time spent pulling hair and debugging cryptic error messages was quite a bit more than we had hoped.

But for someone who has not worked with FPGAs before, IceStudio will give a smooth, easy way for you to become accustomed to what is possible.

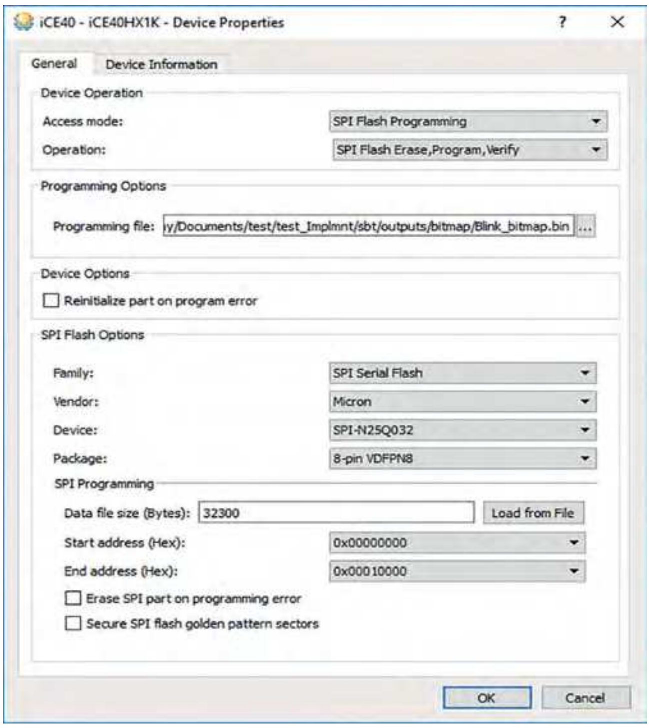
Hardware

In terms of hardware, there are a few development boards around which feature more powerful FPGAs than the iCEstick discussed here. However, for now, the iCEstick suits our purposes, and we think it will be a great starting point for those wishing to try out FPGAs for the first time. It is available from Mouser and Digikey for around £20.

Next month

Finally, just a reminder that next month in *Part 2* we'll put FPGA theory into practice by building an *iCEstick VGA Terminal*.

Reproduced by arrangement with
SILICON CHIP magazine 2020.
www.siliconchip.com.au



Screen7: we'd never have guessed these properties, so we're glad we found a guide to help us out. Make sure you don't select NVCM programming. That is the non-erasable (write-once) memory built into the iCE40HX-1k IC. We use the Flash memory instead, to allow repeated write/erase cycles.

Practical Electronics BACK ISSUES EVERYDAY PRACTICAL ELECTRONICS

APRIL 2019

PROJECTS • Internet-based 'GPS' Time Source • Infrared-sensing Heater Controller • Using Cheap Asian Electronic Modules – Part 15 • MIDI Ultimate Synthesiser – Part 3

FEATURES • Techno Talk • Teach-In 2019 – Powering Electronics – Part 5 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

MAY 2019

PROJECTS • 2× 12V Battery Balancer • Deluxe Frequency Switch • Using Cheap Asian Electronic Modules – Part 16 • MIDI Ultimate Synthesiser – Part 4 • USB Port Protector

FEATURES • The Fox Report • Techno Talk • Teach-In 2019 – Powering Electronics – Part 6 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

JUNE 2019

PROJECTS • Wide-range Arduino-based LC Meter • USB Flexitimer • Using Cheap Asian Electronic Modules – Part 17 • MIDI Ultimate Synthesiser – Part 5

FEATURES • Techno Talk • Teach-In 2019 – Powering Electronics – Part 7 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

JULY 2019

PROJECTS • Full-wave 10A Universal Motor Speed Controller • Recurring Event Reminder • Temperature Switch Mk2 • Using Cheap Asian Electronic Modules – Part 18 • MIDI Ultimate Synthesiser – Part 6

FEATURES • The Fox Report • Techno Talk • Teach-In 2019 – Powering Electronics – Part 8 • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

AUGUST 2019

PROJECTS • Brainwave Monitor • Super Digital Sound Effects Module – Part 1 • Control your PC with an infrared remote • Watchdog Alarm

FEATURES • The Fox Report • Techno Talk • Net Work • Electronics for a car dynamometer • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

SEPTEMBER 2019

PROJECTS • Intelligent Touchscreen Lathe Speed Controller • Twin Dipole Guitar Speaker • Cheap Asian Electronic Modules – Part 19 • Super Digital Sound Effects Module – Part 2 • White Noise Generator

FEATURES • Techno Talk • Net Work • Circuit Surgery • Audio Out • Practically Speaking • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

OCTOBER 2019

PROJECTS • Programmable GPS-synced Frequency Reference – Part 1 • Digital Command Control Programmer for Decoders • Opto-isolated Mains Relay

FEATURES • The Fox Report • Techno Talk • Net Work • Using Stepper Motors • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

NOVEMBER 2019

PROJECTS • Programmable GPS-synced Frequency Reference – Part 2 • Cheap Asian Electronic Modules – Part 20 • Tinnitus & Insomnia Killer • Colour Maximite Computer – Part 1

FEATURES • Techno Talk • Net Work • Using Stepper Motors • Circuit Surgery • PIC n' Mix • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

DECEMBER 2019

PROJECTS • Extremely Sensitive Magnetometer • Useless Box! • Four-channel High-current DC Fan and Pump Controller • Colour Maximite Computer – Part 2

FEATURES • The Fox Report • Techno Talk • Net Work • Circuit Surgery • Using Stepper Motors • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

JANUARY 2020

PROJECTS • Audio DSP – Part 1 • Isolated Serial Link • Four-channel High-current DC Fan and Pump Controller – Part 2 • Colour Maximite Computer – Part 3

FEATURES • The Fox Report • Techno Talk • Net Work • PIC n' Mix • Using Stepper Motors • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

FEBRUARY 2020

PROJECTS • Audio DSP – Part 2 • Motion-Triggered 12V Switch • USB Keyboard and Mouse Adaptor for Micros • Using Cheap Asian Electronic Modules – Part 21 • Colour Maximite Computer – Part 4

FEATURES • The Fox Report • Techno Talk • Net Work • Practically Speaking • Using Stepper Motors • Circuit Surgery • Audio Out • Make it with Micromite • Max's Cool Beans • Electronic Building Blocks

MARCH 2020

PROJECTS • Diode Curve Plotter • Audio DSP – Part 3 • Steam Train Whistle / Diesel Horn Sound Generator • Using Cheap Asian Electronic Modules – Part 22

FEATURES • The Fox Report • Techno Talk • Net Work • PIC n' Mix • Circuit Surgery • Audio Out • Make it with Micromite • Visual programming with XOD • Max's Cool Beans • Electronic Building Blocks

BACK ISSUES – ONLY £5.80

£5.80 per issue for UK incl p&p ■ £6.75 Europe Air Mail ■ £7.50 ROW Air Mail

We can supply back issues of PE/EPE by post.

We stock magazines back to 2006, except for the following:

2006 Jan, Feb, Mar, Apr, May, Jul
2007 Jun, Jul, Aug
2008 Aug, Nov, Dec
2009 Jan, Mar, Apr
2010 May, Jun, Jul, Aug, Oct, Nov
2011 Jan
2014 Jan
2018 Jan, Nov, Dec
2019 Jan, Feb, May, June

Issues from Jan 1999 are available on CD-ROM / DVD-ROM

If we do not have a a paper version of a particular issue, then a PDF can be supplied for the same price – your email address must be included on your order.

Please make sure all components are still available before commencing any project from a back-dated issue.

ORDER FORM – BACK ISSUES

☐ Back issues required (month / year) . . . .

Name . . . .

Address. . . . .

. . . . Tel: . . . .

Email . . . .

☐ I enclose cheque/PO to the value of £ . . . .

☐ Please charge my Visa/Mastercard £ . . . .

Card No . . . .

Valid From . . . . Card Expiry Date . . . .

Card Security Code . . . . (Last three digits on or under the signature strip)

SEND TO: **Practical Electronics, Wimborne Publishing Ltd**
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU.
Tel: 01202 880299 **Fax:** 01202 843233.
Email: stewart.kearn@wimborne.co.uk **On-line Shop**: www.epemag.com

Payments must be in £ sterling – cheque drawn on a UK bank. All items are normally posted within seven days of receipt of order. Send a copy of this form, or order by letter / online if you do not wish to cut your issue.

Win a Microchip Curiosity PIC32MZ EF Dev Board 2.0

Practical Electronics is offering its readers the chance to win a Microchip Technology Curiosity PIC32MZ EF 2.0 Development Board (DM320209).

Worth $47.99 (approx £37) each

The Microchip Technology Curiosity PIC32MZ EF 2.0 Development Board (DM320209) includes an integrated programmer and debugger, which requires no additional hardware to get started.

Many embedded applications are adding better graphics displays, and the PIC32MZ EF, in the LCCG configuration, can support up to a WQVGA display without the added cost of external graphics controllers. An optional, fully featured hardware crypto engine is also available with a random-number generator for high-throughput data encryption/decryption and authentication using AES, 3DES, SHA, MD5 and HMAC.

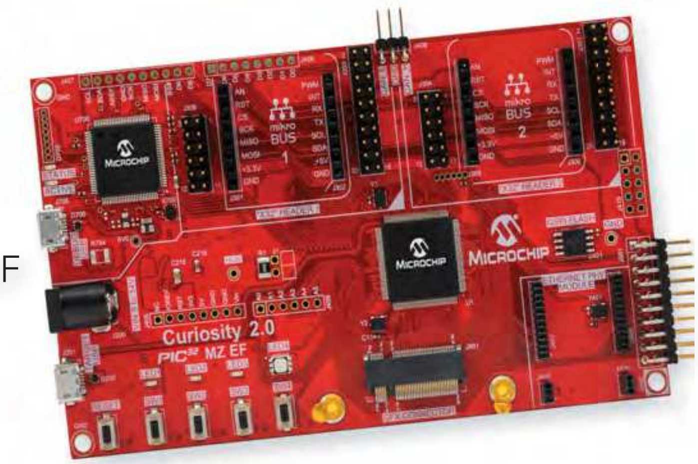
Users can expand functionality through MikroElectronika mikroBUS Click adapter boards, add Ethernet connectivity with the Microchip PHY Daughter Board, add Wi-Fi connectivity capability using the Microchip expansion boards, and add audio input and output capability with Microchip audio daughter boards.

With or without expansion boards, the Curiosity PIC32MZ EF 2.0 Development Board provides the freedom to develop a variety of applications, including Bluetooth audio, CAN, graphics/UI, Internet of Things (IoT), robotics development, and proof-of-concept designs.

The Curiosity PIC32MZ EF 2.0 Development Board includes the following features and more:

- PIC32MZ2048EFM144, 200MHz, 2MB Flash, 512Kb SRAM
- On-board debugger (PKoB4) with:
  - o Real-time programming and debugging
  - o Virtual COM port (VCOM)
  - o Data Gateway Interface (DGI)
- Two X32 audio interfaces supporting Bluetooth and audio
- Xplained Pro extension-compatible interface.

Free-to-enter competition



Microchip
Curiosity PIC32MZ EF Dev Board 2.0

How to enter

For your chance to win a Microchip Curiosity PIC32MZEF Dev Board 2.0, enter your details in the online entry form at:

https://page.microchip.com/PE-PIC32-Board.html

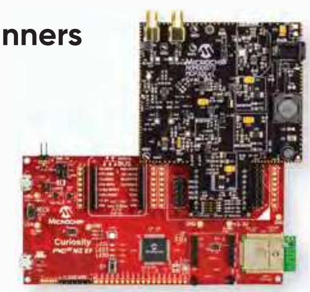
Closing date

The closing date for this offer is 31 March 2020.

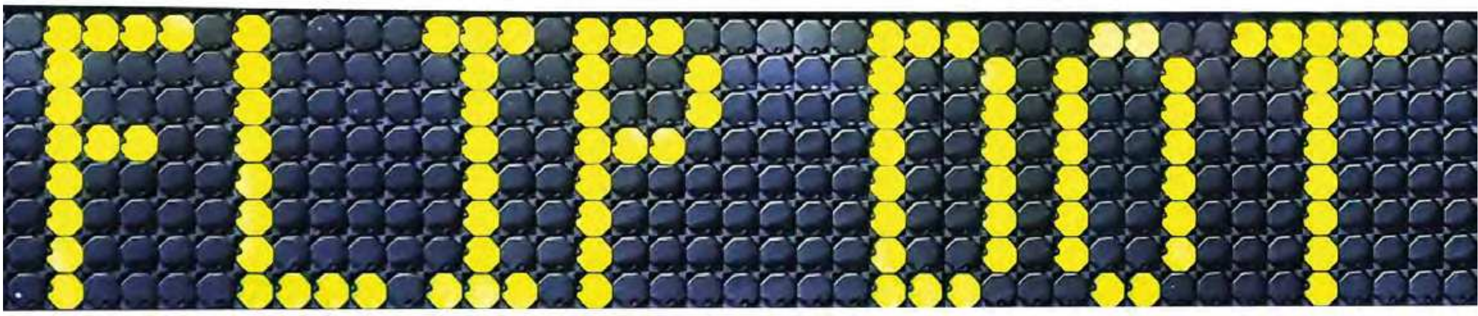
December 2019 winners

Mark Grant

He won a Microchip 1 Msps SAR ADC Evaluation Kit worth £180.00



BUILD YOUR OWN E-X-P-A-N-D-A-B-L-E



If you want a dot matrix display which has digits/letters over 90mm high, is visible under a wide range of lighting conditions and uses no power except when the display is changing, then our new and very cool *Flip-dot Display* is for you. Seeing (and hearing) a *Flip-dot Display* is quite something, so it makes a great conversation starter too!

You've probably seen the large yellow dot displays on the front of many buses and trains, or perhaps in airports. They're highly visible in bright sunlight or under cloudy skies, and they're usually illuminated at night too.

Contrary to what you might believe, they're generally not electronic signs as such: they're actually electromechanical flip-dot displays. They're made from panels that are yellow on one side and black on the other. They rotate to change state, accompanied by a pleasing 'clack-clack-clack' sound.

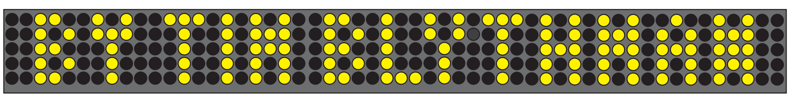
Well, now you can build your very own home *Flip-dot Display!* It's easy to build, uses just a handful of readily available parts and is controlled by an Arduino or MicroMite microcontroller.

You can make it read just about anything you want. If you use a micro with a Wi-FI adaptor, you can even get it to download and display data from the Internet, such as the temperature forecast or sports scores.

So-called 'flip-dot' or 'flip-disc' displays have been around for over 50 years and are still commonly used in countless applications.

Their simplicity and reliability have stood the test of time, and now you can build your own.

For those not familiar with this type of display, each disc or flap which forms a pixel in the dot-matrix display also contains a small permanent magnet. An electromagnet can flip this magnet, and thus the disc, to control which colour is visible from the outside. The polarity of the coil drive current determines which side of the disc appears. When power is removed, the display remains in its last state.



One complete unit – here displaying the letter 'S' – sits upright of its own accord. We have fitted a small length of female header strip to CON1 and CON3 to allow connections to be made with jumper wires. See video: siliconchip.com.au/Videos/Flip-dot

These displays are designed for the discs to remain stationary until commanded to move. Our version has been simplified to make it as easy as possible to build, but it will still make a practical stationary display, and one which can be seen quite well in various lighting conditions and across a large room.

Many commercial flip-dot displays use numerous small coils wound onto tiny armatures – see the photo of one on the next spread.

How our Flip-dot Display works

To simplify our display and make it much cheaper and easier to build, we have formed coils using PCB tracks instead. One PCB contains fifteen such coils on both layers – enough to produce a single character display by itself.

Each board consists of a matrix of fifteen pixels, arranged three wide by five high. This is just enough to display a capital letter, number or symbol. Each pixel consists of a piece of fibreglass that's black on one side and white on the other, with an embedded rare-earth magnet.

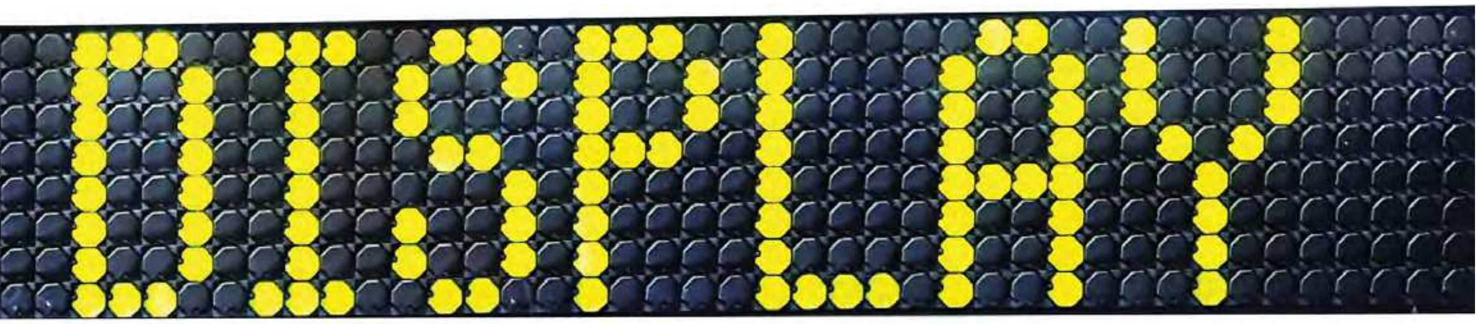
These sit over the PCB-track coils and are attached to that board in such a way that they can rotate through 180° on a pair of simple hinges, allowing either side of the black/white panel to be made visible.

The PCB underneath is also white on one side and black on the other, so that when the panel with the magnet flips, the whole area changes from black to white, or vice versa.

All that the driver board needs to do to cause it to flip is to energise the coil underneath with the correct polarity.

This will repel the magnet initially, causing the panel to swing through 90° until it is at right angles to the panel below. The magnet will then be attracted to the coil and continue moving due to inertia, until it is laying flat on the panel below but with the opposite orientation.

The pixel size (19mm wide and 17mm tall) is a compromise between the magnetic strength of the coil and the weight of the moving elements. Each coil has around 60 turns and measures just over 1.5m in track length, but is packed into an area less than four square centimetres. This



Features

- 15-pixel display per board (three pixels wide, five pixels high)
- Each board can display a single letter, number or symbol
- Display boards can be daisy-chained for multi-character displays
- Customisable colours (BYO paint!)
- 5V/3.3V 4-wire serial interface
- 12V power supply required – 1.5A or higher (see text)
- Each pixel controlled individually
- Stackable for multi-row displays

is about the limit of what is possible with a two-layer board.

We're using 3mm × 1.5mm rare-earth magnets glued into a hole on the flap PCB. It is important that the magnets all face the same way relative to the colours. This ensures that the flaps are interchangeable and consistently display the same colour.

The pixel flaps and the brackets holding the flaps to the panel are small PCBs too. A completed unit including the driver PCB will consist of 23 separate PCB pieces. The bracket PCBs are soldered to the main coil PCB, and the flaps are slotted in place, pivoting around their end tabs.

PCBs are a cheap, convenient way to achieve the correct mechanical dimensions required of multiple identical parts. By using PCBs with a black solder mask and white silkscreen printing, we can use the silkscreen layer to create pixels with very high contrast between the 'on' and 'off' states.

Due to the limited strength of the electromagnets, the display will only work reliably when standing upright, which it will comfortably do without any extra parts.

Driving the display

The display driver circuit is shown in Fig.1. It is designed to be controlled by a microcontroller using a simple serial bus, and is powered from a 12V DC supply. It connects to the coil circuit, shown in Fig.2, via headers CON5-CON8. This circuit represents one set of 3 × 5 pixels that can display a single character; characters can be daisy chained to form larger displays. (We'll explain how that works shortly.)

The driving signals from the microcontroller are fed in via six-pin header CON1. They pass to IC1 and IC2, two 74HC595 shift registers, which decode the serial data stream and use it to control the state of sixteen separate digital outputs (QA-QH on each IC). These control signals will normally be either 0V (low) or 3.3-5V (high).

These digital outputs connect to the control inputs of IC3-IC6, four L293D dual H-bridge motor drivers, which provide the current required to drive the fifteen coils, as well as converting the 0-3.3/5V control-signal voltage swing into a higher 0-12V swing to drive the coils.

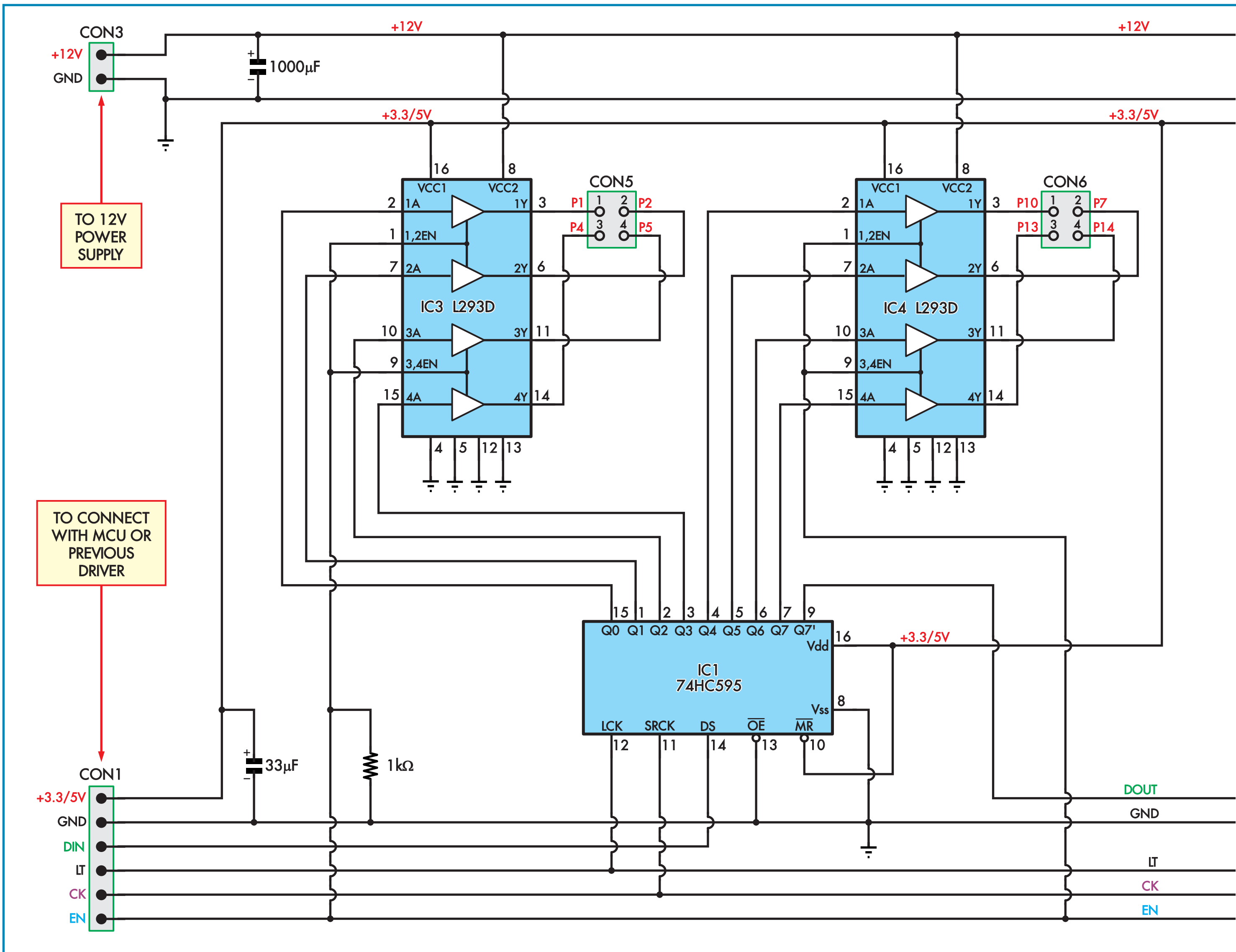
Fifteen of the motor driver outputs connect to one end of each coil, with the sixteenth output driving the other end of all the coils, which are joined together (common or COM).

So to flip a single pixel, the common (COM) output goes either low or high, and one of the other fifteen outputs (P1-P15) is driven with the opposite polarity. This causes current to flow through that one coil in a direction determined by the output polarities.

The direction of current flow determines whether the coil produces a north or south magnetic pole in proximity to the permanent magnet.

The software needs to ensure that only one coil is driven at a time, because all the coil currents return to the same common driver pin. While this pin may be capable of sourcing/sinking enough current to flip more than one pixel at a time, we've found it to be a bit marginal, and it results in IC6 (which drives the COM pin) getting rather hot. So our software flips one pixel at a time.

To achieve this, all outputs are set high or low, except for one, which is set to the opposite polarity. Any output that is set the same polarity as the COM pin will cause no current to flow through the connected coil. Only



Flip-dot Display driver

the single coil that is driven with a different polarity will receive current.

The instantaneous current requirement of the coils is around 1A with a 12V supply, which is above the continuous rating of the L293D. But the coils only need to be pulsed briefly, so the average current is much less than the peak current. The microcontroller pauses briefly between updating each pixel, to keep the average current under the thermal limit and to allow the pixel time to finish its flip manoeuver.

Since the display holds its state with no power applied, the circuit's average operating current is usually not terribly high. Note that no more than two of the four drivers on any IC should be active at a time.

The enable pins of the four L293Ds (pin 1 of IC3-IC6) are joined together and held low by a 1kΩ pull-down resistor, so that the default state of all the outputs is off (high-impedance). It isn't until the microcontroller pulls the enable lines high, via pin 6 of CON1, that IC3-IC6 are activated, and that is only done once the control data has been shifted through IC1-IC2 and latched at their outputs.

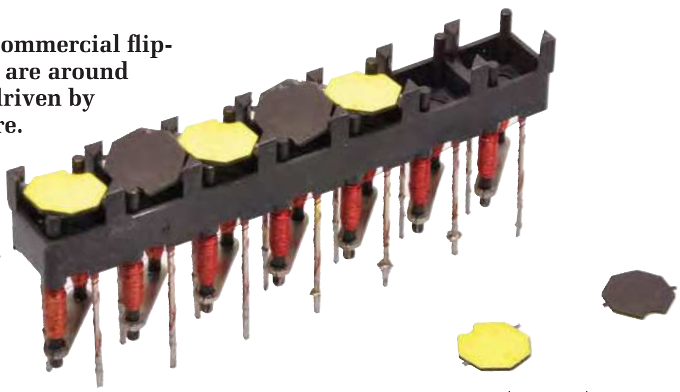
The enable pins are only pulled high for 100ms at a time, to limit the current pulse duration, as explained above. Due to this relatively long drive time, the extra time taken to shift control data from the micro through IC1-IC2 is negligible.

As required by the L293D, the logic ground and power ground are common. Separate connections for 12V power and 3.3V/5V logic supply are available, via CON3 and CON1 respectively.

Construction

This is a mechanical design with moving parts, so a fair degree of precision in the construction is required to ensure proper operation. The primary requirement is that all the parts are put together squarely and lined up correctly before fixing them in place.

The first step is to glue the magnets in the pixel flaps. We highly recommend that the flaps be left in the PCB



The mechanism of a commercial flip-dot display. The discs are around 9mm across and are driven by coils of enamelled wire. The magnetism remaining after the current has ceased is enough to hold the discs in their last position, or even snap them back if they are moved.

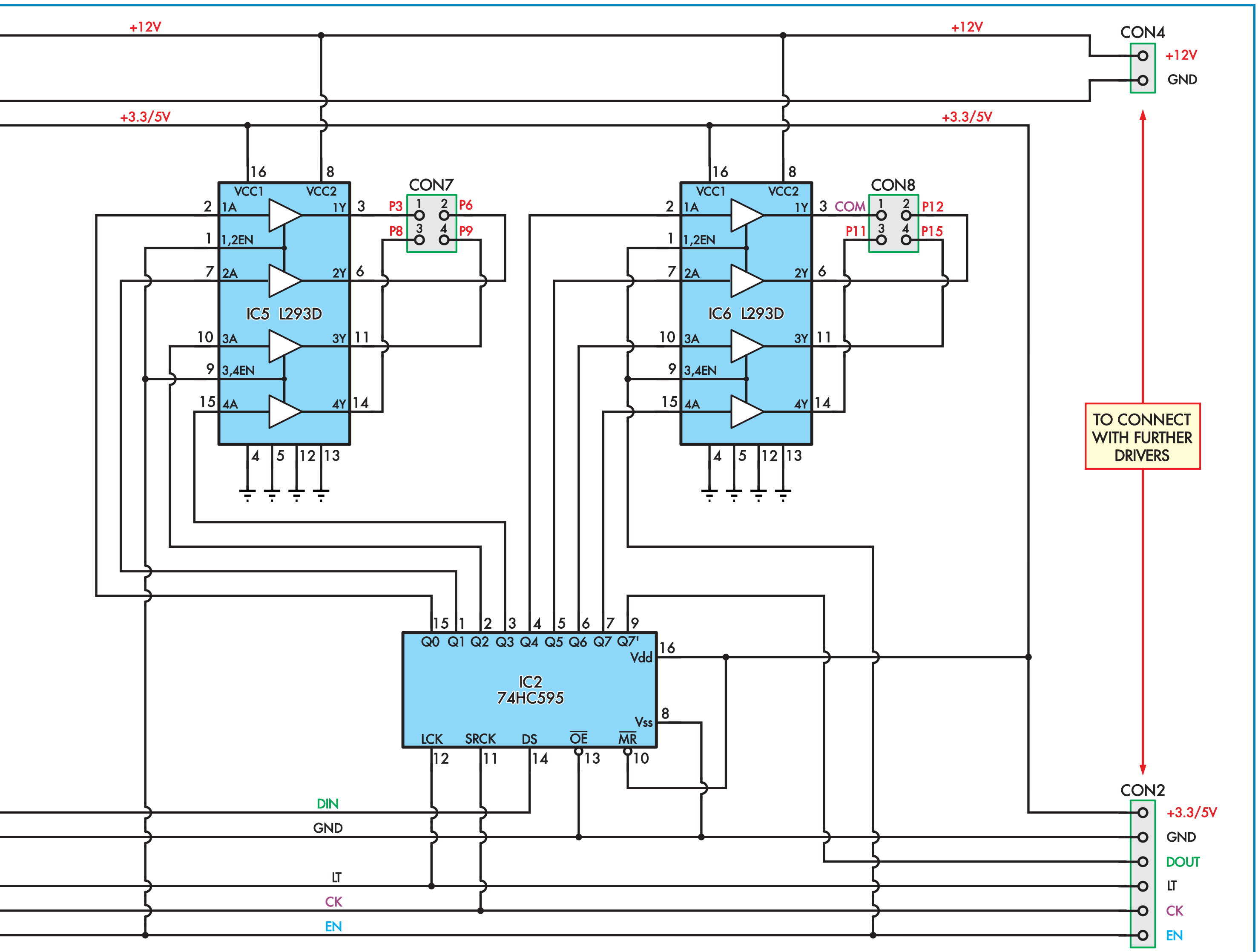


Fig.1: the circuit of the driver for one 3 × 5 pixel *Flip-dot Display*. The control signals and logic supply from CON1 are fed to IC1 and IC2, two 8-bit serial-to-parallel latch ICs. These drive the 16 control inputs of L293D dual H-bridge motor drivers IC3-IC6. Here, they are driving 15 coils etched in a separate PCB, shown in Fig.2.

frame during this step, to avoid pieces getting lost. The flaps are spread out enough that interaction between the magnets is minimal.

We do this step first to allow time for the glue to cure. We used epoxy resin as it has a bit of resilience and is quite strong; cyanoacrylate-type glue (superglue) is probably too brittle and might causing the magnets to come loose after some use.

To make this process easier, you need a disposable, flat plastic surface. The lid from an ice-cream tub or takeaway container is ideal, as epoxy will not stick to this. Another helpful item is a flat sheet of ferrous material (something that a magnet would stick to, such as plain steel). This can be used to help hold the magnets in place. We used a steel case, but you could also use the lid of a 'tin'.

Place the ice-cream tub over the ferrous material, then sit the PCB frame on this. Once you insert the magnets in their holes, they should be held in place by their attraction to the steel, but the ice cream lid will allow them to be removed without too much force. The most critical point of this step is that all the magnets' poles line up.

To achieve this, take the stack of magnets (they'll form into a stack of their own accord), and push the magnet at the end of the stack into one of the holes in the pixels. Then detach it from the stack by sliding the stack to the side, leaving a single magnet sitting in the hole. The PCBs are 1.6mm thick, so the magnets should sit just below the surface of the PCB.

You will see that there are 16 pixel flaps in the frame, but we only need 15, so there is a spare if needed.

Then repeat for the other 14 or 15 pixels, without changing the stack's orientation. When finished, check the magnetic polarity by moving another magnet nearby (not so close that it pulls them out). You should feel all the magnets are attracted to the magnet in your hand without changing its orientation.

