SGS-THOMSON MICROELECTRONICS

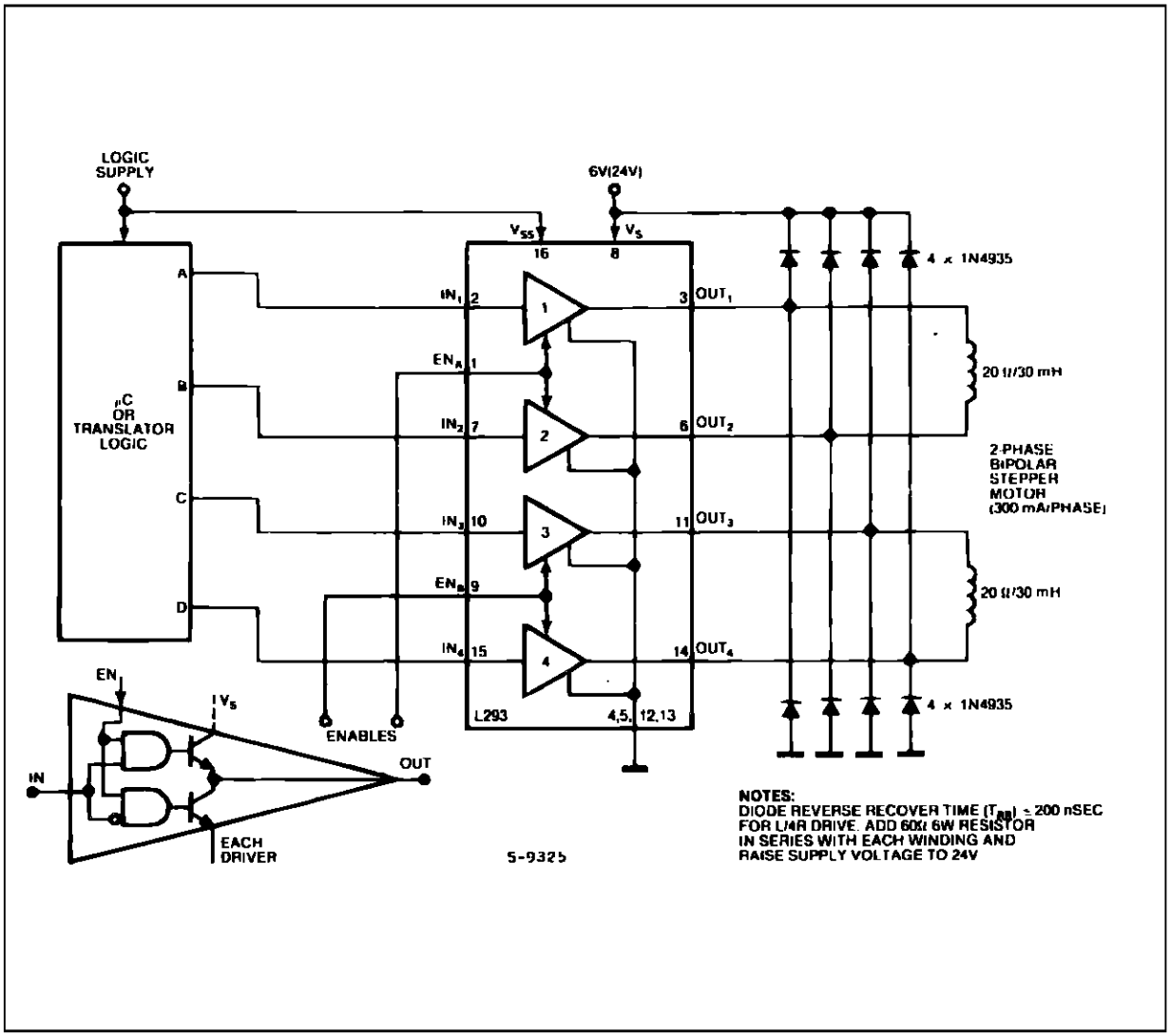
APPLICATION NOTE

HIGH-POWER, DUAL-BRIDGE ICs EASE STEPPER-MOTOR-DRIVE DESIGN

In addition to simplifying design problems, a family of dedicated chips improves stepper-motor drive-circuit reliability by significantly reducing the component count.

