# AN1495
# APPLICATION NOTE

# MICROSTEPPING STEPPER MOTOR DRIVE USING PEAK DETECTING CURRENT CONTROL

by Thomas Hopkins

*Stepper motors are very well suited for positioning applications since they can achieve very good positional accuracy without complicated feedback loops associated with servo systems. However their resolution, when driven in the conventional full or half step modes of operation, is limited by the configuration of the motor. Many designers today are seeking alternatives to increase the resolution of the stepper motor drives. This application note will discuss implementation of microstepping drives using peak detecting current control where the sense resistor is connected between the bottom of the bridge and ground. Examples show the implementation of microstepping drives with several currently available chips and chip sets.*

## Introduction

Microstepping a stepper motor may be used to achieve one or both of two objectives; 1) increase the position resolution or 2) achieve smoother operation of the motor. In either case the basic theory of operation is the same.

The simplified model of a stepper motor is a permanent magnet rotor and two coils on the stator separated by 90 degrees, as shown in Figure 1. In classical full step operation an equal current is delivered to each of the coils and the rotor will align itself with the resulting magnetic vector along one of the 45 degree axis. To step the motor, the current in one of the two coils is reversed and the rotor will rotate 90 degrees. The complete full step sequence is shown in figure 2. Half step drive, where the current in the coil is turned off for one step period before being turned on in the opposite direction, has been used to double the step resolution of a motor. In either full and half step drive, the motor can be positioned only at one of the 4 (8 for half step) defined positions.[4][5] Therefore, the number of steps per electrical revolution and the number of poles on the motor determine the resolution of the motor. Typical motors are designed for 1.8 degree steps (200 steps per revolution) or 7.5 degree steps (48 steps per revolution). The resolution may be doubled to 0.9 or 3.75 degrees by driving the motor in half step. Further increasing the resolution requires positioning the rotor at positions between the full step and half step positions.

**Figure 1. Model of stepper motor**

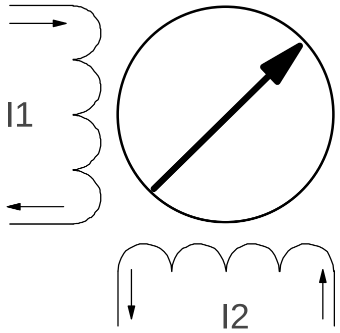
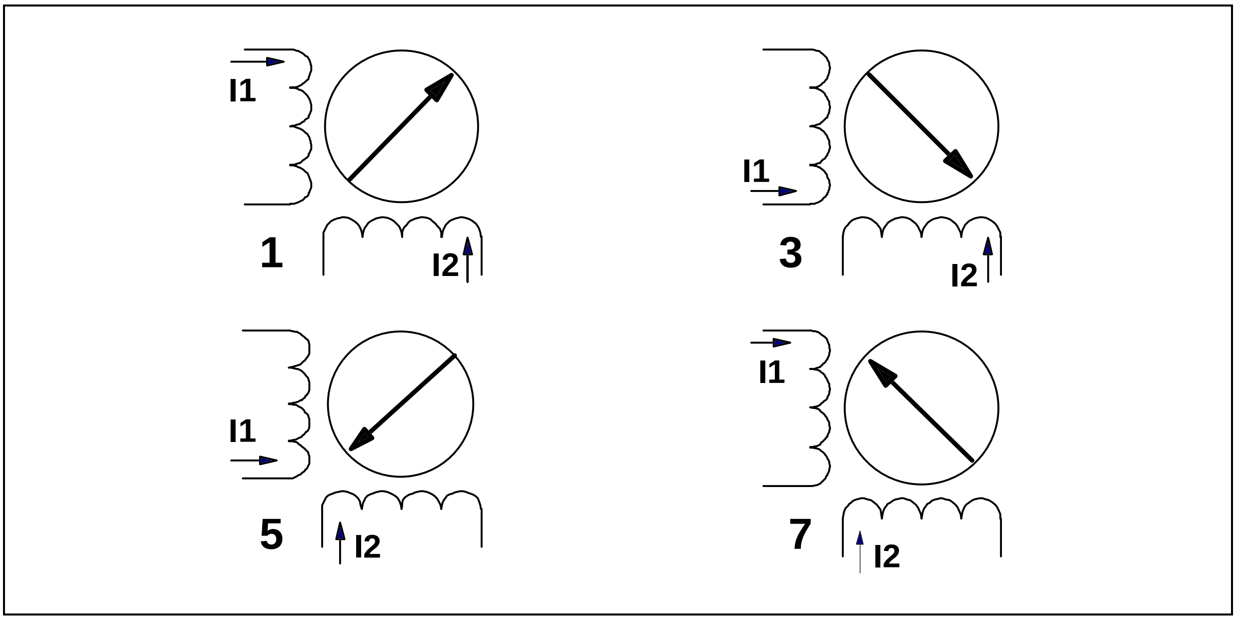


Figure 2. Full step sequence.