Mix up a small amount of epoxy resin, and apply a film to the top of each magnet in its hole. Try to work it down the sides if possible. The rough edges of the PCB will provide good purchase on the glue. Finally, wipe down any excess. This is important – any extra glue may foul and unbalance the mechanism.

You should also ensure that the PCB panel is still flush with the plastic below, as if it is sitting up, the magnets may end up protruding slightly.

Allow the resin to harden. We recommend that you leave it longer than suggested by the manufacturer to ensure it is fully cured. If it is still sticky, it may gum up the mechanism and make handling difficult.

If you wish to change the colour of the flaps, after the resin has cured is an ideal time. A thin coat of paint should be used to ensure that the flaps do not become too heavy. You could use spray paint, one colour on one side, and a second colour on the other side.

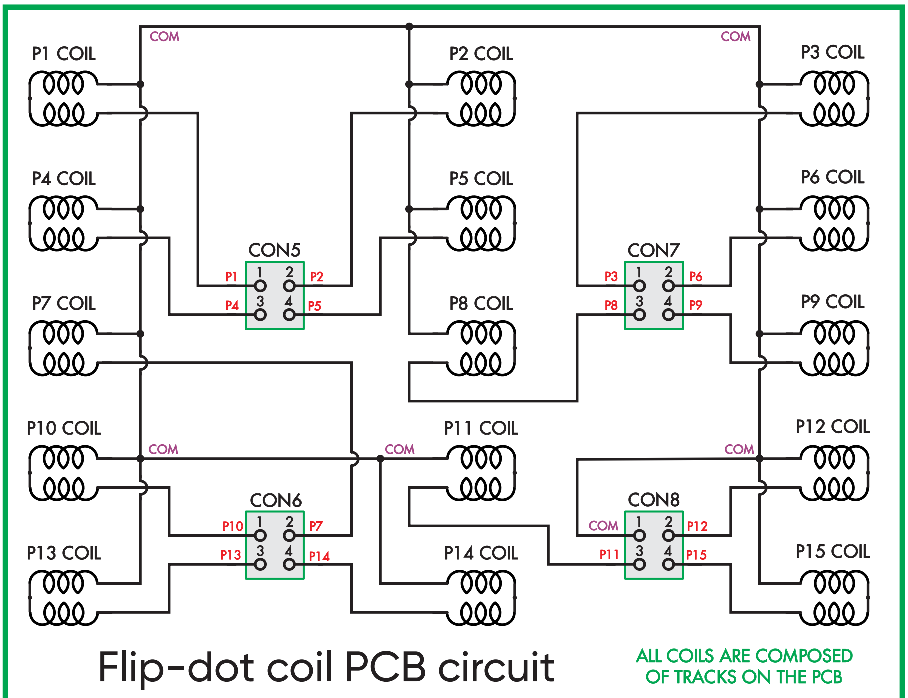


Fig.2: the 15 coils on this PCB are driven by the circuit of Fig.1 and either attract or repel rare-earth permanent magnets mounted in pixel flaps on top of them. Because those rare-earth magnets have a north pole on one side and a south pole on the other side, depending on the direction of current flow through a coil, the flap flips to one side or the other, exposing a different colour.

You could apply the same colours to the coil PCB, although this will need masking to ensure the colours are kept separate. However, we think most constructors will be happy with the black and white as supplied, since it provides good contrast under just about any lighting conditions.

Note that if you are building multiple displays to be ganged together, it's a good idea to ensure that the magnetic polarity is consistent across all the displays, to avoid extra driver software complexity.

If different characters have different pixel black/white orientation, this will need to be programmed into the software, so that it can give a consistent display across characters.

Building the frame

You will need six frame elements to build one fifteen-pixel display. But note that if you are going to be stacking two frames vertically, you will only need eleven in total; one frame will be shared between two boards. The frame pieces are cut from a 72.5 × 75mm PCB that contains eight separate frame pieces, as shown in Fig.3.

Carefully break the frame pieces out of the PCB panel. You may find it easier to cut one side out of the panel with side-cutters before separating each element along the perforated 'mouse-bites'.

The frame pieces do not need to be cleaned up to work correctly, although they can be filed flat along the mouse-bite edges if you prefer. The PCBs are made of fibreglass, so any filing should be done outside with a mask, to avoid breathing in fibres.

The long, flat edge is visible from the front of the display when mounted, so you may wish to colour this black (eg with a marker or paint) to improve the contrast of the display. Note that while our photos show green frames on our prototype, the final boards (available from the *PE PCB Service*) will have a black solder mask instead.

The frames sit on the front of the coil PCB but are soldered at the back, so you won't see any solder when looking at the display later. Line up the edges of the two PCBs; the frame should sit at right-angles to the coil PCB. You will need a fairly large soldering iron tip and be generous with the solder to ensure the fillet bridges the gap.

It's a good idea to solder one of the tabs at the back and check the position before soldering a tab at the other end.

You might like to leave just one tab soldered until the flaps are fitted, as this will give a small amount of flex to the frame, allowing the flaps to be slotted in with less effort.

If you do this, though, make sure to come back later and solder at least one more tab on each frame piece, once you have confirmed that the unit works correctly.

The coil PCB is probably the most delicate part, as the fine copper traces are near the limit of manufacturing tolerances. The traces run quite close to the edge of the board, and if they are damaged, they will be next to impossible to repair and the display may not work correctly – so be careful with it.

On the reverse of the coil PCB, there are pads for four 2×2 pin SMD male headers – see Fig.4. These headers are a similar size overall to their through-hole equivalent.

It's a good idea to push the female header sockets (which will be soldered to the driver board later) over the pins on the SMD headers before soldering them. This way, if you accidentally apply too much heat, they should stay in alignment.

The use of surface-mount headers here means that the front of the display remains unspoiled by soldered joins.

As with any other SMD part, the simplest way to locate the headers correctly is to solder one pin in place, then, after checking that it is in the correct location, solder the remainder. The mating holes for the female headers on the driver PCB are slightly oversize, to allow for minor inaccuracies in the placement of the male headers.

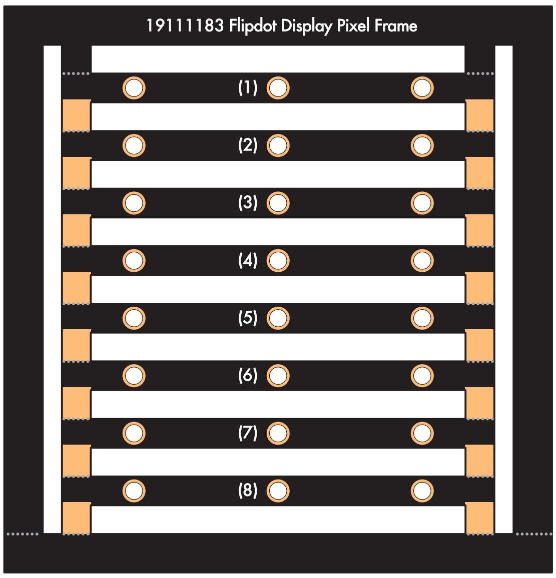


Fig.3: this PCB can be cut apart into eight separate frame pieces - enough to make one 3 × 5 pixel flip-dot display with two pieces left over. The holes form the 'hinges' for the pixel flaps to rotate about, while the exposed copper is soldered to the coil PCB to hold the frame in place.

Cut carefully where shown using a side-cutter to separate the pieces. The frame pieces are quite thin and could be damaged if handled roughly.

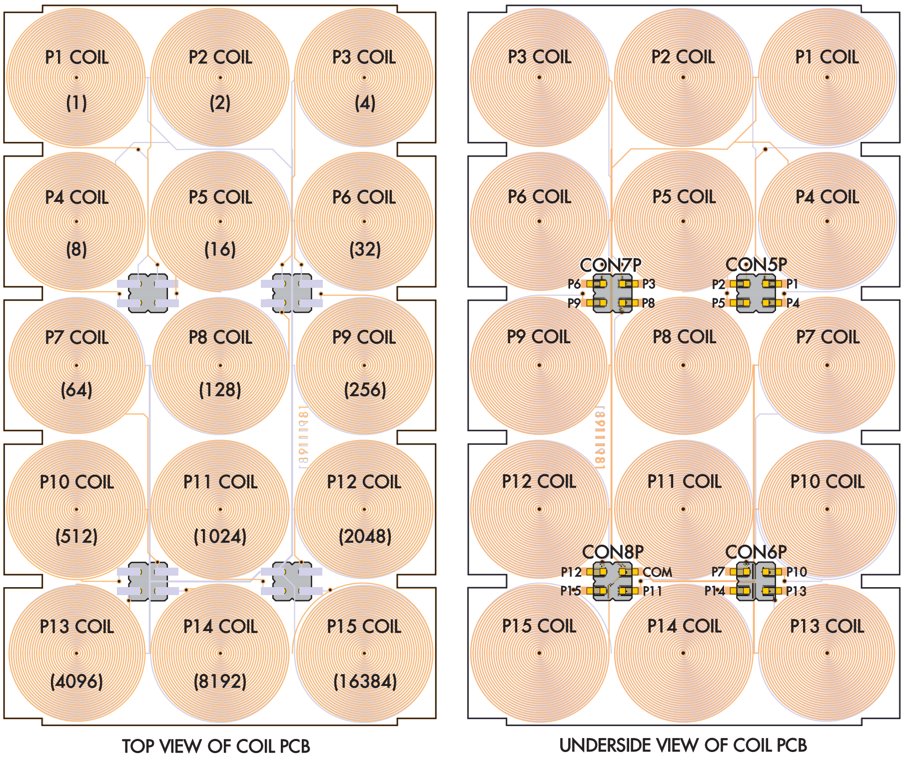


Fig.4: the coil board. Each coil is made from copper on both sides of the board. Solder four 2×2-pin SMD headers to the back side of this board, as shown. The only parts soldered to the top side of the board are the six frame strips which hold the pixel flaps in place. Add numbers in parentheses for each pixel that you want to be 'on' to determine the code used to produce a particular character. For example, 2+8+32 = 42 will give you a caret (^) on the display.

Driver PCB construction

The driver PCB can be built next. We recommend fitting the ICs first, as their placement is not critical. Refer to Fig.5, the PCB overlay diagram, to see which parts go where.

IC1 and IC2 are both 74HC595s, which are fitted at the top of the PCB, with their pin 1 facing down. IC3-IC6 are L293D types, and these go at the bottom of the PCB, with their pin 1 to the left. All six ICs have 16 pins, so take care that they do not get mixed up.

We recommend soldering them all directly to the board, rather than using sockets, for reliability (and because the pins of IC3-IC6 carry fairly high currents). You could use sockets for IC1 and IC2 if you really want to.

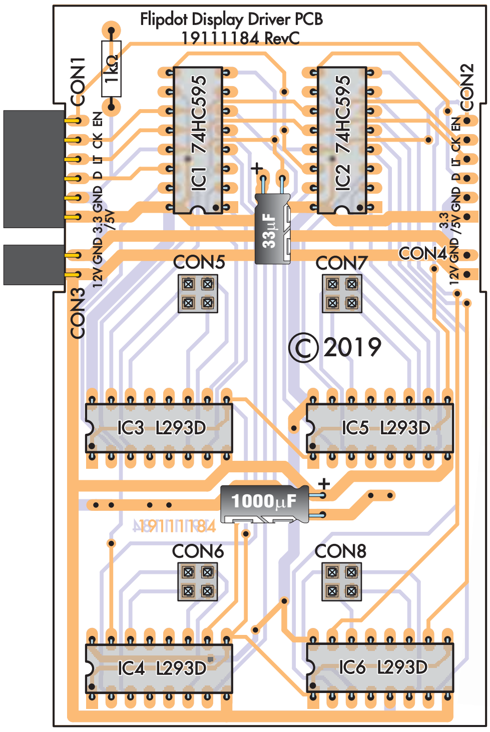
After confirming that the ICs are well seated and correctly oriented, solder all the pins to the PCB, ensuring that you do not put too much heat into the IC. The ground pins on IC3-IC6 (the four pins closest to the centre) sit on a large copper area to provide some heatsinking, so these pins may require extra heat to ensure a good solder joint.

Next, mount the capacitors. Both are the polarised electrolytic type, so observe the polarity marks on the PCB. The longer leads go into the pads marked with a '+' sign, while the striped side of the can is negative. The smaller 10µF capacitor sits between IC1 and IC2. You will need to lay it over on its side, as the coil PCB will sit quite close above it.

The 100µF capacitor fits between IC5 and IC6. It too will need to be laid over. It does not matter which way the capacitors are laid as there is ample space on the PCB.

Fit the female headers next. A good way to ensure that they are mounted square and parallel is to push them over the male header pins on the coil PCB, and use this as a jig to line them up with the holes in the driver PCB.

Note that if you fitted the female headers to the back of the driver board (which we don't recommend) then you could still plug the two boards together. But you would need to modify the software to make it work, since the connections on CON5-CON8 would all be reversed. Our code assumes that these



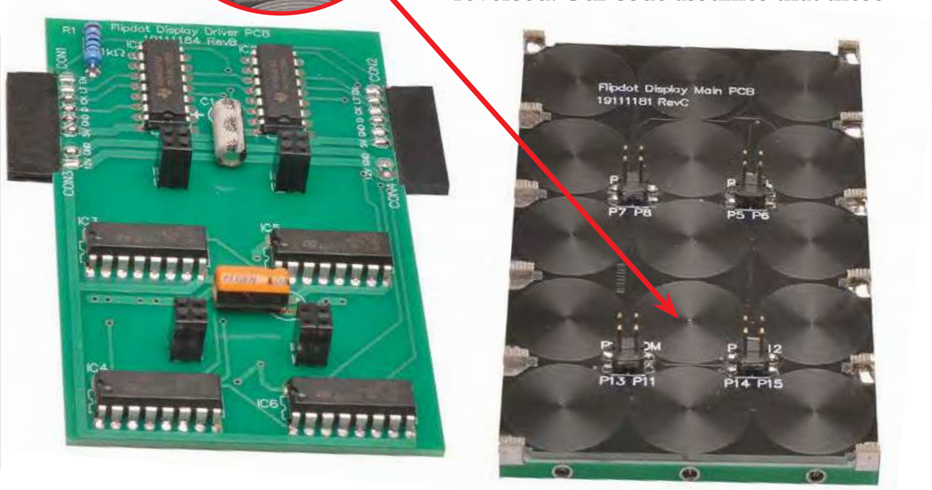


Fig.5: use this PCB overlay diagram and the photo above as a guide to assembling the driver board. Note the location of the headers for CON1 to CON4 and the orientation of the ICs. The two capacitors will need to be laid over to sit under the coil PCB. The female headers are convenient for using jumper wires to a Micromite or Arduino, although you may substitute anything that suits. At right is the *Flip-dot Display* main PCB – it may not be immediately obvious that the circles on this board are in fact coils (see inset) which are responsible for 'flipping' the 'pixel' either white or black.

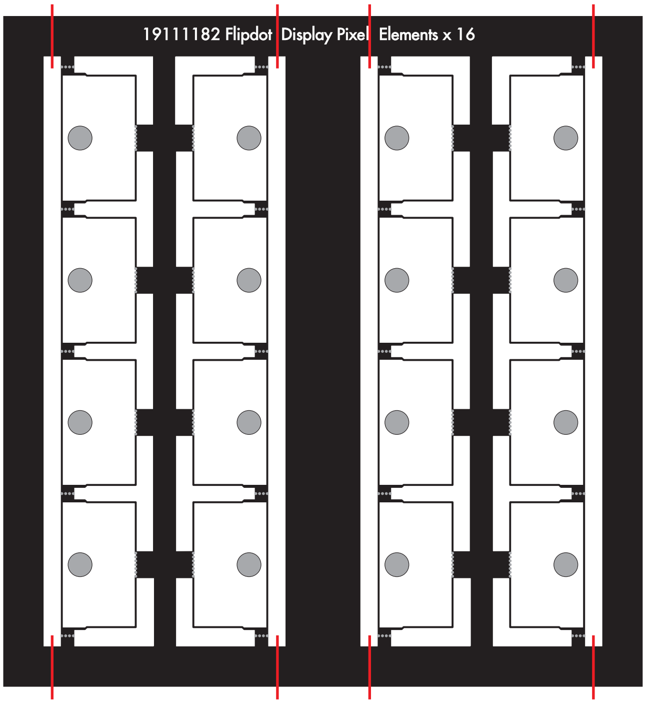


Fig.6: as with the frame pieces, the sixteen pixel flaps are made from PCB material and come joined together.

Cut along the red lines using a sharp pair of side cutters, then separate them at the 'mouse bites'. You can use a file to gently clean up the rough edges if necessary. The magnets are glued into the grey-shaded holes in the middle of each pixel.

headers are on the same side as the other components, so the driver ICs are sandwiched between the two boards.

Ensure that the two boards sit parallel before soldering the female header pins. The holes are slightly oversize, so these pins may need more solder that you might expect.

An alternative to using the female headers is to simply solder the male headers of the coil PCB directly into the driver PCB. You may prefer this if you are building a larger display made of smaller modules, although it will obviously be harder to repair any faults.

Finally, you will need a way to connect the driver PCB's input pins to a microcontroller and power. There are two headers for this. CON3 has two connections for 12V and ground, while CON1 has six connections for 3.3/5V power, ground and logic-level control signals.

CON1 and CON3 are spaced 0.1-inch (2.54mm) apart, so a nine-pin header can be fitted for both, and that is what we've done. It can be broken or cut off a longer header strip if necessary. Solder this to the holes on the left-hand side of the PCB.

For the first board, which will be wired back to the controlling device (eg, Arduino or Micromite) it's best to use female header(s) for CON1 and CON3, to allow male-to-male jumper wires to be used.

But for subsequent boards in a multi-character display, you're better off using a male pin header for CON1 and CON3 instead. This can then be soldered directly to the CON2/CON4 positions on the adjacent board, which holds the two together and allows the PCBs to butt right up to each other, thanks to the two shallow cut-outs on the edges of the board, into which the header's plastic block slots.

Another option would be to fit a female header (socket) for CON2/CON4 on one board, and a male pin header for CON1/CON3 on the next board, and plug them together. This would make it easier to disconnect the boards later if necessary, but they would then have a gap between them. And you would need to come up with a way to hold them together, since the socket won't provide enough friction.

CON2 and CON4 are not needed for a single display. You can leave them off at this point, and fit something later after you have tested the unit, if you decide to combine it with additional display boards.

Final assembly

Now that the glue and paint on the pixel flaps has cured, these can be fitted to the coil PCB's frames. But first, they need to be removed from the PCB panel.

The best way to do this is to carefully cut the panel into smaller pieces using a sharp pair of side-cutters. Take care because the PCB material is quite brittle, and the cut pieces may tend to fly off. Aim away from the body, and use eye protection. Fig.6 shows the recommended cutting locations.

Now, without using any tools, break the flaps by hand from the panel along the mouse-bites. We found that the rough edges were generally not a problem, but they can be filed back a small amount (one or two passes only) with a fine file. Again, beware of breathing the dust from the PCB.

A good test to check that the pixels are all magnetically aligned correctly is to allow them to attract each other into a single stack. If all the flaps show the same colours on the same side, then they are aligned magnetically.

The pixel flaps are simply a firm press fit into the frames. Line up the colours so that the white side of the flap is adjacent to the white side of the coil PCB and the black side of the flap is adjacent to the black side of the coil PCB (see photo).

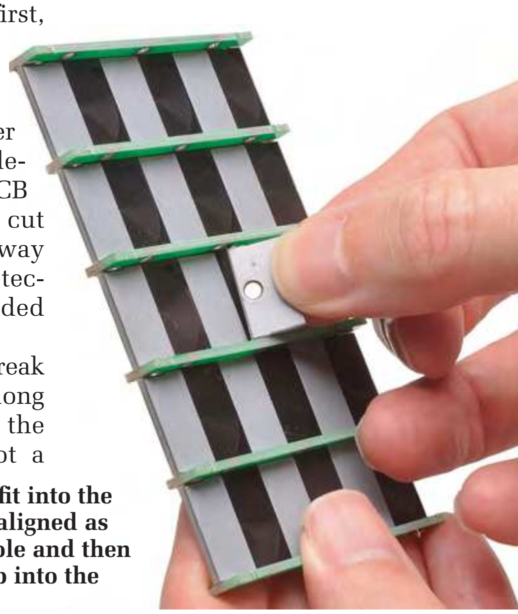
Sit the bottom tab into the hole in the frame, and then gently rotate the upper tab into the hole. Once all the flaps are installed, check that the pixels will all flip freely. This can be done by rotating the entire assembly in your hand and allowing the flaps to move under the influence of gravity.

Connect the coil PCB to the driver PCB by plugging the headers together. The assembly should sit upright on its bottom edge, with a very slight backwards tilt. The backwards tilt will help the flaps to stay in their last driven position.

Connect the micro

The final step for testing is to connect a microcontroller to control the pins. You will also need a source of 12V DC, with preferably at least 1.5A capacity. The ground and 12V supply are connected to CON3, while the 3.3V/5V power and logic signals go to CON1.

See the diagrams for either the Arduino (Fig.7) or Micromite (Fig.8) to suit what you are using. If you are using a microcontroller which has been previously programmed for other purposes, we suggest that you re-program it with the software for this project



The pixel flaps are a simple press-fit into the holes. Ensure that the colours are aligned as shown, slot one tab in the lower hole and then rotate the flap to snap the other tab into the upper hole.

before wiring it up, since if it drives the enable pin high without resetting the latch ICs first, that could cause the driver ICs to overheat.

Testing

Our first test program (available from the April 2020 page of the *PE* website) for either the Arduino or Micromite just cycles between all pixels white and all pixels black. Load this into your micro board (at this point, we're assuming you're comfortable working with Arduino or Micromite modules).

Both programs define which micro output pins control the *Flip-dot Display* via constants at the top of the program code. The pin configuration can be changed by changing the `#define` or `CONST` values. The default pins are grouped together for simplicity of wiring.

Check that the board works as expected and that the driver ICs and the coils don't get hot. They may get warm, but if any are too hot to touch, something is not right. If this is the case, there may be a wiring problem or the driver PCB may be assembled wrong. For example, swapping the clock (CK) and latch (LT) lines between the micro and driver board will cause problems.

If you see multiple pixels flipping at the same time, that is also a sign that the wrong data is being received from the board, pointing to a wiring error between the micro and the driver PCB.

Depending on the rating of your power supply, a fault may cause the L293Ds or the coil PCB to get very hot. Take care when touching the display if you suspect a fault.

Once you have confirmed it's working correctly, check that the pixels flip in sequence. If one or two are not turning over correctly, the tabs at the end of the flaps may be catching against the adjacent pixel. In that case, remove any sticky pixels by gently pushing them down against the frame and tilting them out of the mounting holes. File the ends with just one or two passes of a file, again being wary of the PCB dust.

Double-check that the other pixels are seated correctly in their mounting holes and that they can rotate freely. Then refit the ones you filed, ensuring that the colours line up correctly. You may find that they will operate more smoothly after bedding in (ie, running the test program for a while). Once you are happy with the operation and wiring, try the other example programs.

The **Flipdot_ASCII_2.ino** example sketch also contains a routine that only changes pixels that need to be

Parts list (per each 3 × 5 pixel display)

1 black double-sided PCB* coded 19111181, 96 x 58mm (coil board)
1 green double-sided PCB* coded 19111184, 96 x 58mm (driver board)
6 pieces from black PCB* coded 19111183, each piece 58 x 8mm (frame pieces)
15 pieces from black PCB* coded 19111182, each piece 19 x 10mm (pixels)
*Note: all the PCBs are available fron the *PE PCB Service*.

15 3mm diameter, 1.5mm-thick rare earth magnets (available from eBay or Amazon)
4 2x2-way SMD male header [eg, snapped from Altronics P5415]
8 2-way or 4 2x2-way female header sockets
1 9-pin female or male header (CON1,CON3) (see text for details)
Epoxy resin for gluing magnets into flaps

Semiconductors
2 74HC595 8-bit shift registers, DIP-16 [Altronics Z8924, Jaycar ZC4895]
4 L293D motor driver ICs, DIP-16 [Altronics Z2900, Jaycar ZK8880]

Capacitors and resistors
1 1000µF 16V electrolytic capacitor
1 33µF 6.3V electrolytic capacitor
1 1kΩ 1/4W 1% metal film resistor

Additional parts
1 12V DC 1.5A power supply (*higher current may be needed for multi-character displays*)
1 Arduino or Micromite board for control
1 set of jumper leads to connect to microcontroller and power supply

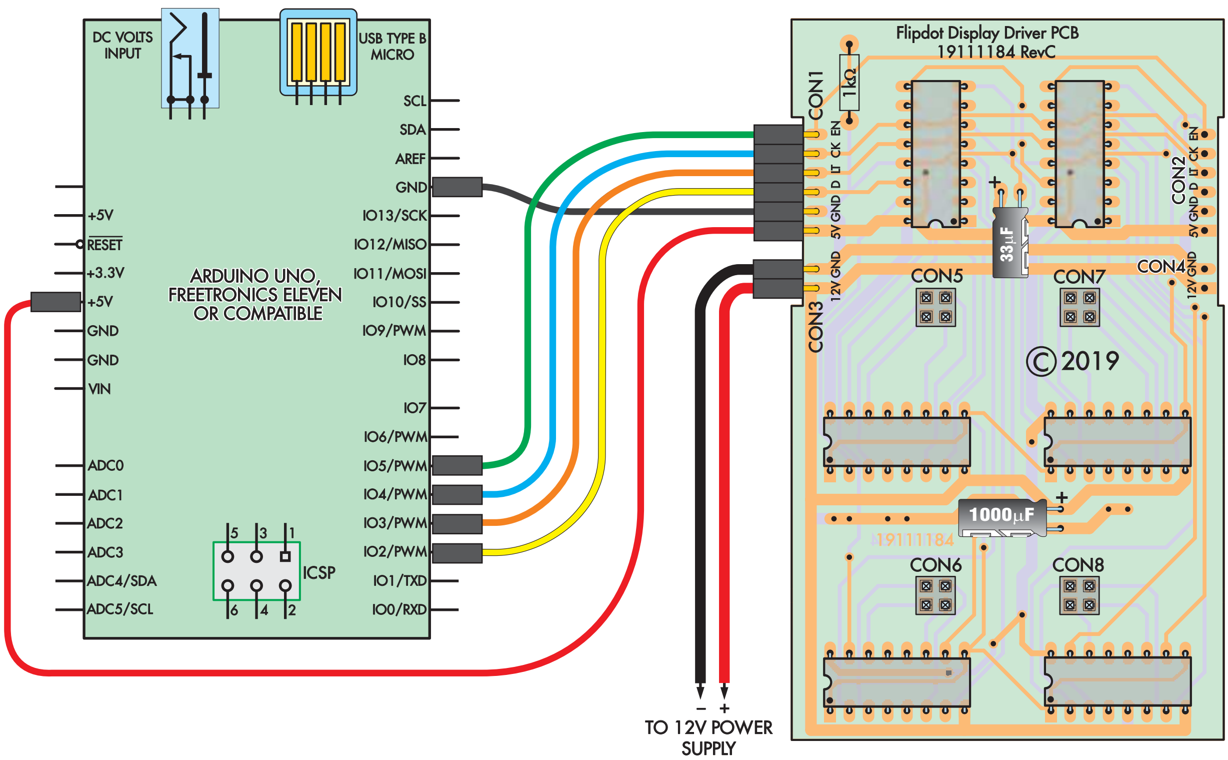


Fig.7: this wiring diagram shows how the *Flip-dot Display* can be connected to just about any Arduino-compatible board. The microcontroller needs just four digital outputs to control the display.

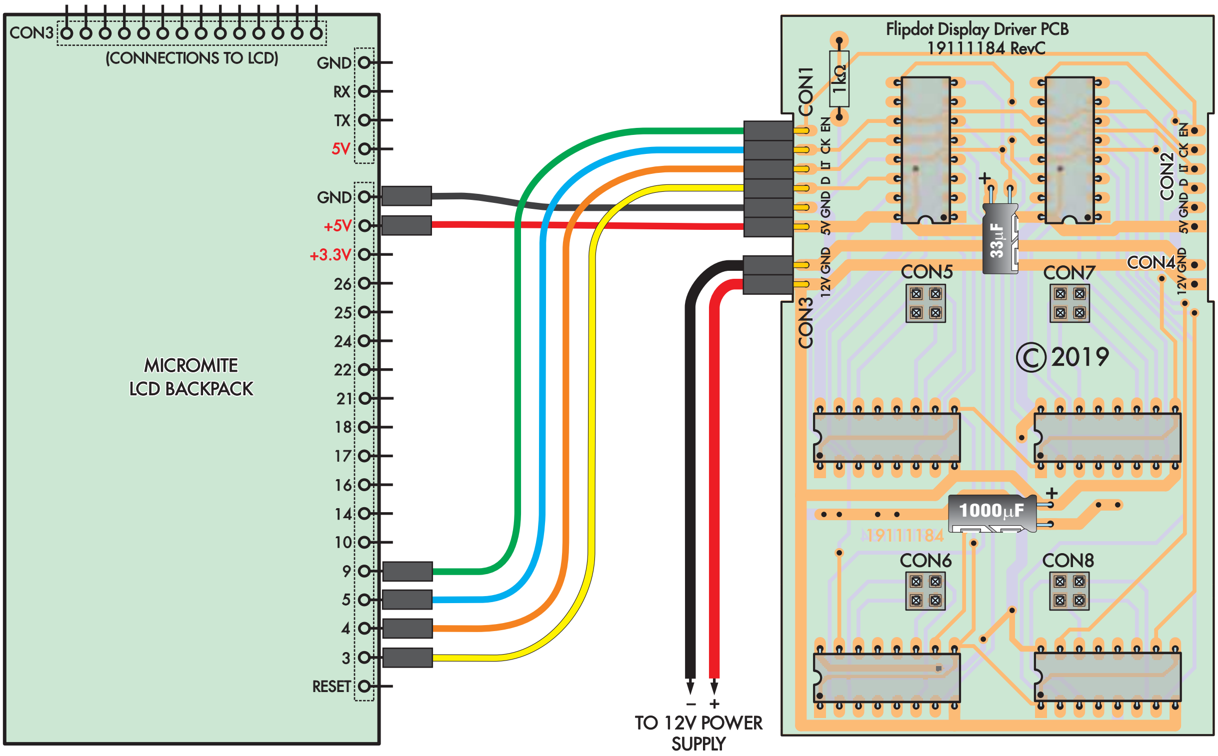


Fig.8: a microcontroller with 3.3V I/O can also control the *Flip-dot Display* directly, such as the Micromite shown here. This is the recommended wiring, which allows you to use our test and sample programs without having to modify them.

changed, improving the update speed and reducing the power requirement.

Using the display

Both the Micromite and Arduino programs make use of a 16-bit value to store the displayed data for a single board. Fig.4 shows the bit mask values of each pixel. To create a particular configuration, add up the values for each pixel that you want to be black and ignore those which you want to be white. The resulting number represents that configuration and can then be used in the software.

If you find the colours are reversed to what you expect, then there are constants defined at the start of the program which can be changed to reverse the colours. Check the comments in the files to see.

This can be caused by all the magnets being reversed relative to what the program expects. So it's entirely possible that you will have to change these constants.

A small amount of epoxy resin is all that is needed to hold the magnets in the flaps. The steel panel (underneath) keeps the magnets flush, and the plastic inbetween stops the magnets sticking to the steel.

Multi-character displays

As mentioned earlier, multiple displays can be chained together to make a larger display by fitting a male header for CON1/CON3 on the second and subsequent boards and soldering these to the CON2/CON4 positions on the adjacent board.

This results in all the control and power pins being connected in parallel, except for the data pin.

The data out signal (pin 3 of CON2) connects to the data in signal (pin 3 of CON1) on the subsequent board, so that serial data passes from one board to the next and therefore, the controlling micro can independently set the state of all pixels in the chain.

Note that the enable pull-down resistors of connected boards are effectively connected in parallel, so you only need to fit this resistor to the first board (ie, the one that will be connected to the micro).

The coil PCBs can also be joined by soldering the tabs of the frame PCBs on adjacent boards. This can also be done to connect multiple rows of boards vertically.

A single *Flip-dot Display* is modestly sized by itself, but with four or six units placed side by side, you could create an attention-demanding clock which gives you a gentle audible alert every time the minutes or seconds digit changes.

With multiple displays, each panel is capable of updating one pixel at a time, so the update time does not increase as you add more characters, as long as your power supply is capable of supplying enough current for all the displays to be driven simultaneously.

12V supply

You need a 12V supply capable of several amps for a multi-character display, and we recommend you parallel the 12V bus with wires that have a decent current-carrying capability, to help deliver that extra current to all the boards.

The software uses the shift registers to shift in the new data for each panel, then toggles the global enable line and they all update in sync.

The largest and most complicated sample program provided allows you to define the number of characters in your display, then update them all with a new text string as required.

Note that lower-case letters in this string are automatically mapped to upper case, since those are much clearer when displayed on a 3 × 5 pixel matrix. Numbers and symbols are left as-is.