The L293, L293E and L298N dual-bridge ICs (see **box**, "inside the dual-bridge ICs") significantly reduce the problems encountered in the design of stepper-motor drive circuitry. They can, for example, simplify the design and increase the efficiency of constant-current choppers. And with a single chip replacing the transistors and predriver stages, circuit performance improves. Best of all, the devices have applications in complex as well as basic driver networks.

Figure 1 : The Simplest Stepper-motor Drive Technique is the Basic L/R Configuration. Adding Series Resistors and Raising the Supply to Make an L/4R Drive Improves Torque at High Steps rates but Reduces Efficiency.



SIMPLEST DESIGN IS AN L/R DRIVE

The simplest motor-drive configuration (fig. 1) consists of a µC that performs the translator function in software (see **box**, "Generating switching sequences") and drives the motor through a L293 dual bridge. Only eight external components are required ; these are diodes that protect the device's output transistors against inductive spikes generated when a winding de-energizes.

The L293 handles 1A continuous (for higher current use and L298N). However, if you plan to run the motor continuously with two phases on, dissipation will be the limiting factor.

You can improve the performance of this basic L/R drive by increasing the series resistance and raising the supply voltage to restore the original phase current. At high speeds, torque improves, but efficiency decreases. Normally, you increase each winding's resistance by a factor of four through the addition of a 3R series resistance, resulting in the L/4R drive.

INSIDE THE DUAL-BRIDGE ICs

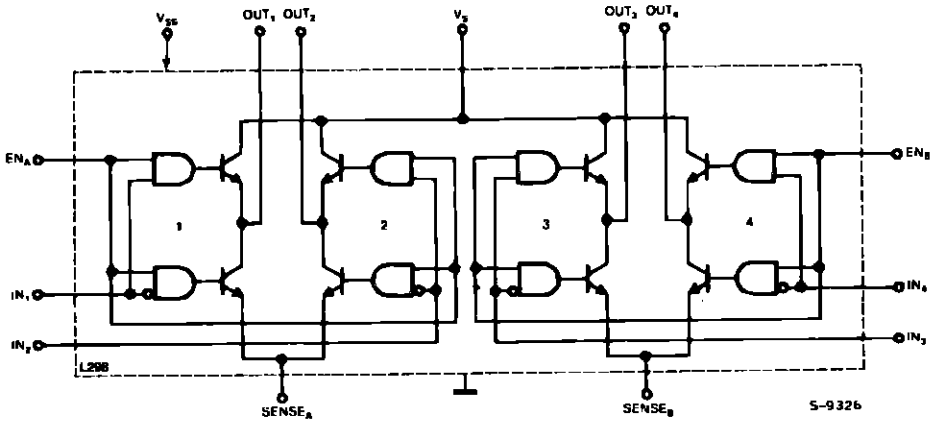
The L293, L293E and L298N (figure) contain two power-transistor bridges, predriver stages, control logic and protection circuitry. There's a control input for each bridge and an enable input for each half bridge ; inputs connect directly to µCs, CMOS or TTL. The ICs integrate level shifters with a separate logic-supply pin. For current sensing, the L293E and L298N have external emitter connections.

A single package drives a 2-phase bipolar stepper motor, challenging the assumption held by many that unipolar motors are easier to drive.

You can use a bipolar motor - simpler and less expensive than a unipolar motor - without building complex power stages. Furthermore, you don't have to worry about simultaneous conduction of a half bridge's source and sink transistors - a basic problem with discrete-component bridge circuits. Chip design makes it impossible for both transistors to be on at the same time.

Designers should also discard the mistaken idea that constant-current chopper drivers are complicated and expensive. You can build one with two bridge ICs and a few passive components.

Dual-bridge driver ICs reduce the parts count of bipolar stepper motors and simplify design. The schematic of the L298N is functionally similar to those of the L293 and L293E.



