# THIS SPEC IS OBSOLETE

Spec No: 001-41949

Spec Title: PSOC(R) 1 STEPPER MOTOR CONTROLLER - AN41949

Sunset Owner:  Wanchun (Ronny) Liu (RLIU)

Replaced by: 001-92562

# PSoC® 1 Stepper Motor Controller

**Author: Wanchun Liu**
**Associated Project: Yes**
**Associated Part Family: CY8C24xxx, CY8C27xxx, CY8C29xxx**
**Software Version: PSoC® Designer™ 5.1**
**Related Application Notes: AN43679**

OBSOLETE

AN41949 describes how to use the Pseudo Random Sequence Generator (PRS) in PSoC® 1 to drive a stepper motor.

## Introduction

Actual technology offers different solutions for movement. Compared to costly high performance motors such as brushless motors or CC motors, stepper motors are able to reach excellent accuracy in angular position and speed. They are ideal for robotics applications and are commonly used by hobbyists. Stepper motors are also preferred over other motors because of their simple functionality.

This application note describes the operation of these common motors and their driving techniques. This application note also proposes an alternative use of PRS. It is used as shift register to generate the excitement motor phases that enables the full control of phase generation by PSoC hardware without CPU time.

Some of the stepper motor's merits and shortcomings are as follows:

**Merits**

- It is possible to use them in open loop systems, that is, without position and speed sensors. High power processing by the CPU is not needed for the accuracy.
- They do not have brushes or contacts.
- They rotate very slowly without the aid of a speed reducer

**Demerits**

- They operate unevenly, thus intensifying rough motor vibrations.
- They have lower efficiency compared to other electric motors.
- They do not have a high speed rotation.
- They cannot make adjustments when encountering extreme speed variations.

The three main types of Stepper motors:

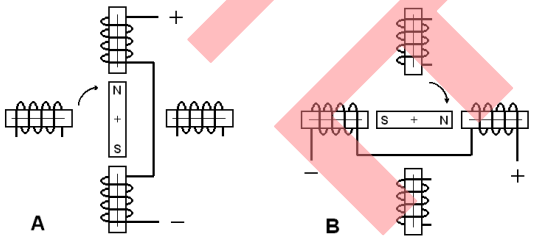
- Variable reluctance motors
- Permanent magnet motors
- Hybrid motors

The most popular motors are the unipolar and the bipolar motors. This application note refers to the use of unipolar motors and is theoretically adapted to other models.

## Functionality of the Steppers

Stepper motors belong to the synchronous motors group. Compared to other motors, when powered, stepper motors try to keep the shaft position in a defined balanced condition. Basically, the movement is achieved from the rotation, by a magnetic field generated into the stator and by the relative rotor chase. Figure 1 shows two excitation phases (A and B) and their consequent rotation and their step position in relation to the motor shaft.

Figure 1. Windings, Excitation, and Rotation

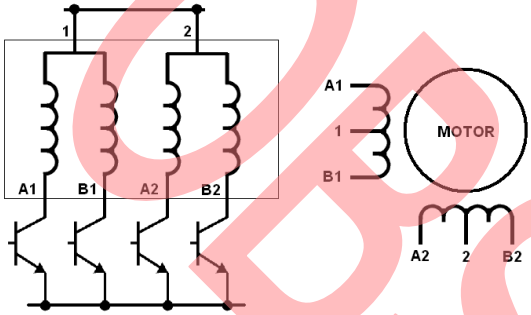



**Note** For simplicity, Figure 1 puts in the relationship between the steps and driving phases. But actually, step numbers vary in size from a few steps to several hundred steps per revolution. This depends on the number of rotor teeth and the distance from the stator poles that is usually two or four.

## Unipolar and Bipolar Stepper Motors

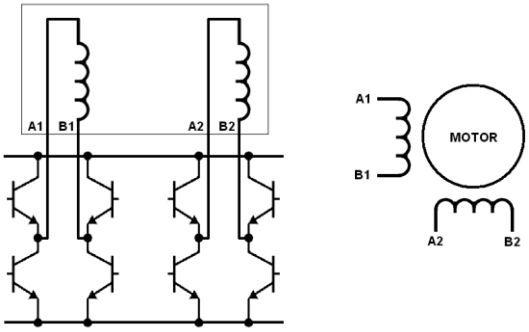
Unipolar stepper motors are the easiest to drive. They are electrically composed of four unique windings of eight wires, or four internally connected windings of six wires. It is possible to drive them with four Metal Oxide Semiconductor (MOS) or Bipolar Junction Transistor (BJT) switches as shown in Figure 2.

