



AN10814

Driving stepper motors using NXP I²C-bus GPIO expanders

Rev. 01 — 11 September 2009

Application note

Document information

Info	Content
Keywords	stepper, stepper motor, GPIO, push-pull, quasi-bidirectional, MOSFET, optical interrupter, Fast-mode Plus, Fm+, I2C-bus
Abstract	The NXP GPIOs offer an alternative to expensive application specific stepper motor driver ICs. This application note outlines how to control multiple stepper motors together with optical position sensor inputs by using the right combination of GPIO from a large portfolio of products.

Revision history

Rev	Date	Description
01	20090911	Application note; initial version.

Contact information

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

A stepper motor is a digital version of the direct current electric motor. The rotor moves in discrete steps as commanded, rather than rotating continuously like a conventional motor. When stopped but windings energized, a stepper (short for stepper motor) holds its load steady with a holding torque and has full torque at standstill.

With steppers, precise positioning and repeatability of movement are achievable. They offer excellent response to starting/stopping/reversing. They are reliable since there are no contact brushes in the motor. Therefore the life of the motor is simply dependant on the life of the mechanical bearing. Since the motors respond to digital input pulses, open-loop control of the motor is possible, making the motor simpler and less costly to control. Due to these reasons steppers are extensively used in many applications including gaming, robotics, industrial control, toys, medical equipment, door control, disc drives, printers, etc.

Controlling a stepper motor requires energizing the motor windings with proper sequence. This requires generating proper digital waveforms, also called stepping pulse sequences (explained in [Section 3](#)). The motor coils require higher current compared to logic circuits. In addition, motor coils are inductive loads. The outputs of a stepping motor controller should have the capability to drive high current inductive loads of motor coils and be able to provide adequate power drive to motor windings.

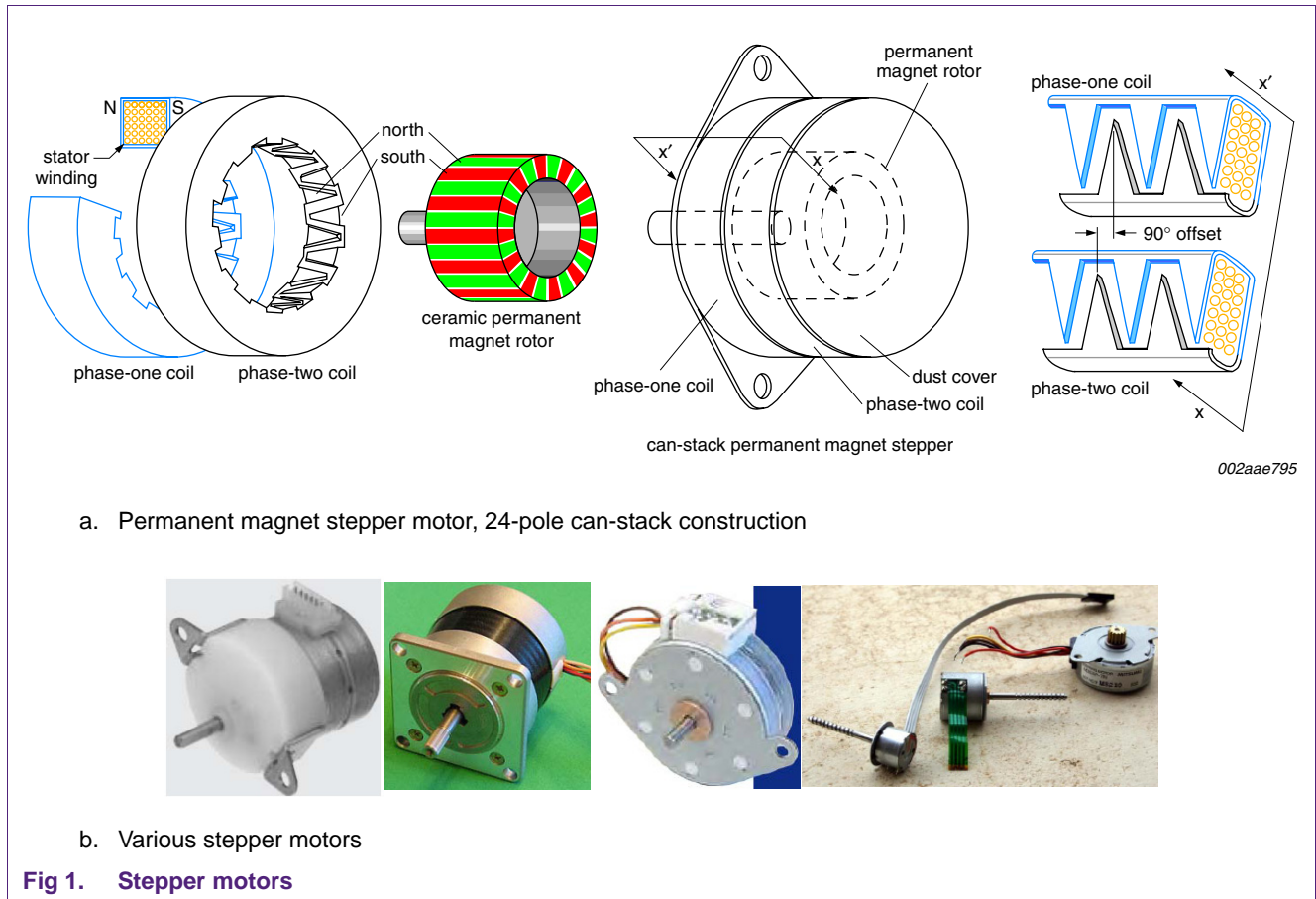
Today, application specific stepper motor drivers are available, but can be costly. This application note explains how to use inexpensive General Purpose I/O (GPIO) expanders from NXP to drive stepper motors.

2. Stepper motor background

A stepper motor, as its name suggests, moves one step at a time, unlike conventional direct current motors, which spin continuously. When commanded, a stepper motor moves some specific number of steps, it rotates incrementally that many number of steps and stops.

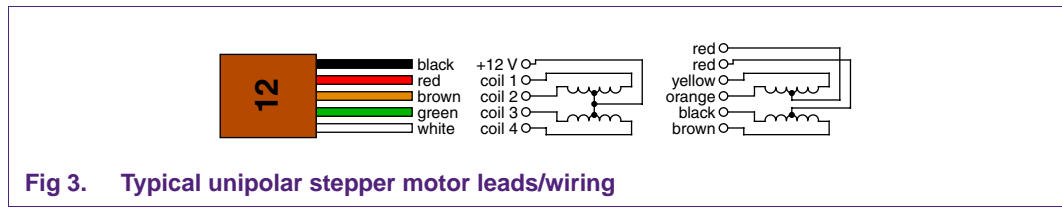
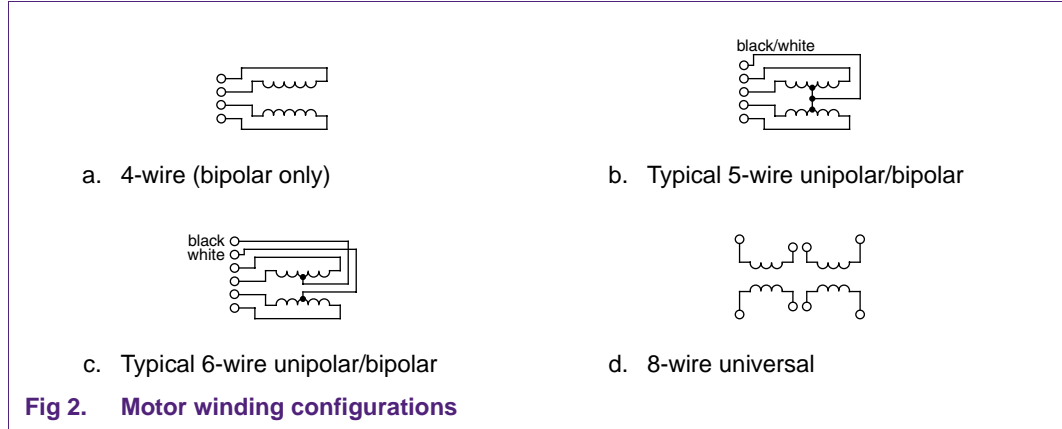
Inside a typical stepper motor are four coils of wire located 90° away from each other. In the middle is the rotor which spins and has permanent magnets fitted around its circumference. As the windings are energized, the rotor spins; each magnet in turn approaches, passes, and moves away from each of the four coils in turn.

Depending on how the coils are connected and driven, steppers fall into two types: unipolar and bipolar. In the unipolar mode of operation, the current flow in the windings always remains in the same direction to achieve rotor movement. A bipolar mode on the other hand, involves an alternate reversal of current flow in the windings to achieve rotation.

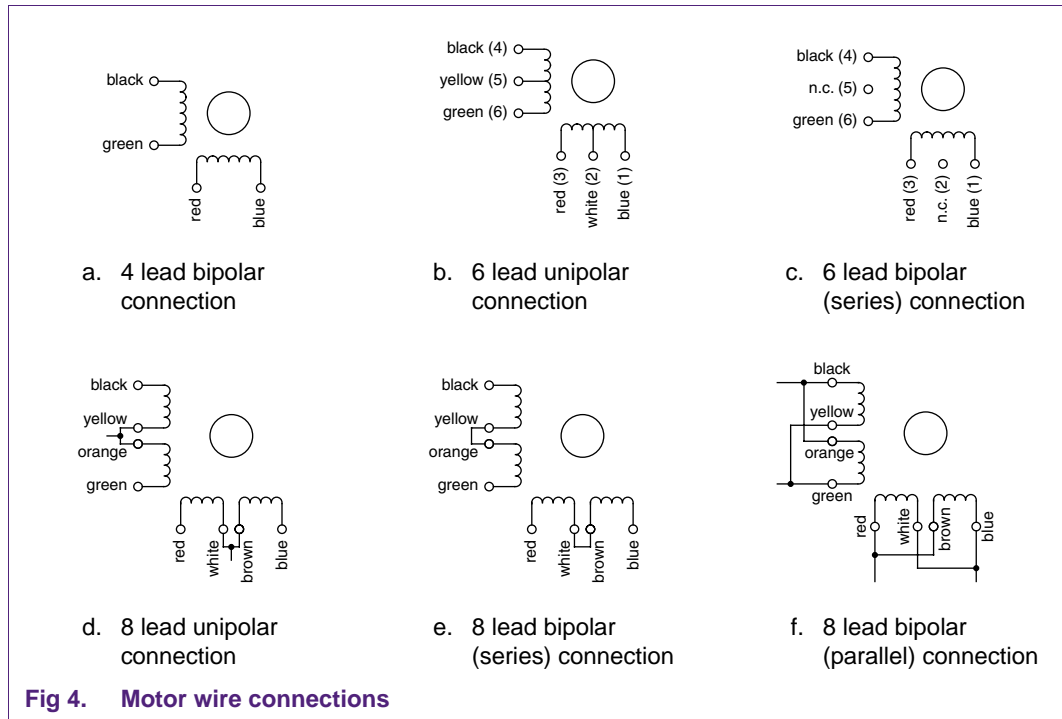


2.1 Motor winding configurations

Figure 2 shows the most common types of motor winding configurations available in the market.