Type	I_O	$I_{O(PEAK)}$	V_S	Package	Sensing Connections
L293	1A	1.5A	36V	DIP16	
L293E	1A	1.5A	36V	DIP20	One per Half Bridge
L298N	2A	2.5A	46V	Multiwatt15	One per Bridge

MULTIPLE SUPPLIES BOOST PERFORMANCE

A dual-level supply also improves the performance of a basic L/R circuit. A high supply voltage yields good torque characteristics when the motor is running. A lower-than-rated voltage provides some holding torque when the motor is at rest, thereby saving power when the motor is idle.

Fig. 2 shows a suitable voltage-switch circuit. R_X sets the holding current, which can be low because a permanent magnet or hybrid stepper motor provides some holding torque at zero current. However, make certain the L293's motor-supply input never goes below the logic-supply voltage. While there's no danger of damaging the device, it's imposssible to drive the output transistors correctly under such conditions.

The dual bridge's enable inputs offer a means of extending the chip's flexibility. For example, you can connect them directly to the logic supply - no resistors are needed - to enable the chip permanently. As an alternative, use the enable inputs to disable the motor during the power-on reset sequence.

In wave-drive and half-step modes, use the enable inputs to increase torque at high speeds. When a winding de-energizes, flux collapse is a function of the current-decay rate. During this decay, the de-energized winding opposes the efforts of the next winding in sequence, partially cancelling the torque. You can minimize this effect by disabling a bridge only when the winding it drives is turned off ; because the $\Delta i/\Delta t$ of an inductor equals E/L, disabling the bridge accelerates the current decay. This action discharges the winding's stored energy through its supply and maintains the terminal voltage E at V_S plus two diode drops. If you were to leave the bridge enabled, the current would flow to ground through one diode and one transistor, and it would lower the terminal voltage. This scheme doesn't apply with drives with two phases on because no winding ever de-energizes.

Figure 2 : Switching the Supply to a Lower Voltage when the Motor is Idle Saves Current without Compromising Driving Power.

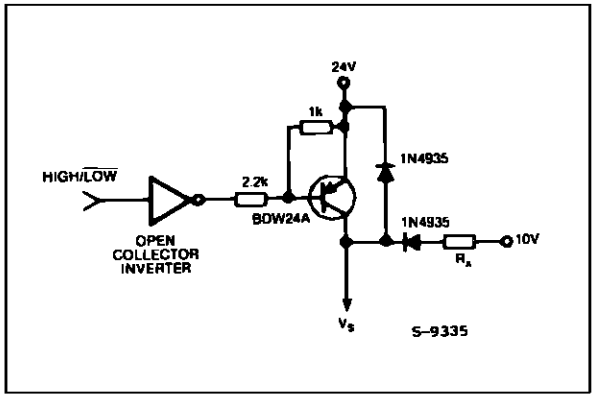
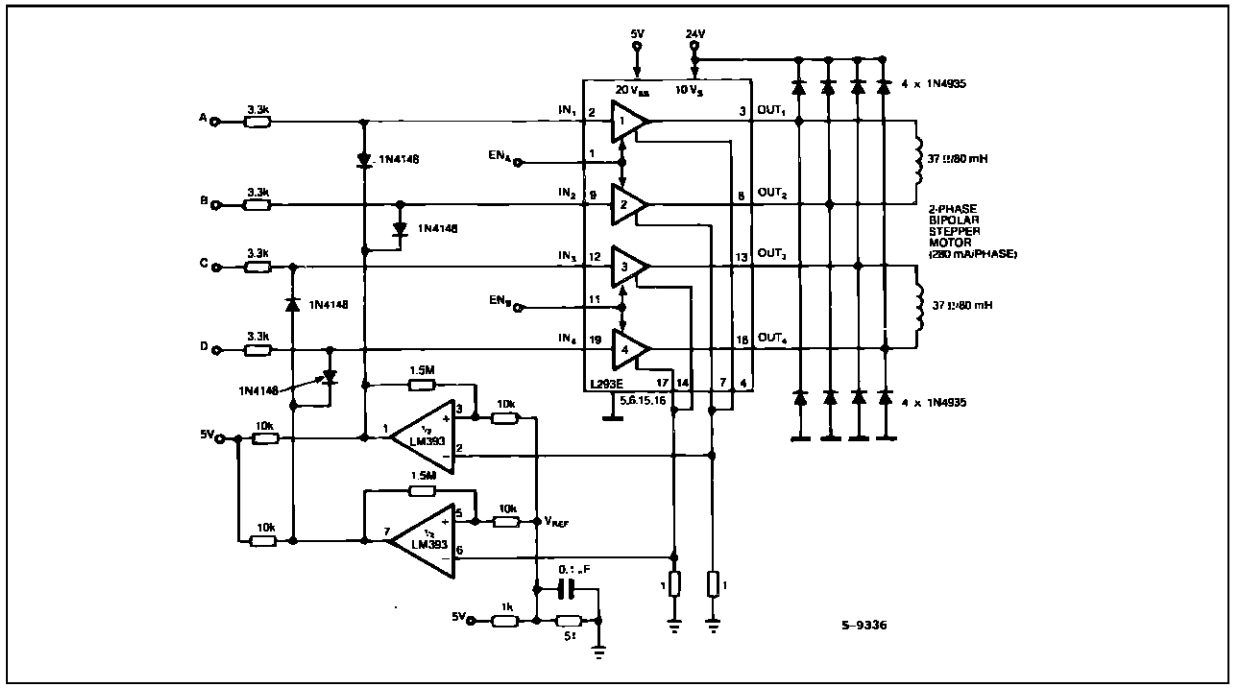


