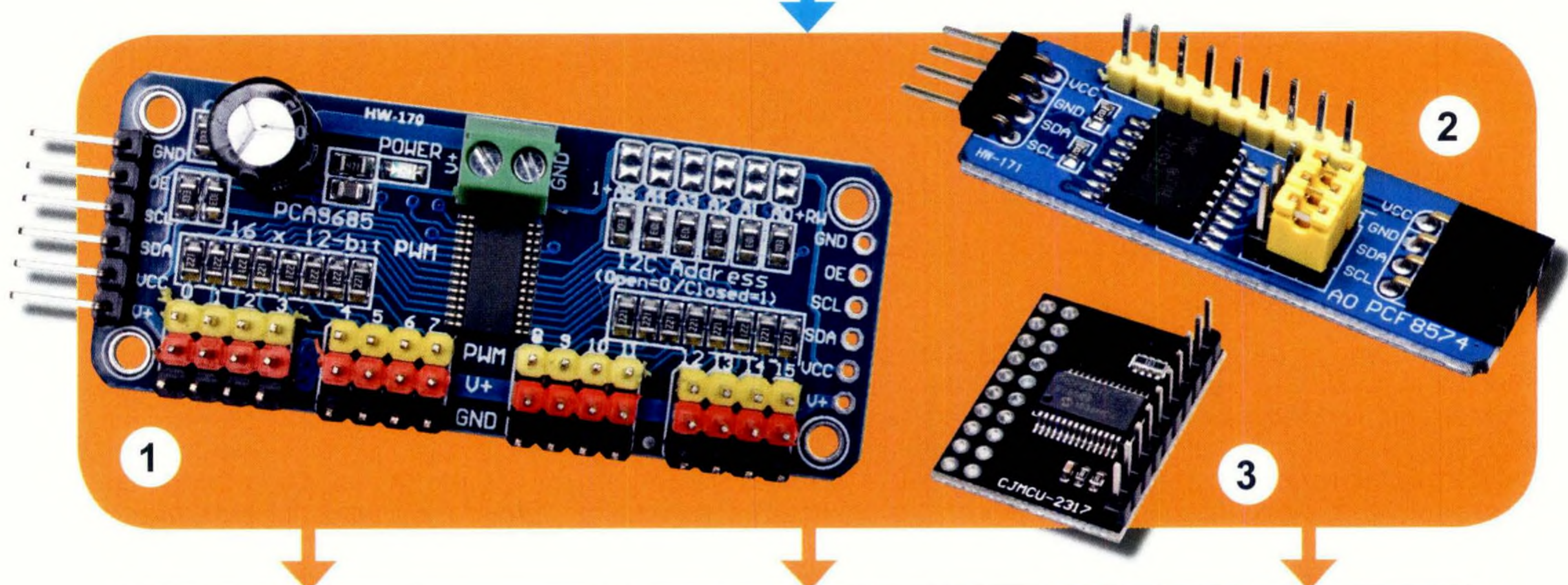


I/O Expander Modules



Sometimes, when working with microcontrollers, you just don't have enough pins to do what needs to be done. You might have started with the idea of a simple design but later found out that you had forgotten some crucial features. Uh oh! It can be a lot of work to change to a bigger, more expensive microcontroller, possibly involving learning some new soldering or programming skills. But there's another way out of this pickle.

by Tim Blythman

If you've been working with microcontrollers for long, you've almost certainly run into the situation where you don't have enough I/O pins to do what you need to. Or you've known in advance that you don't have enough pins, but for whatever reason, you don't want to switch to a bigger part. It can be a conundrum.

The ideal solution is to use an I/O expander module. In this article, we describe three different expander modules. They are all controlled over an I²C serial bus, so at worst, they take up two pins on your micro. If you're already using the I²C bus for other purposes, they won't use up any more pins at all.

That's the great thing about I²C; the addressing scheme means that over 100 devices can be controlled by just two lines. Many microcontroller plat-

forms (including Arduino and Micro-mite BASIC) include native support for I²C.

And all three of the modules we present have the option to change the device address, so multiple expanders can be connected using the same bus. I/O pin counts in the hundreds are easily achievable by using enough of these modules.

The three modules we describe here have a variety of different features, so they have different strengths. We'll describe them according to the IC that they are based around; in each case, the IC data sheet is a great resource to help you fully understand each module's capabilities and quirks.

While it's possible to use the bare ICs in your designs, they are all quite small, so by using a module, you not only save the effort of having to solder

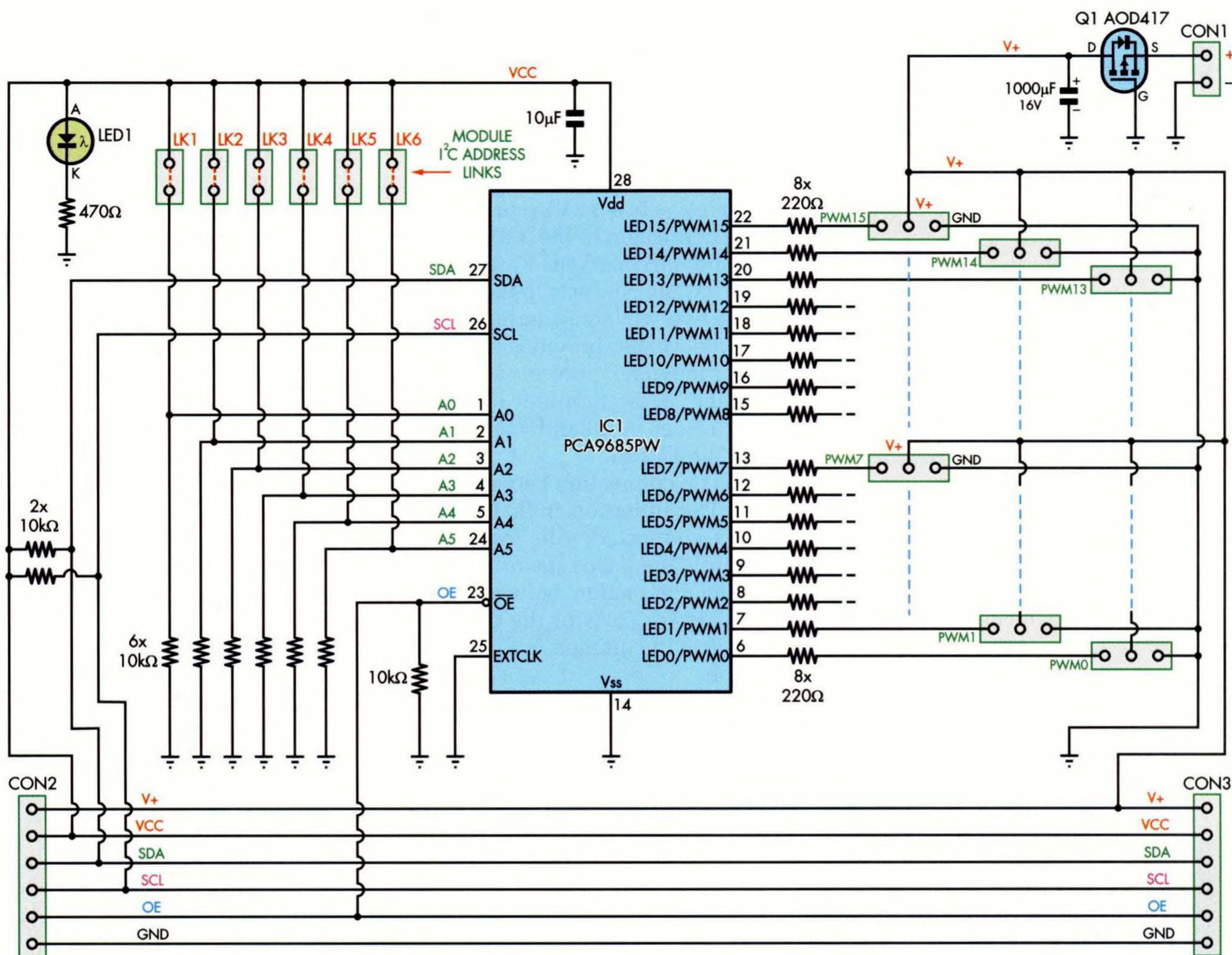
them, you also get all the other necessary support components along with handy headers to make connecting to other devices a cinch.