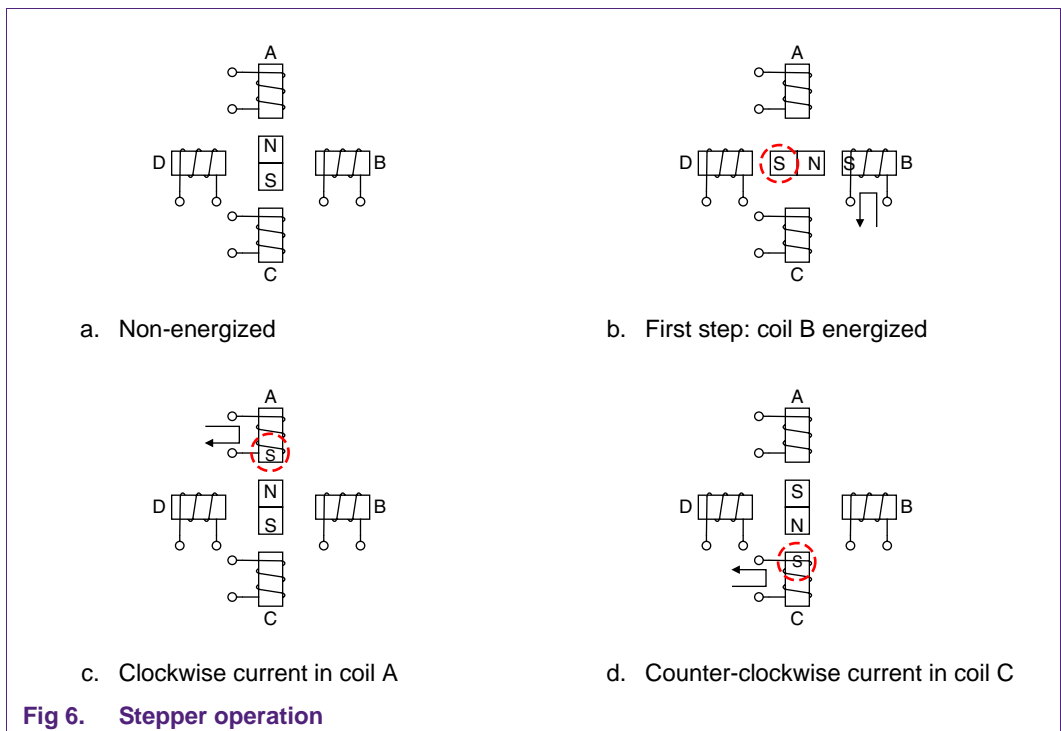
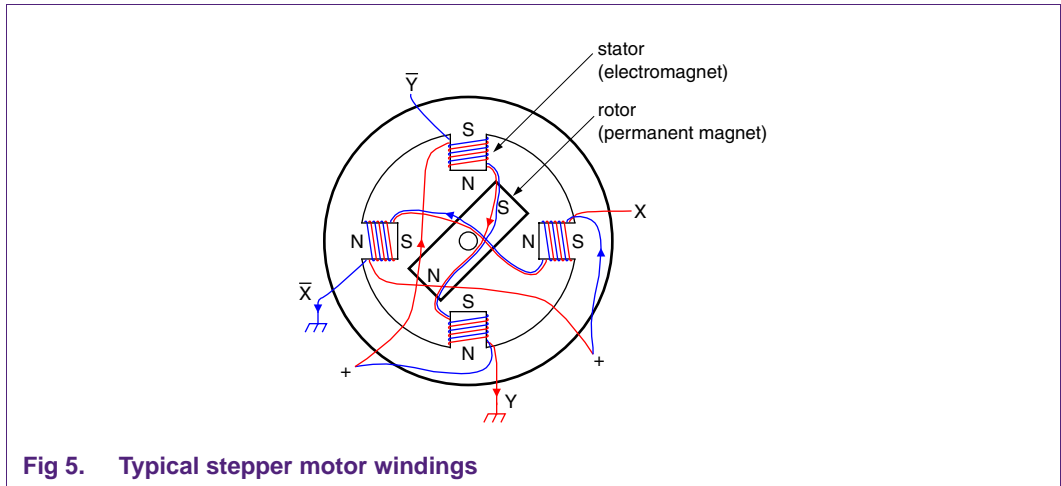


2.2 Motor wire connection diagrams



2.3 Principle of operation

Normally, the stator coils are paired to form two coils as shown in [Figure 5](#). Power is applied to two coils. Two stator cups formed around each of these coils, with pole pairs mechanically offset by $\frac{1}{2}$ of a pole pitch, become alternately energized north and south magnetic poles. Between the two stator-coil pairs the offset is $\frac{1}{4}$ of a pole pitch. The permanent magnet rotor has the same number of pole pairs as the stator coil section. Interaction between the rotor and stator (opposite poles attracting and like poles repelling) causes the rotor to move $\frac{1}{4}$ of a pole pitch per winding polarity change.



The four windings are energized in a sequential manner to rotate the shaft. The number of steps required to make one revolution equals $360 \div (\text{step size})$.

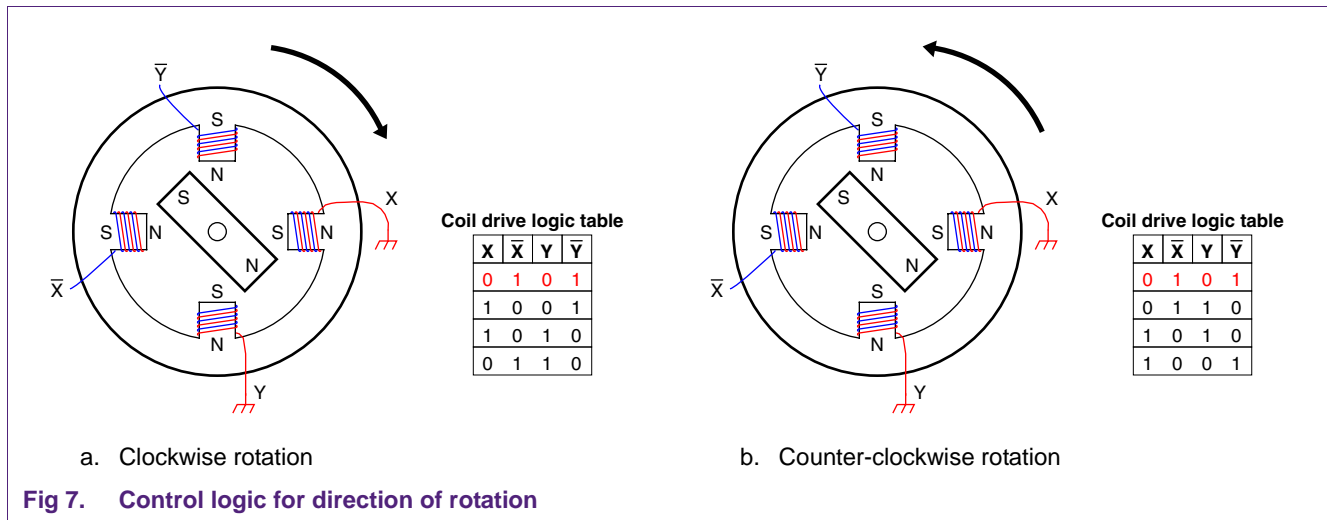
Example: For a 2-phase wound stepper motor with 1.8° mechanical step size (angle), the total number of steps is $360 \div 1.8 = 200$.

The number of steps performed by a stepper (resolution) can be increased by changing its sequence. To generate 200 steps, the full-step mode is employed. To generate 400 steps (i.e., a resolution of 0.9) half-step mode is required.

Common step angles are: 0.75°, 0.9°, 1.25°, 1.8°, 3°, 6°, 7.5°, 15°, 19°.

2.4 Controlling the direction of rotation

The reversal of direction of rotation is achieved by reversing the sequence only as shown in [Figure 7](#).



2.5 Speed of rotation and motor efficiency

The speed of a stepper motor depends on the rate at which the motor coils are turned on and off. This is also called the 'step-rate'. The maximum step-rate, and hence, the maximum speed of a stepper motor, depends upon the inductance of the stator coils. At standstill or low step rates, increasing the supply voltage produces proportionally higher torque until the motor magnetically saturates. Near saturation the motor becomes less efficient so that further increase of supply voltage will not result in greater torque. The maximum speed of a stepper motor is limited by inductance and eddy current losses. At a certain step-rate the heating effect of these losses limits any further attempt to get more speed or torque out of a motor by driving it harder.

3. Overview of stepper motor control

Stepper motor control involves generating and programmed control of step pulse sequences and driving the motor coils via a power driver stage (higher current and higher voltage driver compared to logic stages).

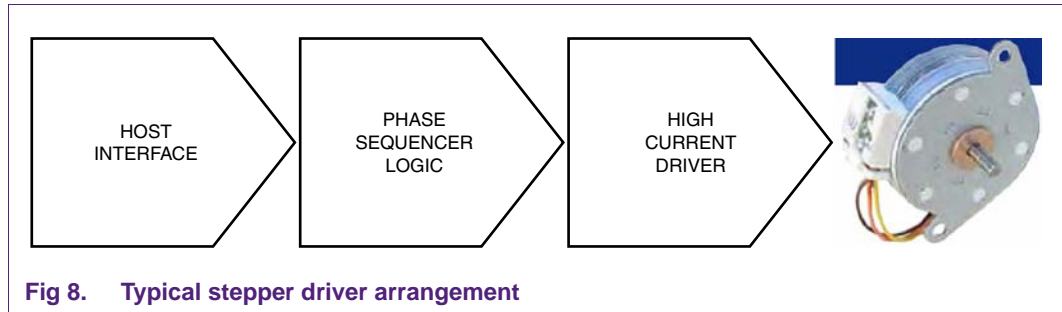


Fig 8. Typical stepper driver arrangement

The following are the most common drive modes.

- Wave drive (1 phase on)
- Full step drive (2 phases on)
- Half step drive (1 and 2 phases on)
- Microstepping (continuously varying motor currents)

Remark: Microstepping drive method will not be discussed in this application note.

3.1 Phase sequence generation

Phase sequence waveforms determine the type of drive method used to control the stepper motors. Generally, these waveforms are used to drive high current power driver stages to provide required drive current to the stepper motor coils.

[Figure 10](#), [Figure 12](#) and [Figure 14](#) show the waveforms for wave drive, two-phase and half-step formats.

Since all signals in each type of drive method are in defined relations with each other, it is possible to generate them using standard logic.

For most stepper motors that employ permanent magnet rotors useful torque is achievable for speeds below 1000 steps per second. Driving steppers at higher speeds results in reduced torque. Therefore, the step pulses are in the millisecond range.

3.2 Wave drive (1 phase on)

In Wave Drive only one winding is energized at any given time.

The advantage of wave drive mode is its simplicity. The disadvantage of wave drive mode is that in the unipolar wound motor only 25 % and in the bipolar motor only 50 % of the total motor winding are used at any given time. This means that maximum torque output from the motor is not made available.