STORE YOUR BACK ISSUES ON CD-ROMS

ONLY £16.45 each
Five years just £29.95
including VAT and p&p

VOL 1	**BACK ISSUES**	– January 1999 to June 1999 Plus bonus material from Nov and Dec 1998
VOL 2	**BACK ISSUES**	– July 1999 to December 1999
VOL 3	**BACK ISSUES**	– January 2000 to June 2000
VOL 4	**BACK ISSUES**	– July 2000 to December 2000
VOL 5	**BACK ISSUES**	– January 2001 to June 2001
VOL 6	**BACK ISSUES**	– July 2001 to December 2001
VOL 7	**BACK ISSUES**	– January 2002 to June 2002
VOL 8	**BACK ISSUES**	– July 2002 to December 2002
VOL 9	**BACK ISSUES**	– January 2003 to June 2003
VOL 10	**BACK ISSUES**	– July 2003 to December 2003
VOL 11	**BACK ISSUES**	– January 2004 to June 2004
VOL 12	**BACK ISSUES**	– July 2004 to December 2004
VOL 13	**BACK ISSUES**	– January 2005 to June 2005
VOL 14	**BACK ISSUES**	– July 2005 to December 2005
VOL 15	**BACK ISSUES**	– January 2006 to June 2006
VOL 16	**BACK ISSUES**	– July 2006 to December 2006
VOL 17	**BACK ISSUES**	– January 2007 to June 2007
VOL 18	**BACK ISSUES**	– July 2007 to December 2007
VOL 19	**BACK ISSUES**	– January 2008 to June 2008
VOL 20	**BACK ISSUES**	– July 2008 to December 2008
VOL 21	**BACK ISSUES**	– January 2009 to June 2009
VOL 22	**BACK ISSUES**	– July 2009 to December 2009
VOL 23	**BACK ISSUES**	– January 2010 to June 2010
VOL 24	**BACK ISSUES**	– July 2010 to December 2010
VOL 25	**BACK ISSUES**	– January 2011 to June 2011
VOL 26	**BACK ISSUES**	– July 2011 to December 2011
VOL 27	**BACK ISSUES**	– January 2012 to June 2012
VOL 28	**BACK ISSUES**	– July 2012 to December 2012
VOL 29	**BACK ISSUES**	– January 2013 to June 2013
VOL 30	**BACK ISSUES**	– July 2013 to December 2013
VOL 31	**BACK ISSUES**	– January 2014 to June 2014
VOL 32	**BACK ISSUES**	– July 2014 to December 2014
VOL 33	**BACK ISSUES**	– January 2015 to June 2015
VOL 34	**BACK ISSUES**	– July 2015 to December 2015
VOL 35	**BACK ISSUES**	– January 2016 to June 2016
VOL 36	**BACK ISSUES**	– July 2016 to December 2016
VOL 37	**BACK ISSUES**	– January 2017 to June 2017
VOL 38	**BACK ISSUES**	– July 2017 to December 2017
VOL 39	**BACK ISSUES**	– January 2018 to June 2018
VOL 40	**BACK ISSUES**	– July 2018 to December 2018
VOL 41	**BACK ISSUES**	– January 2019 to June 2019

FIVE-YEAR DVD/CD-ROMs

No.1 – Jan 03 to Dec 07
No.2 – Jan 04 to Dec 08
No.3 – Jan 05 to Dec 09
No.4 – Jan 06 to Dec 10
No.5 – Jan 07 to Dec 11
No.6 – Jan 08 to Dec 12
No.7 – Jan 09 to Dec 13
No.8 – Jan 10 to Dec 14
No.9 – Jan 11 to Dec 15
No.10 – Jan 12 to Dec 16
No.11 – Jan 13 to Dec 17
No.12 – Jan 14 to Dec 18 – **NEW!**

15 YEARS OF EPE!

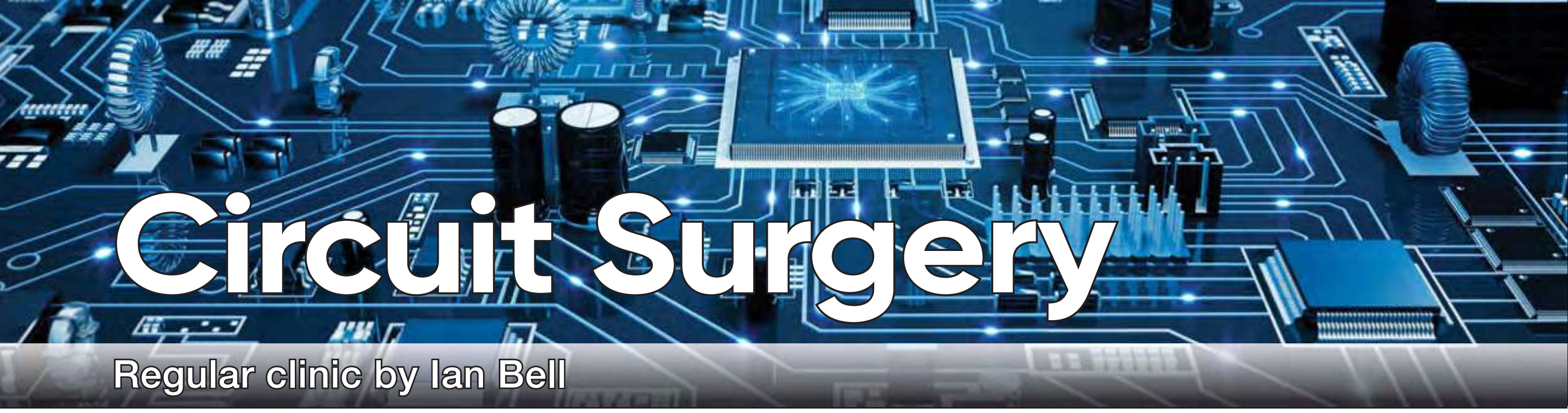
AMAZING BUNDLE OFFER

No.1 – Jan 03 to Dec 07
PLUS
No.6 – Jan 08 to Dec 12
PLUS
No.11 – Jan 13 to Dec 17

ALL FOR JUST
£39.95

All DVD/CD-ROMs are suitable for any PC with a DVD/CD-ROM drive.
They require Adobe Acrobat Reader (free from: www.adobe.com/acrobat)

ORDER YOUR CD-ROMs TODAY!
JUST CALL 01202 880299 – OR VISIT www.epemag.com



Circuit Surgery

Regular clinic by Ian Bell

Problems with SPICE simulations

John Curtin posted a question on the EEWeb forum about a problem simulating a circuit in LTspice. 'I need help for LTSpice, if possible. I'm simulating an RC net to charge an electrolytic capacitor through a resistor (Fig.1). The voltage on the capacitor should be ascending, according "constant of time", instead it's fixed to 12V. Can you help me, please? Thank you.'

As often happens on EEWeb, a solution was provided by another forum member, Giovanni Di Maria in this case, and we will discuss this and other approaches shortly.

This question is an example of an apparently simple simulation appearing not to work properly – a not uncommon experience for new users of analogue simulators. As regular readers know, we make frequent use of LTspice simulation to illustrate *Circuit Surgery* discussions, and since October 2018 several *Circuit Surgery* articles have been dedicated to getting started with using LTspice. So this article is effectively a continuation of that, and we will look at a few things that might go wrong and one or two commands or settings which may resolve problems.

Our version of John's circuit is shown in Fig.2 – the only difference is that we have given names to the nets (wires) in the circuit. This is useful because it makes it easier to see which signal is which in a results plot. LTspice automatically names nets (eg, `n001`, `n002`,...) but it is easy to get confused about which trace you are looking at, and if you accidently clicked in the wrong place when probing it may not be obvious if you are looking at meaningless node names on the waveform plot. However, this is not the problem in this case, and the simple circuit layout and small size make the issue minimal here. In general though, it is good practice to name all the nets you are interested in plotting data for. Use the `Label Net` command button or Edit > Label Net from the menu.

Charging

Before simulating a circuit it is a good idea to have some idea of what to expect based on the theory of operation. It is useful to estimate parameters such as expected voltages, frequencies or timings. These figures help us choose appropriate simulation settings, such as how long to simulate for.

Here, we have a capacitor charging via a resistor, so we expect the voltage to rise following an exponential curve, asymptotic with the applied voltage. There is a well-known formula that we can use to find the time taken for the capacitor to charge. Specifically, for a charging capacitor the time (t) taken to reach voltage V_t charging from 0V towards an applied voltage of V_a is given by:

$$t = -RC\ln\left(1 - \frac{V_t}{V_a}\right)$$

Where 'ln' is the natural logarithm of the item inside the brackets. In John's circuit the capacitors are charging towards 12V, so V_a = 12V. If we choose, say, the time taken to reach 90% of the applied voltage (about 11V), then V_t = 11V. We can then plug all the values into the above equation and get a V_t of about 25s. John's simulation is set to run a transient analysis (time-based simulation) for 60s of simulated time (`.tran 60`) so we should see the voltage

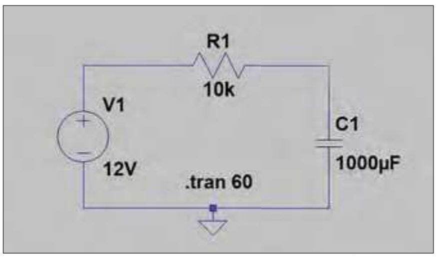


Fig.1. John Curtin's LTspice circuit from the EEWeb forum.

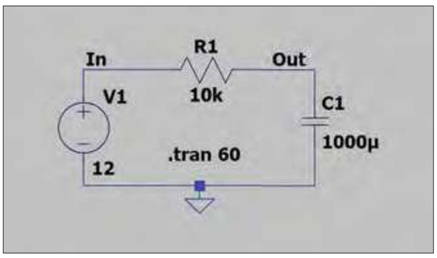


Fig.2. John's circuit with node labels added.

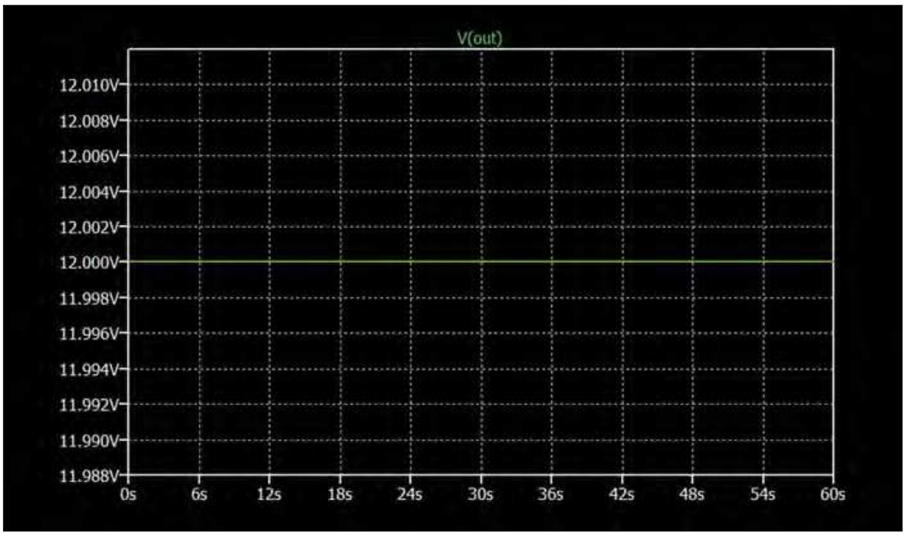


Fig.3. Simulating the circuit in Fig.2 does not produce an RC charging curve.

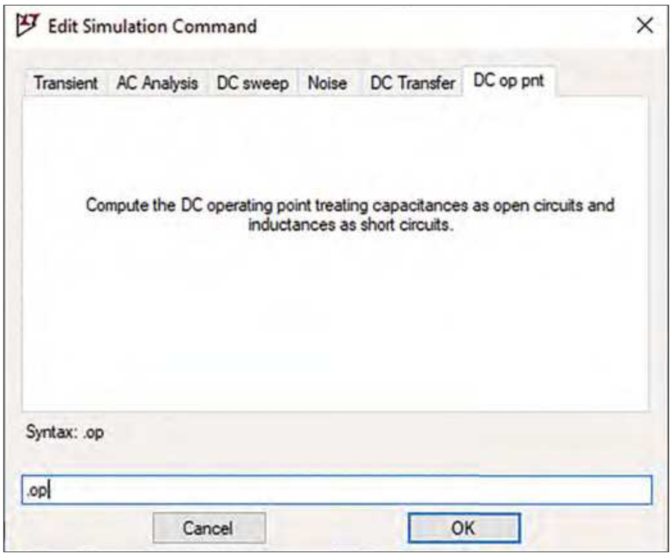


Fig.4. Running a DC operating-point analysis.

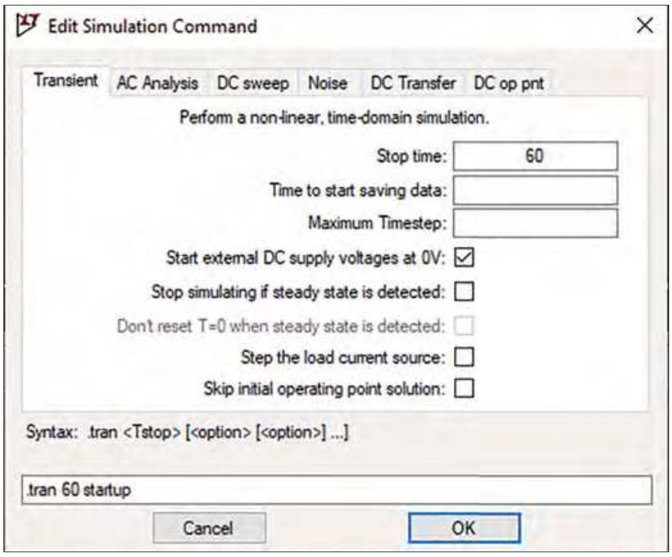


Fig.5. The startup option for transient analysis.

getting very close to 12V about half way through – this is a sensible choice of simulation time and is not the reason for the flat 12V result.

Numerical integration

The result of running the transient simulation for the circuit in Fig.2 is shown in Fig.3. As John described it, the output voltage is indeed fixed at 12V. At this point some people may be inclined to think the simulator has got it wrong – it can happen. In this case, however, it hasn't, but to understand why you need to know what SPICE does when it performs a transient simulation.

In order to perform a transient simulation SPICE has to do numerical integration of the differential equations that model the behaviour of capacitors and inductors. Here, 'numerical' refers to mathematical techniques and algorithms that solve problems by estimating numerical solutions rather than by using symbolic manipulation of equations. It is called 'integration' because it is attempting to solve the differential equations decribing the circuit's behaviour.

Numerical integration methods attempt to find the shape of an unknown curve (such as the waveform in a circuit). This is on the basis that, although the curve is initially unknown, we do know the starting point. Numerical integration techniques estimate the next point along the curve, close to the starting point – for simulation waveforms this point is a small time step into the future of simulated time. Once the new point is obtained it becomes the start point for the next estimate. Of course you have to start somewhere at the very beginning (at simulation time zero) and SPICE simulators find the initial starting point by performing a DC operating-point analysis.

Operating point

A 'DC operating-point analysis' treats capacitances as open circuits and inductances as short circuits and (as its name suggested) calculates the voltages and currents in the circuit with only the DC components of any sources applied. This may be familiar to readers in the context of working out the 'bias point' in circuits, such as transistor amplifiers (also referred to as the 'operating point'), but can be performed on any circuit. Another way to view this is finding the circuit's state after infinite time – that is, any changes that might have occurred when the DC sources 'switched on' have settled down and we have the final steady value.

Looking at the circuit in Fig.2 with just DC voltages considered (which is all we have in this case), and working out what happens with 'capacitances as open circuits and inductances as short circuits', we would expect zero current in R1 due to the open circuit at C1, so there will be zero voltage dropped across R1 and the output would be at 12V. If we think about the circuit settled after infinite time, then C1 will have been fully charged to 12V and there would be no voltage across the R1 – it is the same result. This is the starting point for John's simulation, and, as there is nothing in the circuit to cause any change after this, the voltage across the capacitor remains at 12V, as we see in Fig.3.

LTspice can run just the operating point analysis and give you the results as a list of voltages and currents. If you have already tried the simulation set up as in Fig.2 and Fig.3, then try this by changing the simulation command. With the schematic selected, do Simulation > Edit Simulation Command. In the simulation command dialog (see Fig.4) select the DC op pnt tab and note the comment about computing the operating point (as discussed above). Change the command text to `.op` (it may be `.op 60`), click OK and click on the schematic to place the command. Then run the simulation. The results will appear in a text widow and should be similar to the following:

```
--- Operating Point ---

V(out):  12   voltage
V(in):   12   voltage
I(C1):   1.2e-014 device_current
I(R1):   -1.2e-014 device_current
I(V1):   -1.2e-014 device_current
```

The exact current values may be different as they are approximations to zero (in this case 12fA (one femtoamp = one billionth of a microamp).

There a few ways in which we can obtain a simulation which produces the capacitor-charging curve that John expected. These can be divided into two general approaches. We can use simulator commands (SPICE directives, and `.tran` command options) to change the way in which the existing simulation operates. Alternatively, we can change the simulation to include changing conditions, which result in the capacitor charging.

Startup

The fact that we might want to simulate a situation in which the DC voltages start at zero and switch on (like a power supply) is recognised by an option, which is available for transient simulation. Assuming you have the simulation in Fig.2 and Fig.3 set up and have tried the operating-point analysis, close the results window and open the simulation command dialog

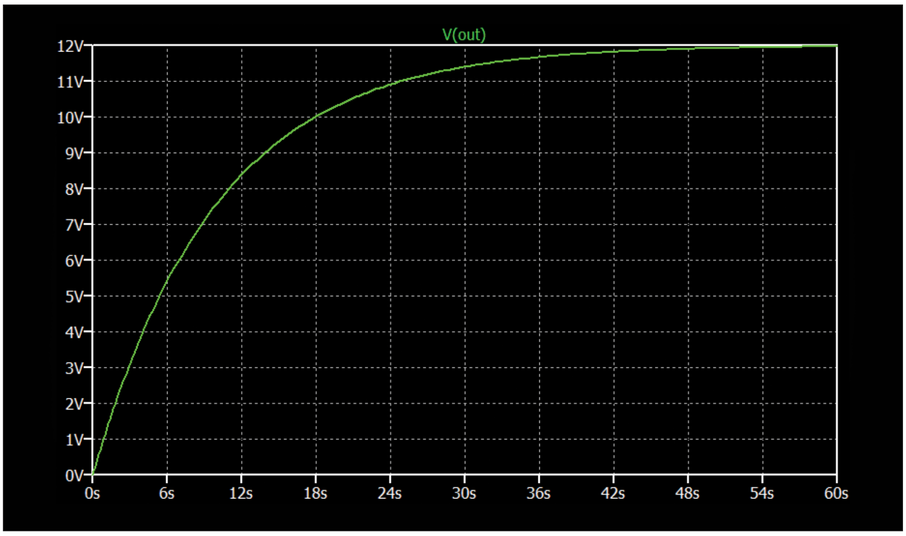


Fig.6. Simulating the circuit in Fig.2 with the start-up option.

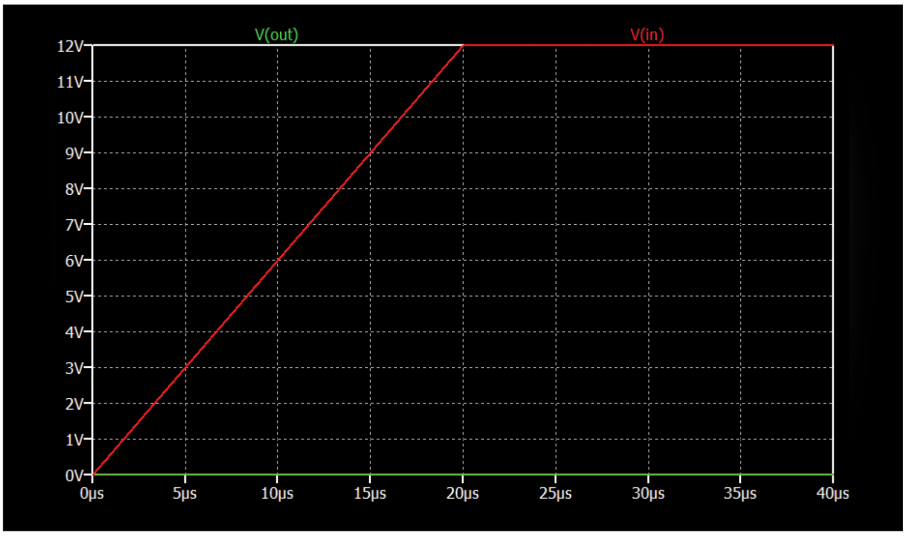


Fig.7. DC source ramping with startup option.

again. Select the Transient tab and re-enter the Stop time: as 60. This will restore the simulation to that used by John. Now also check the Start external DC supply voltage at 0V option (see Fig.5 – note the simulation command is now `.tran 60 startup`). Click OK in the dialog and run the simulation.

The results are shown in Fig.6, where we see the RC charging curve with the timing calculated earlier. The `startup` option causes the DC sources to start at 0V and ramp up to their designated voltage in 20µs. Starting V1 at 0V causes C1 to start with 0V across it, so once V1 reaches 12V, C1 will start charging towards this value. This start-up ramp can be seen by displaying V(in) and zooming in to the first few tens of microseconds of the simulation, as shown in Fig.7. In this case, the ramp time is so short compared with the time constant of the RC circuit that the simulation follows the expected step response, but if the capacitor was much smaller V(out) would basically track the V1 ramp rather than following the exponential charge curve.

Initial condition

Another approach to achieving the RC curve is to set an initial condition in the circuit. This was the approach suggested by Giovanni Di Maria on the EEWeb form. We can do this using a `.ic` (initial condition) SPICE directive. SPICE directives can be used to provide additional instructions to the simulator beyond those contained directly in the simulation command (such as `.tran` and `.op` which we have already seen), which are themselves SPICE directives. The `.ic` directive forces the DC operating point analysis to use the values specified in the directive (it constrains the solution to use these values). It can be used to set initial voltages on nets (and hence across capacitors), or currents in inductors.

Here we can set the initial voltage on the output to 0V, which will start the simulation with C1 discharged. Specifically, we use the directive:

```
.ic V(out)=0
```

This sets the initial output voltage to zero. To do this, starting from the simulation that produced the results in Fig.6 and Fig.7, first edit the simulation command again and uncheck the `startup` option (the command should be just `.tran 60`). Then click on the .op button and enter the `.ic` directive text in the box as shown in Fig.8. Click OK and click on the schematic to add the directive. Then run the simulation. The results will be almost the same as in Fig.6 except that V(in) will be 12V from the very start of the simulation.

Pulse

A different approach to the simulation, as noted earlier, is to create a dynamic situation in the circuit – to somehow switch the voltage applied to the RC circuit during the simulation, rather than producing the curve from a constant DC source. One way to do this is to modify the circuit to include a switch to connect or disconnect the voltage applied to the capacitor, but this means modifying the circuit to add the switch and a control signal for it. A simpler approach is to modify V1 to apply a pulse rather than a fixed DC voltage.

To do this, right click on V1 on the schematic and click the Advanced button on the first dialog that appears. A new window opens which allows us to configure the source. Select `PULSE` from the functions list and set the parameters shown in Fig.9. This will create a single pulse of width 60s, switching from 0 to 12V, starting at 5s. The voltage will switch in 20µs. This is the same as the `startup` ramp used earlier, but here we can choose whatever time we want. Click OK – the

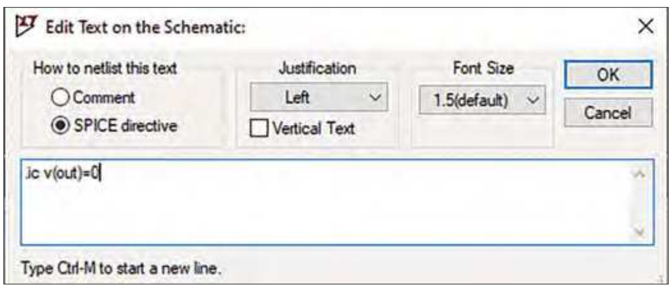


Fig.8. Entering a SPICE directive.

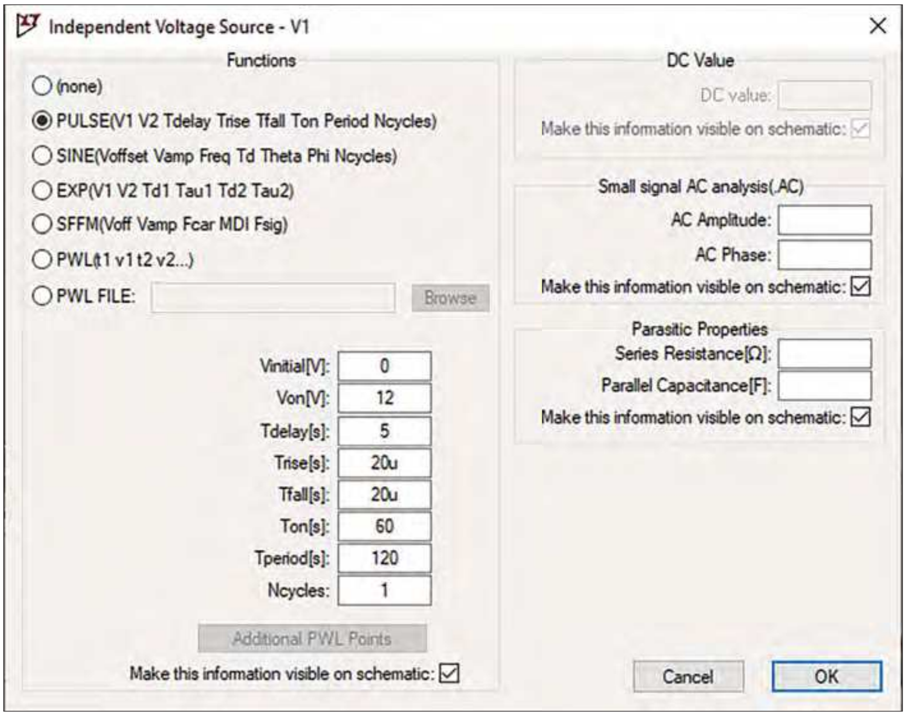


Fig.9. Setting up a pulse from V1.

text describing the pulse: `PULSE(0 12 5 20u 20u 60 120 1)` should appear on the schematic. In order to see the effect of the pulse switching to zero it is useful to extend the simulation, so edit the simulation command to run it for 130s (`.tran 130`).

Running the simulation should produce the results shown in Fig.10. We can see both the charge and discharge curves.

Retro

The circuit in Fig.11 provides another example of a circuit where initial conditions cause a problem with simulation. It is an oscillator; more specifically it is an astable, or free-running multivibrator. This is a 'retro' circuit, which is perhaps little known (and used) now, as we tend instead to employ a variety of special-purpose oscillator or waveform generator ICs, or use microcontrollers to generate timing signals. But it is useful as an example here.

As noted above, before simulating any circuit you should have some understanding of the basic principles of its operation; have some idea of voltages (and currents) likely to occur in the circuit, and be aware of the expected shape and frequency of key waveforms. So we will now look at the operation of the circuit in Fig.11 and obtain an equation for its oscillation frequency.

Operation

Assume that T1 is fully on and T2 is off. This means that one side of C1 is effectively connected to ground (0V), via T1 and the other side will charge towards V_{CC} via R2. Our assumption that T2 is off implies that the voltage at the base of T2 is currently somewhat less than 0.5V. At the same time, the voltage across C2 will be at, or charging towards, something close to V_{CC}. This is because one end of C2 is connected to V_{CC} via R4, and the other is held at around T1's V_{BE} of about 0.7V. T1 does not influence the voltage across C2 at this time because it is fully switched off.

As C1 charges through R2, the voltage at the base of T2 rises and eventually T2 will turn on. At this point, the voltage at the collector of T2 falls very rapidly to close to 0V (as T2 saturates). The voltage across C2 is around V_{CC} with the T2 collector side of C2 positive. This is because C2 will have held its charge during the brief time T2 took to switch on, and means the voltage at the base of T1 will have dropped to around $-V_{CC}$, turning T1 fully off. At this point, the voltage across C1 is quite small (the difference between the V_{CE} saturation voltage of T1 and the V_{BE} of T2, so 0.6V to 0.7V or lower).

The process described above repeats on opposite sides of the circuit. C2 charges up towards $+V_{CC}$ from $-V_{CC}$ via R3. At this time, the voltage across C1 charges from the small voltage it had at the switching point towards V_{CC}. The circuit switches again when C1 charging through R3 brings the V_{BE} of T1 to around 0.6V, switching it on and T2 off. We are back to the start of our description and the whole cycle starts again. The speed at which the circuit switches is determined by how long C1 takes to charge from $-V_{CC}$ via R2 to turn on T2; and how long C2 takes to charge from $-V_{CC}$ via R3 to turn on T1. We can calculate this using the same capacitor-charging equation we used for John's circuit above.

Calculation

The capacitors are charging from $-V_{CC}$ towards $+V_{CC}$, which in effect is an applied voltage of $V_a = 2V_{CC}$. The time we need is when the capacitor voltage is sufficient to turn on the transistor – that is, it has charged from $-V_{CC}$ to $+V_{BE}$. If we ignore the value of V_{BE} for simplicity we can take the voltage of interest as being V_{CC} above the start point, so V_t/V_a is $V_{CC}/2V_{CC}$ = ½, and timing equations for the circuit become:

$$t_1 = -R_2C_1\ln\left(\frac{1}{2}\right) = 0.69R_2C_1$$

$$t_2 = -R_3C_2\ln\left(\frac{1}{2}\right) = 0.69R_3C_2$$

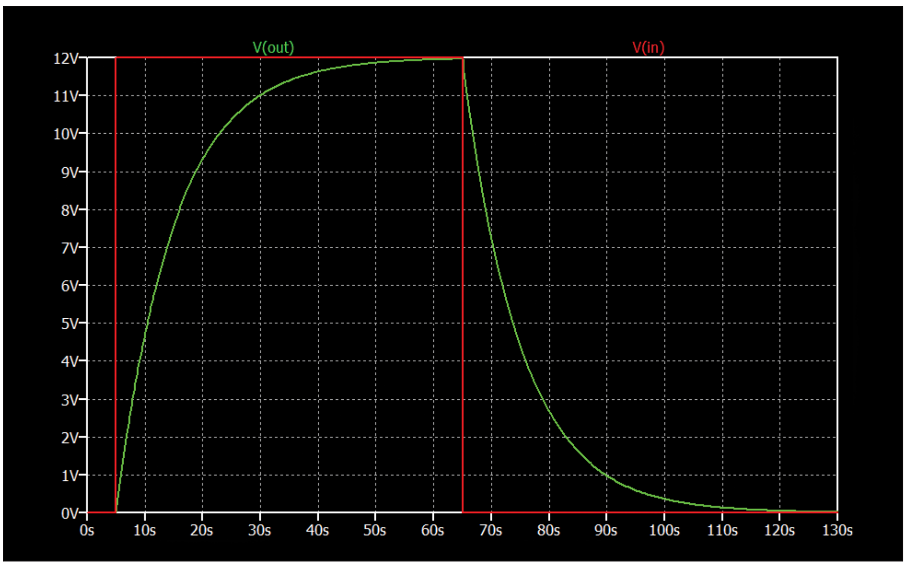


Fig.10. Results of simulation using a pulse source.

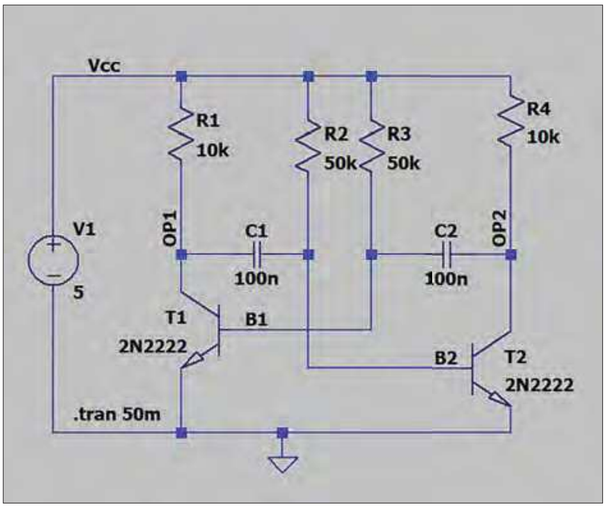


Fig.11. Two-transistor astable multivibrator LTspice schematic.

The period of oscillation is t_1+t_2. If we use equal component values in the two halves of the circuit, so that R1 = R2 = R and C1 = C2 = C, $t_1 + t_2$ is 1.38RC and so the frequency of oscillation is:

$$f = \frac{1}{1.38RC} Hz$$

From the values used in the circuit in Fig.1 we would expect a frequency of oscillation given by: $f = 1/(1.38 \times 1.0 \times 10^{-7} \times 5.0 \times 10^4) = 145$Hz, which is a cycle time of 6.9ms. We would expect the voltage at the collectors of the two transistors to alternately rise from 0V to close to the supply, following a typical capacitor charging curve, and then very rapidly returning to 0V once the transistor switches on, where it will remain for the rest of the cycle. In each cycle we expect the base voltages to suddenly drop to about $-V_{CC}$ and immediately follow a resistor-capacitor charging curve back up to about 0.6V, where the voltage will remain more or less static for half of the oscillation cycle.