Expander 1: PCA9685 module

This module provides up to 16 pulse-width modulated (PWM) or standard digital outputs, which can be used for various purposes including controlling LED brightness or stepper motors.

The PCA9685 module measures 63 x 25mm and features two six-way headers for control, plus 12 three-way I/O headers arranged in groups of four. There's also a two-way screw terminal for power and six pairs of pads which can be bridged to change the IC's I²C address.

The original version of this board was designed by the Adafruit com-



SC
©2019

PCA9685-BASED 16 X 12-BIT PWM EXPANSION MODULE

Fig.1: the circuit of the PCA9685 module as designed by Adafruit. Some variants/clones use different resistor values (eg, 120Ω instead of 470Ω, meaning the power LED is very bright) or omit the reverse polarity protection Mosfet or the electrolytic bypass capacitor.

pany, but has been cloned and is also available from several different online stores too. The circuit diagram of the original Adafruit version is shown in Fig.1.

We sourced a few variants of this board, and found that there were a few variations, including one that omitted the reverse polarity protection and one that used different resistor values. Another lacked the bypass capacitor. But they all did pretty much the same job.

The PCA9685 IC

This IC is manufactured by NXP, and the data sheet can be found at: siliconchip.com.au/link/aas1

It runs from 2.3-5.5V, so can work with both 5V Arduinos and 3.3V Micromites, as well as the increasing number of 3.3V Arduinos.

It comes in a 28-pin SSOP or QFN package (both SMD). While it's possible to hand-solder chips this small, we find it easier to use the module if we have enough space to mount it.

While originally intended to be a LED PWM driver, Adafruit sells their PCA9685 board as a servo motor driver. Its 16 PWM channels can operate at up to 1500Hz with 12 bits of resolution (4096 steps), which is more than good enough to generate servo control pulses.

The three-wide rows of pin headers allow many standard servo motors to plug directly into the board. At 50Hz (20ms between the pulses, as in a typical servo signal), pulses can be generated with a resolution of around 5μs, giving just over 200 steps between the standard servo pulse width limits of 1ms and 2ms.

With these normally corresponding to positions of 0° and 180°, this gives a mechanical resolution of slightly better than 1°.

One interesting feature which may come in useful is that the PWM outputs can be started at different times, giving them different phases throughout the PWM cycle, although all outputs must run at the same frequency.

So for example, if you are driving multiple LEDs at less than full duty, they can be timed to stagger their switch-on times, such that (for example) only one is switched on at a time. This will limit the current steps drawn from the supply and probably reduce EMI too.

With the addition of a high-current buffer (eg, a Darlington array), this board could even be used to drive a stepper motor or brushless DC motor.

By staggering the phases and changing the frequency, the output of the PCA9685 can be set to produce a pulse train sufficient to allow the motor to keep turning without further intervention.

We tested out some possible approaches to generate motor drive signals with this module, and some examples of the waveforms we came up with are shown in oscilloscope grab Scope 1.

Module description

Apart from the 16 sets of output pins (each output is paired with a dedicated GND and power pin), there are also headers for power and I²C bus connections as well as six solder jumpers to allow the address to be set.

An output enable (OE) pin is also broken out on the board, allowing all outputs to be enabled or disabled with a single signal, but an external clock connection is not provided. The module relies on the chip's internal 25MHz oscillator instead. The external clock pin is grounded as per the

data sheet's recommendation for when it is not used.

Referring to the circuit diagram in Fig.1, we see that there are two different supply rails on the board. A nominal 5V rail powers the chip and can be found on the six-way headers at the pin marked V_{CC}. In a 3.3V system, this would be connected to the 3.3V rail.

A second rail marked V₊ is also available at the six-way header as well as the two-way screw terminal. A diode-wired Mosfet provides reverse polarity protection if power is fed into V₊ from the screw terminal but not from the header. A 1000µF capacitor bypasses the V₊ rail.

There is no connection between V₊ and V_{CC}. The intention is that servo motors (if connected) run from the V₊ rail, while the logic runs from V_{CC}, minimising interaction between the logic and power parts of the circuit. All they have in common is a ground connection.

A separate bypass capacitor for the IC and the power indicator LED is also fed from V_{CC}. Apart from the external clock pin, all the IC's pins are broken out.

The six address pins (A0-A5) are normally pulled to ground by 10kΩ resistors, but they can be individually pulled high if the associated solder jumper is bridged.

While this might appear to give up to 64 available addresses, due to I²C reserved addresses and auxiliary addresses for the PCA9685, the actual usable number is 55, using the (7-bit) range 64-119, excepting 112.

By default, with no jumpers set, the board has a 7-bit address of 64 or hexadecimal 0x40. The six jumpers effectively set the value of the six low order address bits.

Address 112 is designated as "All Call" and can be used to address any PCA9685 device regardless of its set address. This allows initialisation of a large number of ICs to occur quickly, by setting all attached devices to the same initial conditions.

During initialisation (or at any other time), the outputs can be set to open-drain (pull low or high-impedance), push-pull or inverted push-pull configurations.

The 16 PWM outputs are brought out to the top (yellow) row of pins on the board, where they are combined with a row of V₊ (red) and GND (black) headers to form a row of servo motor compatible connection points.

The OE (output enable) pin is brought out to the six-way headers but is also pulled to GND by a 10kΩ resistor, so the outputs are enabled by default. This line can be pulled up by a host micro to shut down the outputs if necessary.