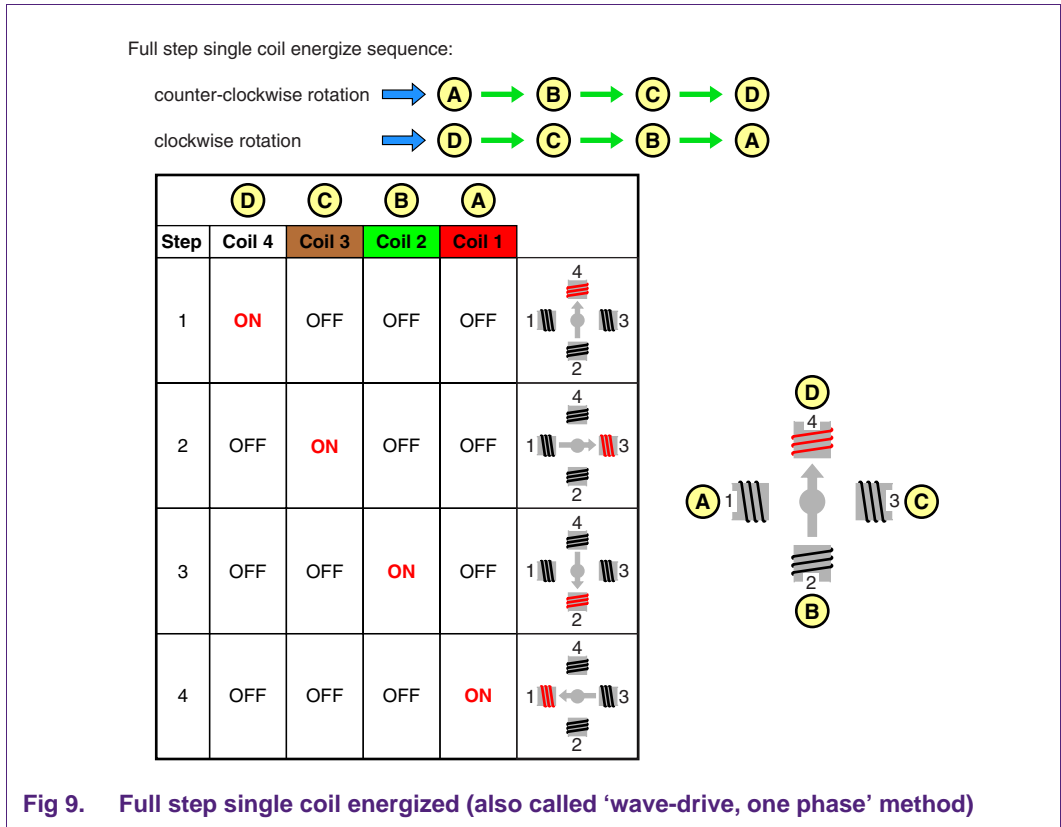


Fig 9. Full step single coil energized (also called 'wave-drive, one phase' method)

Since only one winding is energized, holding torque and working torque are reduced by 30 %. This can, within limits, be compensated by increasing supply voltage. The advantage of this form of drive is higher efficiency, but at the cost of reduced step accuracy.

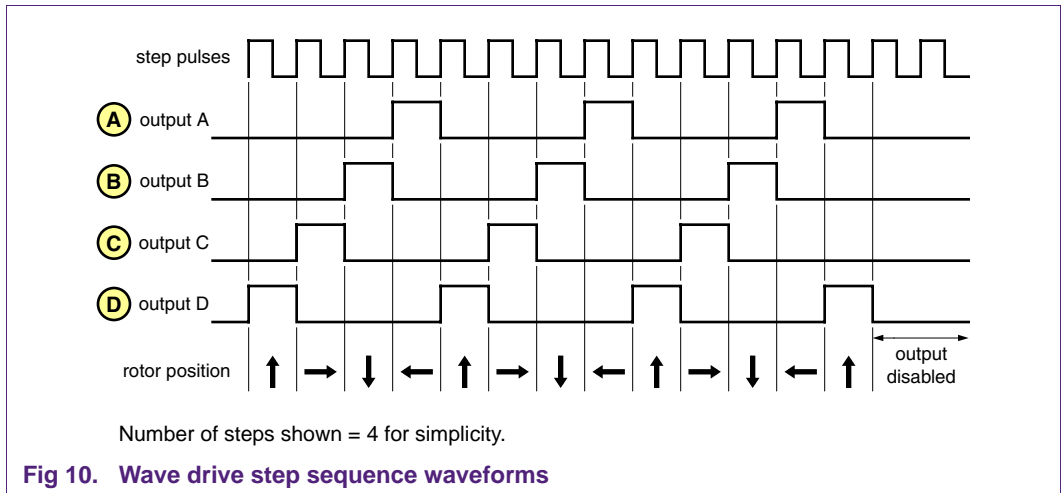


Fig 10. Wave drive step sequence waveforms

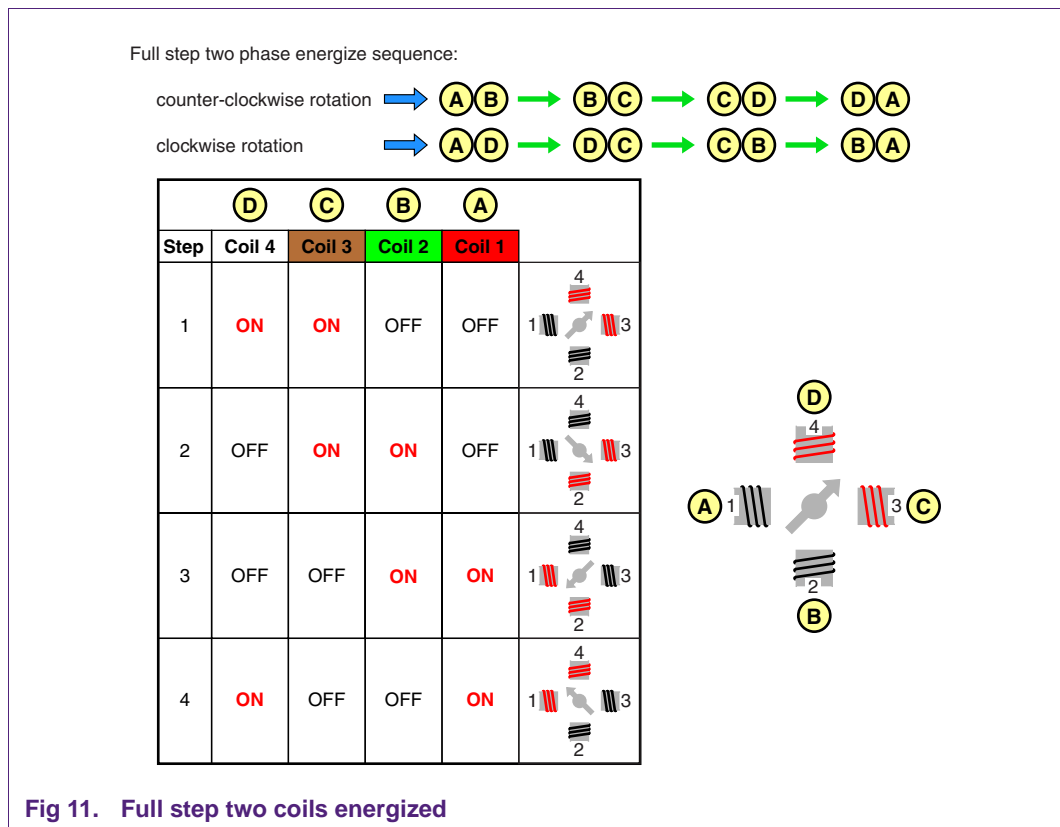
Table 1. Logic output sequence for wave drive

Winding	Step							
	1	2	3	4	5	6	7	8
Winding D	1	0	0	0	1	0	0	0
Winding C	0	1	0	0	0	1	0	0
Winding B	0	0	1	0	0	0	1	0
Winding A	0	0	0	1	0	0	0	1

3.3 Full step two phase drive

In full step two phase drive method, two windings are energized at any given time.

In case of full step drive, the torque output of the unipolar wound motor is lower than the bipolar motor (for motors with the same winding parameters) since the unipolar motor uses only 50 % of the available winding, while the bipolar motor uses the entire winding.



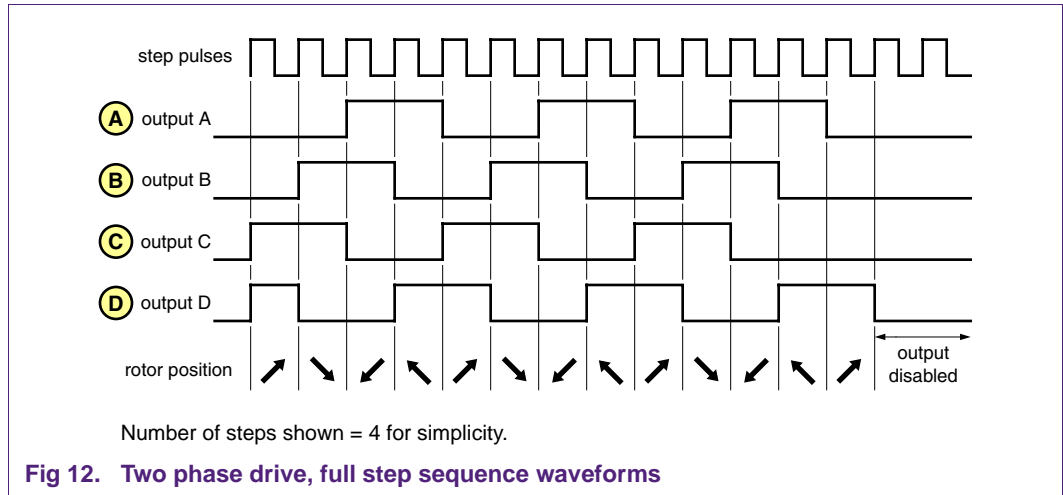


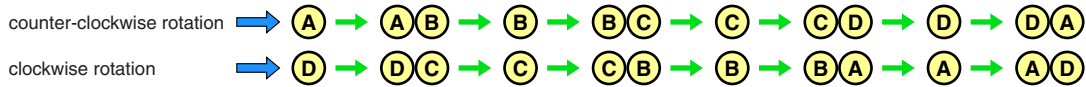
Table 2. Logic output sequence for 2-phase drive

Winding	Step							
	1	2	3	4	5	6	7	8
Winding D	1	0	0	1	1	0	0	1
Winding C	1	1	0	0	1	1	0	0
Winding B	0	1	1	0	0	1	1	0
Winding A	0	0	1	1	0	0	1	1

3.4 Half step drive (1 and 2 phases on)

'Half step drive' combines both wave and two phase full step (1 and 2 phases on) drive modes. This results in angular movements that are half of those in 1- or 2-phases-on drive modes. Half stepping can reduce a phenomena referred to as resonance which can be experienced in 1- or 2- phases-on drive modes.

Full step two phase energize sequence:



	(D)	(C)	(B)	(A)	
Step	Coil 4	Coil 3	Coil 2	Coil 1	
1	ON	OFF	OFF	OFF	
2	ON	ON	OFF	OFF	
3	OFF	ON	OFF	OFF	
4	OFF	ON	ON	OFF	
5	OFF	OFF	ON	OFF	
6	OFF	OFF	ON	ON	
7	OFF	OFF	OFF	ON	
8	ON	OFF	OFF	ON	

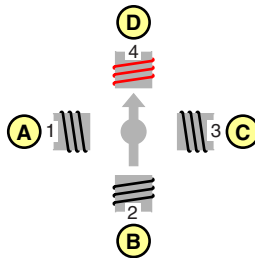


Fig 13. Half step sequencing

As the name implies, in this mode it is possible to step a motor in a half-step sequence, thus producing half steps, for example 3.75° steps from a 7.5° motor. A possible drawback for some applications is that the holding torque is alternately strong and weak on successive motor steps. This is because on full steps only one phase winding is

energized, while on the half-steps two stator windings are energized. Also, because current and flux paths differ on alternate steps, accuracy will be worse than when full stepping.