Flatline

Now we know what to expect, we can set up a suitable transient simulation. Given that one cycle is around 7ms, a transient simulation of 50ms seems reasonable. Unfortunately, if we run this, the expected oscillation does not happen. Both collectors stay at about 30mV and both bases are at about 650mV throughout the simulation. We have another flatline simulation when we expected something more interesting to happen – what is wrong?

The schematic in Fig.11 is perfectly symmetrical – both halves of the circuit are exactly the same. In fact, the circuit will behave in the same way if you removed the capacitors, creating two isolated sub-circuits. The exact symmetry is present in the simulated circuit, but the probability of it happening in a real circuit is so small as to be effectively zero. What we have is a situation of perfect balance in an unstable system, like balancing a ball on the point of a needle; it might be theoretically possible, but in a real system there is always some asymmetry, or external disturbance, which causes imbalance.

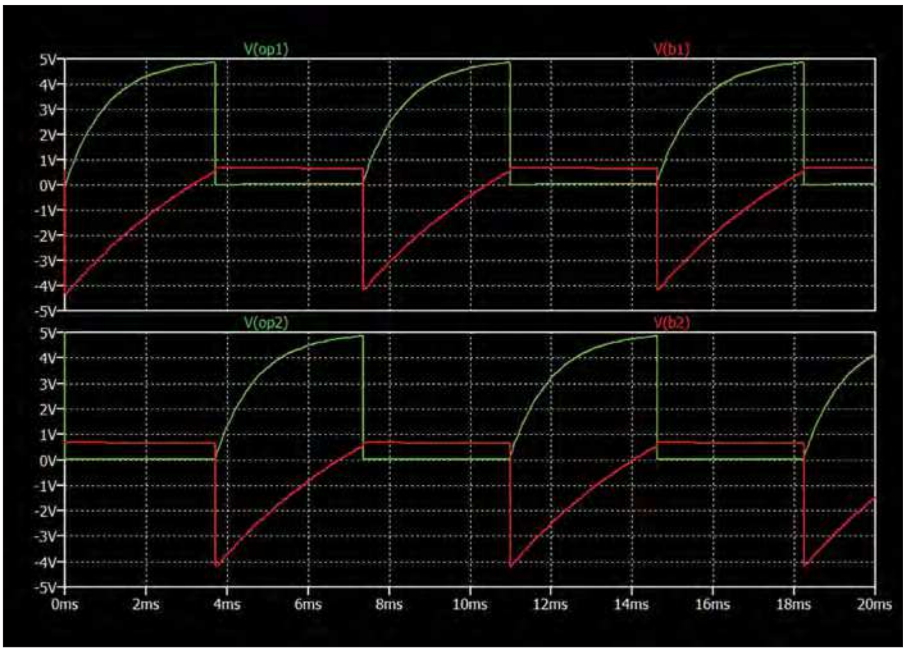


Fig.12. Working simulation of the astable in Fig.11.

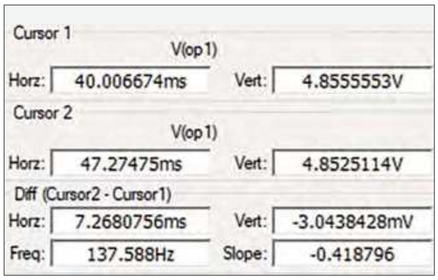


Fig.13. Measuring the frequency using the waveform cursors.

Oscillation

In the real astable, one transistor will switch on slightly faster than the other when power is first applied, the feedback in the circuit will cause the other transistor to tend to switch off and the faster transistor to switch on more. Thus the circuit will quickly tip into one of two possible initial states and will then start to oscillate in the manner already described. We can get the circuit to oscillate in the simulator by forcing an imbalance at the start of the simulation. One way of doing this is to use the `.IC` SPICE directive, discussed above. We can use the following `.IC` statement to set the initial voltages on the two outputs (collectors) to 0V and 5V, thereby ensuring the circuit starts in an asymmetric state. The transient waveform will start from the conditions set by `.IC`.

```
.ic v(op1)=0 v(op2)=5
```

The resulting simulation waveforms (first 20ms) are shown in Fig.12. The shape of the oscillation waveforms fits with our earlier reasoning about the circuit's behaviour, giving us confidence that it is correct.

Using the cursors – obtained by right-clicking the signal names at top of the waveform traces – we can measure the frequency of oscillation. Using `V(op1)` and aligning the cursors with two successive points where the steep voltage drop occurs produces the results shown in Fig.13. We can read the frequency directly as 138Hz, which is close enough to our estimate above to indicate the simulation is working as expected.

Simulation files

Most, but not every month, LTSpice is used to support descriptions and analysis in *Circuit Surgery*.
The examples and files are available for download from the *PE* website.

Practically Speaking

Hands-on techniques for turning ideas into projects – by Mike Hibbett

Introduction to surface mount technology – Part 1

Over the years that this magazine has been in circulation – the first issue came out in 1964 – our projects have been based on a wide variety of electronic components and soldering techniques. Integrated circuits were not widely available in 1964, certainly not for hobbyists; and in the early days we soldered components between lugs on a pegboard, often leaving the components in the air. We certainly used transistors, but also valves ('tubes' for our US friends), which were so large they needed metal work to support them. In those days, some projects were as much mechanical construction as electrical!

It's difficult to accurately define when electronic components were first introduced to the public; it very much depends on your definition of an 'electronic component'. Leyden jar 'capacitors' probably come first (18th century), acting as a primitive electrical power source – but they were important because a circuit cannot function without a power supply. These were followed by piles (early batteries), coils, including transformers, and large plate capacitors, all used extensively by scientists like Faraday, Hertz and later Marconi to create early communication technologies such as telephone, data transmission via Morse code and radio.

The invention of vacuum tube diodes and then triodes using thermionic emission technology (based on X-ray tube and light bulb research) ushered in what we would now call 'electronics', ultimately leading to the birth of what we recognise as our hobby in the late 1920s.

The Second World War was a huge impetus to electronics. The mass production of radio, radar, electronic proximity fuses and a whole host of other devices spurred research that led to huge improvements in electronic component size, quality and price.

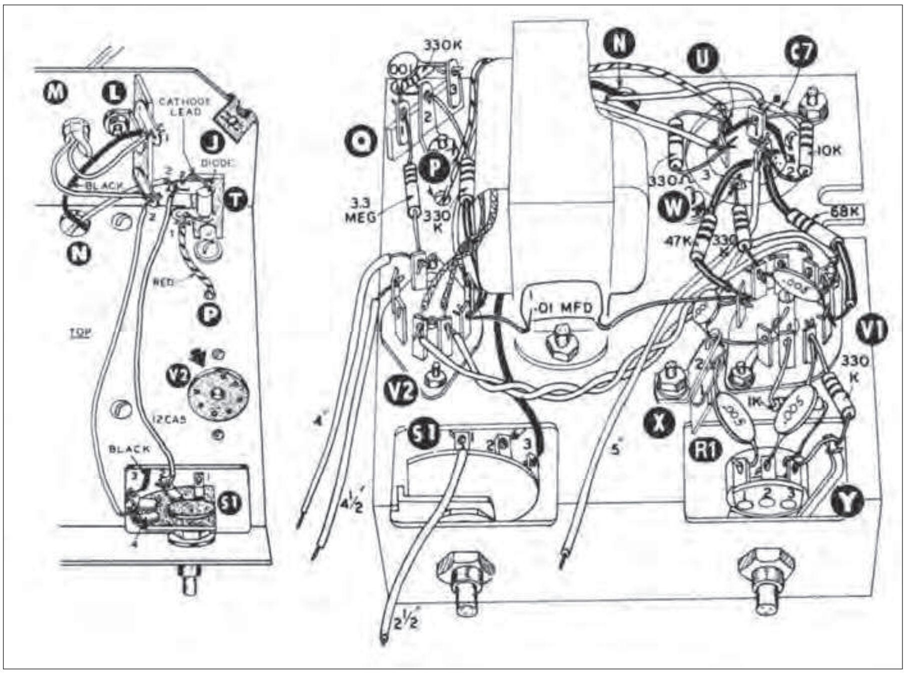


Fig.1. Old-school construction – (top) part of a power amplifier built without the benefit of a PCB (photo courtesy of Alan Winstanley); (bottom) an example of a 1962 home project – a Heathkit Visual-Aural Signal Tracer Model IT-12.

Early kits

Jut like today, early hobbyists were expected to assemble circuits themselves, and kits by companies like Heathkit (who formed in the years after the Second World War) were based on designs created from the surplus of ex-military components sold at low cost at the end of the war. The Heathkit assembly manuals from those days were works of art; their kits include products ranging from Morse code audio oscillators up to televisions and oscilloscopes. Even 30 years later in the 1970s with advanced production technologies available, radically new electronic products were still being offered in kit form – the Acorn System 1 computer being one example (the author still has his.)

The mechanical structure of electronic components has changed enormously over the years. Initially, circuits were based around glass tubes with pins designed for large sockets, or large component packages with long wires, these devices were ideally suited to circuit assembly that more reflected a mechanical assembly project than the designs we build today. Anyone who has seen a 1960s Heathkit assembly manual, like the one in Fig.1, will understand. And the tools we needed to assemble these projects were huge. Not to mention the need to wear a tie like the chap in Fig.2! (For younger readers, in those days it was not uncommon for manual workers to wear two or three-piece suits.)

Over the years components became smaller, wirewrap board assembly and eventually PCB manufacturing processes became available. The vital pivot to the consumer industry however was when surface-mount components and production techniques became available.

Automation

Circuits based on valve technology could be assembled only by hand. Even with today's technology, robotic point-to-point wiring would be a harder problem to automate than driving a car. The precision required to solder point-to-point wired circuits would be at the millimetre level in a three dimensional space. Automated cars are required to navigate roads at the centimetre level. And they are not able to do that reliably yet.

With the introduction of surface-mount components, component part sizes and even the reels that they are supplied on became standardised. That standardisation meant that the designers of machines that will hold, dispense, pick up and place components could be built to a common standard – a standard used by all component suppliers. This focused academic and industrial research into efficient tools to do the job, and with all manufacturers adopting those common standards, there was a business case to justify the R&D investment. If the market had been fragmented by a large number of proprietary component size and delivery packages, automation advances would simply not have been as financially viable, and technology improvements not as fast.

As the accuracy of automated pick-and-place machines has improved over the years, the size of components that can be automatically placed has been reduced, significantly. A 4.7kΩ 1/8-watt wire-ended resistor that we all know and love is not the size it is because it needs to be; as can be seen in Fig.3, it can be a lot smaller – and this is not the smallest size an automated pick and place machine can work with! The wire-ended component is this size because of the wires, nothing else. It's true that for specialised components, working at very high voltages or very high wattage rating, their sizes will be larger – but the vast majority of components placed on circuits for consumer products work at very low voltages and very low power ratings, and these components can be incredibly small, while still reliably placed by machines onto a printed circuit board (PCB).

Fig.2. Building a circuit 1965-style. Want to be on the cover – make sure you wear a tie! (For the benefit of younger and foreign readers, the 1965 price of '2'6' is 12.5p, or £0.125!)

The rise of SMD

The use of surface-mount components (SMD) also results in a significant reduction in the size of the PCB, with a corresponding reduction in weight (if that is important,) and cost, which is always important. With less weight and closer contact to a PCB, SMD components are less susceptible to mechanical stresses, assuming they have been soldered to the PCB correctly in the first place.

With smaller components the soldering process itself becomes a critical part of the overall production process. Academic and industrial research has focused on this for decades, covering:

- Component size standardisation
- Component solder pad design
- Solder mask
- Solder paste
- Component placement
- Soldering process

Let's cover each of those in turn. The first two points are basically the same; if component manufacturers agree on standard component sizes, engineers spend less time designing new footprints for the components they add to their PCBs. The resulting world-wide agreements has had a major impact beyond saving engineers a few hours effort. Optimisation of soldering processes and general reliability improvements has been far more effective and quickly adopted when all companies are able to focus on the same problem. Even minor improvements in the design of a solder pad shape, if it improves soldering reliability by a fraction of 1%, can have a huge impact when every company in the world is using the same design, and can make the same changes. This is the power of standardisation!

Solder mask

The solder mask is a solder-resisting film layer applied to the PCB during manufacture. It is automatically calculated

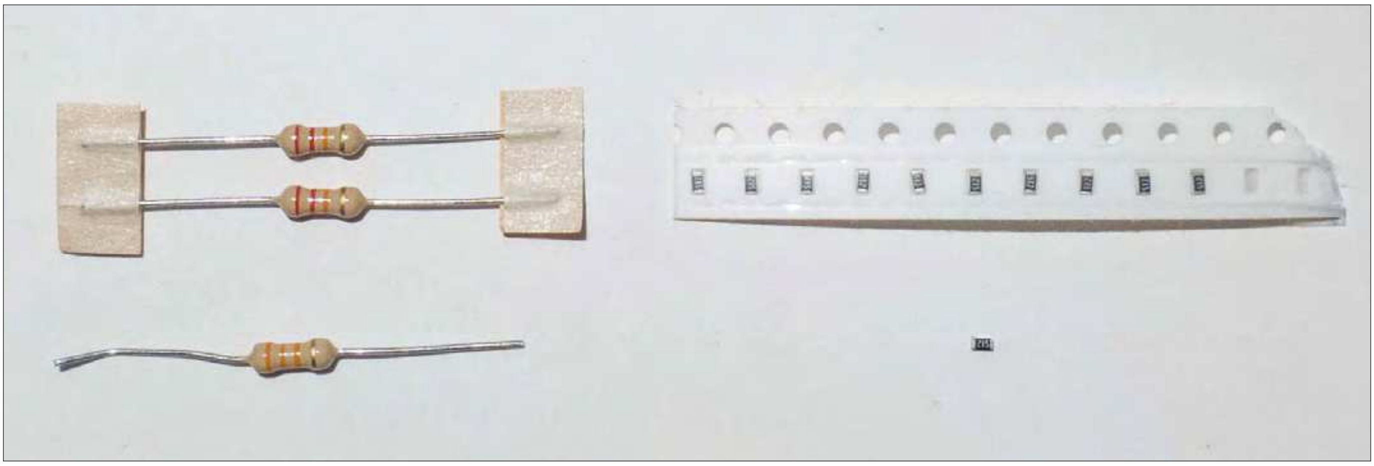


Fig.3. Wire-ended components verses their SMD equivalents: exactly the same device, very different packaging.

by your CAD program based on the standard component outlines that you place on your PCB; it's not something you have to create yourself. The solder mask is a coloured layer (you get to choose the colour) that both protects the copper traces from corrosion and also acts as a repellent to solder – forcing any solder that wants to 'creep' between two tracks away, back to the pad they are supposed to melt on. This is hugely important as component solder pads get closer together.

Solder paste

The solder paste layer is again a layer that is created automatically for you based on the components you place. It defines the areas on your PCB that should be left exposed for the application of a thin layer of solder paste.

While it is possible to add solder paste to PCBs by hand, it is of course far more preferable to be able to apply the paste automatically, or at least by a more efficient process. The solder paste layer is a collection of polygons that define where the solder paste should be deposited; PCB manufacturers take this layer and laser cut a sheet of thin steel with these holes. Then, using the age-old process of screen printing, the thin sheet of steel is placed over the PCB and a squeegee is used – by a robot or manually – to apply a thin coat of solder paste to the PCB. Ink-jet-style printers are now available to do this task, but manual application is still common.

Component placing

Next, components need to be added to the PCB using very fast pick-and-place machines. I could write a whole article on these fascinating robots, which can place parts from reels or 'sticks' of parts at up to 100,000 parts per hour and to an accuracy of within 25µm. They typically represent 50% of the outlay for a complete production line. For the hobbyist side of things they come free – it's you!

Soldering process

The soldering process itself has many critical parameters. The solder paste, consisting of tiny balls of solder in a flux paste, must be heated to a point where the flux melts, time allowed for the flux to clean the surfaces, then the temperature is raised again to melt the solder, and time allowed for the solder to flow over the component and the PCB pad. This process depends on many factors, including the size of the component, it's shape, and how much of a 'heat sink' effect the PCB traces connecting to the component have. These problems have been studied for decades, and the results shared, driving subtle changes to the temperatures used, the cycle times and the specific layout of ground pads on components. It's wonderful that this knowledge has been shared rather than tied down in patents and copyrights.

While this all sounds very complicated, the process is actually very easy to use at home. With low-cost PCB manufacture available in Asia, it's possible to order 10 complex double-sided PCBs with a laser-cut solder paste stencil for less than £30 (US$40). Solder paste can be purchased in small quantities from the usual electronics distributors, and while it has a limited shelf life, for hobbyists a small 30g tub of paste can be stored in the fridge for one to two years – if your partner will tolerate it!

The re-flow oven used by electronics assembly companies can be replaced by a simple desktop convection oven, costing around £40 (US$52). A re-flow timer controller can be added for about $50 (US$65), less if you build one yourself. These are simple devices that will last for decades and are worth the investment if you expect to be making many boards, or would simply like to make assembling surface-mount PCB easier.

This approach to SMD component assembly is well suited to the hobbyist, but if you are looking to expand, perhaps manufacture hundreds of PCBs, then there are better alternatives to simply getting a professional manufacturer to assemble you boards. Automated 'Pick and Place' machines have been created by amateurs and small companies that can place components on PCBs at high rates, yet the equipment is low cost – around €1700 for a machine equipped with an AI-based vision system to accurately place components (see: **www.liteplacer.com**). These kind of machines are only suited to very low production volumes, but they give the benefit of enabling you to quickly produce a large number of PCBs, albeit with a fair amount of post-production 'touching up' (error correction) required. They are desktop machines, and significantly cheaper than the £100k machines used by regular PCB assembly companies.

The development and improvement of these machines has been driven by the open-source community, a group of people keen to expand their experience while sharing it with others. It's through this community that small cottage industry companies have been able to create useful, technologically advanced machines that can help budding entrepreneurs grow their businesses.

Practical Electronics magazine has been in print for 56 years. Ownership may have changed; staff have left or passed on. Over that time the magazine has carried the electronics hobbyist tradition across the era of valves/tubes wired to Paxalon, point-to-point wired components through to robot-assembled surface-mount components measuring just a few millimetres across. It makes us wonder what (and how) hobbyists 50 years hence will be assembling!

Summary

Next time in *Practically Speaking* we will look at passive surface-mount parts, including sockets and switches, and the challenges involved in choosing and placing them on a PCB, both automatically and by hand.

Make it with Micromite

Phil Boyce – hands on with the mighty PIC-powered, BASIC microcontroller

Part 15: Having fun with a colour touchscreen

Fig.1. The 2.8-inch touchscreen used this month lets you create impressive interactive user interfaces.

In a previous column we showed how to connect a tiny 0.96-inch colour IPS screen to the Micromite (see *Part 8* in *PE*, September 2019). We used it to build a clock, and also to demonstrate some of MMBASIC's graphical commands (*Part 9*). With a pixel resolution of 160×80, the IPS screen is the perfect add-on for any project that needs to display simple information such as the time, date, or temperature. However, its small physical size does mean that it is rather limited in what can be displayed at any one time. So this month we are going to show you how to implement a bigger TFT (LCD) screen that not only has a higher pixel resolution (320×240), it also has the option of a resistive touch-panel. This bigger screen will allow you to display much more information, and thereby enable you to create some interesting and useful user interfaces (see Fig.1).

We start this month by building the required TFT adaptor. This comprises just four connectors, some wire links, a resistor, a buzzer (optional), and four support pillars (see Fig.2). You'll then insert an ILI9341 2.8-inch TFT display module (with, or without, touch). Once assembled, you can plug the TFT-adaptor directly into your *MKC*, or better still, plug it into the Bluetooth adaptor board that we built last month. Once a couple of options have been set, and the touchscreen has been quickly calibrated, it will all be ready for use. We will work through several user interface examples so that you can be inspired to develop some ideas for your own projects.

We will end this month by downloading the classic board game, *MasterMind*. It's a great demonstration of how you can use this bigger screen, although we won't use the touch capability in the game. Instead, in the version presented here you will play against the Micromite with an infrared (IR) remote control. The aim of the game is to use logic and skill to solve the secret colour-code as quickly as possible, in no more than ten guesses. If you've never played *MasterMind* before, you will find it a fun, challenging and addictive game!

Micromite code

The code in this article is available for download from the *PE* website.

Building the TFT adaptor

The TFT adaptor is a straightforward circuit comprising just a handful of components assembled onto stripboard; see the schematic in Fig.3. Three pin-strips (J1-J3) allow the adaptor to be plugged into the *MKC* and hence connect to the relevant pins on the Micromite. These also route the *MKC's* 5V power-output pins directly to the TFT screen, which is plugged into the 14-way socket (J4).

Built into the TFT screen itself is an LED backlight which ultimately controls the overall brightness of the screen. In this design, the screen's brightness is fixed by the 10Ω resistor (R1) to provide a decent brightness level. The optional buzzer (LS1) can be added for audible feedback – it can be used as a 'click' effect when the screen is 'touched', or alternatively as a piezo buzzer to make sounds at various frequencies. One side of the buzzer is connected to a PWM output (pin 26), and the other to a general purpose I/O (pin 22) which effectively acts as an enable pin (refer back to *Part 7* in *PE*, August 2019, for details of this technique).

The stripboard layout is shown in Fig.5. Even though there are only a few items to assemble, the stripboard is large enough to securely mount the TFT using four

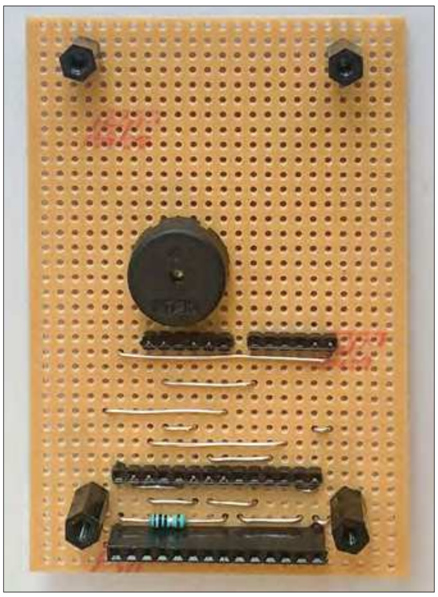


Fig.2. The TFT adaptor used to connect the touchscreen to the MKC.

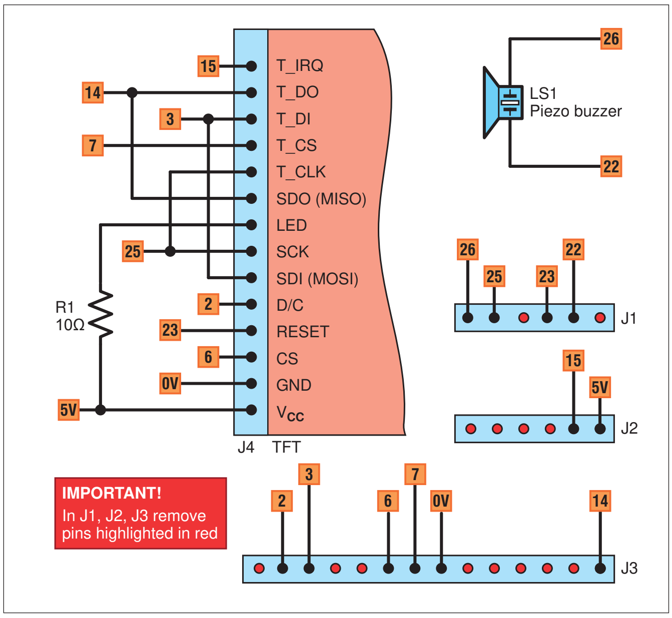


Fig.3. Schematic for the TFT adaptor – the only components required are a single resistor and an optional buzzer.

nylon mounting pillars (M3, 12mm). The positions marked for the mounting pillars are for use with the 2.8-inch version of the screen. You could use the smaller 2.4-inch version, but you will need to relocate the two pillars shown at holes C4 and C21 (and omit the pillars in holes GG4 and GG21).

Begin by cutting out the required size stripboard (24 tracks by 37 holes). Mark the 16 required track cuts, check them (at least twice!), and then manually cut them with a track-cutting tool, or a drill bit. Mark out the mounting pillar holes and use a 3mm drill bit to make the holes.

Next, install the 15 wire links, followed by resistor R1. It cannot be stressed enough that you check the positioning of items at least twice before soldering them.

J1-J3 need to be prepared prior to installation – see Fig.4c. To do this, remove all unused pins and modify the remainder, which are the downward-facing pins that plug into the *MKC*.

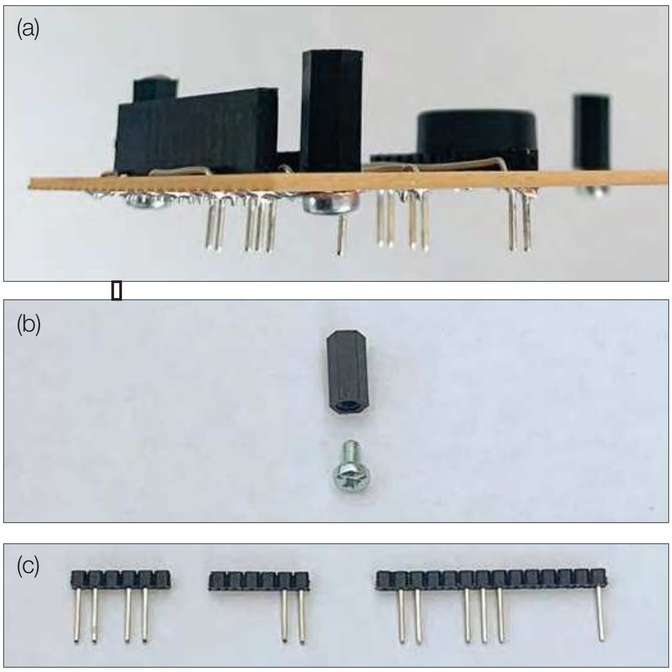


Fig.4. (a) one of the four mounting pillars used to support the TFT module; (b) nylon threaded stand off (12mm, M3) with 6mm M3 screw; and (c) the three modified jumpers that connect the TFT adaptor to the *MKC*.

Next, mount the 14-way socket (J4), and finally solder the buzzer (LS1) into position. Perform the usual visual checks for shorts between neighbouring tracks, and when you're happy everything looks correct, use four 6mm M3 screws to fix the mounting pillars leaving the top hole 'free' in each pillar – see Fig.4a.

Next, carefully insert the TFT screen into the 14-way socket – ensure you align the pins correctly. For now, do not screw the screen into position, just let it rest on the four mounting pillars. Finally, you need to carefully insert the TFT adaptor into your *MKC* (or Bluetooth adaptor and *MKC*).

Configuring and testing

Before the screen will function properly, there are two configuration options that need to be set. However, you may

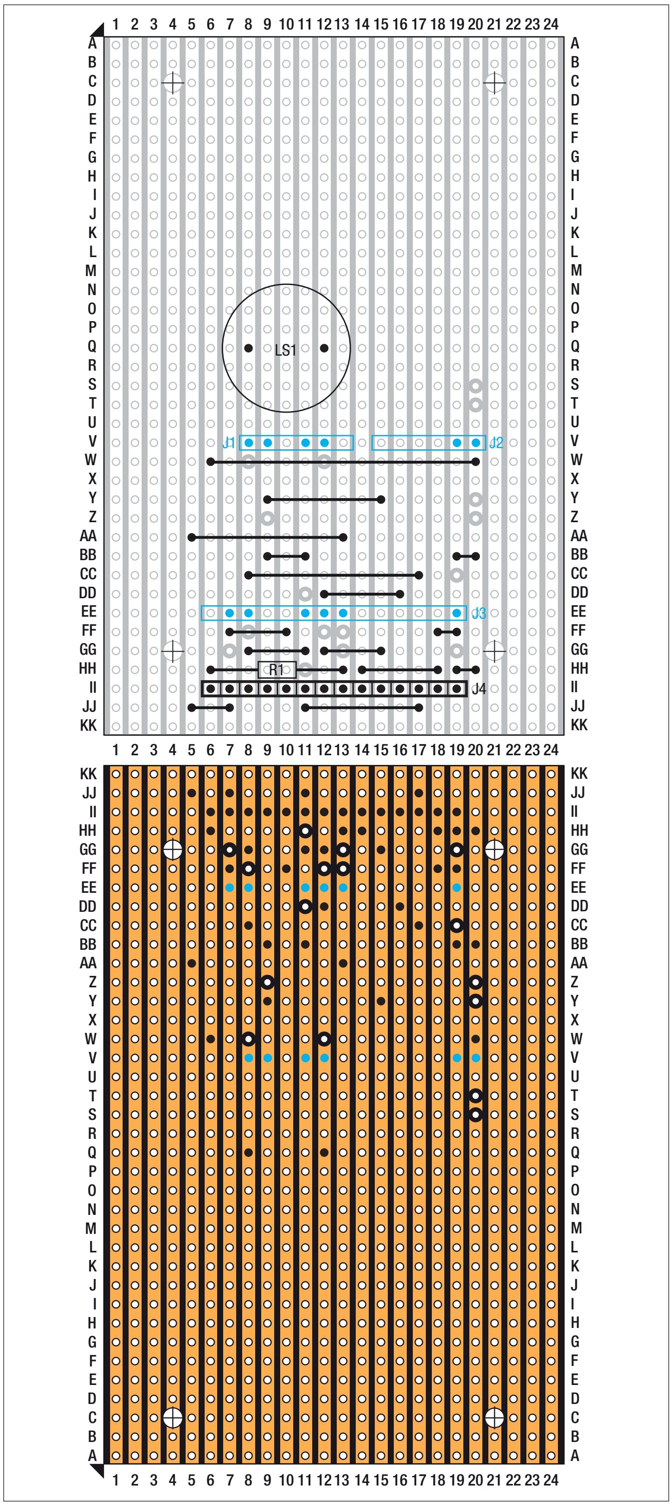


Fig.5. Stripboard layout of the parts used for the TFT adaptor. Note: blue means downward-facing pins – see Fig.4.

Fig.6. Typing the command `GUI TEST LCDPANEL` should result in the display of this animated test-pattern.

Fig.7. The touchscreen is calibrated with the command `GUI CALIBRATE`. Use a stylus (eg, a pen lid) and follow the on-screen instructions; ensure you accurately touch the centre of each 'target'.

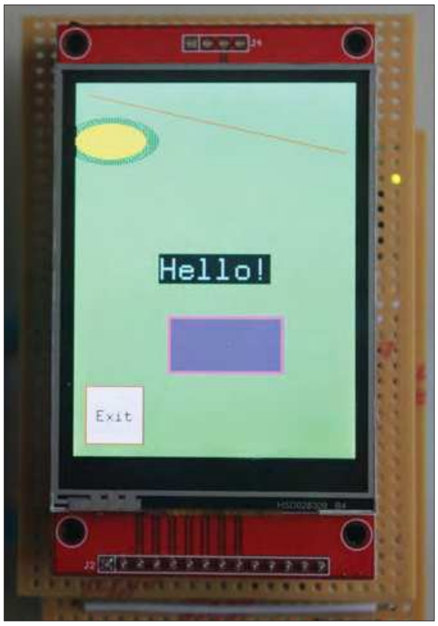


Fig.9. Check that the basic graphical commands (as listed in the text) display on the TFT screen as expected. Try changing the parameters to experiment with the appearance.

still have the IPS screen driver installed, in which case it will need to be removed first (remember that the Micromite can have no more than one external TFT display attached). Now connect your *MKC*/TFT to your computer either via the *DM*/USB lead, or via the Bluetooth link. Launch your Terminal application (eg, TeraTerm), and connect to the correct COM port and at the correct speed. If all is well, you'll see the usual Micromite welcome message (or see your auto-running program, in which case simply use `Ctrl-C` to stop it). If you do not see the command prompt, then unplug everything from your computer, and check that the adaptor boards are indeed inserted correctly.

Once you see the command prompt, you are in a position to remove the IPS driver. Simply type `OPTION LCDPANEL DISABLE` followed by `LIBRARY DELETE` (you can do this even if you're not sure you have the IPS driver installed). It is then advisable to reset the *MKC* before moving on (remove power, then reapply after a couple of seconds).