The I²C SDA and SCL pins are also brought out to the six-way headers and these have 10kΩ pull-up resistors. While this is higher than the recommended 4.7kΩ value for I²C bus lines, we had no trouble without adding external pull-ups. Later, we will look at how these resistors behave when multiple boards are connected.

Cleverly, the two six-way headers have matching pin-outs, so boards can be stacked end to end, for example, by fitting a female header to one end and a male

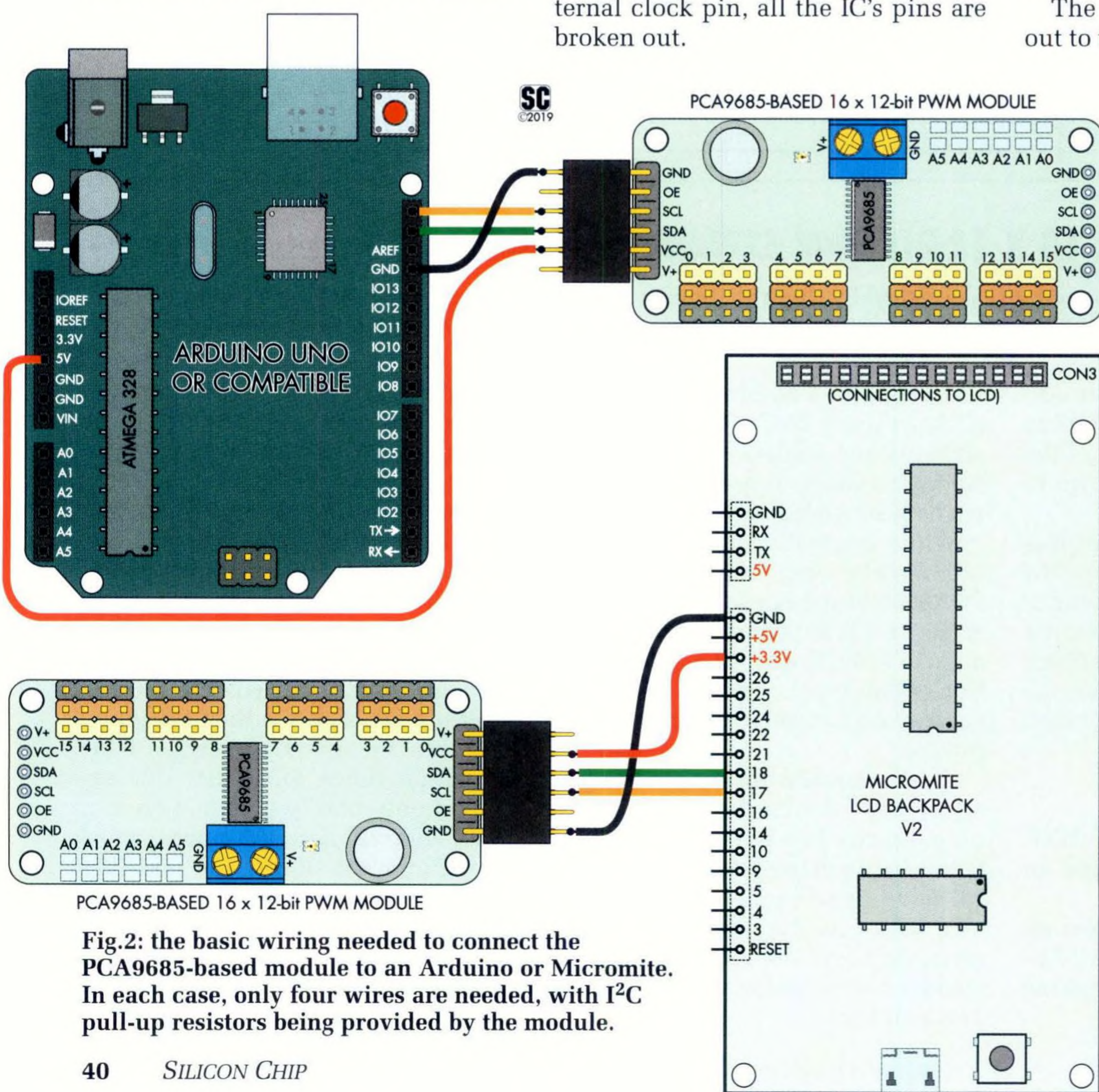
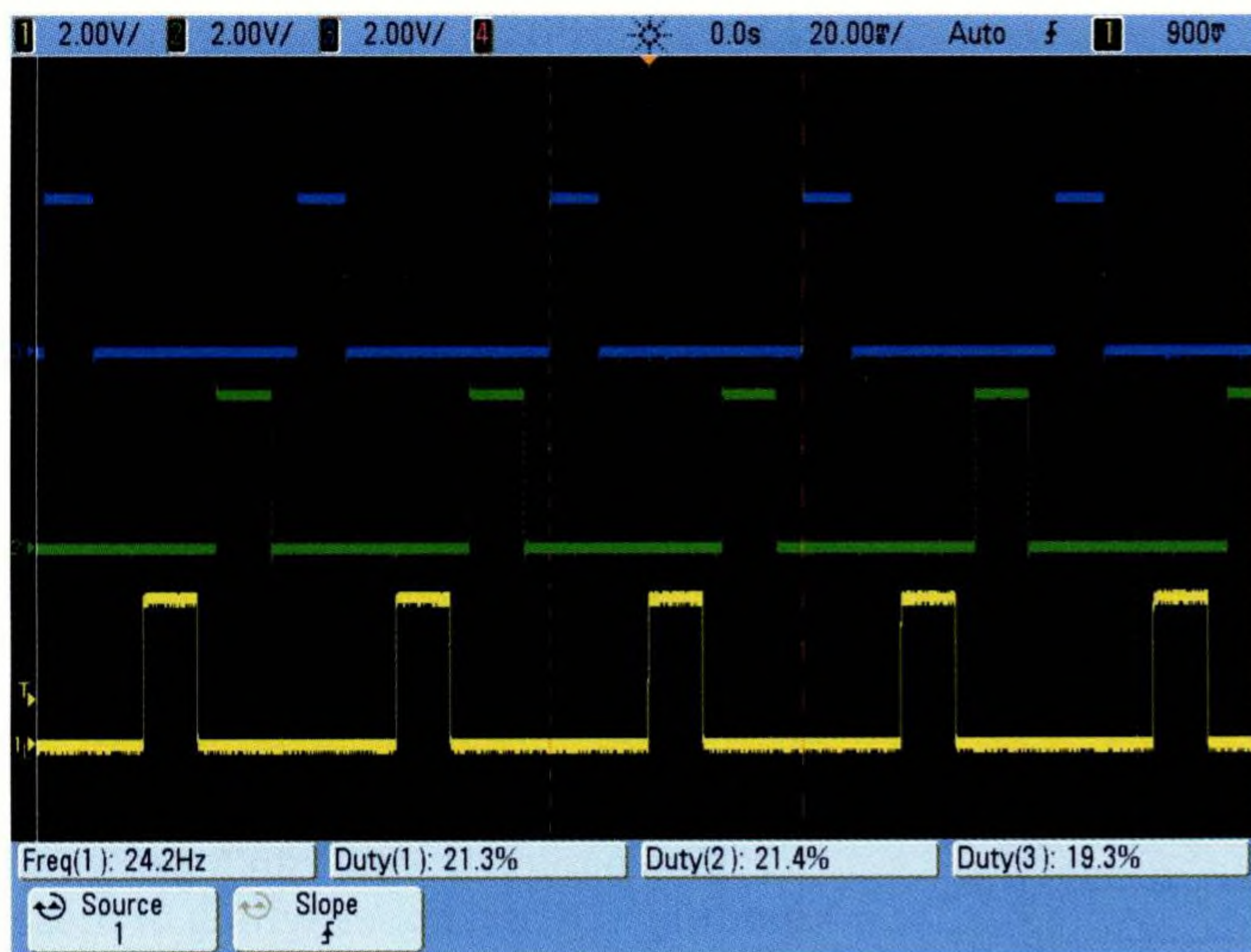
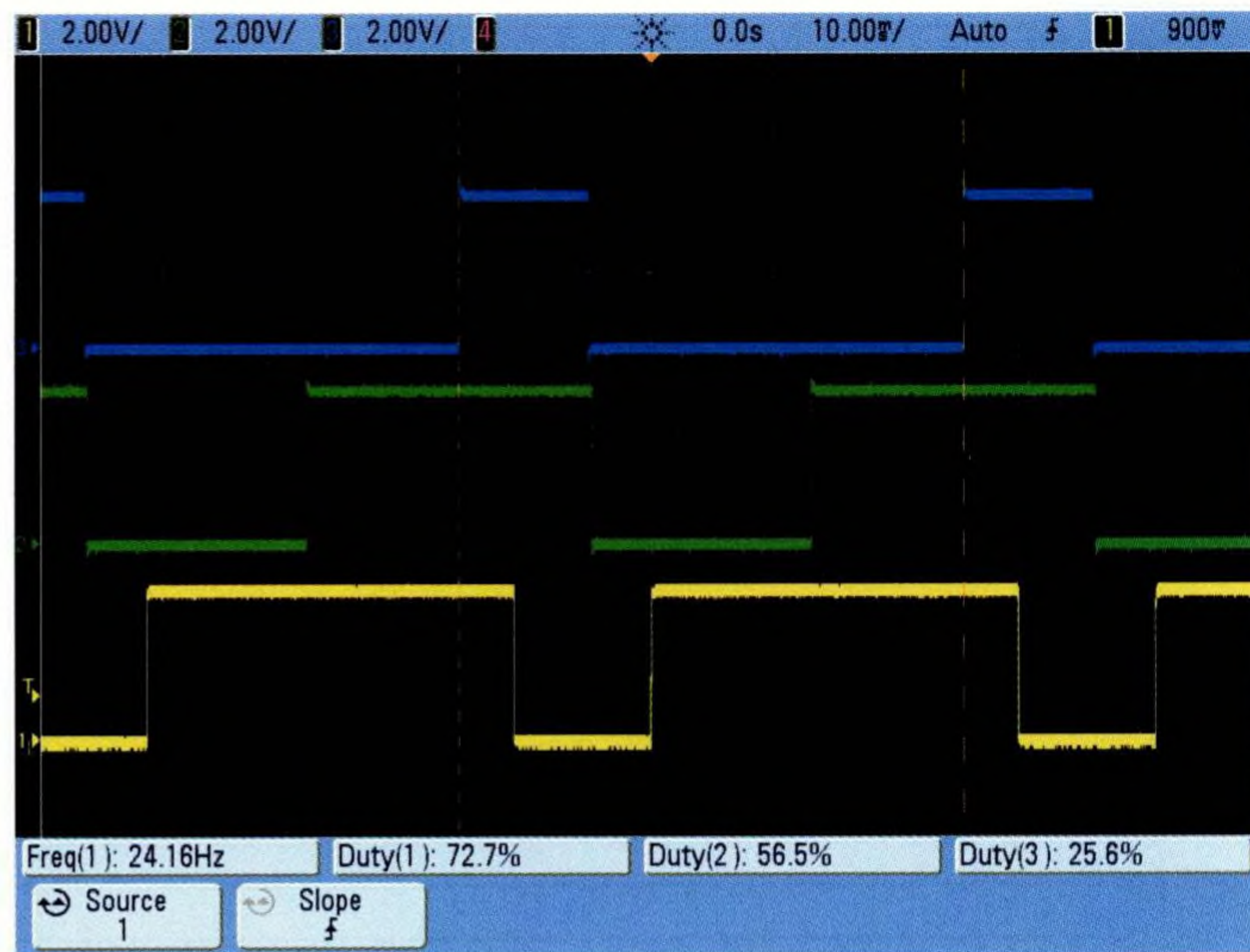