Figure 2. Unipolar Stepper Driving




In the unipolar steppers, the winding's current flow is always in the same direction (unipolar). The bipolar stepper motors have a simplified construction, but they require a more complex drive. A bipolar stepper's current outflows from the power circuit and must cross each winding in two directions. This requires the use of a bridge or an equivalent circuit for each winding as shown in Figure 3.

Figure 3. Bipolar Stepper Driving




## Operation

Independently from the employed motor or from the section of power in use, the excitation phases are identical in both the unipolar and the bipolar steppers. Table 1 reports the excitation phases in the simplest modality defined as **"Wavemode"** or "**One-phase-ON full step**".

Table 1. Wavemode Excitation Phases

| STEP | A1 | A2 | B1 | B2 |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 |
| 4 | 0 | 0 | 0 | 1 |

In this modality, only one phase at a time is energized. For every commutation, there is a corresponding step. In this case, the angular movement is the one confirmed by the constructor.

It also permits energizing two phases simultaneously. In this modality, defined as **"Two Phase ON"**, the rotor finds its balance condition between the two phases. The angular movement carried out by the single step is equivalent to the preceding modality even if it has shifted a half step. Table 2 shows the winding excitation phases.

Table 2. Two Phases ON Excitation Phases

| STEP | A1 | A2 | B1 | B2 |
|---|---|---|---|---|
| **1** | **1** | **1** | **0** | **0** |
| **2** | **0** | **1** | **1** | **0** |
| **3** | **0** | **0** | **1** | **1** |
| **4** | **1** | **0** | **0** | **1** |

The advantage of this second modality is that it is related to the torque that increases 1.4 times against the Wavemode modality. The two excitation phases cause a force that is perpendicular to the rotor. This resulting force is 1.4 times higher (or square root of two). Increased current consumption and motor overheating are factors taken into consideration. On the basis of the preceding modality, it is possible to obtain a third denominated **"Half-step."** Alternating between Wavemode and two Phase ON phases results in an angular advancement equivalent to half steps. The motor angular resolution is then doubled; however, it gains an irregular torque (1 - 1.4 - 1 - 1.4 …).

OBSOLETE

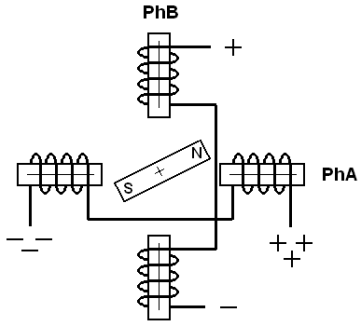
Table 3 shows the excitation phases.

Table 3. Half Step Excitation Phases

| STEP | A1 | A2 | B1 | B2 |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 0 |
| 3 | 0 | 1 | 0 | 0 |
| 4 | 0 | 1 | 1 | 0 |
| 5 | 0 | 0 | 1 | 0 |
| 6 | 0 | 0 | 1 | 1 |
| 7 | 0 | 0 | 0 | 1 |
| 8 | 1 | 0 | 0 | 1 |

A more sophisticated solution consists of the phase driving with current control. This modality is called “microstepping”. In microstepping, mode currents between adjacent phases are regulated according to their position, theoretically infinitesimal, among steps. The higher the phase current, the nearer it is to the adjacent rotor equilibrium position. Figure 4 shows the different energization between phases and relative rotor position.

Figure 4. Microstepping Rotor Position




This modality reduces one of the more unpleasant disadvantages of the stepper motor, the rough erratic motor movement. This also provides more sophisticated electronics to drive PWM pulses and enables an accurate and linear current control in each phase.

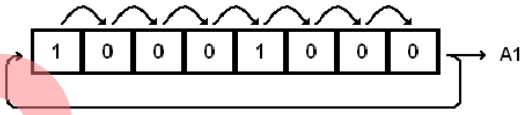
## PSoC Phases Generation

The freeness in peripheral construction and their flexibility makes PSoC devices more than adequate for the driving phase generation for a stepper motor. A specific stepper motor controller peripheral is not available in the PSoC device but it is easily created.

Table 1, Table 2, and Table 3, report the state of the excitation phases. These are considered repetitive. That is, after the last line ends, it restarts with the first. In fact, the bit sequence does not change and they repeat every 4 bits in wavemode in two Phase On and every 8 bits in Half Step mode.