You will probably see the TFT's LED backlight on (with the screen looking like it has a whitish/grey appearance). If not, then it is worth checking that 5V is getting to the TFT's LED pin. When you can see the LED backlight on, you can proceed with setting the Micromite to use the built-in screen

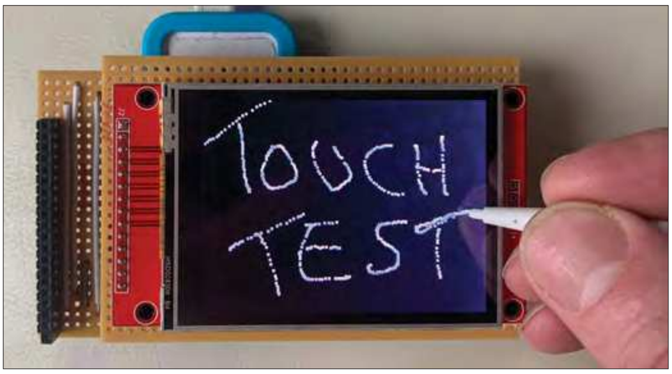


Fig.8. Use the command `GUI TEST TOUCH` to test that calibration has been performed accurately. A good test is writing a few simple words (slowly, and with the aid of a stylus).

driver (which this particular screen uses), along with setting up the 'Touch' feature (if you have the touch-panel version of the screen).

First, to set the Micromite to use the built-in screen driver, type:

```
OPTION LCDPANEL ILI9341,P,2,23,6
```

On pressing `Enter`, the TFT should go blank (it no longer looks whitish/grey). In the above configuration setting, the `P` sets portrait mode (as opposed to landscape), and the numbers `2,23,6` represent the three Micromite pin numbers used to connect to control signals on the TFT module.

Now type `GUI TEST LCDPANEL` and you'll see the usual built-in animated test pattern of circles, as shown in Fig.6. If you see this, then congratulations, you're almost done. If not, check for correct insertion of the adaptor(s), and that there are no shorts between tracks on the TFT adaptor. If you're still not seeing anything, then type `OPTION LIST` to check that the pin numbers shown for `OPTION LCDPANEL` are correct. If not, then type `OPTION LCDPANEL DISABLE` to remove it, and then repeat the above `OPTION` again, this time with the correct pin numbers.

For those that have the touch-panel feature included with the TFT module, you will now need to set it up by typing:

```
OPTION TOUCH 7,15
```

This tells the Micromite which pins are connect to the touch-panel control signals. You can check that you entered it correctly by immediately typing `OPTION LIST`. If you make a error, type `OPTION TOUCH DISABLE` before correcting any typing mistake.

Before you can use touch, you have to perform a calibration. To do this, type: `GUI CALIBRATE` and follow the instructions that appear on the TFT screen. You will see four cross-hairs appear; one at a time (see Fig.7). Using a blunt, pointed implement (plastic stylus), touch over the central point of each 'target', and press down until the target disappears. Be careful not to touch any other parts of the touch-screen when touching the cross-hair. After the fourth 'target', you will see a message (in your terminal app) stating whether or not calibration was successful. If you see, `Done. No errors` then all is done. If you see, `Warning. Inaccurate calibration` then repeat `GUI CALIBRATE`, but this time take extra care to touch the centre of each target.

The next step is to test that the touch-panel works accurately – this is vital, especially when it comes to any touch interface that relies on a user touching 'buttons' placed on the TFT screen. To test the accuracy of the touch-panel simply type: `GUI TEST TOUCH` and the screen will turn blank. Now, use a stylus to draw on the screen and check that the pixels appear at the point of contact with the screen. A good test is to slowly write some words and check that they are recognisable (see Fig.8). Press any key to end the test. If there is 'drift' between where the stylus is touching the screen, and where the pixels illuminate, then re-run the calibration process.

The final thing to test is the buzzer, which is connected between pins 22 and 26. To do this test we will set the PWM output on Pin 26 as a 1kHz square wave, and set pin 22 as a low output to act as 0V. Now type two lines of code:

```
PWM 2,1000,50
SETPIN 22,DOUT
```

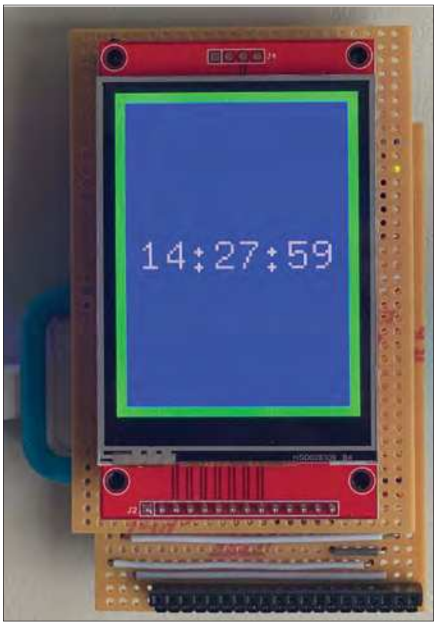


Fig.10. The result of the eleven-line 'clock' program listed in the text. If you have installed the optional buzzer then you will also hear a ticking sound.

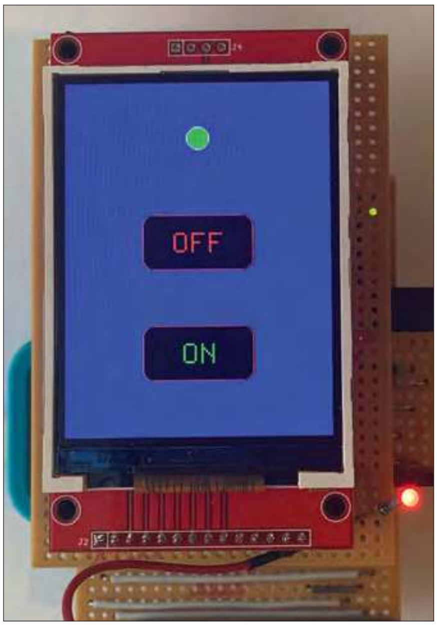


Fig.11. Using two touch-buttons to control external hardware. Here a physical (red) LED on I/O pin 17 is being controlled, along with a virtual (green) LED on the screen.

On pressing `Enter` after the second line, you should hear the piezo buzzer sound. If not, then you will need to check the soldering between the piezo buzzer and pins 22 and 26. To stop the buzzer sounding type `SETPIN 22,OFF` (or `PWM 2, STOP` or `EDIT`).

Graphical commands

Now that you have built and successfully tested the touchscreen adaptor, we can start to use some of MMBASICs graphical commands with this bigger screen. Back in *Part 9* (*PE*, October 2019), we used some of the graphical commands on the IPS screen. All of these are still valid, but we now have more pixels to play with. Let's begin by seeing those previous commands in action. Enter the following, one line at a time, and check that you end up with a screen like that shown in Fig.9:

```
CLS RGB(red)
CLS RGB(0,255,128)
TEXT 120,160,"Hello!",cm,1,2,rgb(green)
BOX 80,200,100,50,3,rgb(magenta),rgb(blue)
LINE 10,10,230,60,1,RGB(red)
CIRCLE 30,50,20,5,2,RGB(black),RGB(yellow)
RBOX 10,260,50,50,1,RGB(red),RGB(white)
text 35,285,"Exit",cm,1,1,rgb(black),rgb(white)
PIXEL 130,225,RGB(black)
```

Now try adjusting some of the parameters to ensure that you understand how to use each command. It is worth referring to the *Micromite User Manual* for full parameter details; also, work through Chapter 8 from the *Getting Started with the Micromite* manual.

The above list shows just some of the building-block examples available for displaying information on the screen; overall, Fig.9 is not a very useful end result, so now let's enter the following short program instead:

```
PWM 2,2000,50
BOX 0,0,240,320,10,RGB(GREEN),RGB(BLUE)
DO
  TEXT 120,160,TIME$,CM,1,3,RGB(WHITE),RGB(BLUE)
  IF oTime$<>TIME$ THEN
    oTime$=Time$
    SETPIN(22),DOUT
    PAUSE 10
    SETPIN(22),OFF
  END IF
LOOP
```

On running the code, you should see the time displayed on the screen in the format shown in Fig.10, along with an audible 'tick' each second. If you don't see or hear this, then check the code has been entered correctly. Note, if you do not have an RTC connected, then you will first need to set the 'clock' by typing the command: `TIME$="hh:mm:ss"` (replacing the relevant values for `hh` (hours), `mm` (minutes), and `ss` (seconds); eg, `TIME$="14:27:00"`

As a challenge, try converting the IPS Clock code (*Part 9*) to display the date on this bigger screen.

Using touch

Having just had a brief look at how to use the touchscreen for visual output, let's now explore how to use the touch capability. As usual, MMBASIC makes this easy. We will keep things simple by working step-by-step through the following 'how to' topics:

- Read the *x* and *y* coordinates of the screen position being touched
- Trigger an interrupt on touch
- Create a touch button
- Respond to a touch button

These ideas will provide building blocks to enable us to put the touchscreen to good use. We will be working towards a short program that demonstrates how to control external hardware via the touch-screen. In our example, we will be controlling an LED (connected to an I/O pin) switching it on or off via two touch-buttons drawn on the screen (see Fig.11). Let's begin.

Reading touch position

Two built-in system variables, `Touch(x)` and `Touch(y)`, contain the screen coordinates of the current position being touched on the screen. The coordinate values returned have a direct correlation to the pixel position; the *x* coordinate responding to the horizontal (left/right) location, and the *y* coordinate for the vertical (up/down) location. If the screen is not being touched then both return a value of `-1`. Note that only a single touch-point can be read; so if you are touching more than one point on the screen, then the 'averaged' coordinate values are returned (which in most cases is meaningless). That said, a single point of touch is ideal for most situations, such as selecting something on the screen.

To see the `Touch(x)` and `Touch(y)` system variables in action, type in, and `RUN` the following five-line program:

```
CLS
DO
  TEXT 10,10,"X="+STR$(Touch(X))+"     ",lt,1,3
  TEXT 10,60,"Y="+STR$(Touch(Y))+"     ",lt,1,3
LOOP
```

The top-left corner represents coordinate $x = 0$, $y = 0$ (and the bottom-right corner is 240,320 – this assumes that the orientation setting in `OPTION LCDPANEL` was set to `P` for portrait). However, you probably won't see these exact numbers returned when touching the relevant corners as it will ultimately depend on how accurately the screen was calibrated (ie, how accurately you touched the 'targets' after `GUI CALIBRATE`). Don't panic if you don't see values of `0,0` and `240,320` returned in diagonally opposite corners; but do ensure that you see something close to them (otherwise the touch feature won't be as accurate as it could be). If necessary, re-calibrate with `GUI CALIBRATE` and try to be as precise as you can be when touching each target centre point.

To make the program more interesting, insert the following single line of code (immediately after the second `TEXT` command, and before the `LOOP`):

```
PIXEL Touch(x),Touch(y)
```

Using a Touch interrupt

MMBASIC makes it very easy to set up an interrupt that is triggered whenever the screen is touched. In fact, it is better to think of the interrupt as being configurable to trigger only when the screen is 'just touched', or only when 'just released' (or to trigger on either of these). To see this in action, start a new program (ie, type `NEW` at the command prompt), then enter and `RUN` the following program (its operation will be explained shortly):

```
SETPIN 15,INTB, TouchInt
DO
  CLS RGB(BLUE)
LOOP
SUB TouchInt
  CLS RGB(RED)
END SUB
```

On running the program you will see the screen turn blue; and whenever you touch the screen, or release your touch, the screen flashes red briefly (indicating the interrupt being triggered). So how does it work? There are three distinct parts to this program.

The first line simply sets up the interrupt by detecting a change of state on pin 15 (the `INTB` parameter). If you refer to the schematic in Fig.4, you will see that I/O pin 15 connects to the TFT's touch-interrupt output pin from the TFT module (and we referred to this pin as the *touch-interrupt* parameter when setting `OPTION TOUCH`). We are simply saying to jump to an interrupt subroutine (that we have named `TouchInt`) whenever there is a change of logic state from the TFT touch-interrupt pin – in other words, whenever the screen is 'just touched' or 'just released' (liken it to 'pen down' and 'pen up' detection if that helps).

The main-program is just three lines long, comprising a simple `DO`/`LOOP` that continually turns the screen blue. Hence, the screen remains blue until the interrupt subroutine is called.

Looking at the single line of code in the interrupt subroutine, you can see that all it does is change the screen colour to red (but the main program immediately then sets it back to blue – resulting in the brief red flash whenever the interrupt is triggered).

Now change the `SETPIN` line of code by altering the `INTB` parameter to `INTH`, and then to `INTL`. Observe the effect it has on the triggering of the interrupt: either a low-to-high change with `INTH` (ie, release only) or a high-to-low change with `INTL` (ie, touch only).

One example of how to use this touch interrupt in a practical situation is to trigger an emergency stop of some external hardware; eg, a running motor.

Creating a touch-button

We will now write a short program to draw a button on the screen. To do this we will use the graphical command `RBOX` to draw the button, and the `TEXT` command to give it a label. Now `RUN` the following program (again, start a new program):

```
CLS RGB(blue)
RBOX 70,120,100,50, ,RGB(red),RGB(black)
TEXT 120,145,"OFF",cm,1,2,RGB(red)
```

The `RBOX` command draws a button that is 100-pixels wide, by 50-pixels high; and it is drawn at the location with coordinates $x = 70$ and $y = 120$ (this is where the top-left of the button is located). The `TEXT` command then adds the '`OFF`' label centre justified at the central position of the button. The *x* coordinate for the `TEXT` command is calculated as 70 pixels along (ie, where the left side of the button is) *plus* half the width of the button (ie, 100/2 = 50) giving a value of $x = 120$ as the centre of the button. The *y* coordinate is from the top of the button ($y = 120$ from the `RBOX` command) *plus* half the height of the button (ie, 50/2 = 25) giving a value of 145. Hence $x = 120$ and $y = 145$ for the centre of the button – and using `CM` justification in the `TEXT` command means that the label will be positioned exactly in the centre of the button. If you didn't quite follow this, just `RUN` the program and check that you see an `OFF` button near the middle of the screen.

Next, let's add a second (`ON`) button by adding these two lines of code:

```
RBOX 70,220,100,50, ,RGB(red),RGB(black)
TEXT 120,245,"ON",cm,1,2,RGB(green)
```

We will be using these buttons to show you how to control an external LED connected to I/O pin 17. In addition, we will mimic the LED status on the touchscreen. To do this, we will use the `CIRCLE` command and fill it with either black (to indicate the LED is off), or green to indicate that it is on. The fill-colour of the LED (ie, the `CIRCLE`'s fill-colour parameter) will be stored in a variable that we call `FCol`. To draw the LED on the screen, insert these two lines at the very start of your program:

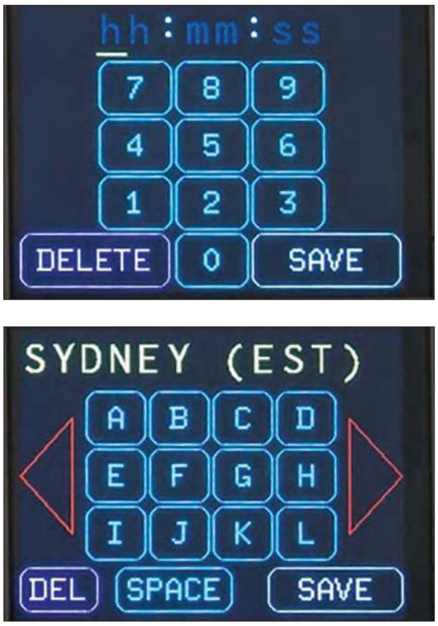


Fig.12. Example screen-shots of numerical and alphabetical touch-keypads.

Fig.13. On running MasterMind you will see this screen (it is animated!)

Fig.14. Black and white scoring pegs provide feedback to each guess you make.

```
DIM FCol as INTEGER
FCol = RGB(black)
```

and add this `DO/LOOP` at the very end:

```
DO
  CIRCLE 120,50,10,1,1,RGB(white),FCol
LOOP
```

Having typed in the above code, `RUN` the program to see the end result. Note that neither the buttons, nor the LED, will actually do anything at the moment (that comes next) – we are simply creating two buttons and an LED on the screen ready for the next stage. Check that on running the code you see something similar to that shown in Fig.11.

Responding to a touch button

Now that we have the required elements drawn on the screen, we need to add the code that responds to a press of either touch button. To do this, we use the 'how-to' techniques learnt above. In essence, we use an interrupt that is triggered whenever the screen is touched, and within the interrupt subroutine we check the values of `Touch(x)` and `Touch(y)` to see if they fall within the position of either button. If so, then we will set the state of the LED (both the on-screen virtual LED, and the external physical LED) and switch it either off or on as required.

To achieve all of this we add two lines of code at the start (to set up the touch-interrupt on pin 15, and to configure an I/O pin for the external LED), and then add the touch-interrupt subroutine code (to test if a button is pressed). The complete listing is shown here – the non-bold code is your current program. Just add the bold code (and check each line shown below does indeed match your code).

```
SETPIN 15,INTL,TouchInt
SETPIN 17,DOUT)
DIM FCol as INTEGER
FCol = RGB(black)
CLS RGB(blue)
RBOX 70,120,100,50, ,RGB(red),RGB(black)
TEXT 120,145,"OFF",cm,1,2,RGB(red)
RBOX 70,220,100,50, ,RGB(red),RGB(black)
TEXT 120,245,"ON",cm,1,2,RGB(green)
DO
  CIRCLE 120,50,10,1,1,RGB(white),FCol
LOOP

SUB TouchInt
  xx=Touch(x)
  yy=Touch(y)
  IF (xx>70) AND (xx<170) THEN
    IF (yy>120) AND (yy<170) THEN
      FCol=RGB(black)
      PIN(17)=0
    ELSE IF (yy>220) AND (yy<270) THEN
      FCol=RGB(green)
      PIN(17)=1
    END IF
  END IF
END SUB
```

Once your code matches the above, connect an LED (and current-limiting resistor) to I/O pin 17. Then RUN the program and check you can control the LED via the two touch buttons. Ensure that both the physical LED and the virtual LED change state correctly. If not, then by now you should be confident enough to check things over – start with the code, and then check physical connections. Do ensure that you understand each line of code 100% as you will then find it very easy to create your own touch-screen GUI to control external hardware.

Keypads

From the above example that demonstrates how to implement a couple of buttons, the next logical step is to create a 'keypad' that allows for data to be inputted (such as a numeric value). To do this, you essentially create several buttons arranged appropriately on the display with relevant text labels. Two examples of such keypads are shown in Fig.12. Rather than use valuable space here repeating the code from the manuals, please refer to Chapter 8 from the *Getting Started with the Micromite* manual. You will find the code for a numeric 'data entry' keypad on pages 83-85. This is an excellent implementation, so do be sure to take the time and try it out.

Some example GUIs

Some of you may prefer to learn from other people's code as opposed to writing it from scratch. If you are one of these

Fig.15. The *BackPack* PCB is a low-cost, yet extremely useful module to explore this month's topics.

(and even if you're not), it is well worth looking through some of the amazing programs that have been written by Geoff Graham, the creator of the Micromite. On his website (**geoffg.net**), you will see various projects that he has created using the 2.8-inch touchscreen to great effect. All his code is available to download for free from his website. I strongly recommend you take the time to explore some, if not all, of the following projects (you will see them listed in the top left corner of his home page):

- Air Quality Monitor
- DDS Signal Generator
- Super Clock
- Boat Computer
- Parking Assistant.

Look at how Geoff uses the graphical commands, as well as how he uses 'touch' in the various user interfaces. Do feel free to change his code so that you can gain a better understanding of just how easy it is to use the touch-screen in your own projects.

The Micromite BackPack

For those of you who want to use a 2.8-inch touchscreen in your own dedicated Micromite project, or who just want to explore MMBASIC graphics and touch features in more detail, I recommend the *Micromite BackPack* as a very useful module. In fact, you will see that the above-mentioned projects on Geoff's website are all based around the *Micromite BackPack* (it has also been used for many projects featured in *PE*). Better still, a full description of building one has been published in *PE*. See May 2017, then updated in May 2018.

You can liken version 2 of the *BackPack* as an *MKC* and a *DM* on one compact PCB, with a 14-way socket into which you directly plug the TFT (see Fig.15). This low-cost PCB means that you are able to keep your *MKC* and *DM* dedicated to development purposes. The *BackPack* PCB mostly uses through-hole components – apart from the USB socket and two transistors. These transistors allow for software control of the backlight brightness (via PWM, just as we did with the IPS display). These three SMD parts are supplied pre-soldered, meaning there is just a handful of through-hole parts required to complete the module: six capacitors, four resistors, three connectors, an LED, a voltage regulator and a button (plus the Micromite PIC, the *MicroBridge* PIC, and the TFT). Think of the *BackPack* as an *MKC*/*DM*/touch-TFT ready to run any project you develop. Readers of *PE* can email me directly for a crazy offer of this fantastic PCB!

Questions? Please email Phil at:
contactus@micromite.org

MasterMind

This month's download demonstrates a practical (and fun) use for the screen in the form of the famous game, *MasterMind*. It brings together several elements that have been covered in this series, including IR remote control, sound and graphics. We are not going to go into an explanation of the code as this is something you can do as a personal exercise. However, for those of you unfamiliar with the game, we will explain the rules enough for you to have fun playing this addictive game.

Put simply, you have to solve the secret colour code as quickly as possible (see Fig.13). The game starts with the Micromite choosing four random coloured 'pegs' and sequencing them in a row – this forms the secret colour code. Each peg can be any one of six colours, and the pegs can have colours repeated. Not only do you need to identify the colour of the four pegs, but also the sequence of the pegs has to be solved. After each 'guess', the Micromite will score you in the form of black and white scoring-pegs. A black scoring peg indicates that you have the correct colour peg in the correct position, and a white scoring peg indicates a correct colour but in the wrong position; so ultimately you're seeking four black scoring pegs! The skill comes in using the scoring pegs to deduce what the secret code is; and you have no more than ten attempts at guessing the secret code correctly.

To give some examples regarding scoring pegs, refer to Fig.14. This shows an actual game played (and won) in which the secret code was red, blue, yellow, red. The first attempt (numbered 1 at the bottom of the screen) shows a guess of cyan, green, red, green. This resulted in a score of just one white peg for the red guess in position 3 (ie, correct colour, wrong position). Likewise, attempt 2 shows a guess of yellow, blue, yellow, blue and scored two black pegs for the correctly guessed coloured pegs in positions 2 and 3. However, during game play, we do not know which pegs in the guess the scoring peg(s) refer to – that is where your skill and powers of deduction come into play. Work through the remaining attempts in Fig.14 to ensure that you understand the scoring pegs for the various guesses entered. You can see that the correct code was entered in attempt 9 (resulting in four black pegs). Note that in attempt 8 we had a total of four scoring pegs meaning that at this point we knew the colours of all four pegs in the secret code – they just needed to be put in the correct order.

A few useful points to bear in mind:

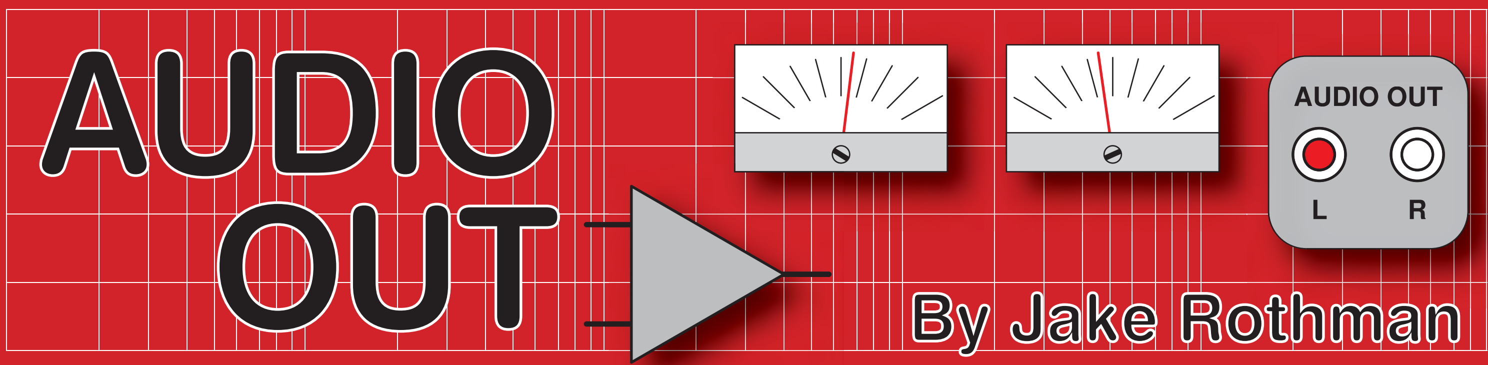
- The quantity of scoring pegs can be anything from none to four
- The position of the scoring pegs has no relevance; they are simply displayed from left to right, with any black peg(s) first, followed by any white peg(s)
- A response of no scoring pegs means that none of the colours in the guess are in the secret code
- A guess can comprise of up to four pegs; hence you can guess a single peg if you want to find out if a particular colour peg is in the secret code.

Download the file **MasterMind.txt** from the April 2020 page of the *PE* website and install it onto your *MKC*. The program code is commented throughout and finished to a level that allows you to enjoy playing the game. However, why not try adding your own features! I hope you enjoy it.

Next month

Over the last year and a bit we have covered numerous topics in the *Make It With Micromite* series. We believe it is now time to put many of the things that we have learnt into a fantastically fun project. So what is this project? On several occasions we have mentioned building a robot buggy, so next month we will start this journey by assembling the basic system. The robot will be built on a Zumo chassis (see Fig.16), it will be operated remotely via Bluetooth, and will have a personality that will be developed over several months. You can either follow step-by-step, or alternatively customise the options for your own personal use – after all, you now have the skills to easily control hardware by using the power and simplicity of MMBASIC.

Fig.16. The *Micromite Robot Buggy* featured next month is based on the excellent Zumo chassis.



Noise about noise – a new slant on analogue noise with tilt control – Part 1

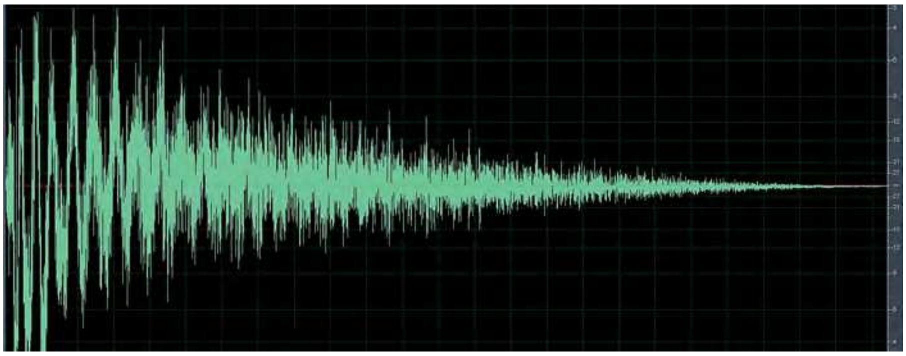


Fig.1. Roland TR909 Snare drum hit – note the large amount of noise in the signal.

Fig.2. Any noisy devices I come across are hoarded in a special drawer for synthesiser noise sources, including some 1965 AF114 germanium transistors that are so noisy they may have radioactive isotopes in their crystal structure!

I've often described my job as either eliminating distortion or generating it. It's the same with electronic noise. However, this time, instead of minimising noise in a pre-amplifier, we'll work on noise generation for audio synthesis.

Noise is an essential component of synthesis, especially for voice, snare drums and cymbals. Fig.1 shows a typical snare hit on a storage scope. The little module in this project can be an ideal addition to the battery-powered analogue synth (see the *Audio Out* series covered in the August 2018 issue of *Practical Electronics* (actually *EPE* back then)). Its tilt tone control circuit is also useful for Hi-Fi in its own right.

Random fluctuations

Normal electronic noise is truly random, caused by many effects, the main one being thermal agitation. For musical use we're much more interested in more dramatic noise produced by crystal lattice defects in semiconductor crystals and chip surface irregularities. Contaminants and reverse bias leakage currents are also significant. Manufactures do their best to avoid making noisy devices; better, more consistent manufacturing helps to produce less noisy devices, but they can't eliminate these effects completely and some noise is inevitable. Most sensible engineers and builders discard noisy devices, but synth builders hoard them – see Fig.2 for my not-so-secret cache of noisy bits and pieces in their own drawer.

The famous Roland TR-808 drum machine used specially selected 'defective' 2SC828 transistors, which were marked with a spot of pink nail-varnish! When the supply of those noisy transistors dried up Roland ceased production – a decision that was doubtless reinforced by a general prejudice against analogue at the time.

In an interesting about turn, in 2017 Roland reissued the TR-808 (as the TR-08) but the original analogue noise is now simulated in software. This technique was described in John Clarke's *White Noise Generator* project in the September 2019 issue of *PE*. As a 'software-wary' analogue engineer, I'd like to present my digital-free solution, which, like many analogue designs came about through a serendipitous accident.

Voltage regulator problem

I designed a power amplifier that seemed to have excessive noise. I tracked down the problem to the centre-rail bias generator shown in Fig.3. I had decided to use a TL431 shunt regulator to replace a 22V Zener diode (which are inherently noisy due to their avalanche breakdown mechanism); I simple assumed the TL431 would be better. Surprisingly, when I put the regulator chip in the circuit the noise level was four-times worse. The new circuit is shown in Fig.4.

I realised that what had happened was the regulator was amplifying its own noise. This can be explained by looking at the regulator's internal circuit shown in Fig.5. The internal voltage reference was 2.5V and two resistors were used to set the output voltage. These were effectively a feedback network setting the gain of the system to almost 10 to get the required 22V output. Consequently, the noise from the reference was amplified by this amount. The solution was to maintain the required DC gain, but reduce the AC gain (to unity) by bypassing

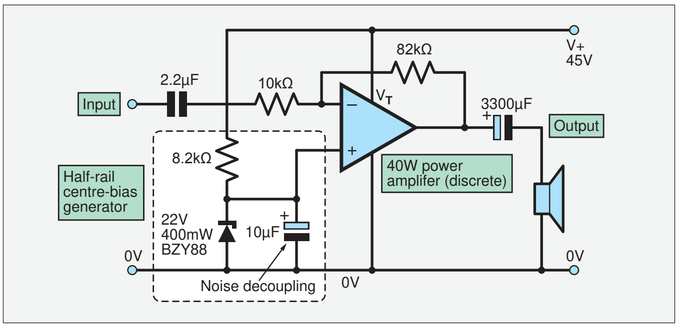


Fig.3. Centre-rail bias generator in a power-amp with a 44V supply. The 22V Zener was too noisy.

the feedback resistor with a capacitor, as shown in Fig.6. This reduced the noise to about half that of the Zener and I was pleased to have fixed the circuit.

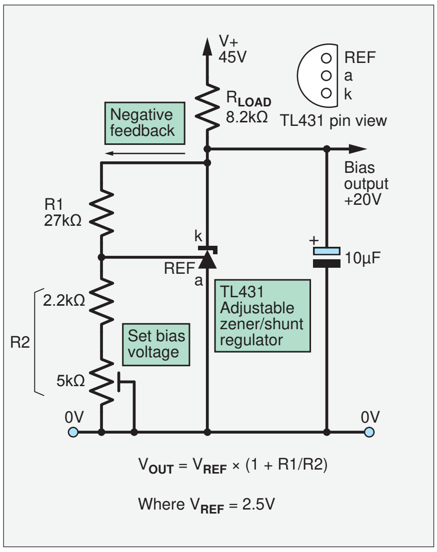


Fig.4. Replacing the Zener with a voltage regulator using this circuit made the noise unexpectedly worse.

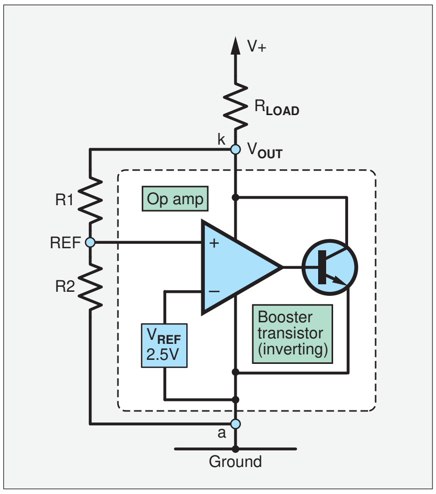


Fig.5. Regulator circuit showing the internal circuit of the TL431. There is a reference generator, a series-pass transistor and an op amp. The resistors set the gain around the loop.

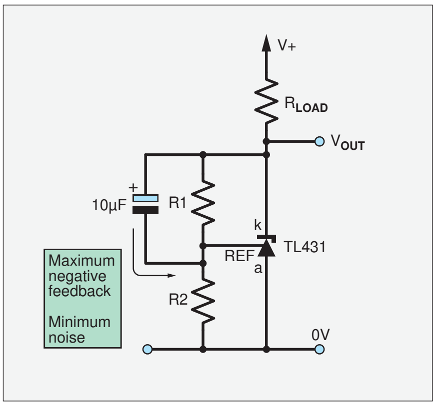


Fig.6. Correct bypass capacitor connection to give minimum noise. Gain at AC is unity, while the DC gain sets the output voltage with R2 and R3.