Fig.2: the basic wiring needed to connect the PCA9685-based module to an Arduino or Micromite. In each case, only four wires are needed, with I²C pull-up resistors being provided by the module.



Scope 1: here we're using the PCA9685 module to generate pulse trains each phase shifted by approximately 120° compared to the last. Waveform like this could be used to drive a brushless motor or spread out the current demand of multiple PWM loads.



Scope 2: this demonstrates using the PCA9685 module to produce three different PWM waveforms with different rise and fall positions, with each duty cycle being fully adjustable. The main restriction is that the repetition frequency of all outputs must be the same.

jumper to the other. The V_+ track is quite thick, and the GND trace consists of a solid copper pour on the back of the PCB, so passing a fair amount of current between boards is possible.

It appears the board is quite well designed and breaks out practically all the useful features of the PCA9685 IC.

What needs to be connected?

For basic testing, only four wires are needed: V_{CC} , GND, SDA and SCL. If you wish to connect a servo motor to the headers, you will need a supply for the V_+ rail too. The basic connections for a Micromite and Arduino are shown in Fig.2.

Software

We have written sample programs for Arduino and Micromite. Both of these allow the PWM frequency to be set, as well as the start and duration times of the pulses.

Internally, the PCA9685 uses start and end variables to define the pulse parameters of each output, as well as specific bits to enable full-on and full-

off states, so some minor translation is done by the code.

In the Micromite example, these variables are set by sliders on an attached ILI9341 LCD (as you would have on a Micromite LCD Backpack), while the Arduino code uses the serial monitor as a menu to enter the parameters, these being a letter for the parameter followed by its value.

Both examples contain some functions to simplify writing your own code to control the module. Adafruit has also written an Arduino library which can be found at <https://github.com/adafruit/Adafruit-PWM-Servo-Driver-Library>

Our sample code is available as a free download from the SILICON CHIP website.