The state of each single phase is given from a shift register properly loaded with the adequate starting point value. See Figure 5.

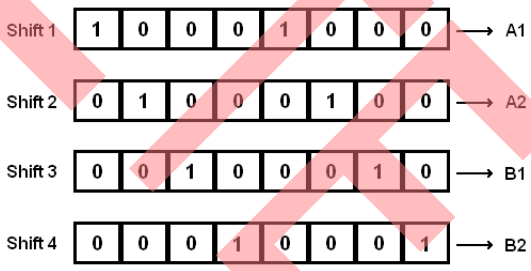
Figure 5. Phase A1 Shift Register




Connecting the shift register exit to its own input obtains a circular buffer. At every single clock pulse, the bit sequence, placed into the module is shifted outside. At the same time, it is reloaded into the shift register. The shift register contains the initial sequence that is preloaded at the start, after the 8 clock pulses. This sequence infinitely continues.

The four stepper motor driving phases are realized with each of four shift registers that is configured as previously described. But it is preloaded with a different value, the one relative to its own phase. Refer to Figure 6.

Figure 6. Four Phases Shift Registers

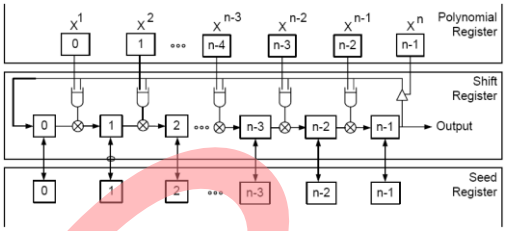



The shift register clocks are driven simultaneously. Each clock pulse produces a different phase on the shift exit. In fact, this is the **STEP** of our stepper motor.

The PSoC device does not have a real shift register among its digital peripheral, but it possesses a PRS that adapts itself to the case. In theory, a PRS is basically formed from a shift register. In this application, the single phase shift register uses a PRS peripheral. See Figure 7.

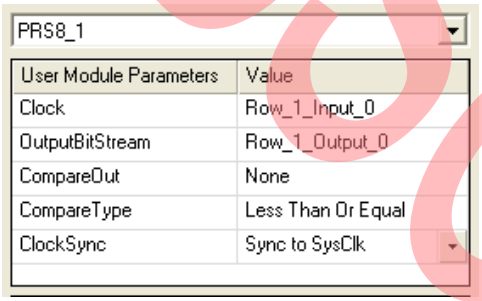
OBSOLETE

Figure 7. Internal PRS Structure




The peripheral connections are freely connectable and it simply produces a shift register as described in Figure 5. The output is connected to its input that is routed to a pin to drive one single motor phase. The clock signal that communicates with the other four PRSes is connected to an external pin or to an internal timer to generate the phase motor pulses inside the PSoC device. Figure 8 shows the first PRS configuration parameters. The remaining three are configured the same way.

Figure 8. PRS Configuration




## PSoC Device's Software

The software is needed only in the peripheral initialization. During the operation, CPU intervention is not required as the clock, internally generated or coming from the outside, directly draws the four phases shift registers.

Code 1, Code 2, and Code 3 show the initialization related to the three modalities described in this application notes. The reader must observe that PRS initial values are obtained from Table 1, Table 2, and Table 3.

Code 1. Wavemode Initialization

```
PRS8_1_WriteSeed(0x11);
PRS8_2_WriteSeed(0x44);
PRS8_3_WriteSeed(0x22);
PRS8_4_WriteSeed(0x88);
```

Code 2. Two Phase ON Initialization

```
PRS8_1_WriteSeed(0x99);
PRS8_2_WriteSeed(0x66);
PRS8_3_WriteSeed(0x33);
PRS8_4_WriteSeed(0xcc);
```