Figure 3 : Maintaining a Constant-average Phase Current this Fixed-ripple Chopper Provides Improved Performance and Efficiency V_{REF} Controls the Phase Current.



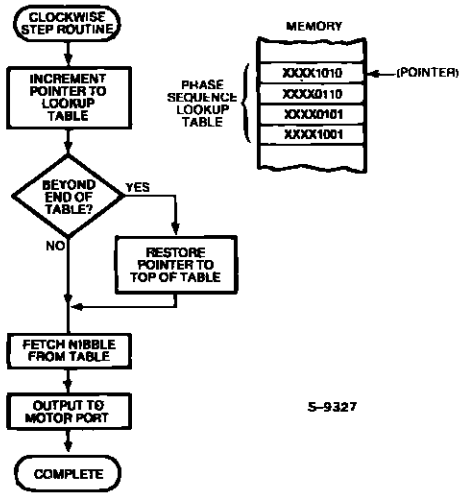
GENERATING SWITCHING SEQUENCES

In addition to selecting a motor and determining power-stage design, you must also decide how to generate the switching sequences that step the motor. Programming a µC or using a special piece of hardware called a transistor accomplishes this task.

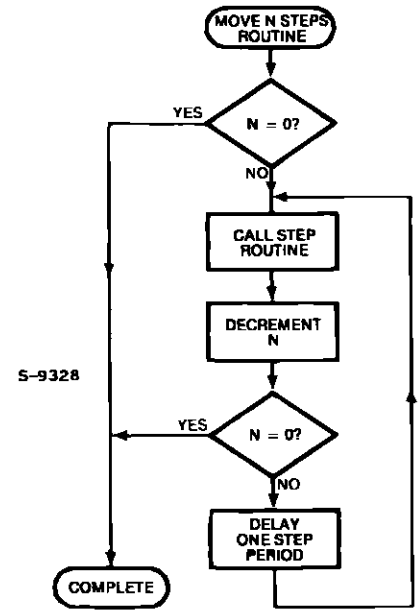
Software translation is more economical, and it is the first choice for large-volume products. Fig. A shows a basic step procedure (a) that you can integrate into a routine (b) ; the routine executes a clockwise rotation of N steps at a fixed rate. The step rate is defined by a software loop, but you can also use programmed timer interrupts.