Expander 2: PCF8574 module

You may have heard of the PCF8574 before, especially if you have ever used any of the I²C-controlled character LCD panels, as described in our March 2017 article (siliconchip.com.au/Article/10584). It is the PCF8574 that provides the I²C-to-parallel con-

version that makes it so easy to use these LCD screens.

The module we are looking at, designated HW-171, measures 48 x 11mm, although other similar modules are available. Its circuit diagram is shown in Fig.3. It has a wide operating voltage range, 2.5-6V, making it suitable for all 3.3V and 5V applications.

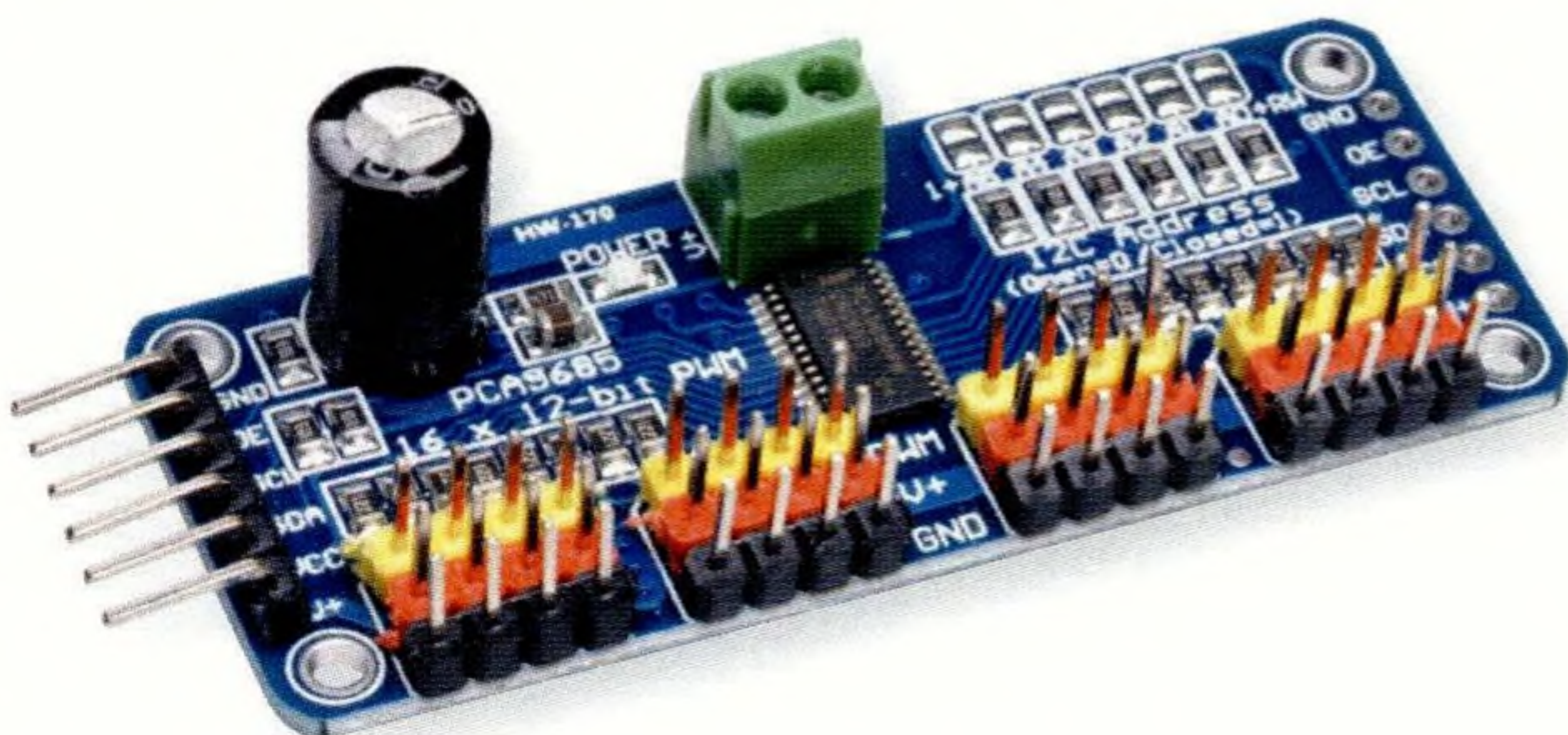
The I²C modules designed to attach to the back of an LCD panel can also be used as I/O expanders, although they usually omit one of the pins as only seven control lines are needed for driving a character LCD.

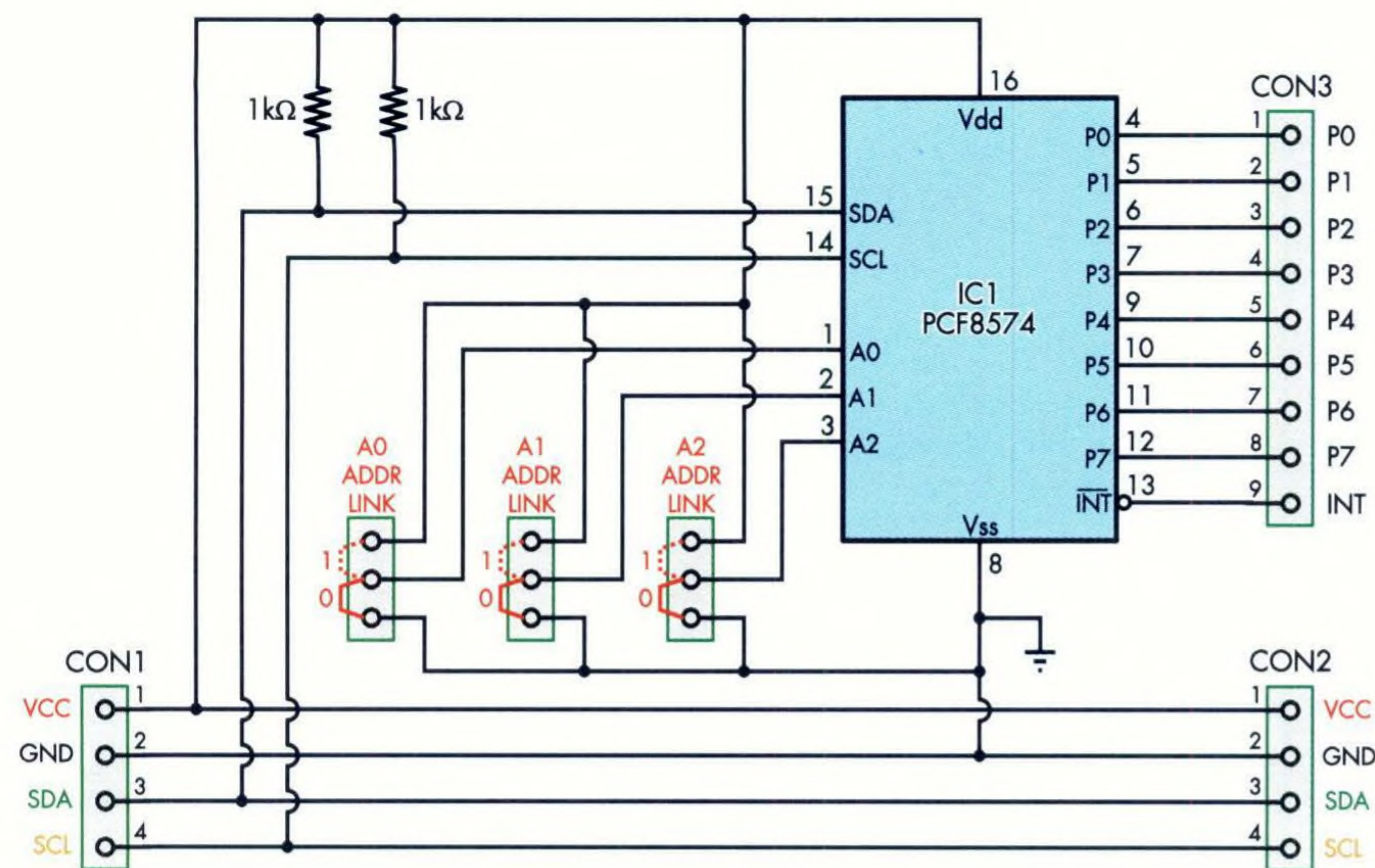
This module has a simple interface, with a four-pin male header at one end and a four-pin female header at the other end for control and daisy chaining. The pins are designated V_{CC} , GND, SDA and SCL, with the last two being the I²C bus. A nine-way header breaks out the I/O ports on one side (the ninth pin provides an interrupt function), while a row of three three-pin headers with jumper shunts are used for address selection.