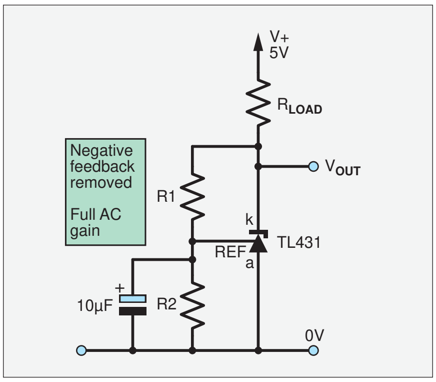


Fig.7. Bypassing the reference pin to ground gives maximum AC gain and maximum amplification of noise.

I was still intrigued by the noise, so I removed all the AC feedback by connecting the capacitor to ground, as shown in Fig.7, and got an amazing noise generator. Now the reference noise was being amplified at full open-loop. I tried a few different TL431s and found the noise to be much stronger and consistent compared to the usual analogue techniques. Also, the generator worked well down to about 3.5V, whereas most analogue systems need at least 9V. This circuit runs at 5V, the rail chosen for the *PE* low-power synthesiser (started in *PE*, August 2018). One word of caution, there is always the risk when using unspecified device characteristics that the manufacturing process might be improved and the 'defect' that is being exploited will be removed. That said, I think we are on stable ground with the TL431 – it was launched in 1978, so Texas Instruments have had over four decades to make changes, and the current version is unlikely to be altered.

Circuit description

Fig.8. Shows the noise generator part of the circuit, consisting of the noisy regulator followed by amplification of 38. This is provided by non-inverting op-amp stage IC1a to increase the output level to around 2.2V peak-to-peak (line-level/0dB). No half-rail potential divider network is needed since this is set by the voltage regulator's output itself, fixed at 2.5V. If the circuit is to be used at 9V instead of its intended 5V, the bias voltage can be changed to 4.5V by adding R4, which is *omitted* for normal 5V operation. To complete the noise generator, a tone control is added, built around the second half of IC1. We'll deal with this part of the circuit later.

A short digital noise diversion

Where predictability or a specific spectral response is needed, we often turn to pseudo-random digital noise generation, where a string of seemingly random zeros and ones (see Fig.9) are generated until the whole sequence repeats again after a long time (from say 20 seconds to months). Incidentally, the problem of excessive randomness was exposed by the early shuffle mode on music players, such as the iPod, where occasionally the same song would be played consecutively. To stop this happening, the random number generation algorithm had to be tweaked to make it appear more 'random' to human perception – ironically, by preventing song repeats and thereby reducing the real randomness.

For sound synthesis a short pseudo-random sequence can be useful, because this can endow a recognisable repeating pattern within it, effective for simulating mechanical noises such as engines. For this, I developed a CMOS hardware-based circuit built around the usual arrangement

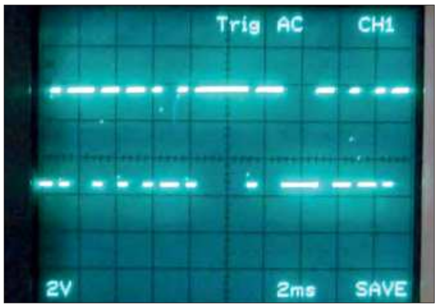


Fig.9. Digital noise: this screen shot shows the random string of 'zeros' and 'ones' which, just like pulse-code modulation can be integrated by a low-pass filter to produce analogue noise.

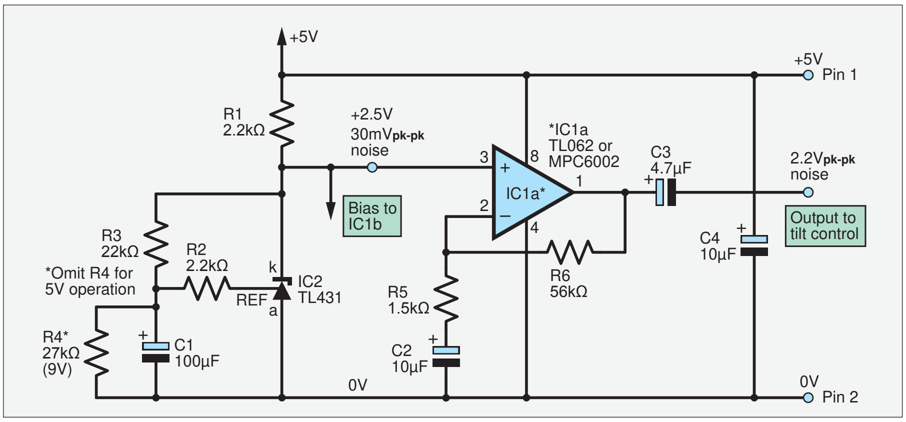


Fig.8. Analogue noise generator with output amplifier. R4 must be added for supply voltages greater than 5V. R2 limits possible capacitor discharge current into the reference pin. (This circuit is connected to a tilt tone control which will be covered next month.)

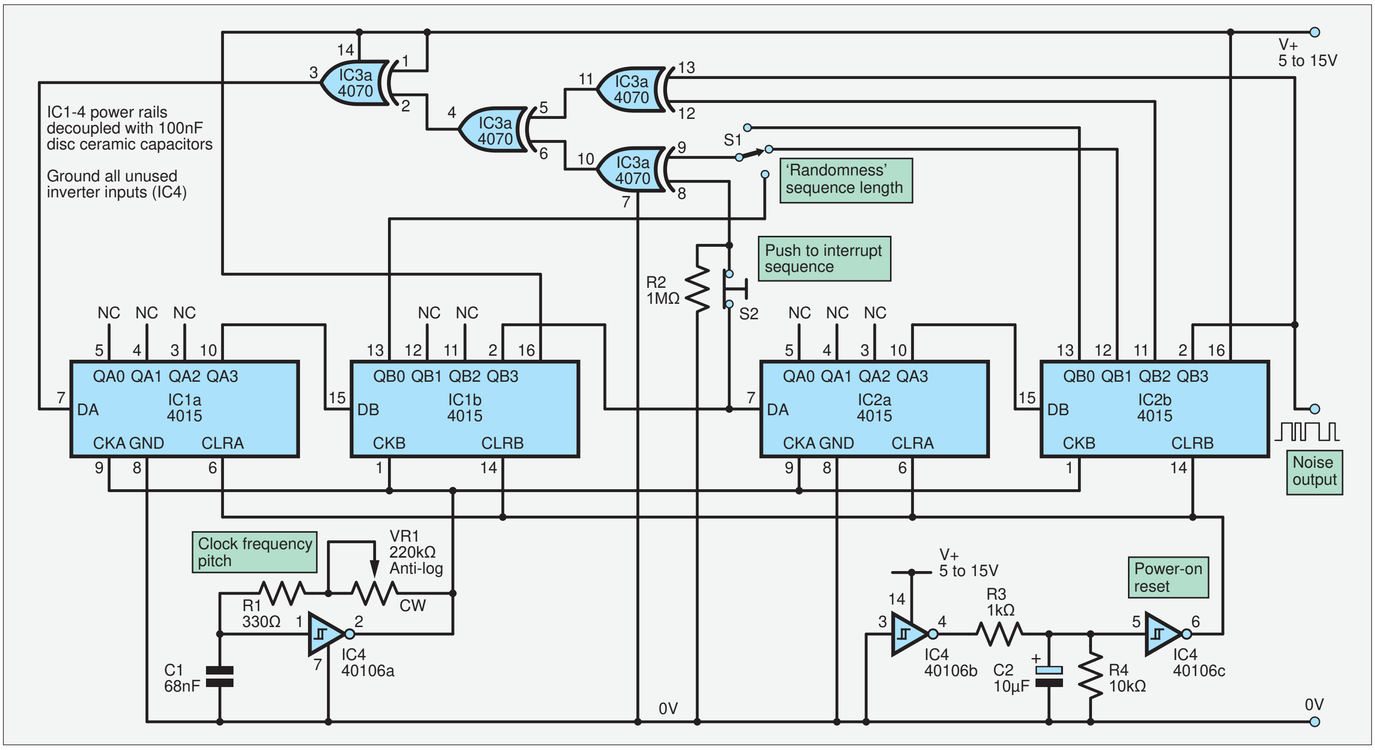


Fig.10. Digital noise generator circuit for synthesisers. It has a frequency control, sequence length switch and a freeze button.

of shift registers and an exclusive OR gate. As John Clarke said in his article, the good old MM5837 chip that integrated the whole lot is now obsolete and expensive. Even the CMOS 4006 18-stage shift-register used in many synthesisers, such as the Wasp, has gone. However, there is still the 4015, a four-stage shift-register, of which two are used in the circuit shown here in Fig.10.

The sound of the digital noise can be changed greatly by varying the clock frequency with VR1. Slowing it down makes the sound gradually change from normal white noise to an interesting tonal noise ('grey' noise) with spaced frequency peaks. Eventually, it just generates random clicks, like a Geiger counter. Modulating the clock with an envelope generator or low-frequency oscillator could give even more variation and control. Different tappings along the shift-register are also employed to give different pattern repeat lengths set by S1. This can range from about 30 seconds down to fractions of a second. Interrupting the feedback with a momentary switch (S2) freezes different bit cycle patterns, giving further weird tonal variations, where you don't know what you are going to get. Musicians love that switch and this circuit is complementary to the analogue circuit.

A breadboard of the circuit is shown in Fig.11. If there is enough demand I'll do a PCB. However, if your soldering iron is burning a hole in your bench, you can at least build the analogue noise generator PCB for now!

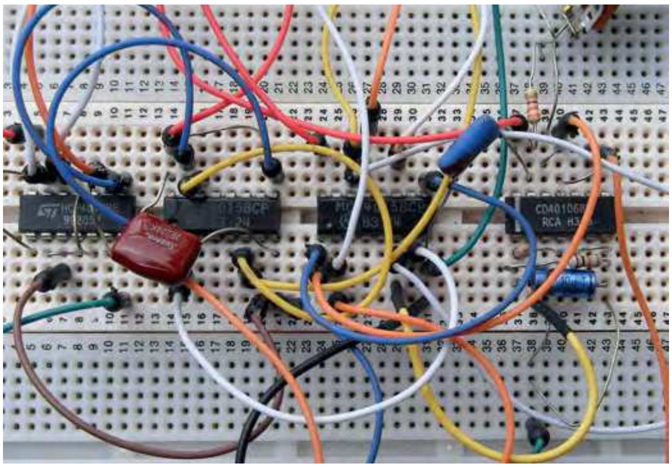


Fig.11. Breadboard of digital noise generator (I'll do a PCB if enough people want it).

Next month

That's enough digital – next month we'll return to strictly analogue when we build the noise generator!

Visual programming with XOD

By Julian Edgar

Fan Speed Controller

How many times have you been working with fan-cooled equipment and then the fan suddenly trips, coming on at full power and disturbing all your concentration? My bench-top power supply is terrible for that – and worse still, when the power supply is running a large load, the fan constantly cycles on-off, on-off. It drives me mad! Or what about a powerful fan-cooled audio amplifier? In quieter spots in the music, the sound of a fan is likely to intrude. Of course, at times you will need the fan working hard, but often when the fan is blasting at full speed, it really only needs to be ticking over – and catching the temperature rise before it goes too far.

That's where this little project comes in. It is based on an Arduino Uno and a small, inexpensive MOSFET PWM control module. The Uno is available from numerous suppliers, and you can find the MOSFET module by searching on eBay for '3-20V MOSFET MOS Transistor Trigger Switch Driver Board PWM Control Module'. For example, at the time of writing item 303491652040 is just £2.80 delivered. However, any similar PWM-controllable MOSFET board will also work fine.

I used a Microchip MCP9700 temperature sensor, but any sensor that can be easily configured to give a temperature reading in degrees Celsius can be used. (The MCP9700 has the advantage of being able to read temperatures of less than zero with a normal 5V supply, making this a controller useful for a wide range of applications.) In many uses, the temperature sensor will be attached to a heatsink (eg, by a clamp), but it can also be used to measure the temperature in free air.

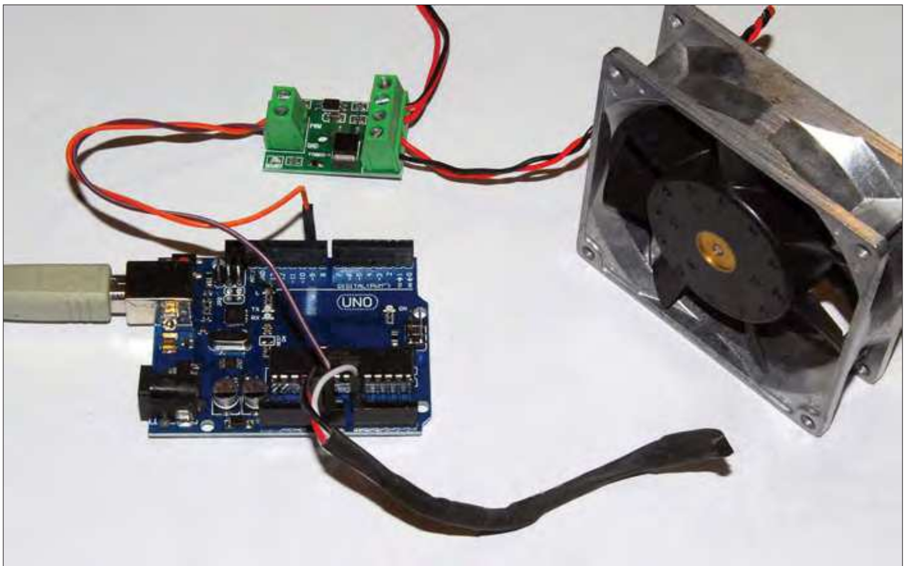


Fig.1. The fan speed controller uses an Arduino Uno board, MOSFET module and temperature sensor. The use of XOD visual programming software allows easy changes to be made to the controller's operation. The temperature sensor is in the foreground. A much more powerful fan than the one shown here can be used if required.

Controlling the fan

The Arduino program ('sketch') is written in Xod (pronounced 'Zod'), a free visual programming software that is easy to follow – and very easy to edit. In fact, to achieve the fan behaviour we want, we will change the values in the program – so you can think of this project as a PC-programmable fan controller. (For an introduction to XOD, see the March 2020 issue of *PE*.)

So what parameters can be changed? There are five settings:

- Period over which temperature reading is averaged
- Temperature at which the fan starts
- Temperature at which the fan reaches maximum speed
- Minimum duty cycle at which the fan can run (note that duty cycle controls fan speed)
- Hysteresis (the difference between the fan switch-on and switch-off temperatures).

Being able to alter all of these is important if the controller is to best suit a specific application. For example, to control the fan in a bench-top power supply, you might set the period over which the temperature is averaged to two seconds, the temperature

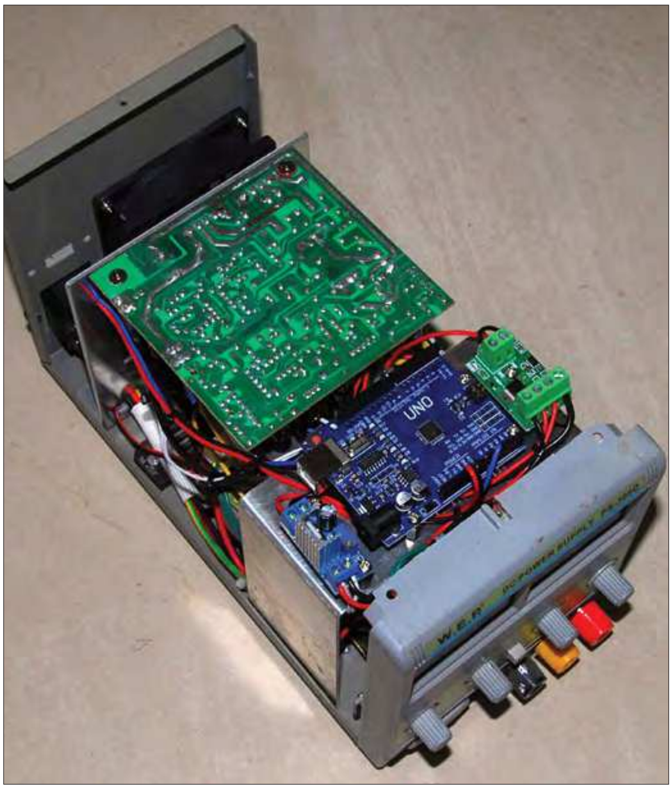


Fig.2. The fan speed controller installed in a bench power supply. The new boards are near the front – from left, voltage regulator for the Arduino (needed here because of the 24V internal supply of the bench supply), Arduino Uno and MOSFET module. The temperature sensor is placed on the heatsink near the rear of the power supply, adjacent to the fan.

at which the fan starts at 25°C, and the temperature at which the fan is running at full power at 50°C. The hysteresis might be set to 3°C. (Note: in operation, the starting temperature and hysteresis are added together, so the actual starting temp in this example will be 28°C.)

And the minimum duty cycle? That depends on the fan – some will run down to 25% duty cycle, others not below 40%. Being able to set this minimum means the fan is never fed (say) a 15% duty cycle, which would be an average voltage too low to turn it.

Note that the fan that is used can be quite powerful: the cited MOSFET module can run a continuous 5A at 12V – 60W. And, if an even more powerful fan is required, you can simply specify a module with a higher current rating or add a heatsink to the shown module (in this form it should be good for 10A).

Wiring

The MCP9700 temperature sensor has three connections: +5V, ground and signal. Refer to Fig.3. Make these connections via header pins to the Arduino, using port A0 for the signal.

The MOSFET module has connections for PWM and ground. Connect Port 9 to the 'PWM' terminal on the module, and the corresponding ground terminal to a ground pin on the Arduino. The fan connects to the Out (+) and (–) terminals and power to the DC (+) and (–) terminals. Depending on the fan voltage (5 or 12V) you can run the entire system on either of these voltages. Refer to Fig.3 for these connections.

Program

After you have installed XOD on your PC (**see https://xod.io/downloads/**), you can download the fan controller program from the April 2020 page of the *PE* website and then upload it to the Arduino. But the real beauty of XOD is it's easy to see how the program works, so let's turn to Fig.4. Don't be daunted – I'll break it down into its parts. The circled numbers are matched on the XOD diagram in Fig.4

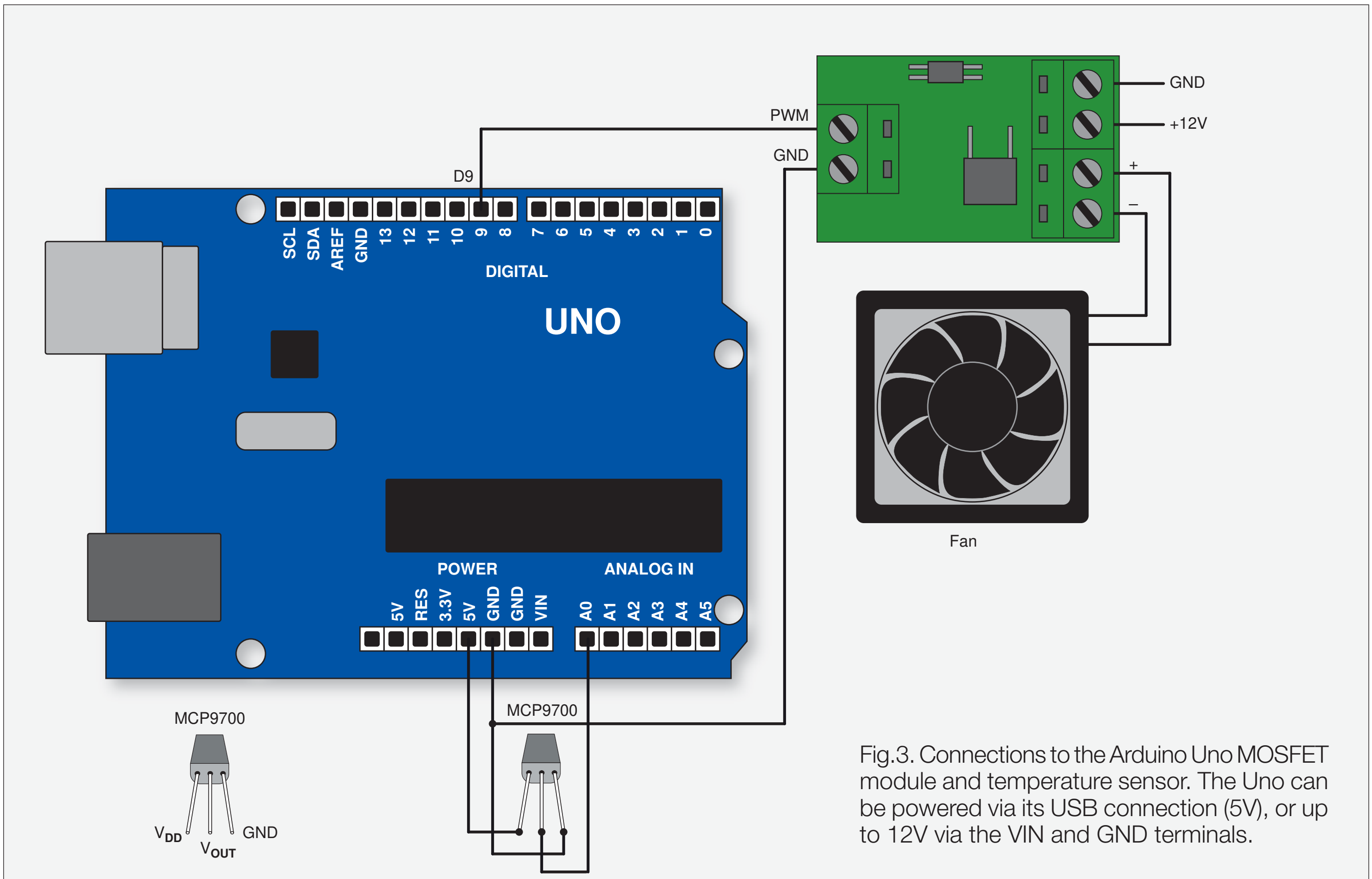


Fig.3. Connections to the Arduino Uno MOSFET module and temperature sensor. The Uno can be powered via its USB connection (5V), or up to 12V via the VIN and GND terminals.

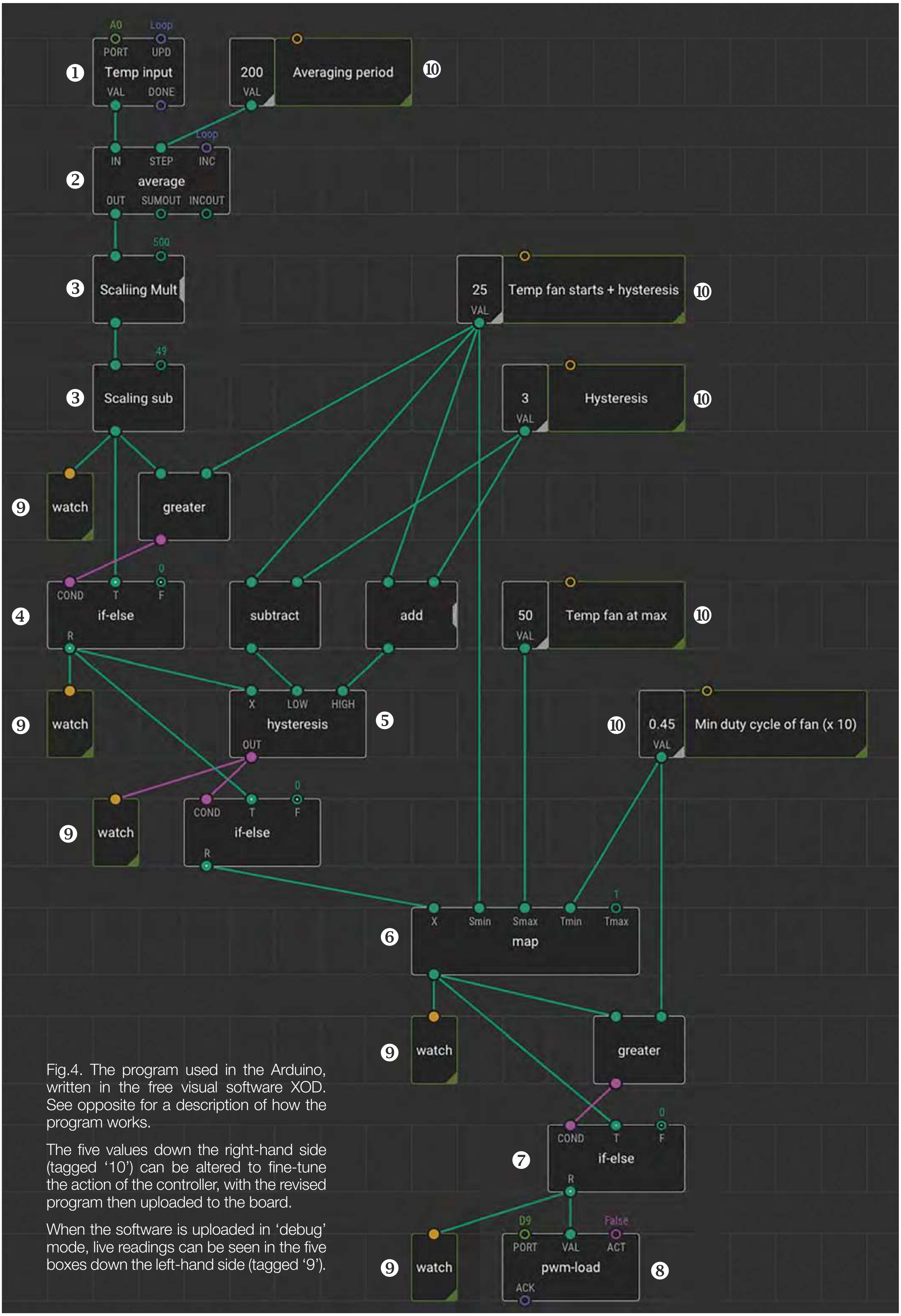


Fig.4. The program used in the Arduino, written in the free visual software XOD. See opposite for a description of how the program works.

The five values down the right-hand side (tagged ‘10’) can be altered to fine-tune the action of the controller, with the revised program then uploaded to the board.

When the software is uploaded in ‘debug’ mode, live readings can be seen in the five boxes down the left-hand side (tagged ‘9’).

Temperature input

❶Starting at the top, the temperature input is continuously watched (ie, looping) at Port A0.

Averaging

❷The next box averages the value. (A higher number equals a longer averaging period.)

Scaling

❸Next, we scale the reading so that it is in degrees Celsius. (The second scaling box can be used to tweak the temperature sensor offset for highest accuracy; 50 is the nominal value.)

Decision

❹After that, there's the first of the program's decisions to be made – is the measured temperature over 25°C? If it is greater than 25 (ie, True), the 'if-else' box passes the signal on; if not (ie, False), it is replaced with a zero.

Hysteresis

❺The hysteresis box follows next. This sets the difference between the switch-on and switch-off temperatures. Here it has been set at 3°C. If the temperature is not within this range of the set point (ie, the fan is permitted to run), the 'if-else' box passes that information on to the 'map' box.

Mapping

❻The 'map' box is a scaling device. It takes an input value range (here set from 25 to 50) and converts that into a 0 to 1 output. (The PWM generator – I'll get to in a minute – requires an input range of 0-1.)

Duty cycle decision

❼Another 'if-else' box follows – this allows the signal to pass only if it is above 0.45 (ie, 45% duty cycle). If it is below that, the signal is again replaced by a zero.

PWM generator

❽The final box is a PWM generator. It uses port D9 – and that's where we connected our PWM MOSFET module. This box outputs 100% duty cycle when fed a 1, and 0% duty cycle when fed 0. (Incidentally, the output frequency is about 400Hz.)

Real-time watching

❾Note the 'watch' boxes down the left-hand side. If you upload the program in debug mode (press the bug-shaped icon at the bottom right of the XOD screen to do this) the numerical values and logic (ie, true/false) at each step of the program will be visible, 'live' in these boxes. By applying heat (eg, with a soldering iron) to the temperature sensor, you can watch the program working.

System tuning

❿This is also the time to fine-tune those values – use the boxes down the right-hand side of the program to do that. Click on each and you can change the value in the left-hand 'Inspector' column of the XOD software before uploading the tuned program to the Arduino board.

Conclusion

In this article the project has been used to control fan speed in electronic equipment, but its range of adjustments, capacity (using a suitable MOSFET module) to drive high current loads and ability to read a wide range of temperatures, means the controller is suitable for many applications. For example, it could also be used to control pump speed in a water-cooled PC, ventilate a garden hothouse, or control the speed of a car radiator fan.

XOD files

The XOD file discussed in this article can be downloaded from the April 2020 page of the *PE* website.

Max's Cool Beans

By Max the Magnificent

Flashing LEDs and drooling engineers – Part 2

Fig.1. 450-piece box of LEDs from Amazon – remember, you can never have too many!

The purpose of this miniseries is to cogitate and ruminate on how we can use LEDs to add to the pizazz of our hobby projects. In my previous column (*PE*, March 2020), we considered some of the things we could do with a single unicolor LED controlled by a single-pole, single-throw (SPST) toggle switch. Just for giggles and grins, if you are a little rusty in this area, you can peruse and ponder my column on *Electrical Switch Terminology* (**https://bit.ly/30ZpToT**).

My initial experiments were performed using a green LED that I pulled out of my treasure chest of spare parts. Since we will be using a bunch of LEDs – and on the basis you can never have too many – I ordered a 450-piece set from Amazon for only $11.99 in the US or £8.99 in the UK (**https://amzn.to/38K7D5q**). I remember when LEDs were rather new and relatively expensive, so being able to pick up a box containing five colors × 90 pieces for less than two UK pennies apiece never fails to amaze me (Fig.1.)

Remember that we're using an Arduino Uno with its 5V input/output pins for our experiments. All of the LEDs in my new goodie box are shown as having maximum forward currents of 20mA (ie, 0.02A). Furthermore, the red and yellow LEDs are shown as having a 2.0 to 2.2V forward-voltage drop. If we assume 2.0V, this means we'll need current-limiting resistors of $(5 – 2)/0.02 = 150\Omega$. Meanwhile, the white, blue, and green LEDs are shown as having a 3.0 to 3.2V forward-voltage drop. If we assume 3.0V, this means we'll need current-limiting resistors of $(5 – 3)/0.02 = 100\Omega$. Happily, I have hundreds of both resistor values lying around.

The mists of time

Just to refresh our memories, let's take a quick peek back through the mists of time to remind ourselves where we're at. We started with a single toggle switch controlling a single unicolor LED (Fig.2.). We also noted that, in the UK, moving the actuator of a toggle switch down or up typically turns it On or Off, respectively (things work the other way in the USA, and you take your chances in the rest of the world.

Using this setup, we explored a number of different scenarios. We started with the simplest case in which the LED tracks the state of the switch:

```
Switch -> On; LED -> On
Switch -> Off; LED -> Off
```

Next, we added a bit of flash (no pun intended) by making the LED flash three times before going hard on when the switch was toggled to its On position:

```
Switch -> On; LED -> Flash-Flash-Flash-On
Switch -> Off; LED -> Off
```

Finally, we experimented with making the LED fade on and fade off, which actually looked surprisingly tasty:

```
Switch -> On; LED -> Fade-On
Switch -> Off; LED -> Fade-Off
```

Of course, we could play with a mixture of these effects. For example, I might be tempted to combine the flashing On from the second scenario with the fading Off from the third scenario (so many fun things to play with; so little time to do them all).

Taking a timeout

Sad to relate, I fear we have to take a bit of a timeout with one of the things we discussed in my previous column. This was doing something special if no one used the switch for a certain amount of time – say 10 minutes – which we called the 'Inactive Timeout Period.' We considered a variety of possibilities here, such as the LED slowly 'breathing' (fading On and Off) if the switch were in its On position, and periodically emitting a tiny flash if the switch were in its Off position.

On reflection, however, I fear I was getting a little carried away here, because I was thinking of a single toggle switch and its accompanying LED in isolation. Suppose we have a control panel festooned with toggle switches and LEDs. If they were all breathing and/or flashing at the same time, the result would be a visual cacophony of chaos, and you don't want to see one of those too often.

System status

On the other hand, let's suppose that we have a master On/Off control in the form of a pushbutton switch, for example. This could be equipped with a single green LED like our toggle switch. Alternatively, it could boast two LEDs, let's say green and red (Fig.3).

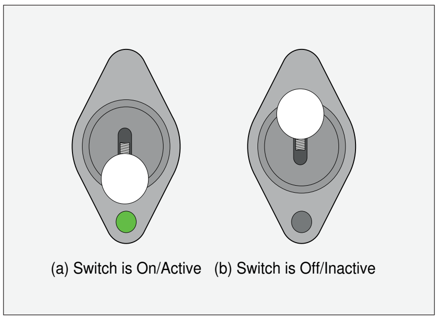


Fig.2. Single LED's active/inactive states.

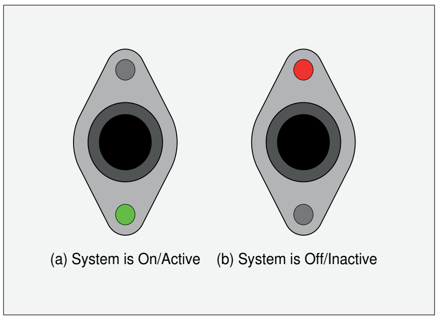


Fig.3. System's master On/Off pushbutton.