Fig. A : A µC can generate the phase sequence (a) for a stepper motor. A routine (b) expands a single-step routine into are that executes a move of N steps.

a)



b)



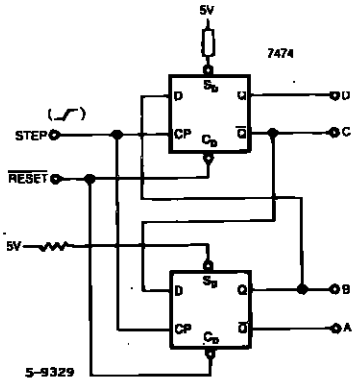
c)

A simpler approach uses the software equivalent of a shift register. For example, you can load a 99 (hex) into a register and take the phases from bits 0 to 3. A Rotate Left instruction yields a clockwise step ; a Rotate Right instruction causes a counterclockwise move.

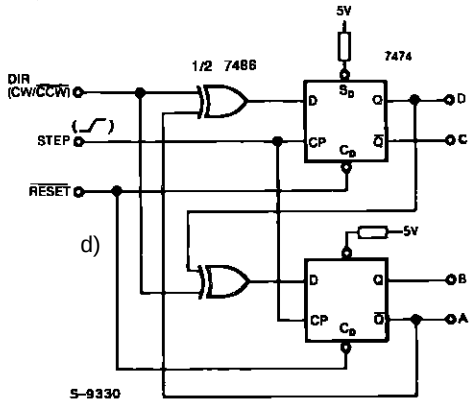
When software translation ties up your µC, lighten the load by adding a hardware translator. In applications involving unidirectional motors, this logic circuit (fig. B) requires only one pulse for each step ; you'll also need a direction signal (b) if your motor rotates in both directions. By adding a 7408 to a 2-phase translator, you can satisfy a wave-drive application (c), while the addition of two OR gates provides fast turn-off in wave-drive mode.

Fig. B : Built a simple 2-phase hardware translator using a dual flip-flop, for single (a) or bidirectional (b) rotation. Add some extra ICs for wave-drive signals (c) and to provide fast turn-off (d).

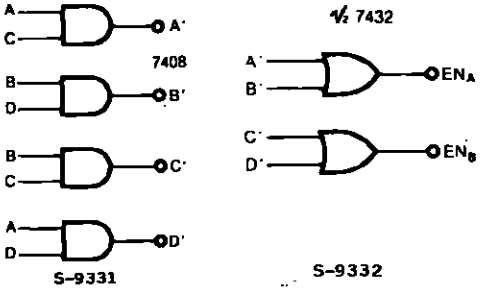
a)



b)



d)



Often a µC controls the translator, setting the direction line and providing a pulse for each step. Software is thus simplified, and if you use a programmed interrupt scheme, the µC is free to handle other tasks. Fig. C describes an absolute-positioning routine for a step with a direction-control translator ; Fig. D outlines how programmed timer interrupts are used to relieve the burden on the C.

Two special cases call for hardware translation. The first is in a system for which you have already designed in control circuitry to provide step and direction signals. The second case involves single-quantity and small-run applications, in which the cost of a few ICs is a small price to pay for simplified software.

Figure C : For use with a Hardware Translator this a Absolute-positioning Routine Sets the Direction Line and Sends the Appropriate Number of Step Pulses.