Another issue occurs at low operating speeds. At low speeds, both the full and half step drive tend to make abrupt mechanical steps since the time the rotor takes to move to the next position can be much less than the step period. This stepping action contributes to jerky movement and mechanical noise in the system. Looking at the simplified model of the stepper motor in Figure 1, it can be seen that if the two coils were driven by sine and cosine waveforms the motor would operate as a synchronous machine and run very smoothly. These sinusoidal waveforms may be produced by a microstepping drive .

Microstepping can be implemented in either a voltage mode or current mode drive. In voltage mode drive, the appropriate duty cycle would be generated by the controller so that the voltage applied to the coil (Vsupply * duty cycle) is the appropriate value for the desired position. In current mode drives, the winding current is sensed and controlled to be the appropriate value for the desired position. This application note will consider only current mode drive implemented using peak detecting current controllers.

To understand the microstepping concept, consider the simplified model of the stepper motor as shown in Figure 1. As previously discussed when the two coils are energized with equal currents, the resulting magnetic vector will be at 45° and the permanent magnet of the rotor will align with that vector. However, if the two coils are energized by currents of different magnitude, the resulting magnetic vector will be at an angle other than 45° and the rotor would attempt to align with the new magnetic vector. If one coil were driven with a current that was twice the current in the second coil the magnetic vector would be at 30°, as shown in Figure 3. For any given desired position, the required currents are defined by the sine and cosine of the desired angle.

To implement a microstepping drive, two D/A converters are used to set the current level in the coils of the motor, as shown in the block diagram in Figure 4.

**Figure 3. Example alignment of microsteping**

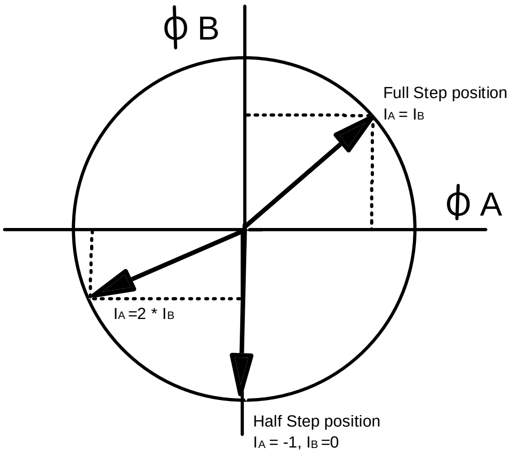
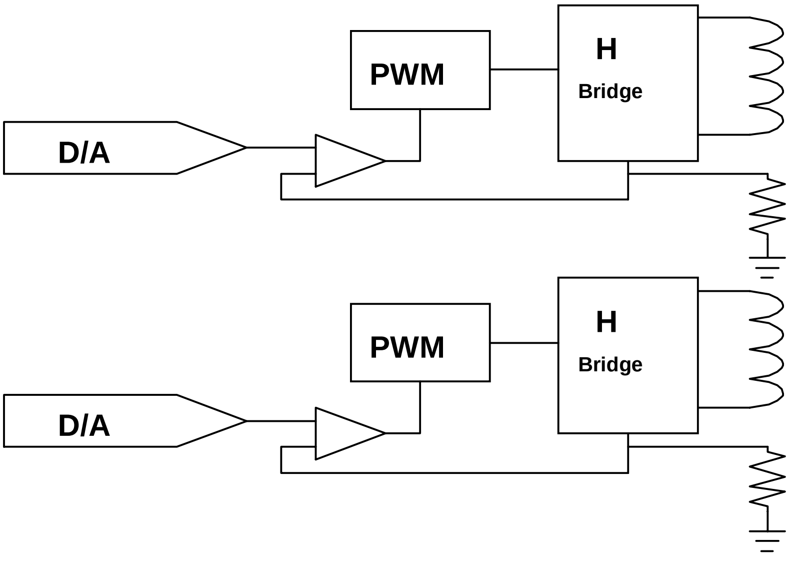


**Figure 4. Block Diagram of microstepping motor drive.**



### Microstepping with the L6208

In a typical application the L6208, which integrates two H-Bridges with the current control, drives a bipolar stepper in either full or half step modes. The internal state machine generates the full step or half step sequence from the clock and direction inputs. [1] Although at first glance it is not obvious that the L6208 may be used in a microstepping application, it is possible since the current control circuits have separate reference inputs.

To implement a microstepping application, a variable voltage proportional to the desired output current must be applied to each of the reference pins. In the block diagrams above, the two required D/A converters provide the required voltages. A simple and inexpensive alternative to a D/A converter chip is to use a counter/timer in the microprocessor to generate a PWM output for each phase and pass this through a voltage divider and low pass filter to get the desired voltage. The Vref input voltage is equal to the microprocessor power supply voltage times the divider ratio of the resistor divider times the PWM duty cycle. Figure 5 shows the connection between a microcontroller and the L6208. The complete circuit schematic for the power section is shown in appendix A.