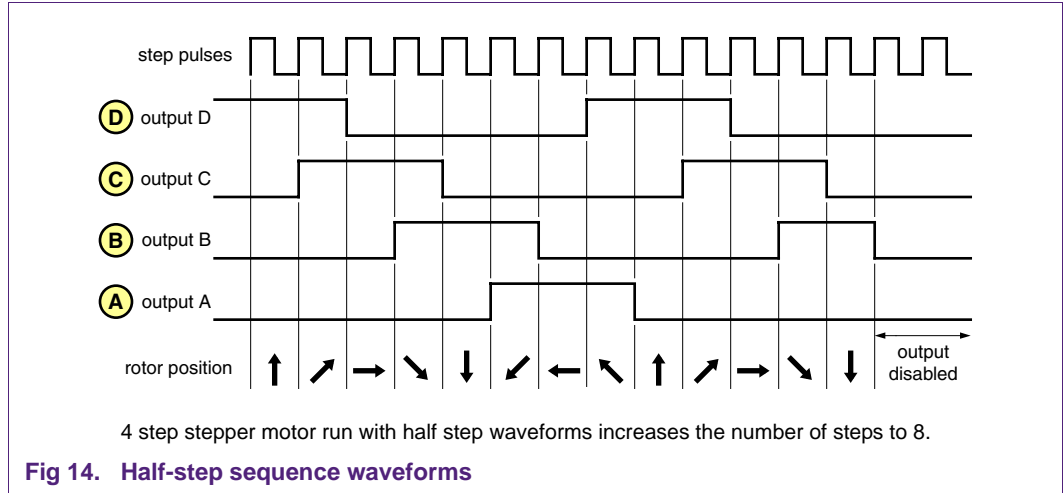


Fig 14. Half-step sequence waveforms

Table 3. Logic output sequence for half-step drive

Winding	Step							
	1	2	3	4	5	6	7	8
Winding D	1	1	0	0	0	0	0	1
Winding C	0	1	1	1	0	0	0	0
Winding B	0	0	0	1	1	1	0	0
Winding A	0	0	0	0	0	1	1	1

4. Motor shaft position sensing

A common method used for sensing the position of rotor shaft is to use an optical interrupter module. An optical interrupter module consists of a light emitter and a light receiver separated by a slot or air gap. The light emitter is an LED and the receiver is a photo transistor.

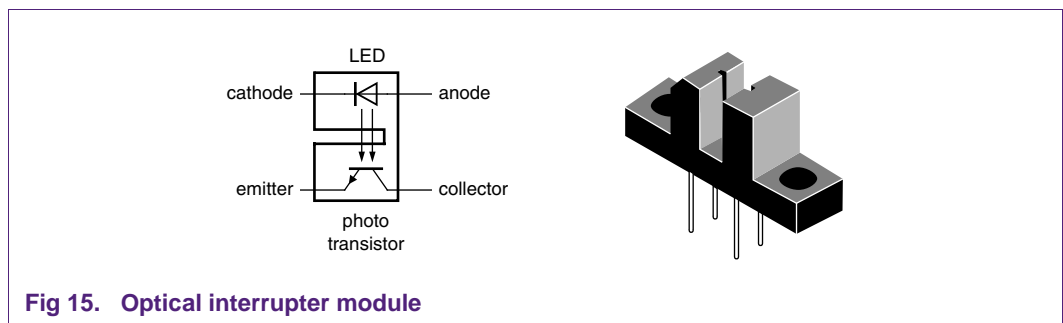


Fig 15. Optical interrupter module

The interruption of the light beam by a mechanical object causes an on/off signal to be generated by the photo transistor.

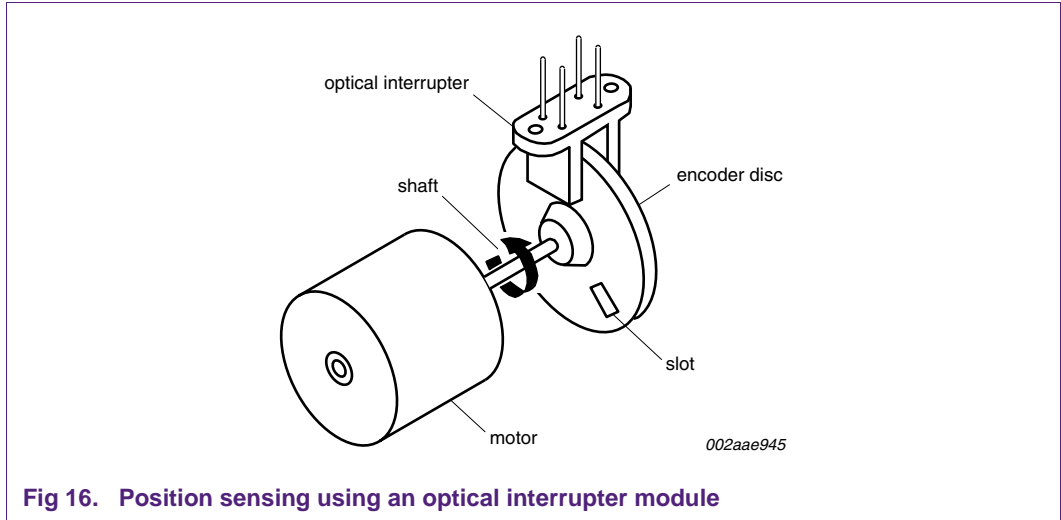


Fig 16. Position sensing using an optical interrupter module

This signal can be used by one of the input ports of an I²C-bus GPIO device to generate an interrupt signal. The interrupt signal from the GPIO can be used by the microcontroller to maintain a reference signal for home position of the rotor shaft.

The LED and photo transistor require additional circuitry to be able to generate a logic level signal compatible with the I²C-bus GPIO.

A typical circuit used for generating an inverting type logic output is shown in [Figure 17](#).

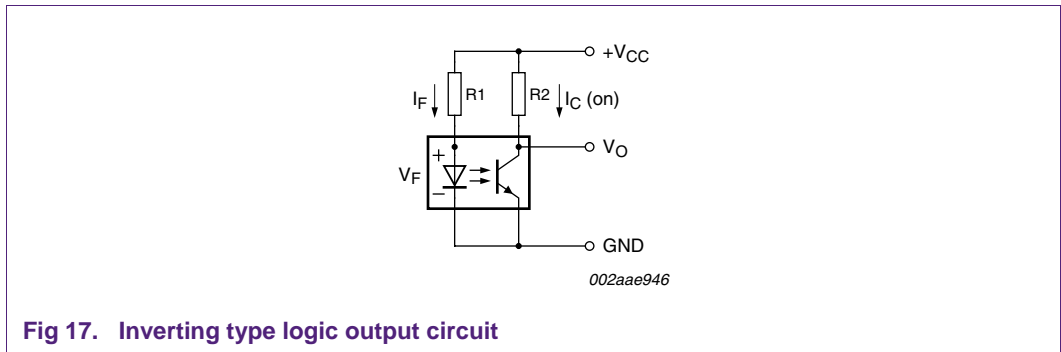


Fig 17. Inverting type logic output circuit

The common-collector circuit will generate an output signal which transitions from LOW to HIGH when the light (or infrared radiation) on the photo transistor is interrupted. This is commonly referred to as an ‘inverting logic condition’.

Table 4. Logic states for inverting type sensor output circuit

Optical interrupter output state	State	
gap blocked	yes	no
photo transistor state	OFF	ON
output voltage (V_O)	HIGH	LOW
output logic	1	0

A typical circuit for generating a non-inverting type logic output is shown in [Figure 18](#).

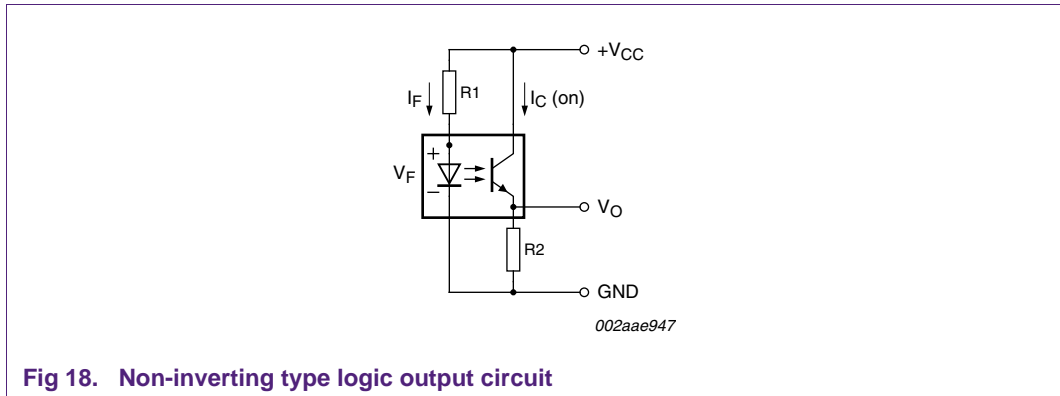


Fig 18. Non-inverting type logic output circuit

The common-emitter circuit will generate an output signal which transitions from HIGH to LOW when the light (or infrared radiation) on the photo transistor is interrupted. This is commonly referred to as an 'non-inverting logic condition'.

Table 5. Logic states for non-inverting type sensor output circuit

Optical interrupter output state	State	
gap blocked	yes	no
photo transistor state	OFF	ON
output voltage (V _O)	LOW	HIGH
output logic	0	1

5. Electrical drive for stepper motor coils

Stepper motor coils require higher current drive compared to logic circuits. Coil current ratings can range from 100 mA to several Amps. Higher the torque, larger the current. Even with smaller stepper motors each coil could draw 350 mA or more. Besides drawing higher current, motor coils typically operate at 12 V, 24 V or 48 V.

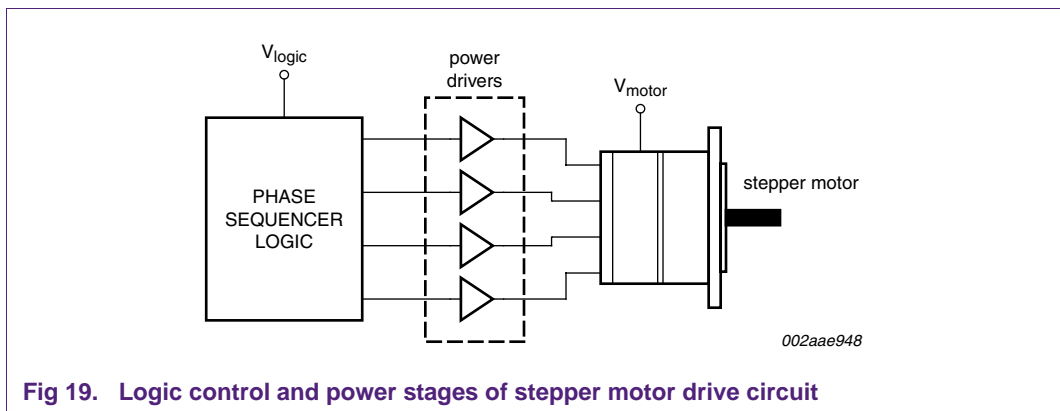


Fig 19. Logic control and power stages of stepper motor drive circuit

5.1 Using transistors to drive motor coils

Bipolar power transistors can be used to drive motor coils.