Notes : Enter with **desired** position **current** position in register or mem-

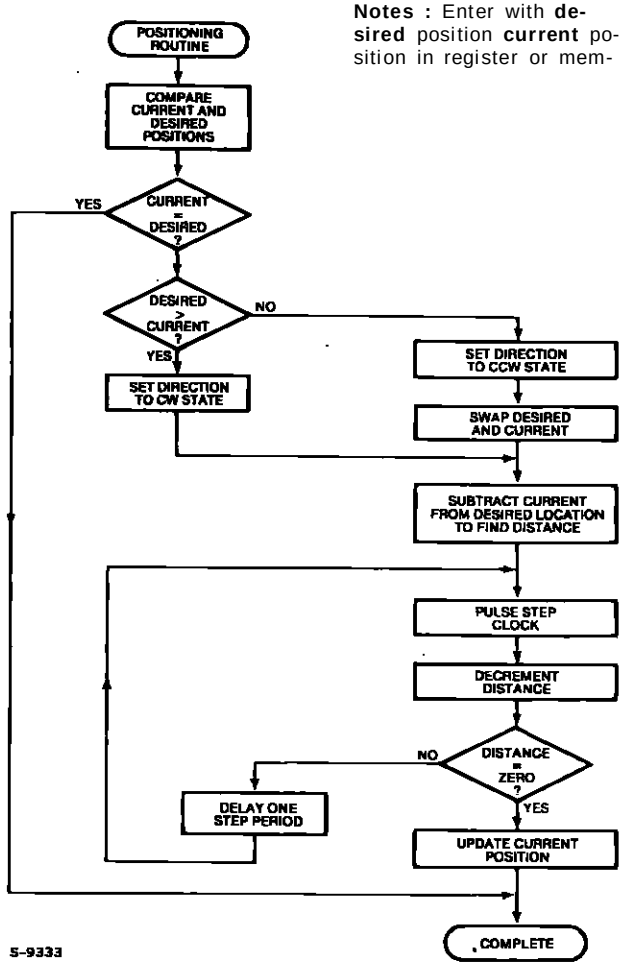
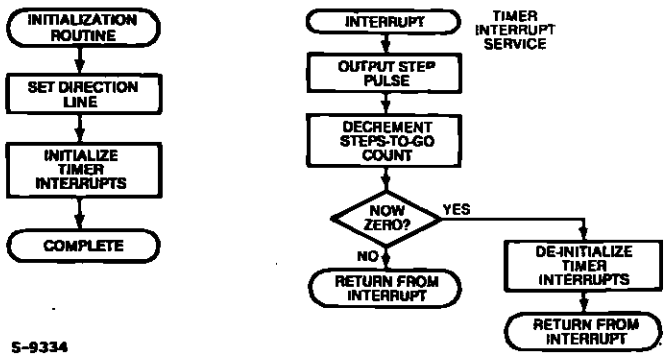


Figure D : To Set a motor Step Rate, used Programmed Timer Interrupts in Place of Software Timing Loops.



CHOPPER CIRCUIT OFFERS MORE ENHANCEMENTS

Adding a chopper circuit to maintain a constant-average phase current improves performance and efficiency. Fig. 3 shown a simple constant-current drive that employs a dual bridge, dual comparator and a few passive components. This circuit requires an L293E, because this dual-bridge IC offers access to the lower emitter connections, thus letting you insert current-sensing resistors.

Operation of this fixed-ripple chopper drive is straightforward. When the µC or translator activates a bridge, the increasing load current raises the voltage across the sensing resistor until it equals the comparator's reference voltage. The comparator then switches, clamping the translator signals through the diodes to deactivate the bridge. As the current decays, the voltage across the sensing resistor decreases until it equals the comparator's lower threshold. The comparator switches again, allowing the µC or translator to activate a bridge and restart the cycle. As long as the translator drives the bridge, this sequence repeats to provide a constant-average phase current with fixed ripple.

V_{ref} adjusts the lower current limit, while the comparator's hysteresis sets the ripple, and hence the peak current. Although a divider establishes the value of V_{ref} in this case, you can employ the dual-supply design approach and switch V_{ref} to a lower value when the motor is idle. In addition, use of a D/A converter establishes V_{ref} for micro-stepping applications. This drive circuit, with its fixed-ripple current characteristics, is well suited for such service.

FIXED-FREQUENCY CHOPPER IS MOTOR INDEPENDENT

Fig. 3's drive has some disadvantages. First, the chopper frequency depends on motor characteristics, and these parameters vary from unit to unit. In addition, it's impossible to synchronize the choppers, and this shortcoming can cause trouble on the ground plane.

Using a flip flop/comparator arrangement (fig. 4) to develop a fixed-frequency chopper overcomes these problems. In this circuit, the NE555 timer generates negative pulses that reset the flip flops to enable the phase-control signals from the translator. If these signals are set to energize a winding, the

Figure 4 : This Fixed-frequency Constant-current Chopper Driver Enables the Synchronization of Several Drives, thus Minimizing Potential Ground-plane Problems.