Since the L6208 includes an internal phase generation circuit, this circuit must be synchronized to the externally provided reference voltages. Again a simple solution is possible. The initial state of the decoding logic after reset is known and may be used as the starting state. After applying a reset to the L6208, either at power up or by forcing a reset from the microprocessor, the full-scale voltage is applied to both Vref pins to align the stepper motor to the known state that corresponds to one of the full step positions. Once the motor is aligned, the references can be reduced to 70.7%, which is the correct value for the currents for the 45-degree position in the microstepping sequence. After the motor is aligned the microcontroller can move through the sine/cosine table to generate the appropriate reference levels to move in either direction. The software also has to set the appropriate direction on the CW/CCW pin and generate a clock pulse for each phase reversal that is required. This occurs whenever the phase crosses a 90° boundary in the sine table. By operating the L6208 in the full step mode and providing clock signals at the appropriate time, the decoding logic will output the correct phase information for the bridges. Using the L6208 in the half step mode with the appropriate clock signals can improve the performance at the zero cross over of the current, as will be discussed later.

**Figure 5. Circuit connections for the L6208**

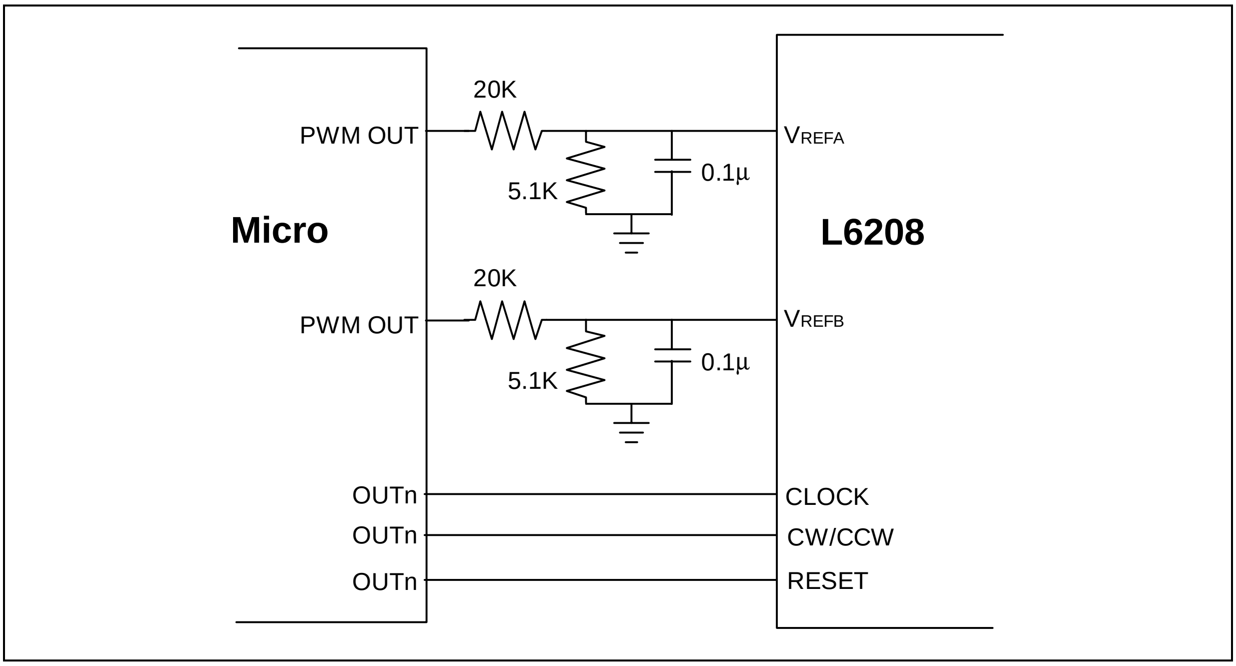
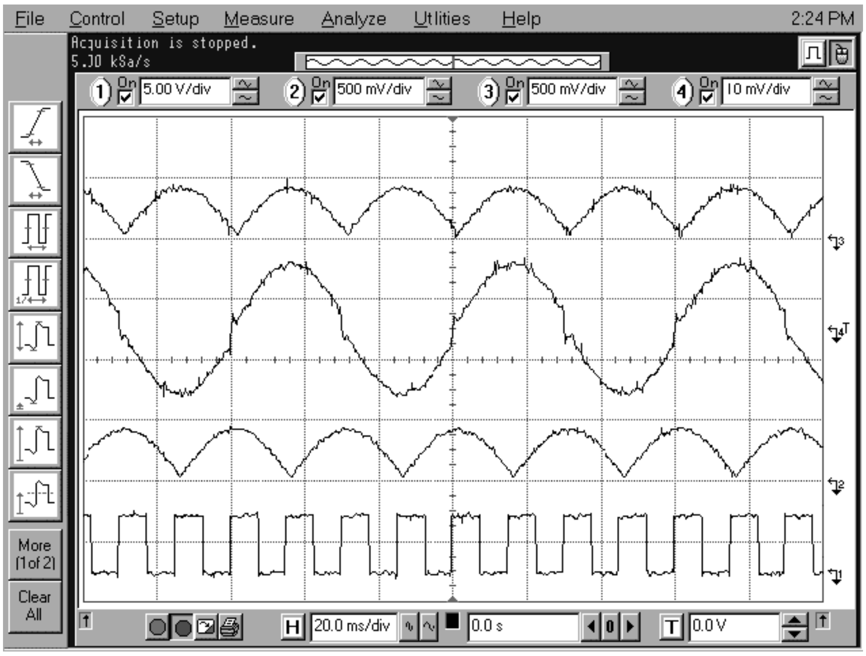