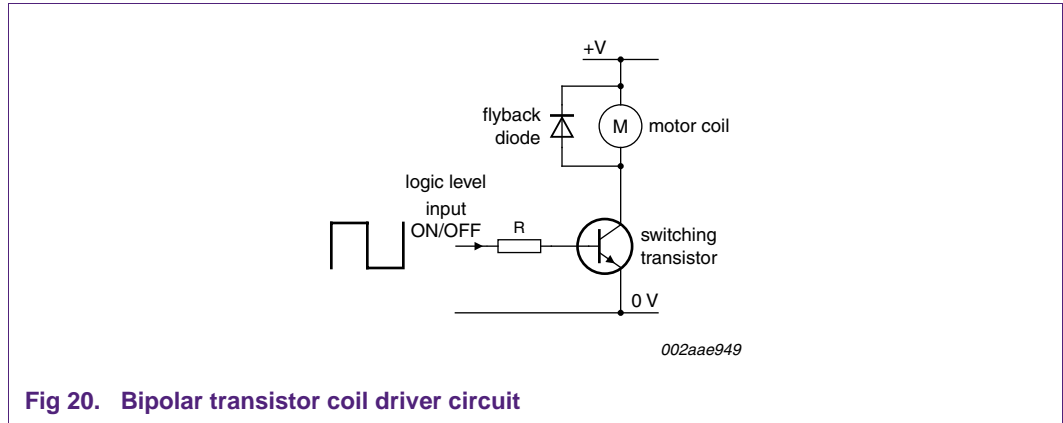


Fig 20. Bipolar transistor coil driver circuit

Motor coils are inductive loads. When the current is removed from the coil, the magnetic field collapses and this causes current to flow through the wire until the magnetic field is completely collapsed or gone. This can cause a damaging voltage spike back towards the logic circuit. Switching inductive loads can generate transient voltages of many times the steady-state value. For example, turning off a 12 V coil can easily create a negative spike of 200 V. To prevent damage to the circuit due to such high voltages, a diode is placed in parallel with the load as shown in [Figure 20](#). This is known as a freewheeling diode or flyback diode. The electric current generated by this high voltage spike is applied to the diode and is prevented from appearing across the transistor switch.

It should be noted that digital logic outputs cannot directly provide the base current required by power transistors. For this reason, Darlington connection-type transistor shown in [Figure 21](#) is used for this purpose.

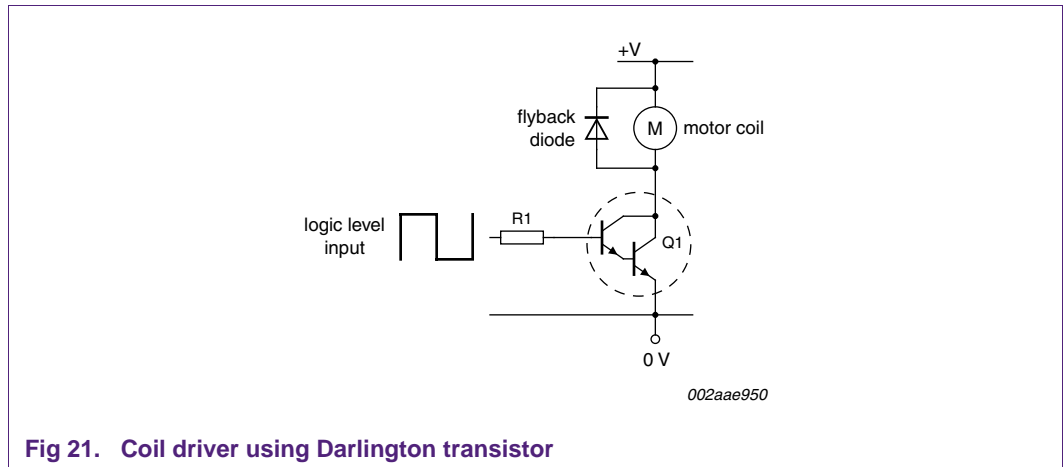


Fig 21. Coil driver using Darlington transistor

The effective h_{FE} of this transistor is the multiplication of the h_{FE} of each transistor inside. The high gain of this configuration results in requiring very small base current from logic outputs.

5.2 Using MOSFETs to drive motor coils

Availability of logic level MOSFETs provides a convenient means to drive motor coils. MOSFETs use voltage type drive as opposed to bipolar transistors.

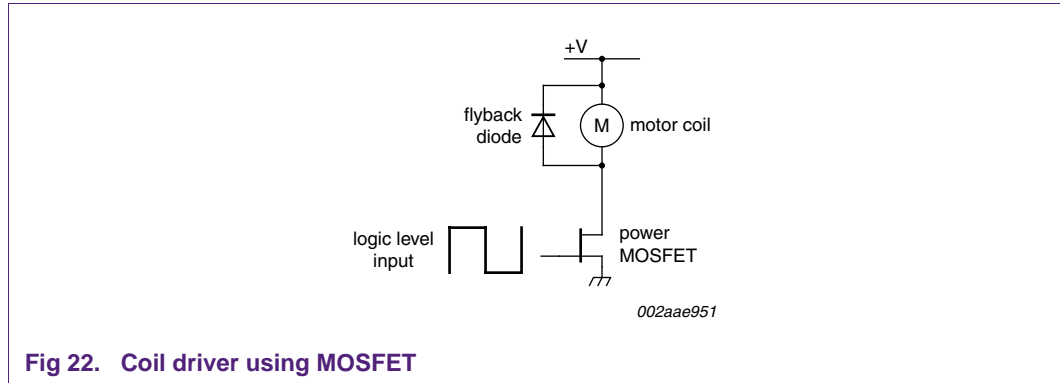


Fig 22. Coil driver using MOSFET

6. Selecting suitable NXP GPIO for stepper motor control

NXP has a large portfolio of I²C-bus GPIOs. Generally they fall into two sub-families, commonly called 'quasi-bidirectional General Purpose I/Os' and 'totem pole General Purpose I/Os'. Devices can be chosen with 8-bit, 16-bit or 40-bit width. Additional features (not available on all the devices) are active LOW interrupt output, active LOW reset input, programmable I²C-bus address pins and low power consumption. Finally, some devices come with additional functions (e.g., EEPROM, DIP switch) providing integrated and price attractive combination solutions. Detailed information on these GPIOs discussed in this section and data sheets can be found at the NXP web site: www.nxp.com/i2clogic.

NXP offers many general purpose I/O expanders (GPIO). These devices are controlled by I²C-bus. Since I²C-bus interface uses only two signal lines and most microcontrollers have I²C-bus available, using I²C-bus controlled GPIOs to generate step sequences for stepper motors is simple and inexpensive.

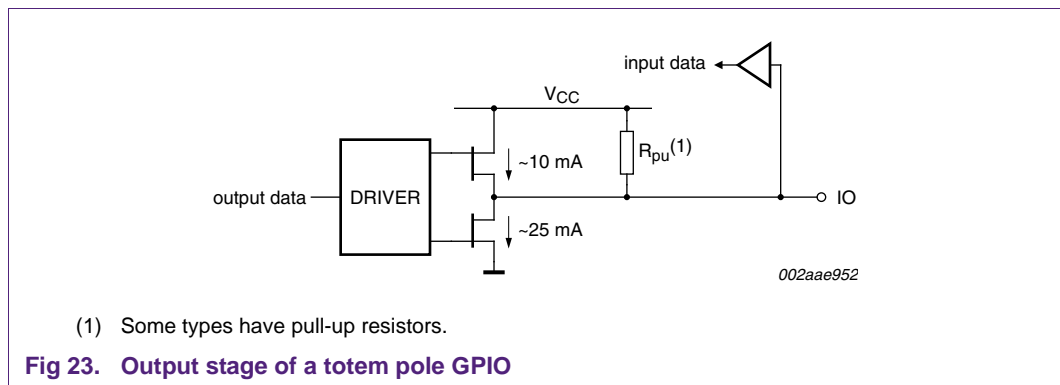
The logic states of these GPIOs are determined by the bits programmed into control registers. Using I²C-bus commands and toggling the I/O pins configured as outputs in proper sequence, various step sequences needed to control a stepper motor can be implemented. Additionally, input ports can be employed to provide motor shaft position information as described in [Section 4 "Motor shaft position sensing"](#).

All NXP I²C-bus general purpose I/O expanders offer similar functionality and most have an $\overline{\text{INT}}$ output, but several newer devices also have a $\overline{\text{RESET}}$ input or $\overline{\text{RESET}}$ and $\overline{\text{OE}}$ input. The $\overline{\text{INT}}$ output is used to signal the microcontroller when any of the inputs change state (1-to-0, or 0-to-1). The $\overline{\text{RESET}}$ input can be used to initialize the device to its default state without de-powering it. This is useful in situations where the I²C-bus has a glitch that prevents proper transmission of data between the microprocessor and I/O expander. Incorrect data in the I/O expander is eliminated through resetting it. Other slave devices without a $\overline{\text{RESET}}$ input require their power supply to be lowered to below 0.2 V and then powered back up to V_{CC} for the slave device to return to its default state, which can be inefficient and time-consuming in the application. The $\overline{\text{OE}}$ is used to 'high-impedance' the outputs without having to use I²C-bus commands.

6.1 Using I²C-bus totem pole GPIOs to generate step sequence waveforms

NXP Totem-pole GPIO devices have the following common features:

- I/O structure: totem pole (push-pull) architecture provides good sinking and sourcing capabilities
- I/O configuration (input or output) is controlled by a Configuration register programmable through the I²C-bus
- An Output register programs the pins configured as outputs to be HIGH (logic 1) or LOW (logic 0)
- A Polarity Inversion register inverts the polarity of the logic level read in the Input register
- I/O current drive capability:
 - sink capability = 25 mA
 - source capability = 10 mA
- Power-up state: devices power up with I/Os configured as inputs
- Internal Power-On Reset (POR)



The I/O output of these totem pole GPIOs can be directly connected to gate of MOSFETs.

6.2 Quasi-bidirectional GPIOs

NXP quasi-bidirectional GPIOs use a push-pull I/O port with an internal weak current-source pull-up to keep the port HIGH since the upper transistor is on for only 1/2 clock cycle.