The male/female pin header combination allows multiple modules to be easily connected to the same I²C bus, and the addressing scheme allows up to eight unique addresses.

Apart from the main IC, the only other electronic components on the module are a pair of 1k Ω pull-up resistors on the I²C lines. These are much lower values than are typically used as I²C pull-ups, but it still seems to work OK. We'll have a look at the effects of these resistors a bit later.

The PCA9685-based module is one of the better designed I/O expander modules. Practically all the available pins are broken out, with the control pins replicated at each end, to allow multiple modules to be daisy-chained.





SC ©2019 PCF8574-BASED I/O EXPANSION MODULE

Fig.3: the circuit of the PCF8574-based module. Apart from the main IC, there are just two extra resistors. It's a great module in that all the useful pins are broken out in a well laid out arrangement.

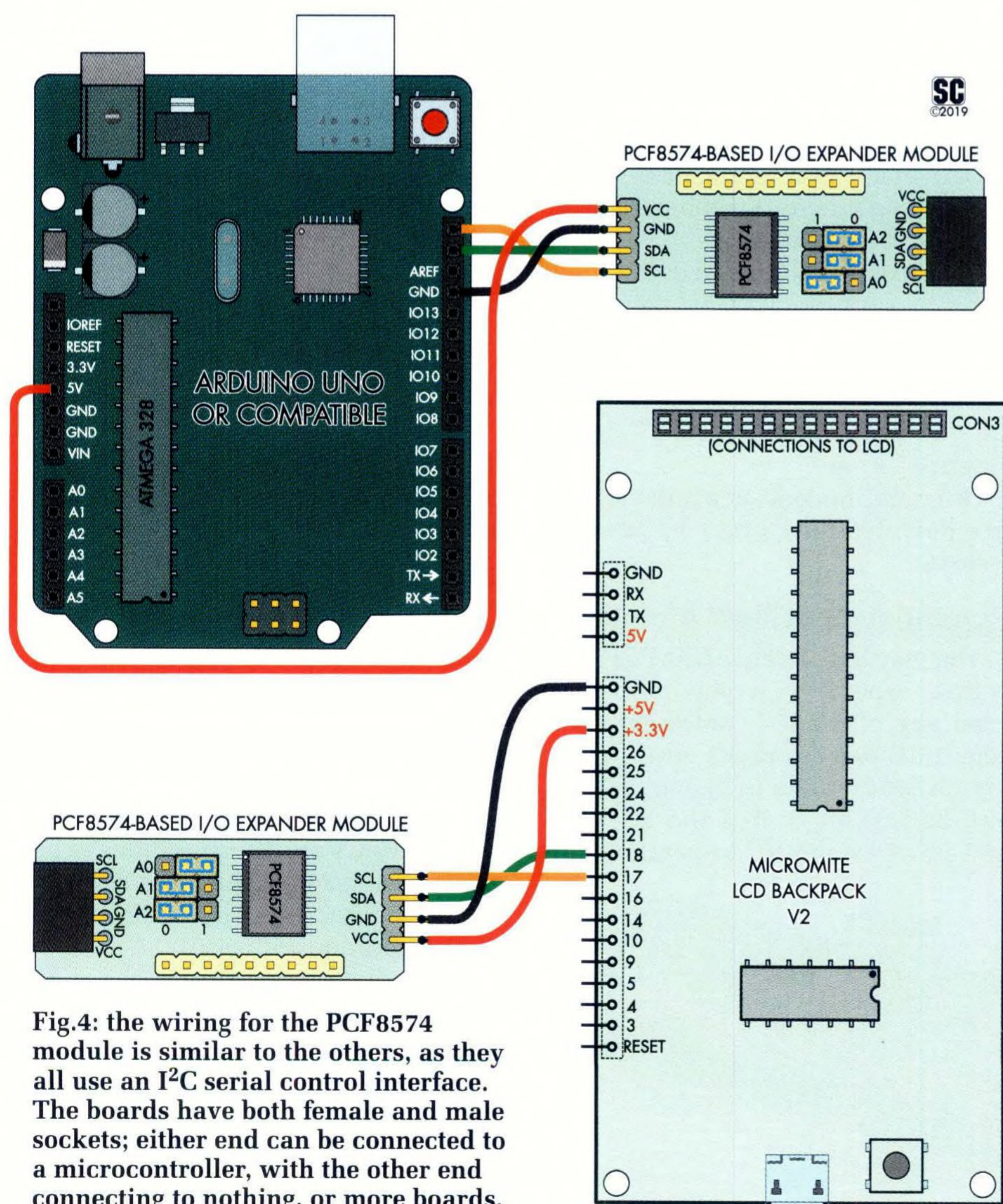


Fig.4: the wiring for the PCF8574 module is similar to the others, as they all use an I²C serial control interface. The boards have both female and male sockets; either end can be connected to a microcontroller, with the other end connecting to nothing, or more boards.

The PCF8574 IC

Like the PCA9685, the PCF8574 is made by NXP. Its datasheet can be found at: siliconchip.com.au/link/aasm

While it can only have eight different addresses, there is a variant called the PCF8574A, which is identical but has a different set of addresses, giving 16 total possibilities.