Figure 6 shows the operating waveforms when using the L6208 in full step mode and varying the reference inputs to achieve microstepping. Trace 1 is the clock input to the L6208. Traces 2 and 3 on the plot are the VrefA and VrefB inputs applied to the L6208. Trace 4 is the motor current in channel B. Although the current has the discontinuities near zero that are typical of a peak detection current control method, the resulting output matches the desired sine wave reasonably well.

**Figure 6. Microstepping waveforms: Typical Operation**



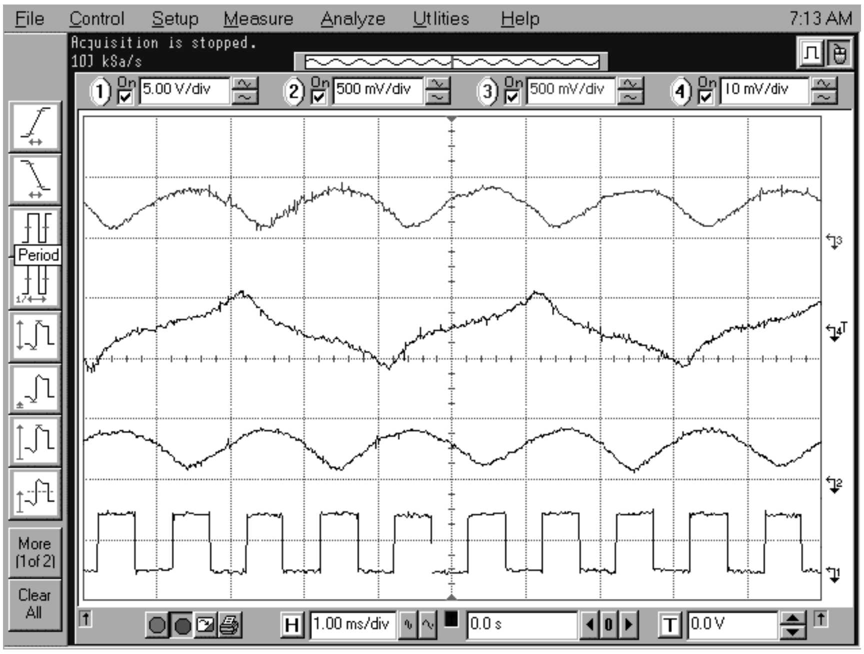
2:24 PM

Acquisition is stopped.
5.00 kSa/s

1 On 5.00 V/div
2 On 500 mV/div
3 On 500 mV/div
4 On 10 mV/div

H 20.0 ms/div 0.0 s
T 0.0 V

**Figure 7. Microstepping waveforms: Current can not follow desired sine wave**



7:13 AM

Acquisition is stopped.
100 kSa/s

1 On 5.00 V/div
2 On 500 mV/div
3 On 500 mV/div
4 On 10 mV/div

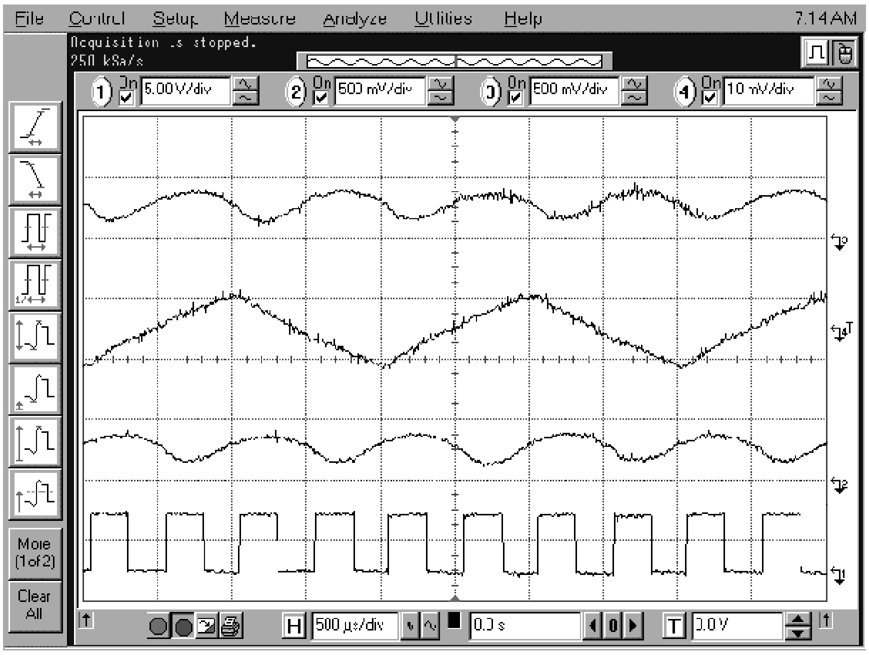
H 1.00 ms/div 0.0 s
T 0.0 V

## Speed Limitations

Since the motor coil is primarily an inductance, the rate of current change in the coils is limited by the L/R time constant of the motor. As the motor is operated at higher speeds, the L/R time constant of the motor limits the rate of current change and the current can no longer follow the desired sine wave. Figure 7 shows the motor current at a higher rotational frequency. On this scope trace, we see two effects. First, the filter on the reference voltage is starting to roll off the reference signal and second, the motor current is limited by the motor time constant and it begins to look more like a triangle waveform than the desired sine wave. Although moving the pole of the filter on the reference voltage will make the reference signal appear more ideal, it will have little effect on the motor current at this point since the motor current is primarily limited by the L/R characteristics of the motor. When approaching this point, the motor will run smoothly in full step mode and the microprocessor could easily change to full step drive.