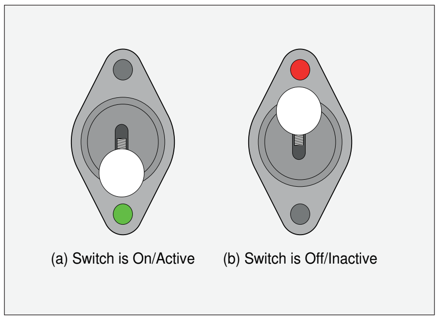


Fig.4. Toggle switch with two unicolor LEDs.

As usual, there are lots of different things we could do here. For example, if the system is On, the green LED could be On and the red LED could be Off; contra wise, if the system is Off, the green LED could be Off and the red LED could be on.

Funnily enough, following my first column, one of my friends – we'll call him Tom (because that's his name) – emailed me to say:

'As a wee bairn of an engineer, I remember joking about adding an Off light to products. My fellow designers would chuckle at this, knowing it was intended to poke fun at the sales/marketing folks who were interested in new 'features' that the competitors didn't have, regardless of how engineeringly senseless they were. When talking to sales/marketing, we positioned this as a product differentiator that would force our competitors to follow suit.

'You can only imagine my surprise when this throwaway idea became real. These days, it is no longer fashionable to have an LED lit to indicate On and unlit to indicate Off. That would be too simple and pedestrian and not waste any extra power. Nowadays, you need an Off light illuminated, usually red, when the device is Off. Or, at least, mostly off, since you need actual power to light the Off light. You also need an On light, usually green, to indicate 'Not OFF.' I obviously missed my calling. I should have gone into sales/marketing (heavy sigh).'

Now, I know that it may seem a little counterintuitive to have a special LED to indicate that the device is Off – obviously having the On LED turned Off performs a similar service – but having a special Off LED can be handy. When I plug a power supply into the wall, for example, I find it reassuring when its green LED lights up to indicate it's seeing power, even if I'm not yet using it for anything. In a similar vein, in the case of something like a TV, having a red Off LED not only informs me that the TV is Off, it also informs me that the system is still receiving power and is poised to leap into action at my beck and call.

Returning to the idea of an inactive mode; suppose the system is On, but no one does anything with it for a while. In this case, maybe the green LED associated with the system's master On/Of pushbutton could enter a breathing mode, while all of the LEDs associated with the various toggle switches could either turn off completely or fade down to a low glow.

Feel the attraction

Another friend called Steve reminded me that pinball machines have something called the 'attraction mode,' in which an unoccupied game sings a siren song to passing punters using flashing lights and inviting sounds.

I can easily see implementing this sort of thing on my own projects, like the *Prognostication Engine* pictured in my previous column. Assuming the beast is on, but no one operates it for the 'Inactive Timeout Period,' then the system's master On/Off pushbutton LED could start breathing, and all of the LEDs associated with the toggle switches could fade down to a dull glow.

Periodically, however, all of the toggle-switch LEDs could enter a period of frenzied activity to attract the attention of anyone who might be in the vicinity. Furthermore, we could add special 'attraction mode' sound effects. I'm also planning on giving the *Prognostication Engine* proximity detection so it knows if someone is close by and can respond accordingly. In the fullness of time, I'm planning on equipping it with AI and facial recognition, so it will only respond to commands from yours truly (at which point I'll be ready to take over the world).

One is the loneliest number

In the early-1970s, the band Three Dog Night recorded a lot of great songs that still have me humming along to this day, including *Joy to the World*, *An Old-Fashioned Love Song*, and *Never Been to Spain*. On the other hand, they were also responsible for *One is the Loneliest Number*, which I loathe with a passion that defies mere words. Suffice it to say that this song makes me want to gnash my teeth and rend my garb.

Having said this, they had a point when it comes to LEDs; one is a very lonely number. Two is a much happier quantity. So, let's suppose that we now equip our toggle switches with two unicolor LEDs – one green and one red (Fig.4.)

Let's name our LEDs GLED and RLED. In this case, the simplest scenario would be to simply turn them On and Off as follows:

```
Switch -> On; GLED -> On; RLED -> Off
Switch -> Off; GLED -> Off; RLED -> On
```

Alternatively, we could make things look a tad more sophisticated using simple fade effects:

```
Switch -> On; GLED -> Fade On; RLED -> Fade Off
Switch -> Off; GLED -> Fade Off; RLED -> Fade On
```

Since we now have two LEDs, we could also modify our original power-on effect. If the switch is in its Off position when power is applied to the system, we could flash the red LED three times, then flash the green LED three times, then flash both LEDs three times, then fade both LEDs on, and finally fade the green LED off, leaving the red LED to indicate that the switch is currently Off. Alternatively, if the switch is in its On position when power is first applied, we could use a similar sequence, but start by flashing the green LED and end with the green LED illuminated.

If you wish, you can download the sketch for the first scenario (**https://bit.ly/2u3O7SV**) and the second scenario (**https://bit.ly/2ShNLjB**), and also watch a video of these tests in action, including the power-on sequences discussed above (**https://bit.ly/2tuOghW**).

Using these two LEDs, I'm sure you can come up with some interesting effects of your own. If so, please email me to tell me about them. For myself, I'm chomping at the bit to perform some experiments with bicolor LEDs, which will be the topic of my next column.

Over-extended

Several folks have emailed me to ask about the small green PCB plugged into the breadboard in my videos. Well, I have a friend called Mike Pelkey. Believe it or not, Mike is the grandfather of BASE Jumping, where BASE is an acronym for building, antenna, span (bridge), and earth (cliff). In 1966, Mike and his friend Brian Schubert made the first parachute jumps from the top of the El Capitan mountain in California's Yosemite National Park.

Mike is also the founder of LogiSwitch.net and the creator of the LogiSwitch switch debounce chips I now use in all of my projects. A couple of months ago, we were chatting on the phone, and I was bemoaning the way I kept on inadvertently pulling out the flying leads associated with my Arduino-Breadboard prototyping projects. You can only imagine my surprise a couple of weeks later when Mike's LogiSwitch Arduino Uno Workbench Proto-Extender Kit (try saying that ten times quickly) arrived in the post (Fig.5.). These little rascals are now available for purchase in the LogiSwitch store (**https://bit.ly/36HS8cU**).

This board is a shield that plugs into an Arduino Uno. Observe the headers in the middle, which allow you to stack additional shields on top. What Mike has done is to take all of the Arduino's digital and analogue input/output (I/O) pins – along with the 5V, 3V3, and GND pins – and break them out to three sides of the board.

Underneath the board on these three sides are downward-pointing pins that you can plug into breadboards, thereby allowing you to create prototype projects that access the Arduino's pins without using any flying leads. In the video accompanying this article, I'm using only one breadboard, but you can use two or three as required.

Just to give you an idea as to how useful this can be, I created a video showing an Arduino driving 45 LEDs using flying leads (**https://bit.ly/2u5Rg4I**). I then created a second video showing the same Arduino using the Workbench board to drive the same 45 LEDs without any flying leads whatsoever (**https://bit.ly/31jjoNR**).

Fig.5. LogiSwitch's Arduino Uno Workbench Proto-Extender Kit.

The additional components on the board include three tactile pushbuttons that are fed through a LogiSwitch debounce IC to three header pins, along with three LEDs and associated current-limiting resistors that are also fed to three header pins. You can use these switches and LEDs to help control and/or debug your projects. You can discover more about the switch debounce IC in this column (**https://bit.ly/2Qxvqzr**) and more about the Workbench in this column (**https://bit.ly/37KPrIf**).

Cool bean Max Maxfield (Hawaiian shirt, on the right) is emperor of all he surveys at **CliveMaxfield.com** – the go-to site for the latest and greatest in technological geekdom.

Comments or questions? Email Max at: **max@CliveMaxfield.com**

Max's Cool Beans cunning coding tips and tricks

Last month I explained that since I'm providing some downloadable Arduino sketches (programs) to accompany the current miniseries of *Cool Beans* column I thought it might be a good idea to provide a rationale for the coding style I use. In my previous column we talked about cases, spaces, comments, `#define` statements, and why using 'magic numbers' is a bad idea. Now let's take a slightly deeper dive into things like naming conventions.

Variable names

A general rule of thumb is to use nouns or noun phrases (nouns with modifiers) for variable names because it makes the code easier to read.

I use 'camel case' for my variable names. This means that compound words or phrases are written such that each word in the middle of the phrase begins with a capital letter with no intervening spaces or punctuation.

In the case of global variables (which are declared outside of any functions and which are available to all functions), I use upper camel case (aka 'Pascal case'), in which the first character is also uppercase; for example:

```
int  MainLoopCounter    = START_COUNT;
bool MainDoneCounting   = false;
```

For local variables (declared inside a function and which are available only to the function in which they are declared), I use lower camel case, in which the first character is in lowercase:

```
int  localLoopCounter  = START_COUNT;
bool localDoneCounting = false;
```

Rationale: I like the look of camel case and I find it conveys a lot of information in an easy-to-read format. Using upper and lower camel case for global and local variables, respectively, helps me know 'what's what' when I'm reading my code.

Function names

With regard to my own functions (we have to use the Arduino's `setup()` and `loop()` functions 'as is'), I use the same upper camel case for function names as for global variables because I think of functions as being global. A general rule of thumb is to use nouns or noun phrases (nouns with modifiers) as names for functions that are 'getters' or 'questioners', and to use verbs or verb phrases for functions that are 'doers' or generate side effects. Functions that return Boolean (true/false) values usually have names that start with 'Is', 'Has' to make reading expressions easier.

Function calls and declarations

Consider the following function call and function declaration:

```
// Body of program
    // Do stuff here

    UpdateCounter();    // Call function
    // Do stuff here
// Body of program

void UpdateCounter ()  // Declare function
{
    // Do stuff here
}
```

Observe that I don't include a space between the function name and the '`()`' parentheses when I'm *calling* the function in the body of the program, but I do include this space when I *declare* the function.

Rationale: This makes it easy to search the code for instances where the function is called (without space) versus the function itself (with space).

Use of curly brackets: { }

When declaring a function, some people place the opening '`{`' immediately after the parentheses; for example:

```
void UpdateCounter () {
    // Statements go here
}
```

By comparison, I prefer to have the opening '`{`' on its own line directly above its corresponding '`}`'; for example:

```
void UpdateCounter ()
{
}
```

I also follow the same practice with conditional statements:

```
if (doneCounting == true)
{
    UpdateCounter();
}
```

Similarly, I follow the same practice for control structures like `for ()` loops; for example:

```
for (int i = 0; i < MAX_COUNT; i++)
{
    // Do stuff here
}
```

Rationale: When you have nested statements, it can quickly become difficult to work out which '`{`' goes with which '`}`'. Placing the opening '`{`' on its own line directly above its corresponding '`}`' makes it much easier to work out what's going on.

More on curly brackets: { }

In the case of a conditional statement that has only a single associated action statement, some people omit the '`{`' and '`}`' altogether:

```
if (doneCounting == true) UpdateCounter();
```

or

```
if (doneCounting == true)
    UpdateCounter();
```

By comparison, I always use '`{`' and '`}`', even if I have only a single action statement; for example:

```
if (doneCounting == true)
{
    UpdateCounter();
}
```

Rationale: When you are debugging code, you often want to add additional statements, and having the '`{`' and '`}`' already in place greatly aids this process while also preventing you from adding new problems while you are trying to resolve existing issues.

Indentation

Like many C programmers, I used to use two spaces for indentation; for example:

```
void BigRedButton (int whatState)
{
  if (whatState == BUTTON_OFF)
  {
    UpdateColor(OFF_COLOR);
  }
  else if (whatState == BUTTON_ON)
  {
    doUpdateColor(ON_COLOR);
  }
  else
  {
  }
}
```

Now, however, based both on my reading of the Barr Group's *Embedded C Coding Standard* (**https://bit.ly/2MP3ftB**) and on my experiences with Python, I prefer to indent using four spaces; for example:

```
void BigRedButton (int whatState)
{
    if (whatState == BUTTON_OFF)
    {
        UpdateColor(OFF_COLOR);
    }
    else if (whatState == BUTTON_ON)
    {
        doUpdateColor(ON_COLOR);
    }
    else
    {
    }
}
```

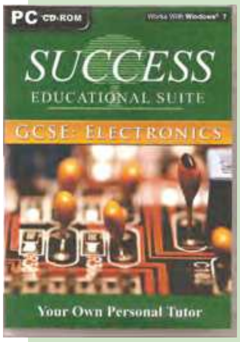
Rationale: Using four-space indentation makes the code much easier to read and understand, both for the originator and for whoever has to work out what this all means at some time in the future. (There's the old programmer's joke about a comment found in code saying, 'When I wrote this, only God and myself knew how it worked, now God only knows!')

Next time

In my next column we'll look at declaring variables as part of `for ()` loops, naming control variables used in `for ()` loops, and data types in general.

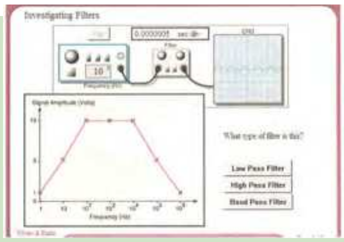
PRACTICAL ELECTRONICS IS PLEASED TO OFFER YOU THESE

ELECTRONICS CD-ROMS



£12.50
inc. VAT and P&P

GCSE ELECTRONICS



Suitable for any student who is serious about studying and who wants to achieve the best grade possible. Each program's clear, patient and structured delivery will aid understanding of electronics and assist in developing a confident approach to answering GCSE questions. The CD-ROM will be invaluable to anyone studying electronics, not just GCSE students.

*** Contains comprehensive teaching material to cover the National Curriculum syllabus * Regular exercises reinforce the teaching points * Retains student interest with high quality animation and graphics * Stimulates learning through interactive exercises * Provides sample examination ques-tions with model solutions * Authored by practising teachers * Covers all UK examination board syllabuses * Caters for all levels of ability * Useful for self-tuition and revision**

SUBJECTS COVERED

Electric Circuits – Logic Gates – Capacitors & Inductors – Relays – Transistors – Electric Transducers – Operational Amplifiers – Radio Circuits – Test Instruments

Over 100 different sections under the above headings

CIRCUIT WIZARD

Circuit Wizard is a revolutionary software system that combines circuit design, PCB design, simulation and CAD/CAM manufacture in one complete package. Two versions are available – Standard or Professional.

By integrating the entire design process, Circuit Wizard provides you with all the tools necessary to produce an electronics project from start to finish – even including on-screen testing of the PCB prior to construction!

* Circuit diagram design with component library (500 components Standard,1500 components Professional) * Virtual instruments (4 Standard, 7 professional) * On-screen animation * Interactive circuit diagram simulation * True analogue/digital simulation * Simulation of component destruction * PCB Layout * Interactive PCB layout simulation * Automatic PCB routing * Gerber export * Multi-level zoom (25% to 1000%) * Multiple undo and redo * Copy and paste to other software * Multiple document support

This software can be used with the *Jump Start* and *Teach-In 2011* series (and the *Teach-In 4* book).

Standard **£61.25** inc. VAT. Professional **£75** plus VAT.

From £49.00

TINA Design Suite V12

FEATURED IN OUR TEACH-IN 2015 SERIES

Analogue, Digital, Symbolic, RF, MCU and Mixed-Mode Circuit Simulation and PCB Design with TINA

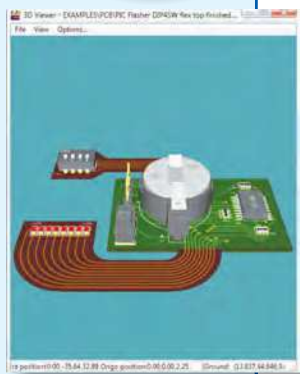
TINA Design Suite V12 is a powerful yet affordable software package for analysing, designing and real time testing analogue, digital, MCU, and mixed electronic circuits and their PCB layouts. You can also analyse RF, communication, optoelectronic circuits, test and debug microcontroller applications.

Enter and analyse any circuit up to 100 nodes (student), or 200 with the Basic (Hobbyist) version within minutes with TINA's easy-to-use schematic editor. Enhance your schematics by adding text and graphics. Choose components from the large library containing more than 10,000 manufacturer models. Analyse your circuit through more than 20 different analysis modes or with 10 high tech virtual instruments.

Present your results in TINA's sophisticated diagram windows, on virtual instruments, or in the live interactive mode where you can even edit your circuit during operation.

Customise presentations using TINA's advanced drawing tools to control text, fonts, axes, line width, colour and layout. You can create and print documents directly inside TINA or cut and paste your results into your favourite word procesing or DTP package.

TINA includes the following Virtual Instruments: Oscilloscope, Function Generator, Multimeter, Signal Analyser/Bode Plotter, Network Analyser, Spectrum Analyser, Logic Analyser, Digital Signal Generator, XY Recorder.



This offer gives you a CD-ROM – the software will need registering (FREE) with Designsoft (TINA), details are given within the package.

Get TINA 12 Design Suite (Hobbyist) for £129 or Student V12 version for £49

Prices include VAT and UK postage

+ get a 1 year free subscription for TINACloud the breakthrough cloud version of TINA which you can run on most operating systems and computers, including PCs, Macs, thin clients iPads and other tablets – even on many smart phones, smart TVs and e-book readers.

To order please either fill out and return the order form, or call us on 01202 880299

Alternatively you can order via our secure online shop at: www.epemag.com

PICmicro TUTORIALS AND PROGRAMMING

HARDWARE

PICmicro Multiprogrammer Board and Development Board

Suitable for use with the three software packages listed below

This flexible PICmicro microcontroller programmer board and combination board allows students and professional engineers to learn how to program PICmicro microcontrollers as well as program a range of 8, 18, 28 and 40 pin devices from the 12, 16 and 18 series PICmicro ranges. For those who want to learn, choose one or all of the packages below to use with the hardware.

- Makes it easier to develop PICmicro projects
- Supports low cost Flash-programmable PICmicro devices
- Fully featured integrated displays – 16 individual LEDs, quad 7-segment display and alphanumeric LCD display
- Supports PICmicro microcontrollers with A/D converters
- Fully protected expansion bus for project work
- USB programmable
- Compatible with the E-blocks range of accessories

£159.99 including VAT and postage (worldwide)

SOFTWARE

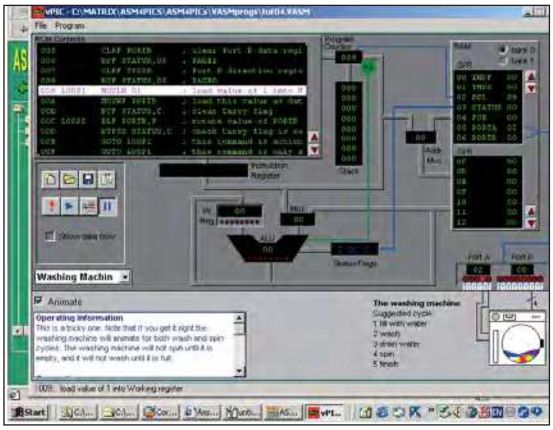
ASSEMBLY FOR PICmicro V7

(Formerly PICtutor)

Assembly for PICmicro microcontrollers V7.0 (previously known as PICtutor) by John Becker contains a complete course in programming the PIC16F84, 16F88 and 16F877a PICmicro microcontroller from Arizona Microchip. It starts with fundamental concepts and extends up to complex programs including watchdog timers, interrupts and sleep modes.

The CD makes use of the latest simulation techniques which provide a superb tool for learning: the Virtual PICmicro microcontroller, this is a simulation tool that allows users to write and execute MPASM assembler code for the PIC16F84 microcontroller on-screen. Using this you can actually see what happens inside the PICmicro MCU as each instruction is executed, which enhances understanding.

- Comprehensive instruction through 45 tutorial sections
- Includes Vlab, a Virtual PICmicro microcontroller: a fully functioning simulator
- Tests, exercises and projects covering a wide range of PICmicro MCU applications
- Includes MPLAB assembler
- Visual representation of a PICmicro showing architecture and functions
- Expert system for code entry helps first time users
- Shows data flow and fetch execute cycle and has challenges (washing machine, lift, crossroads etc.)
- Imports MPASM files.

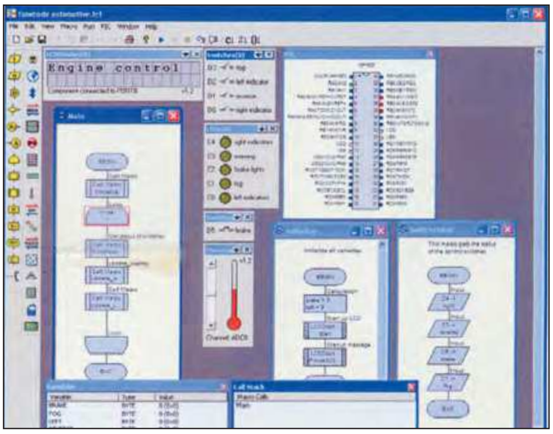


FLOWCODE FOR PICmicro V8

Flowcode is a very high level language programming system based on flowcharts. Flowcode allows you to design and simulate complex systems in a matter of minutes. A powerful language that uses macros to facilitate the control of devices like 7-segment displays, motor controllers and LCDs. The use of macros allows you to control these devices without getting bogged down in understanding the programming. When used in conjunction with the development board this provides a seamless solution that allows you to program chips in minutes.

- Requires no programming experience
- Allows complex PICmicro applications to be designed quickly
- Uses international standard flow chart symbols
- Full on-screen simulation allows debugging and speeds up the development process.
- Facilitates learning via a full suite of demonstration tutorials
- Produces code for a wide range of devices
- 16-bit arithmetic strings and string manipulation
- Pulse width modulation
- I2C.

Please note: Due to popular demand, Flowcode is now available as a download. Please include your email address and a username (of your choice) on your order. A unique download code will then be emailed to you.



PRICES

Prices for each of the CD-ROMs above are:

(Order form on next page)

(UK and EU customers add VAT to 'plus VAT' prices)

Single License . . . . £99 plus VAT

Site Licence . . . . £499 plus VAT

Flowcode . . . . Contact us for pricing
(choose PIC-8b, PIC-16b, PIC-32b, AVR/Arduino,ARM)

ELECTRONICS TEACH-IN 2

ELECTRONICS TEACH-IN 2 CD-ROM
USING PIC MICROCONTROLLERS A PRACTICAL INTRODUCTION

This *Teach-In* series of articles was originally published in *EPE* in 2008 and, following demand from readers, has been collected together in the *Electronics Teach-In 2* CD-ROM.

The series is aimed at those using PIC microcontrollers for the first time. Each part of the series includes breadboard layouts to aid understanding and a simple programmer project is provided.

Also included are 29 *PIC N' Mix* articles, also republished from *EPE*. These provide a host of practical programming and interfacing information, mainly for those that have already got to grips with using PIC microcontrollers. An extra four part beginners guide to using the C programing language for PIC microcontrollers is also included.

The CD-ROM also contains all of the software for the *Teach-In 2* series and *PIC N' Mix* articles, plus a range of items from Microchip – the manufacturers of the PIC microcontrollers. The material has been compiled by Wimborne Publishing Ltd. with the assistance of Microchip Technology Inc.

CD-ROM **Order code ETI2 CD-ROM** **£9.50**

ELECTRONICS TEACH-IN 3

ELECTRONICS TEACH-IN 3 CD-ROM

The three sections of this CD-ROM cover a very wide range of subjects that will interest everyone involved in electronics, from hobbyists and students to professionals. The first 80-odd pages of Teach-In 3 are dedicated to *Circuit Surgery*, the regular *EPE* clinic dealing with readers' queries on circuit design problems – from voltage regulation to using SPICE circuit simulation software.

The second section – *Practically Speaking* – covers the practical aspects of electronics construction. Again, a whole range of subjects, from soldering to avoiding problems with static electricity and indentifying components, are covered. Finally, our collection of *Ingenuity Unlimited* circuits provides over 40 circuit designs submitted by the readers of *EPE*.

The CD-ROM also contains the complete *Electronics Teach-In 1* book, which provides a broad-based introduction to electronics in PDF form, plus interactive quizzes to test your knowledge, TINA circuit simulation software (a limited version – plus a specially written TINA Tutorial).

The Teach-In 1 series covers everything from Electric Current through to Microprocessors and Microcontrollers and each part includes demonstration circuits to build on breadboards or to simulate on your PC.

CD-ROM **Order code ETI3 CD-ROM** **£8.50**

ELECTRONICS TEACH-IN 4

ELECTRONICS TEACH-IN 4 CD-ROM
A BROAD-BASED INTRODUCTION TO ELECTRONICS.

The Teach-In 4 CD-ROM covers three of the most important electronics units that are currently studied in many schools and colleges. These include, Edexcel BTEC level 2 awards and the electronics units of the Diploma in Engineering, Level 2.

The CD-ROM also contains the full Modern Electronics Manual, worth £29.95. The Manual contains over 800 pages of electronics theory, projects, data, assembly instructions and web links.

A package of exceptional value that will appeal to all those interested in learning about electronics or brushing up on their theory, be they hobbyists, students or professionals.

CD-ROM **Order code ETI4 CD-ROM** **£8.99**

ELECTRONICS TEACH-IN BUNDLE – FOR PARTS 3, 4 & 5

CD-ROM **Order code ETIB3 CD-ROM** **£18.95**

ORDERING

ALL PRICES INCLUDE UK POSTAGE

Standard/Student/Basic (Hobbyist) Version price includes postage to most countries in the world EU residents outside the UK add £5 for airmail postage per order

ELECTRONICS TEACH-IN 5

ELECTRONICS TEACH-IN 5
JUMP START

FREE CD-ROM

15 design and build circuit projects dedicated to newcomers or those following courses in school and colleges. The projects are: Moisture Detector, Quiz Machine, Battery Voltage Checker, Solar-Powered Charger, Versatile Theft Alarm, Spooky Circuits, Frost Alarm, Mini Christmas Lights, iPod Speaker, Logic Probe, DC Motor Controller, Egg Timer, Signal Injector Probe, Simple Radio Receiver, Temperature Alarm.

PLUS:

PIC' N MIX – starting out with PIC Microcontrollers and PRACTICALLY SPEAKING – the techniques of project construction.

FREE CD-ROM – The free CD-ROM is the complete Teach-In 2 book providing a practical introduction to PIC Microprocessors plus MikroElektronika, Microchip and L-Tek PoScope software.

160 Pages **Order code ETI5** **£8.99**

Please send me: **CD-ROM ORDER FORM**

☐ **Assembly for PICmicro V6**
☐ **'C' for 16 Series PICmicro V6**

Version required:
☐ **Single Licence**
☐ **Site licence**

Note: The software on each version is the same, only the licence for use varies.

☐ **PICmicro Multiprogrammer Board and Development Board (hardware)**

☐ **Circuit Wizard – Standard**
☐ **Circuit Wizard – Professional**
☐ **GCSE Electronics**
☐ **TINA Design Suite V11 Basic (Hobbyist)**
☐ **TINA Design Suite V11 (Student)**

☐ **Teach-In 2**
☐ **Teach-In 3**
☐ **Teach-In 4**
☐ **Teach-In Bundle (3, 4 and 5)**

Full name . . . .

Address . . . .

. . . . Post code . . . . Tel . . . .

Email: . . . .

☐ I enclose cheque/PO in £ sterling payable to Practical Electronics for £ . . . .
☐ Please charge my Visa/Mastercard: £ . . . .

Valid From: . . . . Card expiry date: . . . .

Card No: . . . .

Card Security Code . . . .

(The last 3 digits on or just under the signature strip)

Single License and Site License Versions – overseas readers add £5 to the basic price of each order for airmail postage (**do not add VAT** unless you live in an EU (European Union) country, then add VAT at 20% or provide your official VAT registration number).

Send your order to:
Electron Publishing Limited
113 Lynwood Drive
Wimborne
Dorset BH21 1UU

To order by phone call: **01202 880299**

Goods are normally sent within seven days

E-mail: shop@electronpublishing.com

Online shop:

www.epemag.com

DIRECT BOOK SERVICE

Teach-In 2017

Introducing the BBC micro:bit

The books listed here have been selected by the *Practical Electronics* editorial staff as being of special interest to everyone involved in electronics and computing. They are supplied by mail order direct to your door.

FOR A FULL DESCRIPTION OF THESE BOOKS AND CD-ROMS SEE THE SHOP ON OUR WEBSITE

www.epemag.com

All prices include UK postage

PYTHON CODING ON THE BBC MICRO:BIT
Jim Gatenby

Python is the leading programming language, easy to learn and widely used by professional programmers. This book uses MicroPython, a version of Python adapted for the BBC Micro:bit.

Among the many topics covered are: main features of the BBC micro:bit including a simulation in a web browser screen; various levels of programming languages; Mu Editor for writing, saving and retrieving programs, with sample programs and practice exercises; REPL, an interactive program for quickly testing lines of code; scrolling messages, creating and animating images on the micro:bit's LEDs; playing and creating music, sounds and synthesized speech; using the on-board accelerometer to detect movement of the micro:bit on three axes; glossary of computing terms.

This book is written using plain English, avoids technical jargon wherever possible and covers many of the coding instructions and methods which are common to most programming languages. It should be helpful to beginners of any age, whether planning a career in computing or writing code as an enjoyable hobby.

118 Pages **Order code PYTH MBIT** **£7.99**

GETTING STARTED WITH THE BBC MICRO:BIT
Mike Tooley

Not just an educational resource for teaching youngsters coding, the BBC micro:bit is a tiny low cost, low-profile ARM-based single-board computer. The board measures 43mm × 52mm but despite its diminutive footprint it has all the features of a fully fledged microcontroller together with a simple LED matrix display, two buttons, an accelerometer and a magnetometer.

Mike Tooley's book will show you how the micro:bit can be used in a wide range of applications from simple domestic gadgets to more complex control systems such as those used for lighting, central heating and security applications. Using Microsoft Code Blocks, the book provides a progressive introduction to coding as well as interfacing with sensors and transducers.

Each chapter concludes with a simple practical project that puts into practice what the reader has learned. The featured projects include an electronic direction finder, frost alarm, reaction tester, battery checker, thermostatic controller and a passive infrared (PIR) security alarm.

No previous coding experience is assumed, making this book ideal for complete beginners as well as those with some previous knowledge. Self-test questions are provided at the end of each chapter, together with answers at the end of the book. So whatever your starting point, this book will take you further along the road to developing and coding your own real-world applications.

108 Pages **Order code BBC MBIT** **£7.99**

MICROPROCESSORS

INTERFACING PIC MICROCONTROLLERS – 2nd Ed
Martin Bates
298 pages **Order code NE48** **£34.99**

PROGRAMMING 16-BIT PIC MICROCONTROLLERS IN C – LEARNING TO FLY THE PIC24
Lucio Di Jasio (Application Segments Manager, Microchip, USA)
496 pages + CD-ROM **Order code NE45** **£38.00**

INTRODUCTION TO MICROPROCESSORS AND MICROCONTROLLERS – 2nd Ed
John Crisp
222 pages **Order code NE31** **£29.99**

THE PIC MICROCONTROLLER YOUR PERSONAL INTRODUCTORY COURSE – 3rd Ed
John Morton
270 pages **Order code NE36** **£25.00**

PIC IN PRACTICE – 2nd Ed
David W. Smith
308 pages **Order code NE39** **£24.99**

MICROCONTROLLER COOKBOOK
Mike James
240 pages **Order code NE26** **£36.99**

THEORY AND REFERENCE

PRACTICAL ELECTRONICS HANDBOOK – 6th Ed
Ian Sinclair
440 pages **Order code NE21** **£33.99**

STARTING ELECTRONICS – 4th Ed
Keith Brindley
296 pages **Order code ELSEV100** **£18.99**

ELECTRONIC CIRCUITS – FUNDAMENTALS & APPLICATIONS – Updated version
Mike Tooley
400 pages **Order code TF43** **£32.99**

FUNDAMENTAL ELECTRICAL AND ELECTRONIC PRINCIPLES – 3rd Ed
C.R. Robertson
368 pages **Order code TF47** **£21.99**

A BEGINNER'S GUIDE TO TTL DIGITAL ICs
Robert Penfold
142 pages **OUT OF PRINT BP332** **£5.45**

UNDERSTANDING ELECTRONIC CONTROL SYSTEMS
Owen Bishop
228 pages **Order code NE35** **£36.99**

BOOK ORDERING DETAILS

All prices include UK postage.

For postage, add £3 per book to Europe, £4 for rest of the world per book.

CD-ROM prices include VAT and/or postage to anywhere in the world.

Send a cheque, (£ sterling only) made payable to: **Practical Electronics** or credit card details (Visa or Mastercard) to:

Electron Publishing Limited,
113 Lynwood Drive, Wimborne, Dorset BH21 1UU

Books are normally sent within seven days of receipt of order.

Please check price (see latest issue of Practical Electronics or website) before ordering from old lists.