The PCF8574 can have a 7-bit address from the range 32 to 39, while the PCF8574A can have an address from 56 to 63. Our units had a default address of 32. Since the chips are interchangeable, if you can't get your module to work, check which of these two chips it has.

While NXP does not make a DIP version of this IC, Texas Instruments does, so it is possible to replicate the functions of this module on a breadboard with the addition of two pull-up resistors for the I²C bus.

The datasheet mentions the PCF8574's suitability for driving LEDs, but unlike the PCA9685, this device is quite minimalist and so can only switch them on or off. But it does provide the ability to read the state of each pin, allowing them to be used as digital inputs, which the more complex PCA9685 does not.

Each of the eight I/O pins can be set to one of two states. The default power-up state is for the pins to be pulled up by a 100µA current source. In this state, the pin can be used as an input, detecting when a connected device pulls the pin low. The 100µA current source is also sufficient to drive a logic pin high, such as when the PCF8574 is used to drive alphanumeric LCD screens.

The other state is to pull the pin low. Each pin can sink up to 10mA. A brief 1mA pull-up current is applied on a transition from low to high, supplementing the weak 100µA pull-up and speeding up transitions.

While this scheme appears very basic, it allows all the pins to be written and/or read with a single byte command. Since repeated reads or writes can occur during the same I²C transaction, complex wave trains can be generated as easily as port writes on a microcontroller.

This is perfect for controlling devices such as the character LCDs we mentioned earlier, as a stream of digital data is often needed to update a series of characters on the display.



The PCF8574 modules are designed to be stacked end-on-end, meaning that it's trivial to connect multiple such modules to a single microcontroller. Note that the address jumpers are set here to give each module a different I²C bus address, to avoid conflicts.

The interrupt pin is an open-drain active low output, and goes low on any change of input pin level. It is reset when a read occurs. It is intended to signal to the microcontroller that the input state(s) have changed and require reading. The interrupt pins of multiple modules can be paralleled, as any device can assert a low without conflicting with other modules.

With such a simple control scheme, no initialisation or command codes are needed; the data that is written or read corresponds precisely to the pin states.

Module description

The module itself is quite simple, as noted above, with only two resistors in addition to the main IC.

While the stackable feature of the modules is handy, it's a pity that the interrupt function is not brought out to a fifth pin at each end, to make it easy to feed this signal back to the controller.

V_{CC} and GND pins near the I/O pins would have been nice too; as it is, there is nowhere convenient to connect the controlled device to the power supply.

As for the other module, only four connections are needed: V_{CC}, GND, SDA and SCL. See Fig.4 for the recommended connections to either a Micromite or an Arduino.

Software

As for the PCA9685 module, we have created both an Arduino and Micromite example program. The Micromite program uses a touch panel interface, while the Arduino program uses a serial interface.

Entering any of the numbers 0-7 will toggle the state of that output pin. The pin states are also read and the current state displayed. A read can also be performed by pressing the "READ" button or entering "R" on the Arduino software.

To help with troubleshooting, we've found some small I²C scanner pro-

grams (for Arduino and Micromite) and included them in our software download for this article.

These scan all addresses on the I²C bus and determine which addresses are actually in use. That might help you figure out which address your module is set for, if you can't figure it out from the jumpers and IC code.

Expander 3: MCP23017/S17 module

The MCP23017 IC is produced by Microchip, the same company responsible for PIC microcontrollers. It has 16 bi-directional digital I/O ports and is controlled over an I²C bus. There is an SPI version, which is called the MCP23S17. The module suits either version of the IC, as some of the pins are marked with designators for both I²C and SPI signals.

The MCP23017 IC has a working range of 1.8-5.5V, so this module is suitable for use with both Micromites and Arduinos. It is quite compact, measuring just 25 x 20mm, although this means that it only has space to label the functions on the back of the module. It has 30 pins in total, although they do not come fitted with headers.

It supports full bi-directional I/O operation on all pins. The register set is reminiscent of a PIC microcontroller, with control bytes for direction, pull-ups, output latches, port reading and interrupt enable. There's also another byte which can be used to invert the polarity of the port.

Given this many registers, there's a greater level of control than for the PCF8574-based module, including full push-pull output drivers, although it lacks the PWM feature of the PCA9685.

Just like a PIC microcontroller, all the I/O pins start as inputs but can be set to be outputs. The commands are simple, and consist of the IC address (as for all I²C transactions) followed by a command (register) byte and data byte. Port writes up to eight bits wide

are possible. Its data sheet can be found at: siliconchip.com.au/link/aasn

Module description

There are two rows of ten pins at one end of the module with the connections to the controlled I/O ports (16 pins) plus connections for interrupt signals and power. There is another single row of 10 pins with the connection to the host for control and power; other non-I/O pins such as the address pins are broken out here too.

But the small size of the module means that some of the nicer features found on the other boards are omitted. For example, although the MCP23017 has three address pins to allow addressing up to eight modules, these pins aren't broken out to jumpers. To use them, you have to solder a wire from one or more of the address pins to the ground pin.