If the step rate is increased further, the motor will stall when the current can no longer reach a value large enough to produce the required torque. Figure 8 shows a typical current waveform when the motor has stalled. The almost pure triangular current waveform is similar to the triangular waveform that would result if the motor were being driven in the full step mode at this step rate. At this operating point the current is entirely controlled by the L/R time constant of the motor and no chopping is occurring.

**Figure 8. Waveforms when motor has stalled**



Acquisition is stopped.
250 kSa/s

1 On 5.00 V/div
2 On 500 mV/div
3 On 500 mV/div
4 On 10 mV/div

H 500 µs/div
0.0 s
T 0.0 V

### Slow vs. Fast Decay mode

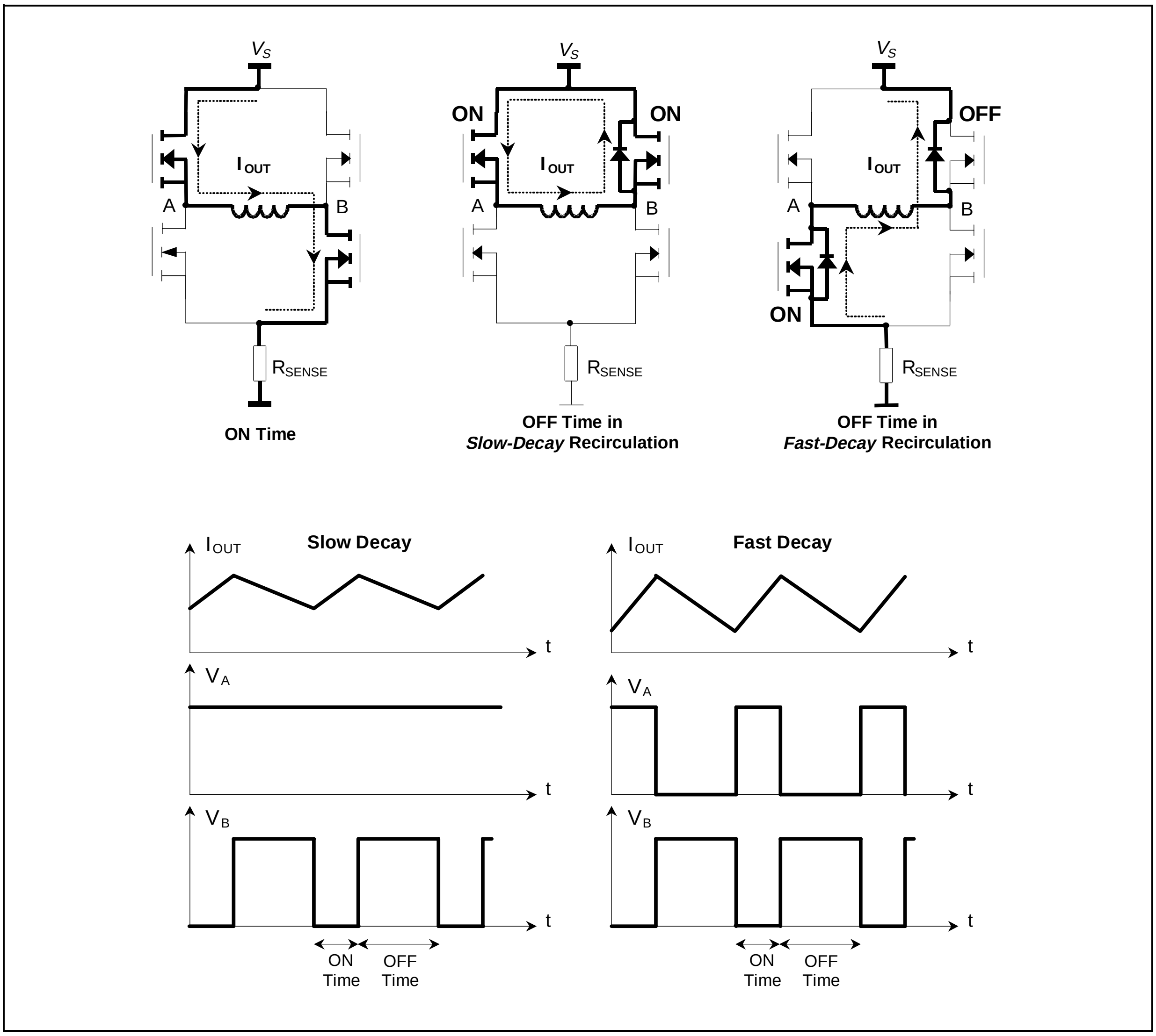
When implementing current controlled motor drives, the designer has a choice of the recirculation path the current flows in during the "off" time. Figure 9 shows the two recirculation options implemented in the L6208. Applying the chopping to only one side of the bridge allows the current to recirculate around a low voltage loop, in the upper transistors with the L6208. Since the rate of change of the current is controlled primarily by the L/R time constant of the motor, the current decays relatively slowly, hence the designation of slow decay mode. However applying the chopping to both sides of the bridge results in the current recirculating back to the power supply and a higher voltage across the coil, hence a fast decay mode. The L6208 also implements a type of synchronous rectification that turns on the MOS transistor in parallel with the conducting diode to reduce the power dissipation. [1]