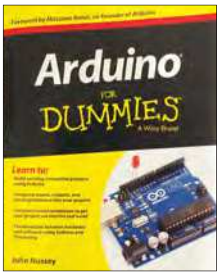
For a full description of these books please see the shop on our website.
Tel: 01202 880299 – Email: shop@electronpublishing.com

Order from our online shop at: www.epemag.com

ARDUINO

Teach-In 2016

See opposite for our popular introduction to the Arduino

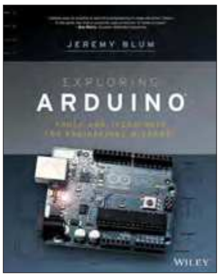


ARDUINO FOR DUMMIES
John Nussey

Arduino is no ordinary circuit board. Whether you're an artist, a designer, a programmer, or a hobbyist, Arduino lets you learn about and play with electronics. You'll discover how to build a variety of circuits that can sense or control real-world objects, prototype your own product, and even create interactive artwork. This handy guide is exactly what you need to build your own Arduino project – what you make is up to you!

- **Learn by doing** – start building circuits and programming your Arduino with a few easy examples – right away!
- **Easy does it** – work through Arduino sketches line by line, and learn how they work and how to write your own.
- **Solder on!** – don't know a soldering iron from a curling iron? No problem! You'll learn the basics and be prototyping in no time.
- **Kitted out** – discover new and interesting hardware to turn your Arduino into anything from a mobile phone to a Geiger counter.
- **Become an Arduino savant** – find out about functions, arrays, libraries, shields and other tools that let you take your Arduino project to the next level
- **Get social** – teach your Arduino to communicate with software running on a computer to link the physical world with the virtual world

438 Pages | Order code ARDDUM01 | £19.99



EXPLORING ARDUINO
Jeremy Blum

Arduino can take you anywhere. This book is the roadmap.

Exploring Arduino shows how to use the world's most popular microcontroller to create cool, practical, artistic and educational projects. Through lessons in electrical engineering, programming and human-computer interaction, this book walks you through specific, increasingly complex projects, all the while providing best practices that you can apply to your own projects once you've mastered these. You'll acquire valuable skills – and have a whole lot of fun.

- **Explore** the features of commonly used Arduino boards
- **Use** Arduino to control simple tasks or complex electronics
- **Learn** principles of system design, programming and electrical engineering
- **Discover** code snippets, best practices and system schematics you can apply to your original projects
- **Master** skills you can use for engineering endeavours in other fields and with different platforms

357 Pages | Order code EXPARD01 | £26.99

COMPUTING AND ROBOTICS

NEWNES INTERFACING COMPANION
Tony Fischer-Cripps
295 pages | Order code NE38 | £41.00

HOW TO BUILD A COMPUTER MADE EASY
Robert Penfold
120 pages | Order code BP707 | £8.49

EASY PC CASE MODDING
Robert Penfold
192 pages + CD-ROM | Order code BP542 | £8.99

FREE DOWNLOADS TO PEP-UP AND PROTECT YOUR PC
Robert Penfold
128 pages | Order code BP722 | £7.99

WINDOWS XP EXPLAINED
N. Kantaris and P.R.M. Oliver
264 pages | Order code BP514 | £7.99

THE INTERNET – TWEAKS, TIPS AND TRICKS
Robert Penfold
128 pages | Order code BP721 | £7.99

eBAY – TWEAKS, TIPS AND TRICKS
Robert Penfold
128 pages | Order code BP716 | £7.50

AN INTRODUCTION TO eBAY FOR THE OLDER GENERATION
Cherry Nixon
120 pages | Order code BP709 | £8.49

HOW TO FIX YOUR PC PROBLEMS
Robert Penfold
128 pages | Order code BP705 | £8.49

AN INTRODUCTION TO WINDOWS VISTA
P.R.M. Oliver and N. Kantarris
120 pages | Order code BP703 | £8.49

WINDOWS 8.1 EXPLAINED
180 Pages | Order code BP747 | £10.99

AN INTRODUCTION TO THE NEXUS 7
118 Pages | Order code BP744 | £8.99

COMPUTING FOR THE OLDER GENERATION
Jim Gatenby
308 pages | Order code BP601 | £8.99

ANDROIDS, ROBOTS AND ANIMATRONS
Second Edition – John Iovine
224 pages | Order code MGH1 | £16.99

ROBOT BUILDERS COOKBOOK
Owen Bishop
366 pages | Order code NE46 | £26.00

INTRODUCING ROBOTICS WITH LEGO MINDSTORMS
Robert Penfold
288 pages + | Order code BP901 | £14.99

MORE ADVANCED ROBOTICS WITH LEGO MINDSTORMS
Robert Penfold
298 pages | Order code BP902 | £14.99

HOW TO FIX YOUR PC PROBLEMS
Robert Penfold
128 pages | Order code BP705 | £8.49

WINDOWS 7 – TWEAKS, TIPS AND TRICKS
Andrew Edney
120 pages | Order code BP708 | £8.49

GETTING STARTED IN COMPUTING FOR THE OLDER GENERATION
Jim Gatenby
120 pages | Order code BP704 | £8.49

WINDOWS 8.1 EXPLAINED
Noel Kantaris
180 Pages | Order code BP747 | £10.99

COMPUTING WITH A LAPTOP FOR THE OLDER GENERATION
Robert Penfold
120 pages | Order code BP702 | £8.49

AN INTRODUCTION TO EXCEL SPREADSHEETS
Jim Gatenby
18 pages | Order code BP701 | £8.49

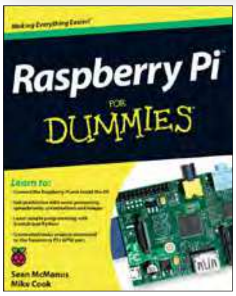
KINDLE FIRE HDX EXPLAINED
118 Pages | Order code BP743 | £8.99

AUDIO & VIDEO

VALVE AMPLIFIERS – 4th Ed
Morgan Jones
288 pages | Order code ELSEV33 | £46.99

BUILDING VALVE AMPLIFIERS
Morgan Jones
368 pages | Order code NE40 | £29.00

RASPBERRY PI



RASPBERRY Pi FOR DUMMIES
Sean McManus and Mike Cook

Write games, compose and play music, even explore electronics – it's easy as Pi! The Raspberry Pi offers a plateful of opportunities, and this great resource guides you step-by-step, from downloading, copying, and installing the software to learning about Linux and finding cool new programs for work, photo editing, and music. You'll discover how to write your own Raspberry Pi programs, create fun games, and much more!

Open this book and find:
What you can do with Python; Ways to use the Raspberry Pi as a productivity tool; How to surf the web and manage files; Secrets of Sonic Pi music programming; A guide to creating animations and arcade games; Fun electronic games you can build; How to build a 3D maze in Minecraft; How to play music and videos on your Raspberry Pi.

400 Pages | Order code RPiDUM01 | £17.99

RASPBERRY Pi MANUAL: A practical guide to the revolutionary small computer
176 pages | Order code H001 | £17.99

RASPBERRY Pi USER-GUIDE – 4th Ed
262 pages | Order code JW001 | £20.90

PROGRAMMING THE RASPBERRY Pi
192 pages | Order code MGH4 | £10.99

GETTING STARTED WITH RASPBERRY Pi
164 pages | Order code OR01 | £11.50

TEACH-IN BOOKS

ELECTRONICS TEACH-IN 6

ONLY AVAILABLE ON CD-ROM

ELECTRONICS TEACH-IN 6
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

Mike & Richard Tooley

Teach-In 6 contains an exciting series of articles that provides a complete introduction to the Raspberry Pi, the low-cost computer that has taken the education and computing world by storm.

This latest book in our Teach-In series will appeal to electronic enthusiasts and computer buffs wanting to get to grips with the Raspberry Pi.

Anyone considering what to do with their Pi, or maybe they have an idea for a project but don't know how to turn it into reality, will find Teach-In 6 invaluable. It covers: Programming, Hardware, Communications, Pi Projects, Pi Class, Python Quickstart, Pi World, and Home Baking.

The CD-ROM also contains all the necessary software for the series so that readers can get started quickly and easily with the projects and ideas covered.

160 Pages | **Order code ETI6** | **£8.99**

ELECTRONICS TEACH-IN 7 (Includes free CD-ROM)

RUNNING LOW – ORDER NOW!

ELECTRONICS TEACH-IN 7
DISCRETE LINEAR CIRCUIT DESIGN

Mike & Richard Tooley

Teach-In 7 is a complete introduction to the design of analogue electronic circuits. Ideal for everyone interested in electronics as a hobby and for those studying technology at schools and colleges. Supplied with a free cover-mounted CD-ROM containing all the circuit software for the course, plus demo CAD software for use with the Teach-In series

Discrete Linear Circuit Design* Understand linear circuit design* Learn with 'TINA' – modern CAD software* Design simple, but elegant circuits* Five projects to build: Pre-amp, Headphone Amp, Tone Control, VU-meter, High Performance Audio Power Amp.

PLUS

Audio Out – an analogue expert's take on specialist circuits
Practically Speaking – the techniques of project building

160 Pages | **Order code ETI7** | **£8.99**

ELECTRONICS TEACH-IN 8

ONLY AVAILABLE ON CD-ROM

ELECTRONICS TEACH-IN 8
INTRODUCING THE ARDUINO

Mike & Richard Tooley

Hardware – learn about components and circuits; Programming – powerful integrated development system; Microcontrollers – understand control operations; Communications – connect to PCs and other Arduinos

This exciting series has been designed for electronics enthusiasts who want to get to grips with the inexpensive, immensely popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. Teach-In 8 will provide a one-stop source of ideas and practical information.

The Arduino offers a remarkably effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle through wireless or the Internet. Teach-In 8 is based around a series of practical projects with plenty of information to customise each project.

This book also includes *PIC n' Mix*: PICs and the PICkit 3 - A Beginners guide by Mike O'Keefe and *Circuit Surgery* by Ian Bell - State Machines part 1 and 2.

The CD-ROM includes files for Teach-In 8 plus Microchip MPLAB IDE XC8 8-bit Compiler and PICkit 3 User Guide. Also included is Lab-Nation Smartscope software.

160 Pages | **Order code ETI8** | **£8.99**

VISIT OUR WEBSITE FOR MORE BOOKS AND FAST, EASY ONLINE ORDERING WWW.EPEMAG.COM

BOOK ORDER FORM

Full name: ....

Address: ....

....

.... Post code: .... Telephone No: ....

Email: ....

☐ I enclose cheque/PO payable to Practical Electronics for £ ....

☐ Please charge my card £ .... Card expiry date..

Card Number ....

Valid From Date .... Card Security Code .... (The last three digits on or just below the signature strip)

Please send book order codes: ....

....

....

Please continue on separate sheet of paper if necessary

THE BASIC SOLDERING GUIDE

LEARN TO SOLDER SUCCESSFULLY!
ALAN WINSTANLEY

The No.1 resource for learning all the basic aspects of electronics soldering by hand.

With more than 80 high quality colour photographs, this book explains the correct choice of soldering irons, solder, fluxes and tools. The techniques of how to solder and desolder electronic components are then explained in a clear, friendly and non-technical fashion so you'll be soldering successfully in next to no time! The book also includes sections on reflow soldering and desoldering techniques, potential hazards, useful resources and a very useful troubleshooting guide.

Also ideal for those approaching electronics from other industries, the *Basic Soldering Guide Handbook* is the best resource of its type, and thanks to its excellent colour photography and crystal clear text, the art of soldering can now be learned by everyone!

86 Pages | **Order code AW1** | **£9.99**

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
APRIL 2020		
Flip-dot Display black coil board	19111181	
Flip-dot Display black pixels	19111182	£14.95
Flip-dot Display black frame	19111183	
Flip-dot Display green driver board	19111184	
MARCH 2020		
Diode Curve Plotter	04112181	£10.95
Steam Train Whistle / Diesel Horn Sound Generator	09106181	£8.50
Universal Passive Crossover (one off)	UPC0320	£12.50
Crossover component set for Wavecor speaker (one off)	WAVXO	(see website)
FEBRUARY 2020		
Motion-Sensing 12V Power Switch	05102191	£5.95
USB Keyboard / Mouse Adaptor	24311181	£8.50
DSP Active Crossover (ADC)	01106191	
DSP Active Crossover (DAC) ×2	01106192	
DSP Active Crossover (CPU)	01106193	£29.95
DSP Active Crossover (Power/routing)	01106194	
DSP Active Crossover (Front panel)	01106195	
DSP Active Crossover (LCD)	01106196	
JANUARY 2020		
Isolated Serial Link	24107181	£8.50
DECEMBER 2019		
Extremely Sensitive Magnetometer	04101011	£16.75
Four-channel High-current DC Fan and Pump Controller	05108181	£8.75
Useless Box	08111181	£11.50
NOVEMBER 2019		
Tinnitus & Insomnia Killer (Jaycar case – see text)	01110181	£8.75
Tinnitus & Insomnia Killer (Altronics case – see text)	01110182	£8.75
OCTOBER 2019		
Programmable GPS-synced Frequency Reference	04107181	£11.50
Digital Command Control Programmer for Decoders	09107181	£8.75
Opto-isolated Mains Relay (main board)	10107181	£11.50
Opto-isolated Mains Relay (2 × terminal extension board)	10107182	
AUGUST 2019		
Brainwave Monitor	25108181	£12.90
Super Digital Sound Effects Module	01107181	£5.60
Watchdog Alarm	03107181	£8.00
PE Theremin (three boards: pitch, volume, VCA)	PETX0819	£19.50
PE Theremin component pack (see p.56, August 2019)	PETY0819	£15.00
JULY 2019		
Full-wave 10A Universal Motor Speed Controller	10102181	£12.90
Recurring Event Reminder	19107181	£8.00
Temperature Switch Mk2	05105181	£10.45
JUNE 2019		
Arduino-based LC Meter	04106181	£8.00
USB Flexitimer	19106181	£10.45
MAY 2019		
2× 12V Battery Balancer	14106181	£5.60
Deluxe Frequency Switch	05104181	£10.45
USB Port Protector	07105181	£5.60
APRIL 2019		
Heater Controller	10104181	£14.00
MARCH 2019		
10-LED Bargraph Main Board	04101181	£11.25
+Processing Board	04101182	£8.60

PROJECT	CODE	PRICE
FEBRUARY 2019		
1.5kW Induction Motor Speed Controller	10105122	£35.00
NOVEMBER 2018		
Super-7 AM Radio Receiver	06111171	£27.50
OCTOBER 2018		
6GHz+ Touchscreen Frequency Counter	04110171	£12.88
Two 230VAC MainsTimers	10108161	£12.88
	10108162	
SEPTEMBER 2018		
3-Way Active Crossover	01108171	£22.60
Ultra-low-voltage Mini LED Flasher	16110161	£5.60
AUGUST 2018		
Universal Temperature Alarm	03105161	£7.05
Power Supply For Battery-Operated Valve Radios	18108171	
	18108172	£27.50
	18108173	
	18108174	
JULY 2018		
Touchscreen Appliance Energy Meter – Part 1	04116061	£17.75
Automotive Sensor Modifier	05111161	£12.88
JUNE 2018		
High Performance 10-Octave Stereo Graphic Equaliser	01105171	£15.30
MAY 2018		
High Performance RF Prescaler	04112162	£10.45
Micromite BackPack V2	07104171	£10.45
Microbridge	24104171	£5.60
APRIL 2018		
Spring Reverberation Unit	01104171	£15.30
DDS Sig Gen Lid	Black	£8.05
DDS Sig Gen Lid	Blue	£7.05
DDS Sig Gen Lid	Clear	£8.05
MARCH 2018		
Stationmaster Main Board	09103171	£17.75
+ Controller Board	09103172	
SC200 Amplifier Module – Power Supply	01109111	£16.45
FEBRUARY 2018		
GPS-Synchronised Analogue Clock Driver	04202171	£12.88
High-Power DC Motor Speed Controller – Part 2		
+ Control Board	11112161	£12.88
+ Power Board	11112162	£15.30
JANUARY 2018		
High-Power DC Motor Speed Controller – Part 1	11112161	£12.88
Build the SC200 Amplifier Module	01108161	£12.88
DECEMBER 2017		
Precision Voltage and Current Reference – Part 2	04110161	£15.35
NOVEMBER 2017		
50A Battery Charger Controller	11111161	£12.88
Micropower LED Flasher (45 × 47mm)	16109161	£8.00
(36 × 13mm)	16109162	£5.60
Phono Input Converter	01111161	£8.00
SEPTEMBER 2017		
Compact 8-Digit Frequency Meter	04105161	£12.88

PCBs for most recent *PE*/*EPE* constructional projects are available. From the July 2013 issue onwards, PCBs with eight-digit codes have silk screen overlays and, where applicable, are double-sided, have plated-through holes, and solder mask. They are similar to photos in the project articles. Earlier PCBs are likely to be more basic and may not include silk screen overlay, be single-sided, lack plated-through holes and solder mask.

Always check price and availability in the latest issue or online. A large number of older boards are listed for ordering on our website.

In most cases we do not supply kits or components for our projects. For older projects it is important to check the availability of all components before purchasing PCBs.

Back issues of articles are available – see Back Issues page for details.

Double-sided | plated-through holes | solder mask

PROJECT	CODE	PRICE
AUGUST 2017		
Micromite-Based Touch-screen Boat Computer GPS	07102122	£10.45
Fridge/Freezer Alarm	03104161	£8.00
JULY 2017		
Micromite-Based Super Clock	07102122	£10.45
Brownout Protector for Induction Motors	10107161	£12.90
JUNE 2017		
Ultrasonic Garage Parking Assistant	07102122	£10.45
Hotel Safe Alarm	03106161	£8.00
100dB Stereo LED Audio Level/VU Meter	01104161	£17.75
MAY 2017		
The Micromite LCD BackPack	07102122	£11.25
Precision 230V/115V 50/60Hz Turntable Driver	04104161	£19.35
APRIL 2017		
Microwave Leakage Detector	04103161	£8.00
Arduino Multifunctional 24-bit Measuring Shield	04116011	£17.75
+ RF Head Board	04116012	
Battery Pack Cell Balancer	11111151	£9.00
MARCH 2017		
Speech Timer for Contests & Debates	19111151	£16.42
FEBRUARY 2017		
Solar MPPT Charger/Lighting Controller	16101161	£17.75
Turntable LED Strobe	04101161	£7.60
JANUARY 2017		
High-performance Stereo Valve Preamplifier	01101161	£17.75
High Visibility 6-Digit LED Clock	19110151	£16.42
DECEMBER 2016		
Universal Loudspeaker Protector	01110151	£12.88
9-Channel Infrared Remote Control	15108151	£16.42
Revised USB Charger	18107152	£5.36
NOVEMBER 2016		
Fingerprint Access Controller – Main Board	03109151	£12.88
Fingerprint Access Controller – Switch Board	03108152	
OCTOBER 2016		
Arduino-Based USB Electrocardiogram	07108151	£9.79
100W Switchmode/Linear Bench Supply – Part 2	18104141	£20.83
SEPTEMBER 2016		
LED Party Strobe	16101141	£9.80
Speedo Corrector	05109131	£12.00
AUGUST 2016		
Low-cost Resistance Reference	04108151	£5.36
USB Power Monitor	04109121	£12.00
JULY 2016		
Driveway Monitor – Detector Unit	15105151	£11.80
Driveway Monitor – Receiver Unit	15105152	£7.50
USB Charging Points	18107151	£5.00
JUNE 2016		
Infrasound Snooper	04104151	£7.50
Audio Signal Injector and Tracer	04106151	£9.64
Audio Signal Injector and Tracer – Demodulator Board	04106152	£5.36
Audio Signal Injector and Tracer – Shield Board	04106153	£7.48
Champion Preamp	01109121/22	£8.29

PROJECT	CODE	PRICE
MAY 2016		
2-Channel Balanced Input Attenuator for Audio Analysers and Digital Scopes – Main Board	04105151	£16.40
Analysers and Digital Scopes – Front Panel	04105152	£20.75
Analysers and Digital Scopes – Rear Panel	04105153	
Appliance Earth Leakage Tester – Main Board	04203151	£16.40
Appliance Earth Leakage Tester – Insulation Board	04203152	
Appliance Earth Leakage Tester – Front Panel	04203153	£16.40
4-Output Universal Voltage Regulator	18105151	£7.50
APRIL 2016		
Appliance Insulation Tester	04103151	£11.80
Appliance Insulation Tester – Front Panel	04103152	£11.80
Low Frequency Distortion Analyser	04104151	£7.50
FEBRUARY 2016		
Spark Energy Meter – Main Board	05101151	£20.75
Spark Energy Meter – Zener Diode Board	05101152	£7.50
Spark Energy Meter – Calibrator Board	05101153	
JANUARY 2016		
Isolating High Voltage Probe For Oscilloscopes	04108141	£11.80
The Currawong – Part 3 – Remote Control Board	01111144	£6.95

PE/EPE PCB SERVICE

Order Code	Project	Quantity	Price

Name ....

Address ....

....

Tel ....

Email ....

I enclose payment of £ .... (cheque/PO in **£ sterling only**)

payable to: **Practical Electronics**

Card No ....

Valid From ....Expiry Date ....

Card Security No ....

You can also order PCBs by phone, Fax, Email or via the shop on our website on a secure server: www.epemag.com

No need to cut your issue – a copy of this form is just as good!

All prices include VAT and UK p&p. Add £4 per *project* for post to Europe; £5 per *project* outside Europe.

Orders and payment should be sent to:

Practical Electronics, Electron Publishing Ltd
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU
Tel 01202 880299 Email: shop@electronpublishing.com
On-line Shop: www.epemag.com

Cheques should be made payable to 'Practical Electronics' **(Payment in £ sterling only)**.

NOTE: Most boards are in stock and sent within seven days of receipt of order, please allow up to 28 days delivery if we need to restock.

ETI BUNDLE (1) Teach-In 3, 4 and 5 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 3 – CD-ROM

Mike & Richard Tooley

The three sections of the *Teach-In 3* CD-ROM cover a huge range of subjects that will interest everyone involved in electronics – from newcomers to the hobby and students to experienced constructors and professionals.

The first section (80 pages) is dedicated to *Circuit Surgery*, *EPE/PE's* regular clinic dealing with readers' queries on circuit design problems – from voltage regulation to using SPICE circuit simulation software.

The second section – *Practically Speaking* – covers hands-on aspects of electronics construction. Again, a whole range of subjects, from soldering to avoiding problems with static electricity and identifying components is covered. Finally, our collection of *Ingenuity Unlimited* circuits provides over 40 circuit designs submitted by readers.

The CD-ROM also contains the complete *Electronics Teach-In 1* book, which provides a broad-based introduction to electronics in PDF form, plus interactive quizzes to test your knowledge and TINA circuit simulation software (a limited version – plus a specially written TINA Tutorial).

The *Teach-In 1* series covers everything from electric current through to microprocessors and microcontrollers, and each part includes demonstration circuits to build on breadboards or to simulate on your PC.

ELECTRONICS TEACH-IN 4 – CD-ROM
A BROAD-BASED INTRODUCTION TO ELECTRONICS

Mike & Richard Tooley

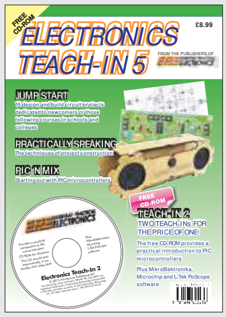
The *Teach-In 4* CD-ROM covers three of the most important electronics units that are currently studied in many schools and colleges. These include, Edexcel BTEC level 2 awards and the electronics units of the Diploma in Engineering, Level 2.

The CD-ROM also contains the full *Modern Electronics Manual*, worth £29.95. **The Manual** contains over 800 pages of electronics theory, projects, data, assembly instructions and web links.

A package of exceptional value that will appeal to anyone interested in learning about electronics – hobbyists, students or professionals.

ELECTRONICS TEACH-IN 5 – CD-ROM
JUMP START

Mike & Richard Tooley



15 design and build circuit projects for newcomers or those following courses in school and colleges.

The projects are: ■ Moisture Detector ■ Quiz Machine ■ Battery Voltage Checker ■ Solar-Powered Charger ■ Versatile Theft Alarm ■ Spooky Circuits ■ Frost Alarm ■ Mini Christmas Lights ■ iPod Speaker ■ Logic Probe ■ DC Motor Controller ■ Egg Timer ■ Signal Injector Probe ■ Simple Radio Receiver ■ Temperature Alarm.

PLUS

PIC'n' Mix – starting out with the popular range of PIC microcontrollers and *Practically Speaking* – tips and techniques for project construction.

The CD-ROM also contains:

■ Complete *Teach-In 2* book, a practical introduction to PIC microprocessors

■ MikroElektronika, Microchip and L-Tek PoScope software.

Three Teach-ins for the great price of £18.95

PLUS you also get the contents of the free CD-ROM from each issue...

... so that's another TWO Teach-Ins and The Full Modern Electronics Manual!

What a Bargain!!

ETI BUNDLE (2) Teach-In 6, 7 and 8 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 6 – CD-ROM
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

Mike & Richard Tooley

Teach-In 6 contains an exciting series of articles that provides a complete introduction to the Raspberry Pi, the low cost computer that has taken the education and computing world by storm.

This latest book in our *Teach-In* series will appeal to electronics enthusiasts and computer buffs who want to get to grips with the Raspberry Pi.

Teach-In 6 is for anyone searching for ideas to use their Pi, or who has an idea for a project but doesn't know how to turn it into reality. This book will prove invaluable for anyone fascinated by the revolutionary Pi. It covers:

- ■ Pi programming
- ■ Pi hardware
- ■ Pi communications
- ■ Pi Projects
- ■ Pi Class
- ■ Python Quickstart
- ■ Pi World
- ■ ...and much more!

The *Teach-In 6* CD-ROM also contains all the necessary software for the series, so that readers and circuit designers can get started quickly and easily with the projects and ideas covered.

ELECTRONICS TEACH-IN 7 – CD-ROM
DISCRETE LINEAR CIRCUIT DESIGN

Mike & Richard Tooley

Teach-In 7 is a complete introduction to the design of analogue electronic circuits. It is ideal for everyone interested in electronics as a hobby and for those studying technology at schools and colleges. The CD-ROM also contains all the circuit software for the course, plus demo CAD software for use with the *Teach-In* series.

- ■ Discrete Linear Circuit Design
- ■ Understand linear circuit design
- ■ Learn with 'TINA' – modern CAD software
- ■ Design simple, but elegant circuits
- ■ Five projects to build:
  - i) Pre-amp
  - ii) Headphone Amp
  - iii) Tone Control
  - iv) VU-meter
  - v) High Performance Audio Power Amp.

PLUS

Audio Out – an analogue expert's take on specialist circuits

Practically Speaking – the techniques of project building.

ELECTRONICS TEACH-IN 8 – CD-ROM
INTRODUCING THE ARDUINO

Mike & Richard Tooley

Hardware: learn about components and circuits
Programming: powerful integrated development system
Microcontrollers: understand control operations
Communications: connect to PCs and other Arduinos.

Teach-In 8 is an exciting series designed for electronics enthusiasts who want to get to grips with the inexpensive, popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. It will provide a one-stop source of ideas and practical information.

The Arduino offers a truly effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle through wireless or the Internet.

Teach-In 8 is based around a series of practical projects with plenty of information for customisation.

This book also includes *PIC n' Mix*: 'PICs and the PICkit 3 – A Beginners guide' by Mike O'Keefe and *Circuit Surgery* by Ian Bell – 'State Machines part 1 and 2'.

The CD-ROM includes the files for:

- ■ *Teach-In 8*
- ■ Microchip MPLAB IDE XC8 8-bit compiler
- ■ PICkit 3 User Guide
- ■ Lab-Nation Smartscope software.

ORDER YOUR BUNDLE TODAY!
JUST CALL 01202 880299 – OR VISIT www.epemag.com

CLASSIFIED ADVERTISING

Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price then our classified page offers excellent value. The rate for semi-display space is £10 (+VAT) per centimetre high, with a minimum height of 2·5cm. All semi-display adverts have a width of 5.5cm. The prepaid rate for classified adverts is 40p (+VAT) per word (minimum 12 words).

Cheques are made payable to 'Practical Electronics'. **VAT must be added**. Advertisements with remittance should be sent to: Practical Electronics, 113 Lynwood Drive, Wimborne, Dorset, BH21 1UU.

Tel 01202 880299 Email: stewart.kearn@wimborne.co.uk

For rates and further information on display and classified advertising please contact our Advertisement Manager, Stewart Kearn – see below.

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for the last twenty-seven years.

MISCELLANEOUS

ADVERTISING INDEX

BRUNNING SOFTWARE . . . 3

CRICKLEWOOD ELECTRONICS . . . 61

ESR ELECTRONIC COMPONENTS . . . 9

HAMMOND ELECTRONICS Ltd . . . 8

JPG ELECTRONICS . . . 80

MICROCHIP . . . Cover (ii), Cover (iii), 5

PEAK ELECTRONIC DESIGN . . . Cover (iv)

PICO TECHNOLOGY . . . 11

POLABS D.O.O. . . . 31

QUASAR ELECTRONICS . . . 2

STEWART OF READING . . . 9

TAG-CONNECT . . . 9

Advertisement offices

Electron Publishing Ltd
113 Lynwood Drive
Merley
Wimborne,
Dorset BH21 1UU

Tel 01202 880299

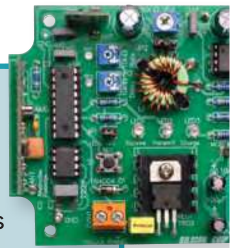
Fax 01202 843233

Email stewart.kearn@wimborne.co.uk

Web www.epemag.com

For editorial contact details see page 7.

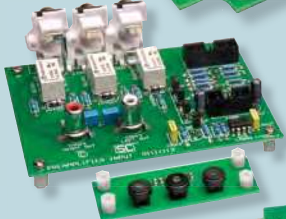
Next Month – in the May issue



433MHz Wireless Data Range Extender
Many remote-control devices rely on a 433MHz data link – you may have one and not even realise it, perhaps an alarm or garage door controller. Is the range a bit less than you'd like? Perhaps the remote unit is too far away or are there trees in the way? Here's the answer: a small, solar-powered repeater for added range.

Bridge-mode Audio Amplifier Adaptor
Want to unlock more power from an audio amplifier and speaker? You can combine this easy-to-build unit with a standard stereo amplifier to deliver the power you need into a single 8Ω speaker.

Ultra-low-distortion Preamplifier with Tone Controls – Part 2
We've introduced our state-of-the-art stereo *Preamplifier*, with its almost unmeasurable noise and distortion – typically 0.0003% THD+N! Now we'll turn to the remote-controlled input selection boards and power supply.



iCEstick VGA Terminal
Want to give a project the retro-computer look? Or do you need a convenient way to display a screen full of text on a low-cost monitor? If so, then this nifty project is for you! It works with a low-cost iCEstick FPGA development board.

Visual programming with XOD
Build a programmable rate-of-change switch module that monitors an input voltage of 0-5V and activates an output when the input changes at a rate that exceeds a pre-set level. We've found lots of uses for it!

PLUS!
All your favourite regular columns from *Audio Out, Cool Beans and Circuit Surgery,* to *Electronic Building Blocks*, *PIC n' Mix* and *Net Work*.

On sale 2 April 2020

Content may be subject to change

NEW subscriptions hotline!

Practical Electronics

We have changed the way we sell and renew subscriptions. We now use 'Select Publisher Services' for all print subscriptions – to start a new subscription or renew an existing one you have three choices:

1. Call our NEW print subscription hotline: 01202 087631, or email: pesubs@selectps.com
2. Visit our online shop at: www.epemag.com
3. Send a cheque (payable to: 'Practical Electronics') with your details to:

   Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Remember, we print the date of the last issue of your current subscription in a box on the address sheet that comes with your copy.

Digital subscribers, please continue to call 01202 880299 or visit: www.epemag.com

Published on approximately the first Thursday of each month by Electron Publishing Limited, 1 Buckingham Road, Brighton, East Sussex BN1 3RA. Printed in England by Acorn Web Offset Ltd., Normanton WF6 1TW. Distributed by Seymour, 86 Newman St., London W1T 3EX. Subscriptions UK: £26.99 (6 months); £49.85 (12 months); £94.99 (2 years). EUROPE: airmail service, £30.99 (6 months); £57.99 (12 months); £109.99 (2 years). REST OF THE WORLD: airmail service, £37.99 (6 months); £70.99 (12 months); £135.99 (2 years). Payments payable to 'Practical Electronics', Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom. Email: pesubs@selectps.com. PRACTICAL ELECTRONICS is sold subject to the following conditions, namely that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price shown on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter whatsoever.

Faster Programming, Wider Voltage Ranges and Enhanced Interface Options

PICkit™ 4 enables faster design development

With five times faster programming and a wider 1.2V to 5V range, the Microchip MPLAB® PICkit™ 4 Development Tool supports low-cost development of even more Microchip controllers.

PICkit™ 4 also introduces improved USB connectivity with a hi-speed USB 2.0 interface and advanced debugging over 4-wire JTAG and serial wire debug with streaming data gateway in addition to legacy interfaces.

The new advanced interfaces support in-circuit programming and debugging of Microchip's CEC1702 hardware cryptography-enabled MCUs, in addition to dsPIC® Digital Signal Controllers and PIC® microcontrollers from 8- to 32-bit.

MICROCHIP

www.microchip.com/PICkit4eu