Code 3. Half Step Initialization

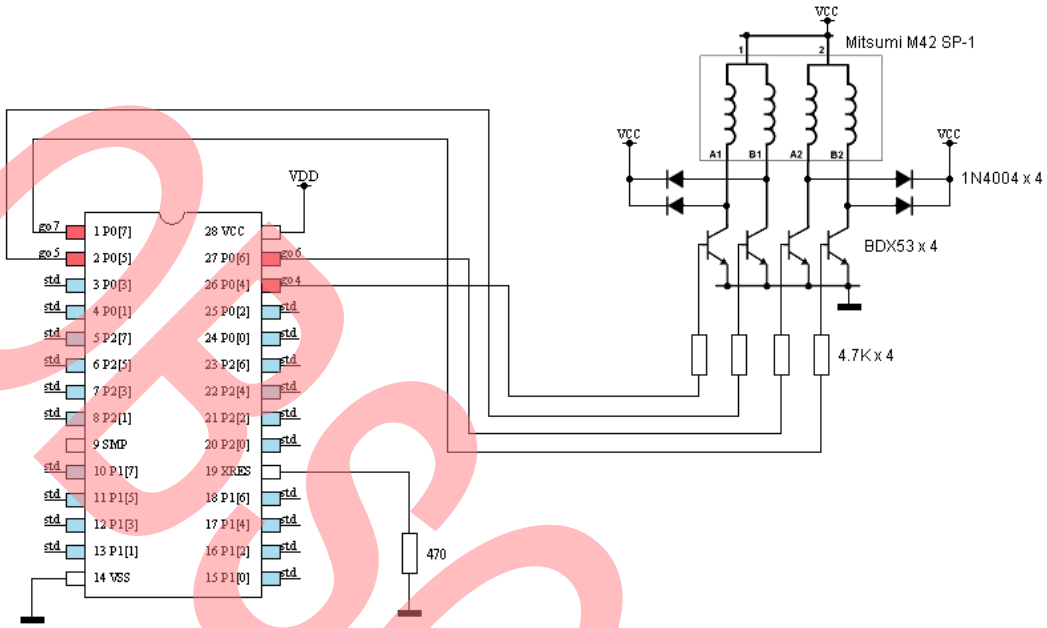
```
PRS8_1_WriteSeed(0x83);
PRS8_2_WriteSeed(0x38);
PRS8_3_WriteSeed(0x0e);
PRS8_4_WriteSeed(0xe0);
```

## PSoC Device's Hardware

The external controller hardware is concentrated on the power section. The adopted solution in this project is described in Figure 2 . The scheme and the related components are shown in Figure 9. More complex solutions are adopted as necessary. SPIS slave are used instead of PRS. Besides these, it demonstrates the PSoC's digital block versatility.

OBSOLETE

Figure 9. PSoC Device's Hardware Configuration




OBSOLETE

## Summary

This application note describes an alternative use of modules typically employed in other fields. It automatically produces bit sequences to drive or command any kind of external device connected to the PSoC device without the intervention of CPU time.

## About the Author

**Name:** Alessandro Molini

**Background:** HW-SW Industrial Automation designer with motion control, PLC, and motor control experience.

**Contact:** alex.mol@tiscali.it

## Document History

Document Title: PSoC® 1 Stepper Motor Controller - AN41949

Document Number: 001-41949

| Revision | ECN | Orig. of Change | Submission Date | Description of Change |
|---|---|---|---|---|
| ** | 1487485 | KANT | 11/12/2007 | New application note |
| *A | 3181372 | KANT | 02/25/2011 | Updated title, abstract, and introduction.<br>Updated the associated project and compiled with latest PSoC Designer software. |
| *B | 3306649 | KANT | 07/08/2011 | Removed references to an absolute part CY8C26xxx.<br>Removed associated project with part CY8C26xxx.<br>Updated associated project. Updated title. |
| *C | 4469430 | DIMA | 08/08/2014 | Sunset Review.<br>Removed AN56551 from associated application notes list as it is obsolete.<br>Updated template. |
| *D | 4821776 | RLIU | 07/03/2015 | Obsolete as per SHAS#128. |

OBSOLETE

## Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

## Products

| | |
|---|---|
| Automotive | cypress.com/go/automotive |
| Clocks & Buffers | cypress.com/go/clocks |
| Interface | cypress.com/go/interface |
| Lighting & Power Control | cypress.com/go/powerpsoc<br>cypress.com/go/plc |
| Memory | cypress.com/go/memory |
| PSoC | cypress.com/go/psoc |
| Touch Sensing | cypress.com/go/touch |
| USB Controllers | cypress.com/go/usb |
| Wireless/RF | cypress.com/go/wireless |

## PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

## Cypress Developer Community

Community | Forums | Blogs | Video | Training

## Technical Support

cypress.com/go/support

OBSOLETE

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2007-2014. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.
This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.
Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.
Use may be limited by and subject to the applicable Cypress software license agreement.