The selection of the decay mode influences the operation of a microstepping drive in several ways. The most obvious is the magnitude of the ripple current. Drives implemented using the fast decay mode will have, for the same off time or chopping frequency, a higher ripple current than drives implemented using a slow decay mode. This difference in itself is not significant for most stepper motor drives. Issues with the stability of the current control loop are discussed elsewhere [3].

When microstepping at a relatively high speed, the selection of the decay mode affects the ability of the drive to follow the desired current level. At any time, the rate of change of current is determined by the inductance of the motor and the voltage across the coil. In the slow decay mode, the voltage across the coil during the off time is only the drop across one transistor and one diode so the current changes very slowly. As the desired current level is lowered, it is the rate of change during the off time that determines how quickly the current transitions to the new level. At low speeds, the effect may not be too noticeable. However, at higher speeds, the motor current cannot decay fast enough to follow the desired decreasing slope of the sine wave. During this time the current change is limited by the time constant imposed by the motor inductance and the slow decay path and can remain higher than the set value. The current will continue to decay at the slow rate until a phase reversal occurs, at which point the bridge reverses, applying the full supply voltage across the coil, effectively putting the bridge in a fast decay mode and the current will decay quickly to zero. Selecting the fast decay mode can improve the ability of the drive to follow fast decreases in the current. The waveforms in Figure 6 are achieved using the fast decay mode.

The ability of the drive to increase current on the upper slope of the sine wave is not affected by the choice of the decay mode since the voltage applied to the coil during the on time is the same.