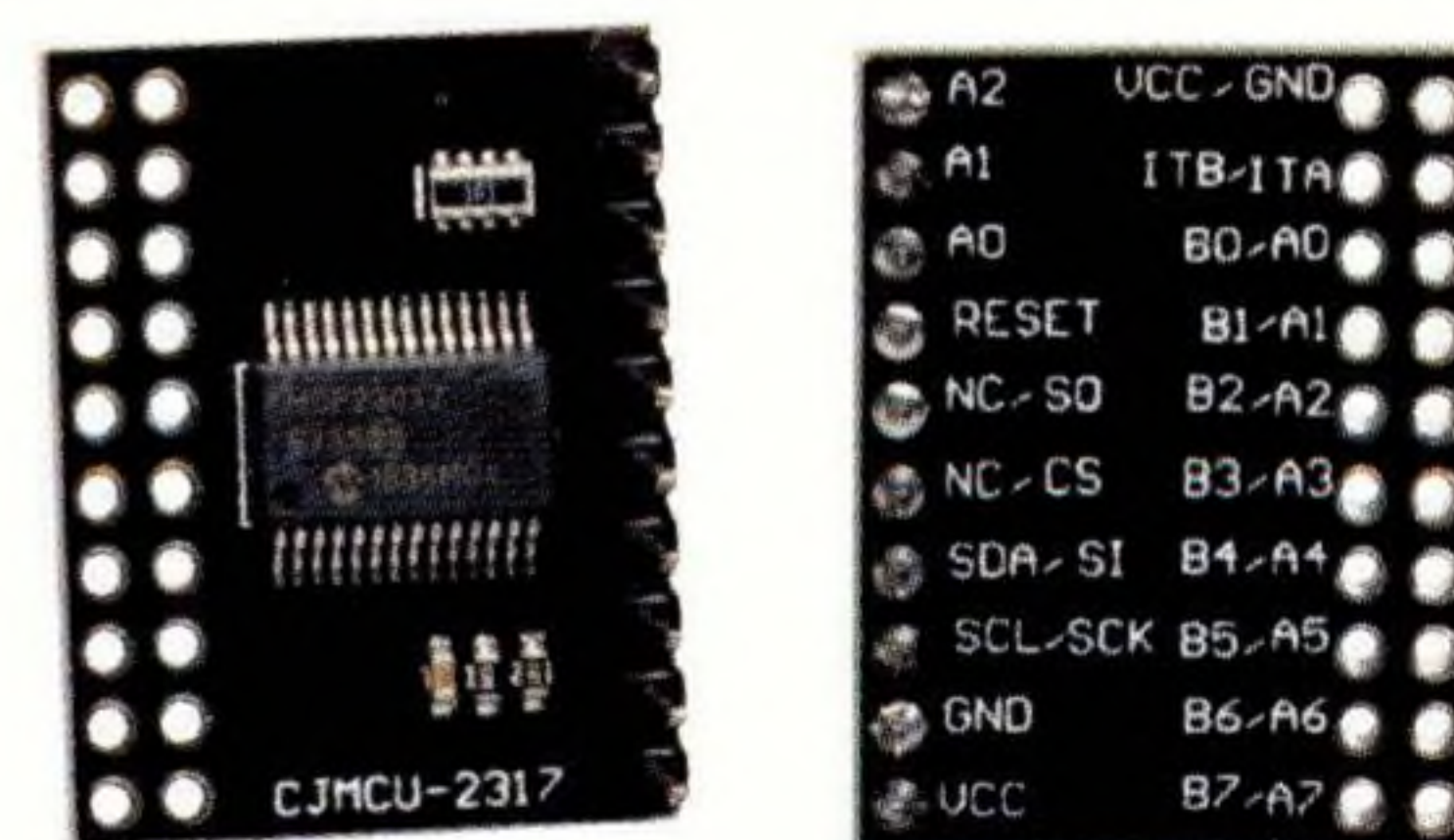
Similarly, there isn't a header to allow multiple modules to be easily stacked. So it's most easily used when it's the only expander module connected to the micro.

The fact that the two rows of output pins are adjacent means that the module does not lend itself well to being used on a breadboard, unless you're happy using just one row of the output pins.

The circuit

The circuit diagram for this module is shown in Fig.5. Apart from the main IC, there are two 10kΩ resistors, one four-way 10kΩ resistor array and a 100nF ceramic capacitor, used to bypass the IC's supply.

The two individual resistors are the I²C pull-ups, while the resistor array is connected to pull the RESET ($\overline{MR}$) pin high (so the chip will operate as soon as power is supplied) and the address pins low (setting the default address).



The MCP23017 module does not feature stackable headers or address jumpers, but it is very compact and provides full digital I/O control of 16 pins, similar to that of a microcontroller. Due to its small size, the pins are labelled on the back of the module.



44 SILICON CHIP

siliconchip.com.au

15mA being sunk from the 3.3V supply when the pins are driven low. This is the absolute maximum pin current of the PIC32, and even under these conditions, I²C communications at 400kHz (the Micromite's upper speed limit) worked flawlessly.

So a maximum of four PCF8574-based modules or 45 PCA9685-based modules can be connected to a Micromite, based on current draw on the I²C pins. This does not take into account extra capacitance which may be added to the bus lines when extra modules are added, so these numbers may not be achievable in practice.

Removing the resistors from some of the modules will decrease this load, as will adding a second I²C bus.

Similarly, the ATmega328 processor on an Arduino Uno supports a maximum of 40mA on each pin, which corresponds to 125Ω pull-ups to the 5V supply. So we tested using 150Ω pull-up resistors.

This too proved to work fine for both modules, suggesting up to six PCF8574-based modules or 66 PCA9685-based modules can be connected to an Arduino board. This includes the same assumptions as earlier, and these results may not be achievable in practice.

It appears that the I²C bus is quite robust, and can work well if it's operating slightly outside its recommended conditions.

Although we didn't run any tests on the MCP23017 based module, based on these results, it should work fine with up to eight modules (the maximum that would be addressable).

Level shifting

Another interesting possibility that arises in using I/O expander modules is that it allows for parts of the circuit to operate at different voltages.

As I²C is an open-collector bus, devices either pull the SCL and SDA lines to ground or let them rise to a higher voltage due to the pull-up resistors. So it isn't necessary for all devices on the bus to have an identical logic high voltage.

If the bus pull-ups are connected to the lowest voltage supply used, no damage can occur through over-voltage. As long as this level is detected as high by the device with the highest logic voltage, then it will still work, although with reduced margin in clearly defined logic levels.

It's important in this sort of situation to ensure that the pull-up resistors that are connected to the bus go only to the lower voltage supplies (although most chips have internal clamp diodes which will clamp the high voltage to a safe level anyway).

So for example, you could connect an I/O expander module running off 5V to a 3.3V Micromite and it should work just fine.

You would then have 3.3V I/Os available direct from the Micromite, and 5V I/Os from the expander. Ideally, the I²C pull-ups should go to the 3.3V supply.

Similarly, you could connect a 3.3V I/O expander to a 5V Arduino micro. In this case, you would want to use the pull-ups on the expander module. The Arduino will read 3.3V as a high level, and while it will have its own 5V I/Os, you can also use the 3.3V I/Os of the expander module to communicate with other devices running at 3.3V.

One of these expander modules may even be the easiest and cheapest way to communicate with a chip that has a digital interface operating at a different level to your micro.

Note though that the resulting I/O speeds will not be very high; this is another factor to be considered.

Summary

Each module described here provides quite a different set of features, so which one is best for you will depend on your needs. You may even find it handy to connect multiple different expander modules to a single micro to perform different jobs.

For PWM or servo control, or LED brightness control, the PCA9685 module is the most useful. Its large number of possible addresses is also a strength. But it doesn't provide you with any extra digital inputs.

The PCF8574 module is the simplest and easiest to use.

If you need more full-fledged microcontroller type I/O pins, then the MCP23017 module has the advantage. There is extra overhead in controlling it compared to the PCF8574, but this is offset by extra features and more I/O pins.

As mentioned above, you can mix and match the modules, although it is an unlucky coincidence that the MCP23017 and the PCF8574 both share the same address space. **SC**

Wagner

ELECTRONICS SUPER STORE

OVER 40 THOUSAND PRODUCTS
ONLINE 1500+ PAGE CATALOGUE
AUSTRALIA WIDE SHIPPING



AUDIO/VISUAL ACCESSORIES
SPEAKERS / PRO AUDIO
ELECTRONIC COMPONENTS
DATA & ELECTRICAL
TOOLS & TEST EQUIPMENT
BATTERIES & MORE!

BUY ONLINE AT
WAGNERONLINE.COM.AU

VISIT OUR SYDNEY STORE
84-90 PARRAMATTA ROAD,
SUMMER HILL, NSW 2130
02 9798 9233