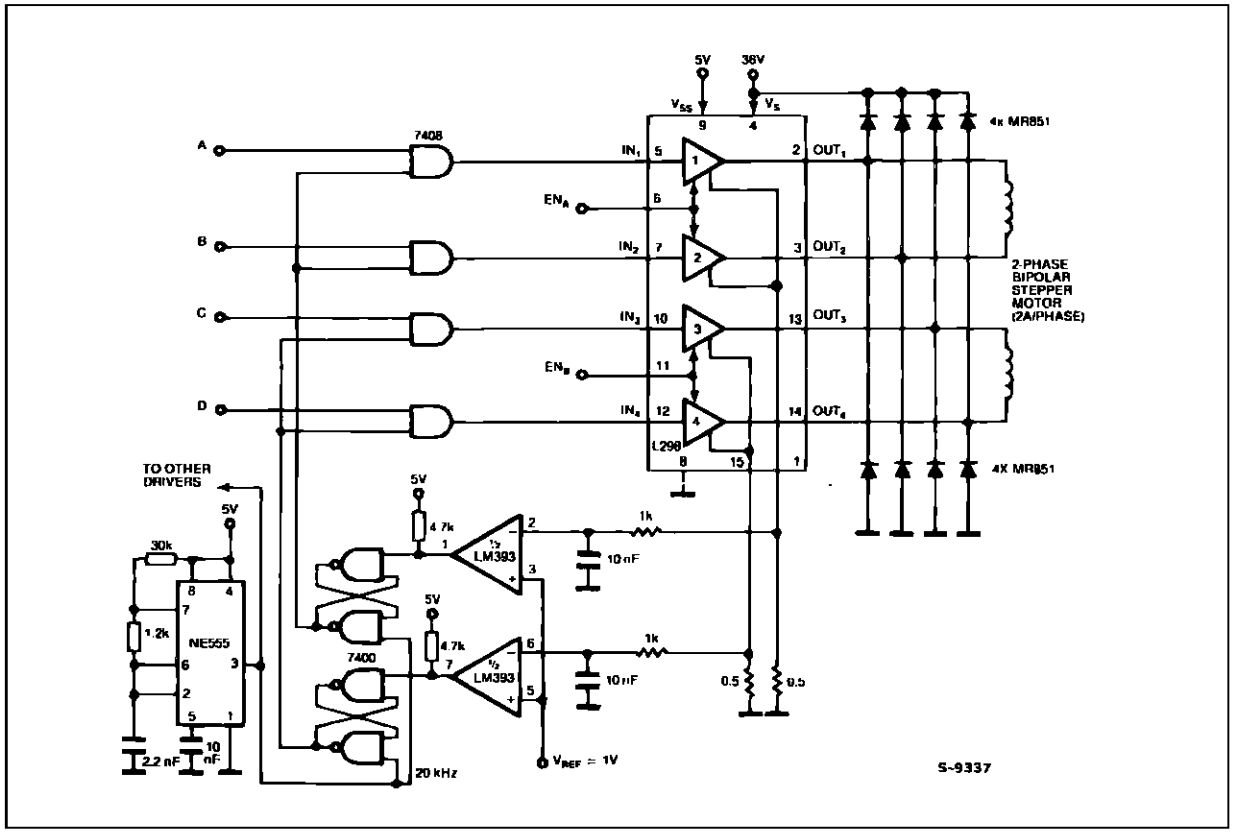
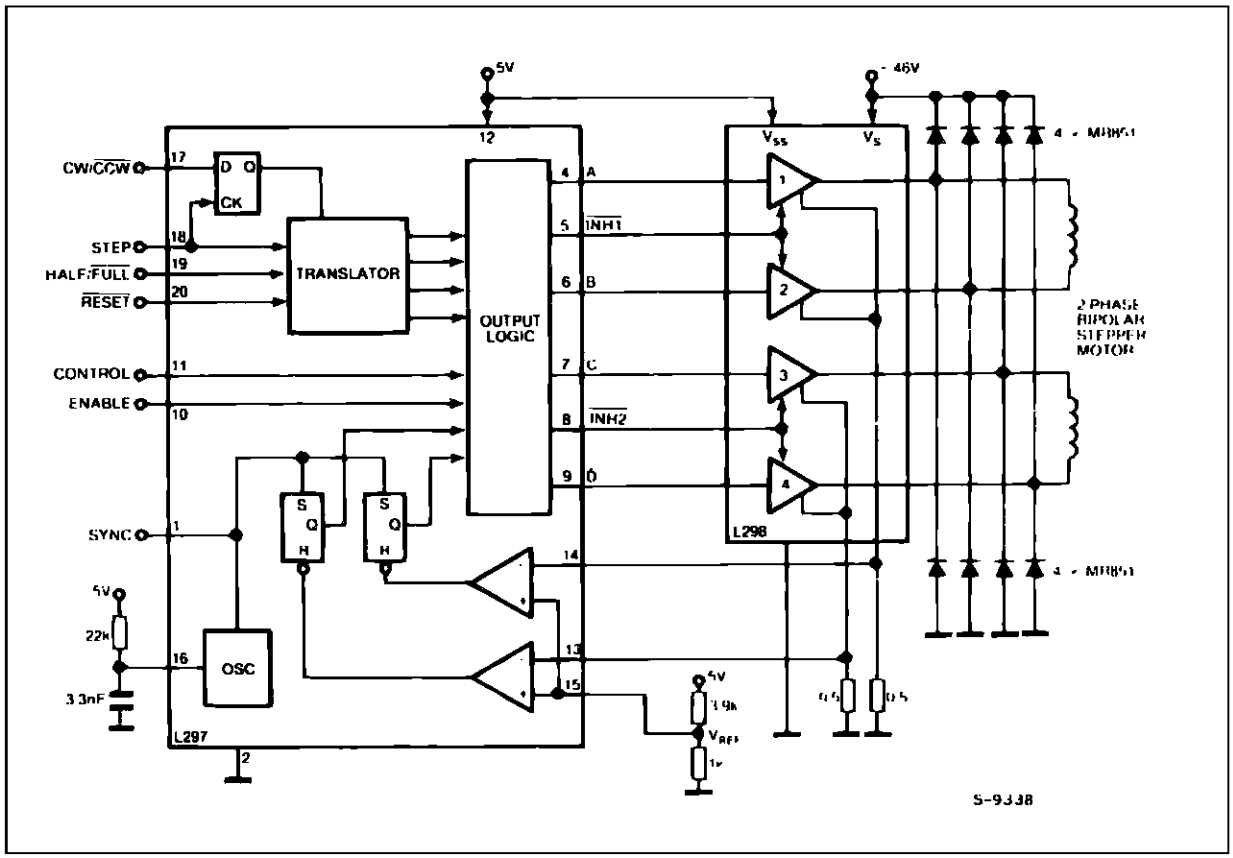


Figure 5 : A Special Translator-chopper Control Circuit cuts the Drivers Components Count to the Minimum.



current in that winding rises until the voltage across the sensing resistor switches the comparator, thus setting the flip flop. This disables the phase signals and deactivates the bridge. Current in the winding falls until the next clock pulse resets the flip flop, and the sequence repeats to maintain a constant current.

CONTROLLER IC REDUCES COMPONENT COUNT

If you're using a hardware translation and constant-current choppers, you can further reduce the component count by using a controller chip such as the L297 – a 20-pin DIP that houses a translator and a dual fixed-frequency chopper circuit. Under the control of step and direction inputs, the L297 generates normal, wave-drive and half-step sequences.

As shown in fig. 5, the controller connects directly to a dual bridge. External component requirements are minimal : and RC network to set the chopper frequency and a resistive divider to establish the comparator reference voltage (V_{ref}).

To accommodate motors with a phase current as great as 3.5 A, replace the single dual-bridge IC with two devices configured in parallel (input to input, enable to enable, etc) to form a single bridge. It's extremely important that you pair the half bridges – 1 with 4 and 2 with 3 – to ensure optimum current sharing.

Reprinted from EDN. 11/24/83