**Figure 9. PWM current control decay modes.**



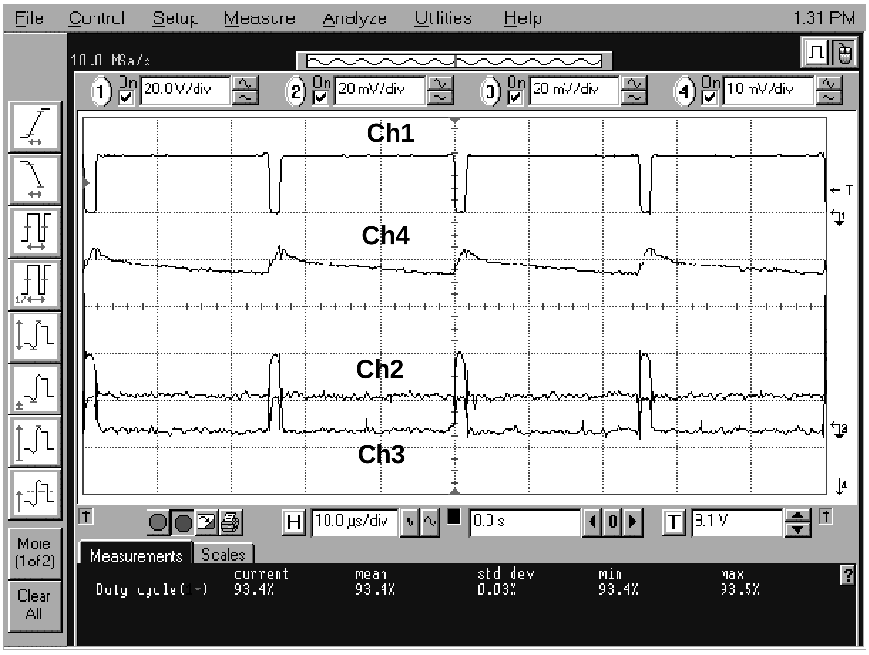
### Minimum current issues

When operating a chopping current control that has a minimum duty cycle, the current cannot be taken below a level that is effectively set by the motor resistance and the minimum duty cycle. Constant off time controls, like the L6208, have a minimum on time that is set primarily by the propagation delays from the end of the off time until the comparator detects a current above the threshold and retriggers the monostable putting the bridge in the recirculation mode again. This minimum on time and the off time set by the monostable set a minimum working duty cycle for the circuit. When this duty cycle is applied to the motor, a current will be established. If a reference corresponding to a current lower than this minimum is set on the input, the circuit will detect that the motor current is above the reference. However, since the IC is already operating at its minimum duty cycle, the current can not go any lower and thus will not reach the current level desired by the reference level. The minimum duty cycle in other controllers can some times be adjusted. The minimum on time in the L6506, for example, is set by the width of the sync pulse. By varying the duty cycle of the oscillator, the minimum duty cycle of the output can also be changed. Since the sync pulse is also used to mask the switching noise in the system, reducing the minimum duty cycle is not always possible. [3]

Figure 10 shows the operating waveforms at the minimum current level. The traces in the oscilograph are:

| | |
|---|---|
| Ch 1 : | Voltage on output pin |
| Ch 2 : | $V_{ref}$ |
| Ch 3 : | $V_{sense}$ |
| Ch 4 : | Load current (20mA/div) |

**Figure 10. Oscillograph of the minimum current**



10.0 MSa/s

1 On ☑ 20.0V/div
2 On ☑ 20 mV/div
3 On ☑ 20 mV/div
4 On ☑ 10 mV/div

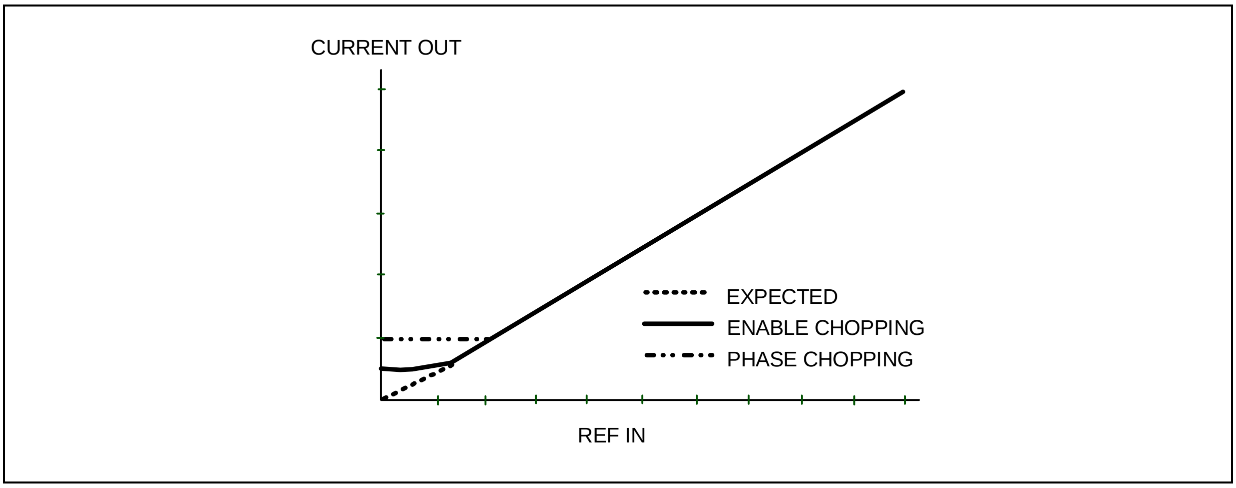
H 10.0 µs/div
0.0 s
T 3.1 V

```
                  current   mean    std dev   min     max
Duty cycle(1-)    93.4%     93.4%   0.03%     93.4%   93.5%
```

At the start of each cycle the bridge is turned on and the motor current flows through the sense resistor to produce the voltage $V_{sense}$. However at this operating point the sense voltage is already greater than the Vref input voltage, as can be seen in Figure 10. The comparator will detect that $V_{sense}$ is greater than Vref and cause the circuit switch the bridge into the recirculation mode and the output is switched off after a delay that is determined by the response time of the circuit. The output pulse width, and hence the operating current, are set by the response of the circuit to a condition where the current sense comparator detects a current above the set value as soon as the drive is turned on. Since this pulse width can not be reduced further, the current that flows is the minimum that the device can regulate. In Figure 10, the minimum current is approximately 100mA.

The minimum current level means a nonlinear transfer function exists between reference in (usually a voltage) to current out. Figure 11 shows the resulting transfer function between reference and output current.