Quasi-bidirectional GPIO devices have the following common features:

- I/O structure provides good current sinking but weak (100 μ A) sourcing capabilities
- No need for configuration (Input or Output) programming: IO register Read operation sets IO pins as inputs and Write operation sets IO pins as outputs
- Power-up state: devices power up with I/Os configured as inputs
- Internal Power-On Reset (POR)

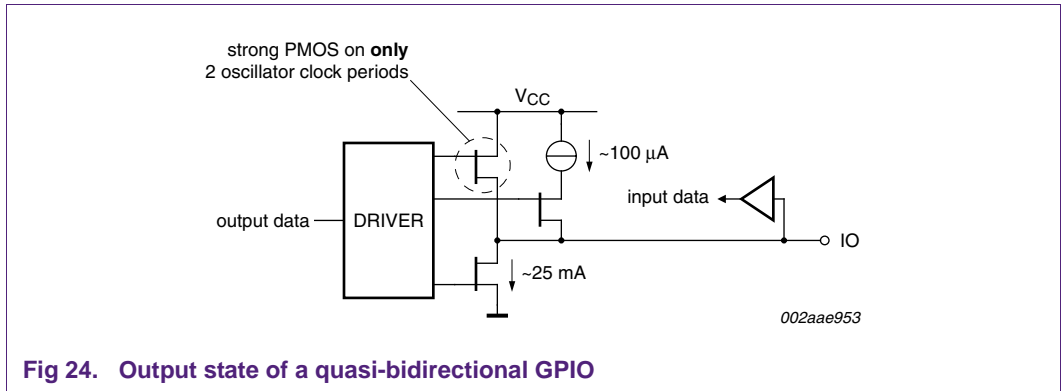


Fig 24. Output state of a quasi-bidirectional GPIO

This family of GPIOs can sink up to 25 mA, but can only source 100 µA. The current sourced by these GPIOs will not be able to drive Darlington transistors used in motor coil drive circuits unless additional external components are used.

Even though MOSFETs use voltage drive method, the MOSFET gate presents a capacitive load to GPIO outputs. The outputs of GPIO needs to provide adequate current source to charge the gate capacitance of the MOSFETs for proper operation. For this reason, 'quasi-bidirectional GPIOs' need additional external components to drive power stage of coil drivers.

Some vendors offer coil driver ICs that directly accept CMOS outputs. In this case, use of quasi-bidirectional GPIOs is a possibility.

Quasi-bidirectional GPIOs are well-suited for position sensing where the outputs of optical interrupters need to be checked. Using a separate (GPIO) chip for position sensing simplifies the software as compared to combining coil drive logic and position sensing in one GPIO device.

6.3 Suggested NXP GPIOs for up to 400 kHz I²C-bus applications

Table 6. GPIOs for 400 kHz I²C-bus operation

Device	Number of I/Os	Slave addresses	RESET	INT	OE	Number of motors	Packages
PCA9537	4	1	◆	◆		1	TSSOP10
PCA9538	8	4	◆	◆		2	SO16, TSSOP16, HVQFN16
PCA9557[1]	8	8	◆			2	SO16, TSSOP16, HVQFN16
PCA9539	16	4	◆	◆		4	SO24, TSSOP24, HVQFN24
PCA9506	40	64	◆	◆	◆	10	TSSOP56, HVQFN56

[1] IO0 on PCA9557 is open-drain type; rest of I/Os are push-pull type.

All GPIOs shown in [Table 6](#) have push-pull outputs that enable glue-less connection to MOSFETs or power Darlington transistors. In determining numbers of motors that can be driven by one of these devices, consider that each motor requires a minimum of 4 IO from the GPIO. So, as an example, the PCA9537 can drive only one stepper motor. When position sensors are used, additional inputs will be needed. It is common practice to use one or two position sensors. As such, a PCA9538 can support two motors with no position sensors or one motor with position sensors.

6.4 Suggested NXP GPIOs for up to 1000 kHz I²C-bus applications

GPIOs shown in [Table 7](#) and [Table 8](#) support Fast-mode Plus (Fm+) I²C-bus speeds. Fast-mode Plus (Fm+) GPIO devices offer an increase in I²C-bus transfer speeds. They can be used at bit rates of up to 1 Mbit/s, yet they remain fully downward compatible with Fast-mode or Standard-mode devices for bidirectional communication in a mixed-speed bus system. The same serial bus protocol and data format is maintained as with the Fast-mode or Standard-mode system. Fm+ GPIO devices also offer increased I²C-bus drive current over Fast-mode or Standard-mode devices, allowing them to drive longer and more heavily loaded buses.

The PCA9698 has push-pull type of outputs that can source up to 10 mA of current to allow direct connection to MOSFETs. Up to 10 stepper motors can be driven by a single PCA9698.

Table 7. Fast-mode Plus GPIO with push-pull outputs

Device	Number of I/Os	Slave addresses	$\overline{\text{RESET}}$	$\overline{\text{INT}}$	$\overline{\text{OE}}$	Number of motors	Packages
PCA9698	40	64	◆	◆	◆	10	TSSOP56, HVQFN56

For applications requiring less than 40 I/Os and Fast-mode Plus I²C-bus, the quasi-bidirectional GPIOs listed in [Table 8](#) can be used. Refer to application design-in section for examples.

Reminder: Quasi-bidirectional GPIOs need additional external components to drive power stage of coil drivers.

Table 8. Fast-mode Plus quasi-bidirectional GPIOs

Device	Number of I/Os	Slave addresses	$\overline{\text{RESET}}$	$\overline{\text{INT}}$	Number of motors	Packages
PCA9670	8	64	◆		2	SO16, TSSOP16, HVQFN16
PCA9672	8	16	◆	◆	2	SO16, TSSOP16, HVQFN16
PCA9674(A)	8	64		◆	2	SO16, TSSOP16, DIP16, SSOP20
PCA9671	16	64	◆		4	SO24, SSOP24, QSOP24, TSSOP24, HVQFN24, DHVQFN24
PCA9673	16	16	◆	◆	4	SO24, SSOP24, QSOP24, TSSOP24, HVQFN24, DHVQFN24
PCA9675	16	64		◆	4	SO24, SSOP24, QSOP24, TSSOP24, HVQFN24, DHVQFN24

7. Application design-in information

A typical power stage used to drive a 12 V unipolar stepper motor with a current rating of 1.25 Amps per winding is shown in [Figure 25](#). This functional block will be used in the following application examples and will be referred to as MOSFET drive module.

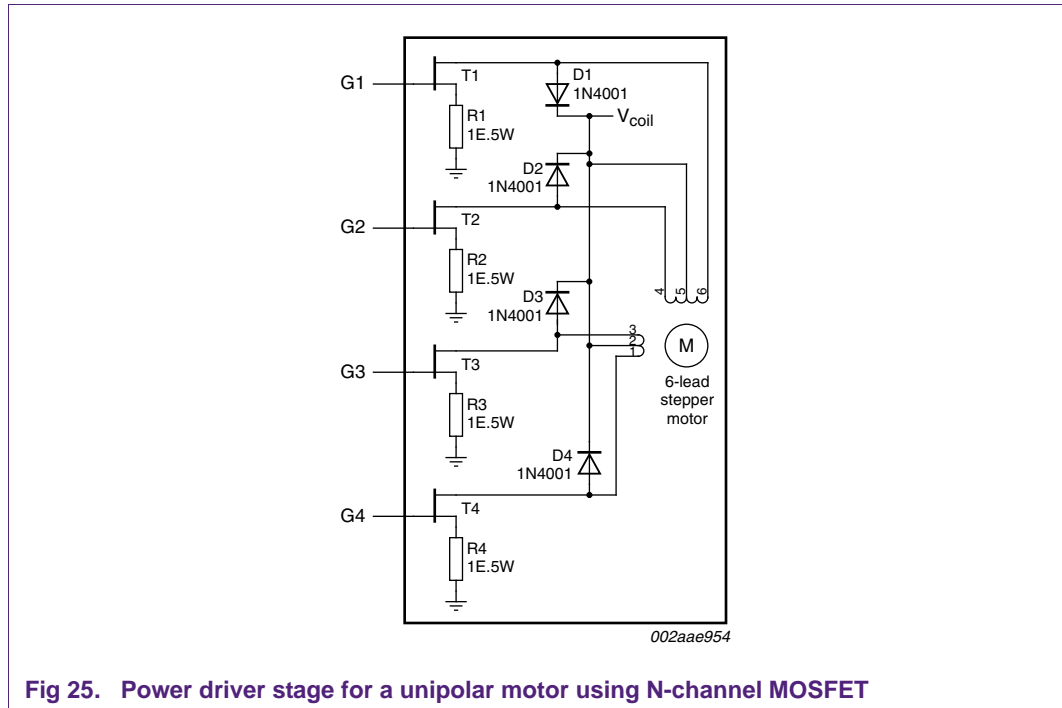


Fig 25. Power driver stage for a unipolar motor using N-channel MOSFET

This example uses MOSFETs (T1 to T4) capable of handling continuous current of 1.25 Amps required by the motor coils and more than 12 V of source to drain voltage rating. The push-pull GPIO outputs will be connected to G1 through G4, the gates of MOSFETs.

Because the MOSFET carries a high current, proper heat sink must be provided. The power resistors R1 to R4 are optional, but recommended to balance the L / R ratio of the windings and to limit the current depending on the resistance of windings.

7.1 Example unipolar stepper motor driver circuit using a PCA9537

Figure 26 shows an example application circuit to control a 12 V, 1.25 Amps unipolar stepper motor.

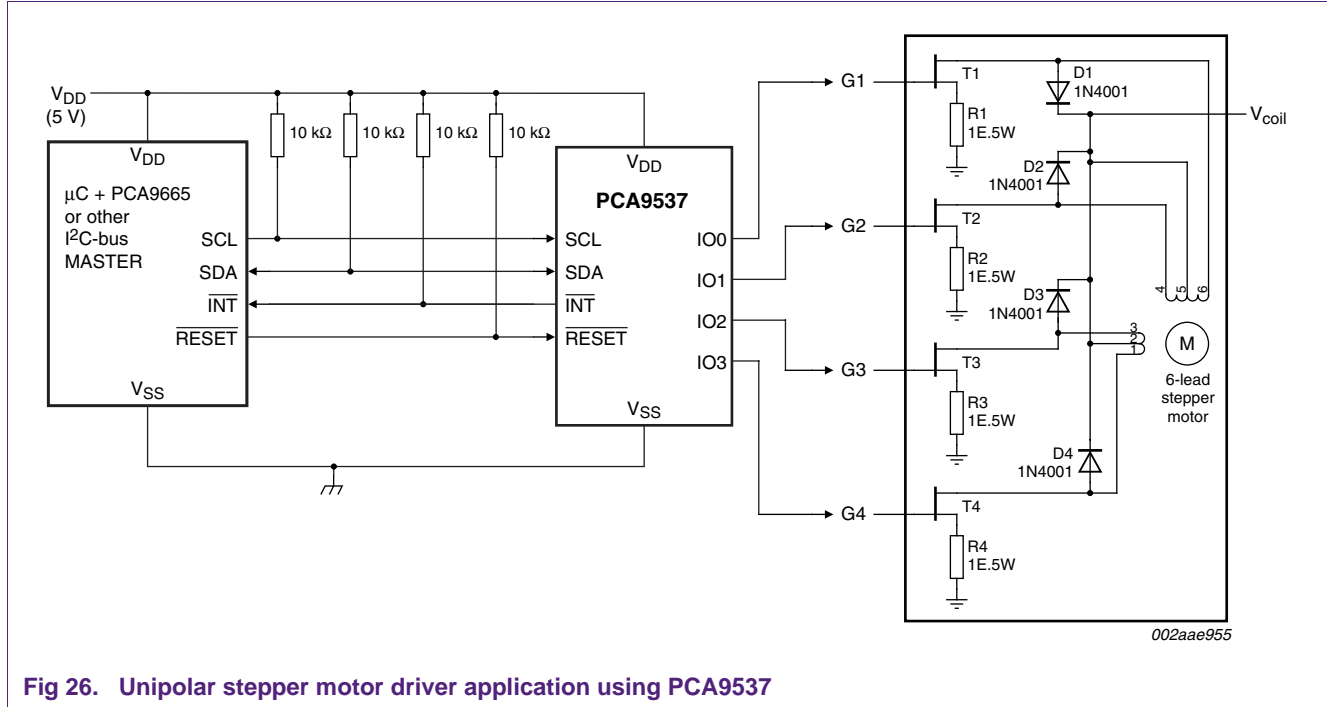


Fig 26. Unipolar stepper motor driver application using PCA9537

The PCA9537 totem pole GPIO used in this example has a fixed I²C-bus slave address, 92h. The host controller's firmware generates the I²C byte sequences needed to toggle the outputs and provide the waveforms at the gate inputs G1 to G4. The type of waveform will be one corresponding to wave, two-phase or half-step drive that is chosen by the user. The duration of the pulses is controlled by time delay implemented in the host controller firmware. There is no need to generate the square wave labeled 'step pulses' shown in Figure 10, Figure 12 and Figure 14. It is shown as a reference for the various waveforms. The maximum I²C-bus speed supported by PCA9537 is 400 kHz.

The following I²C bytecode sequence used with the application circuit shown in [Figure 26](#) will drive a stepper motor at approximately 18 RPM.

The 'S' stands for I²C-bus START condition, and 'P' stands for STOP condition. DLY1 and DLY2 are time delays implemented by the microcontroller firmware. Numbers are represented in hexadecimal notation.

```
S, 0x92, 0x03, 0x00, P // Set all 4 IO pins of PCA9537 as OUTPUT pins;
                        // IO Configuration Register = 0x00
Set DLY1 = 300 ms      // Time delay between steps; 18 RPM for a 7.5° motor

//Start of waveform loop
S, 0x92, 0x01, 0x08, P // Start the half-step sequence waveform
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x0C, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x04, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x06, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x02, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x03, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x01, P // outputs for the next step
Execute time delay between steps = DLY1
S, 0x92, 0x01, 0x09, P // outputs for the next step
Execute time delay between steps = DLY1
// Loop through waveform sequence to keep running motor

// OR execute next two steps if desired
Execute time delay = DLY2 // Hold in current position for time = DLY2
S, 0x92, 0x01, 0x00, P // Turn off motor
```


7.3 Example circuit to drive a stepper motor with two position sensors

Figure 28 shows an example application circuit to control one stepper motor that uses two optical sensors.

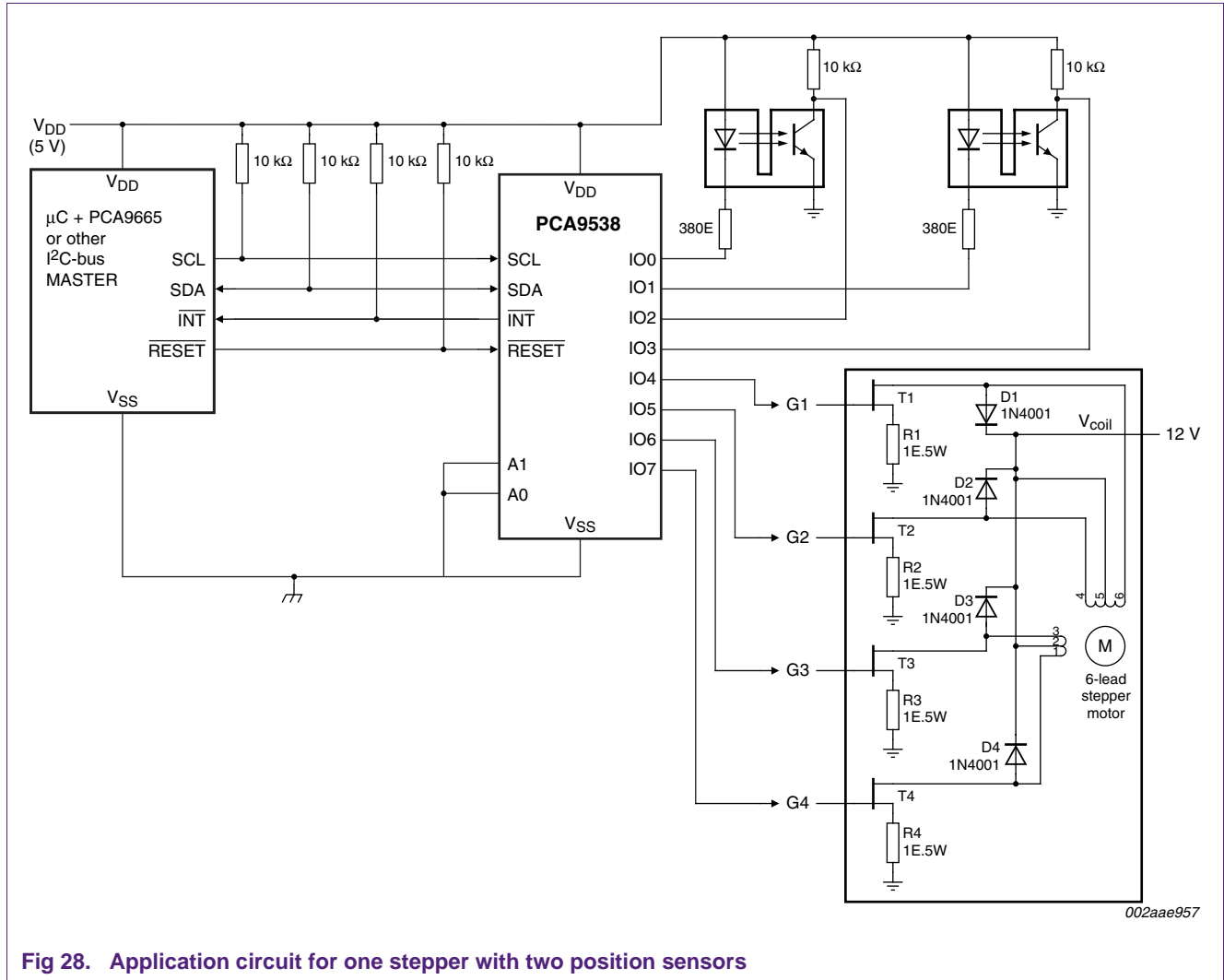


Fig 28. Application circuit for one stepper with two position sensors

The PCA9538 GPIO used in this example is hardware configured to have an I²C slave address = 0x40 by connecting both the address pins A0 and A1 to ground. The push-pull IO4, IO5, IO6 and IO7 are configured as output pins using the Configuration register. These pins are connected to G1 through G4, the gates of MOSFETs to drive the motor coils. The maximum I²C-bus speed supported by PCA9538 is 400 kHz.

The IO0 and IO1 are also configured as output pins to drive the LEDs inside the optical interrupter modules. These internal LEDs require around 10 mA drive. In some cases these LEDs are always on and can free up I/O pins. The advantage and flexibility of using the I/O pins is that these LEDs can be turned off when position sensing is not needed thus saving power. It offers an indirect way of masking the interrupts.

IO2 and IO3 are configured as inputs to sense the output from the photo-transistor inside the optical interrupter. When the optical interrupter outputs a pulse, an interrupt signal is generated by the PCA9538.

7.4 Example circuit to drive a stepper motor with a position sensor and control panel indicators

Figure 29 shows an example application circuit to control 1 stepper motor with one position sensor module.

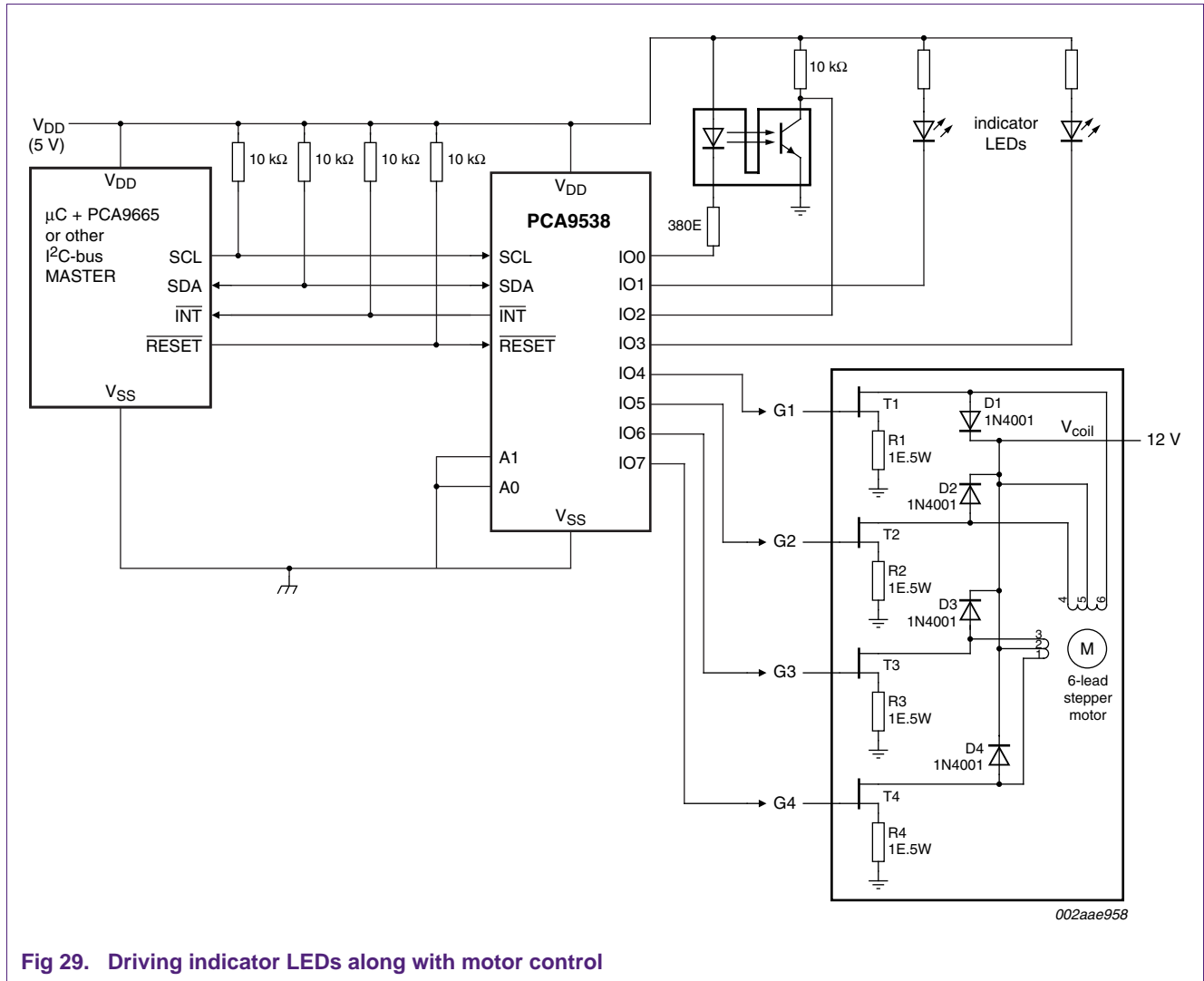


Fig 29. Driving indicator LEDs along with motor control

In this example, the PCA9538 drives two indicator LEDs in addition to controlling the stepper motor.

7.5 Example circuit to drive a stepper motor using Fast-mode Plus quasi-bidirectional GPIO

Figure 30 shows an example application circuit to control 2 stepper motors using a PCA9673 Fast-Mode Plus quasi-bidirectional GPIO.

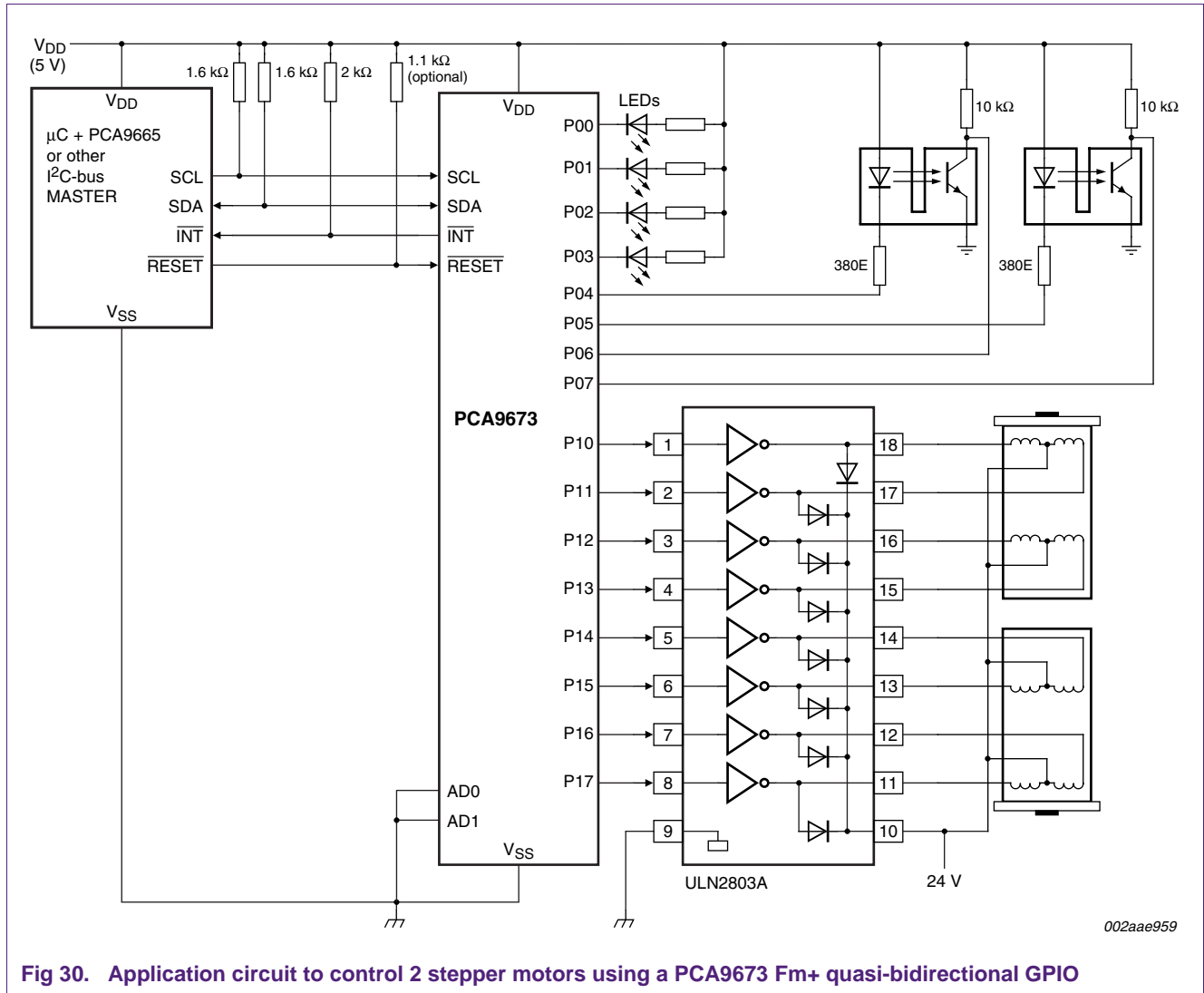


Fig 30. Application circuit to control 2 stepper motors using a PCA9673 Fm+ quasi-bidirectional GPIO

The PCA9673 Fast-mode Plus quasi-bidirectional GPIO outputs are CMOS compatible. This allows the use of power driver ICs that have the capability to accept CMOS compatible outputs. As an example, ULN2803A has 8 high-voltage, high-current Darlington arrays with open-collector outputs and integral clamp diodes (flywheel diodes) to deal with inductive loads. Coil voltages can be as high as 48 V.

7.6 Example circuit to drive a stepper motor using a PCA9537 with position sensing using a separate GPIO

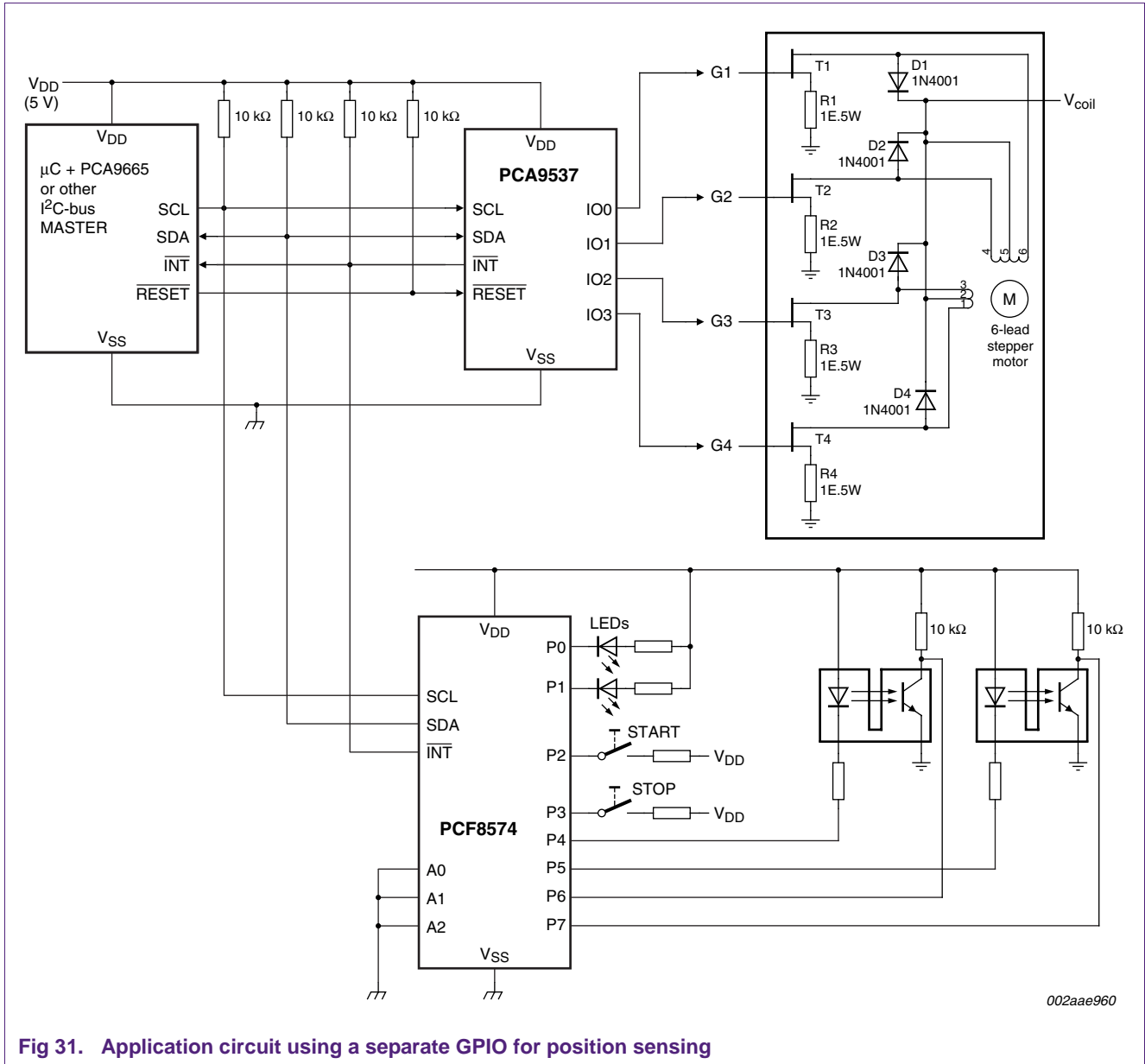


Fig 31. Application circuit using a separate GPIO for position sensing

8. Summary

The NXP GPIOs offer an alternative to expensive application specific stepper motor driver ICs. This application note outlined how to control multiple stepper motors together with optical position sensor inputs by using the right combination of GPIO from a large portfolio of products.

9. Additional information

Detailed information on PCA family of GPIOs and other I²C-bus products can be found at the NXP Semiconductor's web site: www.nxp.com/i2clogic

10. Abbreviations

Table 9. Abbreviations

Acronym	Description
CMOS	Complementary Metal-Oxide Semiconductor
DIP	Dual In-line Package
EEPROM	Electrically Erasable Programmable Read-Only Memory
GPIO	General Purpose Input/Output
I ² C-bus	Inter-Integrated Circuit bus
I/O	Input/Output
IC	Integrated Circuit
LED	Light-Emitting Diode
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor
POR	Power-On Reset
RPM	Revolutions Per Minute

11. Legal information

11.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

11.2 Disclaimers

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

11.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

I²C-bus — logo is a trademark of NXP B.V.

12. Contents

1	Introduction	3	11	Legal information	30
2	Stepper motor background	4	11.1	Definitions	30
2.1	Motor winding configurations	5	11.2	Disclaimers	30
2.2	Motor wire connection diagrams	5	11.3	Trademarks	30
2.3	Principle of operation	6	12	Contents	31
2.4	Controlling the direction of rotation	7			
2.5	Speed of rotation and motor efficiency	7			
3	Overview of stepper motor control	8			
3.1	Phase sequence generation	8			
3.2	Wave drive (1 phase on)	8			
3.3	Full step two phase drive	10			
3.4	Half step drive (1 and 2 phases on)	12			
4	Motor shaft position sensing	13			
5	Electrical drive for stepper motor coils	15			
5.1	Using transistors to drive motor coils	16			
5.2	Using MOSFETs to drive motor coils	17			
6	Selecting suitable NXP GPIO for stepper motor control	17			
6.1	Using I ² C-bus totem pole GPIOs to generate step sequence waveforms	18			
6.2	Quasi-bidirectional GPIOs	18			
6.3	Suggested NXP GPIOs for up to 400 kHz I ² C-bus applications	19			
6.4	Suggested NXP GPIOs for up to 1000 kHz I ² C-bus applications	20			
7	Application design-in information	21			
7.1	Example unipolar stepper motor driver circuit using a PCA9537	22			
7.2	Example circuit to drive 10 stepper motors using a PCA9698	24			
7.3	Example circuit to drive a stepper motor with two position sensors	25			
7.4	Example circuit to drive a stepper motor with a position sensor and control panel indicators	26			
7.5	Example circuit to drive a stepper motor using Fast-mode Plus quasi-bidirectional GPIO	27			
7.6	Example circuit to drive a stepper motor using a PCA9537 with position sensing using a separate GPIO	28			
8	Summary	29			
9	Additional information	29			
10	Abbreviations	29			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.



© NXP B.V. 2009.

All rights reserved.

For more information, please visit: <http://www.nxp.com>
 For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 11 September 2009
 Document identifier: AN10814_1