**Figure 11. Transfer function showing nonlinearity**

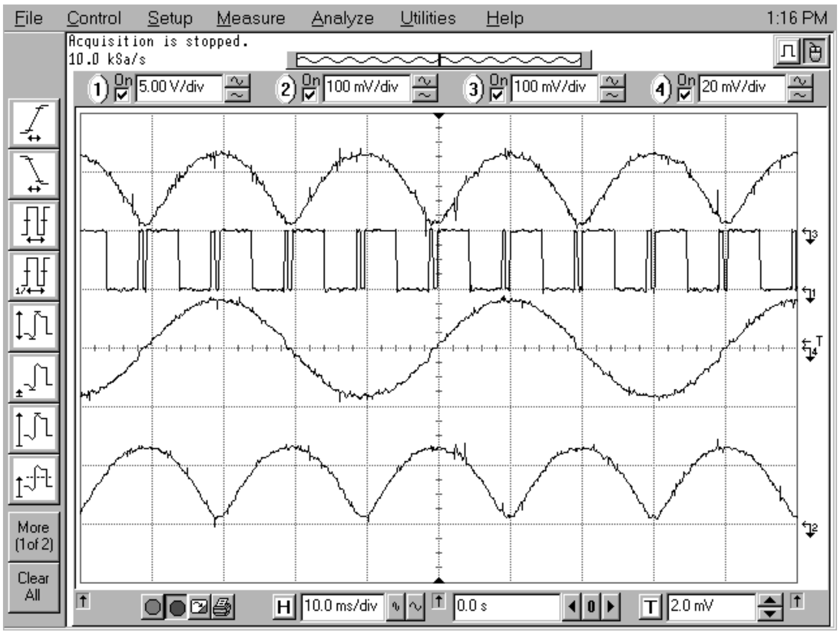


The transfer function also depends on the chopping mode, fast decay (enable chopping) or slow decay (phase chopping) as shown in Figure 11. In slow decay mode the current changes very slowly during recirculation and has a small ripple value. When operating in fast decay the transfer function also has a discontinuity in the slope at low levels. At the minimum current level, the duty cycle is small and when operating at this point the current typically is discontinuous, that is the current rises to a peak value and decays back to zero during each cycle. The flat section of the current transfer function corresponds to this minimum current. When the reference is increased, the device begins to regulate current however the device will still operate in the discontinuous mode. Continuing to increase the reference, the device will begin to operate in the continuous current mode, where the current does not decay to zero in each cycle. When the current changes from discontinuous to continuous, the slope of the transfer function changes. The result is that there are two discontinuities in the transfer function, one set by the minimum current and one set by the change in slope. In theory the slow decay mode could also have two discontinuities, however in practical examples the minimum current is reached before the current goes discontinuous.

The minimum achievable current effectively sets a limit on the number of microsteps per step by setting minimum current for the first microstep. Since the fast decay mode has a lower minimum current, fast decay can be used to minimize the effect of the minimum current, but will introduce another error due to the change of the slope. The latter can be compensated for by adjusting the DAC value.

It is, however, possible to get zero current in a phase by disabling the bridge when zero current is desired in that phase. When using drivers that have an enable input for each bridge simply disabling the bridge will force the current to zero. The L6208, however, does not have a separate enable input for each bridge so we need to use another trick of the logic to disable the bridge at the appropriate time. When driven in the half step mode, one bridge is disabled in each of the even states [2]. This operating sequence can be used to disable the bridge at the appropriate times. To achieve this, operate the L6208 in half step mode and apply a pulse to the clock input at the same time that the desired current is set to zero. At the next change of current apply a second pulse to the clock input and set the current value for the first microstep. The step sequence generator in the L6208 will cause the change from current in one direction in the bridge, to the bridge being disabled, to current in the reverse direction as shown in Figure 12.

**Figure 12. Microstepping waveforms with improved performance at zero current.**



Acquisition is stopped.
10.0 kSa/s

1 5.00 V/div
2 100 mV/div
3 100 mV/div
4 20 mV/div
H 10.0 ms/div
0.0 s
T 2.0 mV

The effects of the minimum current can be seen in the motor movement as errors in the motor position or as a jerky movement in a constant speed movement. How much the minimum current affects the drive depends primarily on the number of microsteps implemented per step. Since zero current can be achieved as described above, the positions at the 90 degree intervals where one coil is driven by zero current and the other is driven by the full scale current can easily be implemented. However, the next microstep where the current in one coil is small is most affected. If the desired current for any position is less than the minimum current, an error occurs. If the current required for the first microstep after the zero current position is greater than the minimum current, no error is contributed. Fortunately, since the desired current profile is a sine wave, the first step after the zero crossing has the largest relative increase in current of any microstep. If the required current for this first microstep is greater then the minimum current the device can regulate, there will be no error in the current to the motor due to the minimum current.

If the design required that one step (90 Deg.) be divided into 16 microsteps, the angle for the first step would be 5.625 Deg. The sine of 5.625 degrees is 0.098. When using an 8-bit D/A, the closest available value would correspond to an input of 25 out of 255. No other microstep needs a current less than this (except the 0 as discussed above). As long as the minimum current is less than the value corresponding 25/255 of the peak current, there will be no noticeable error contributed by the minimum current. Another way to express this that no error will be noticeable if the output current can be regulated to plus or minus 1 LSB over the range 24 to 255. There is no system level requirement to maintain the accuracy for inputs less than 24.

### L6506 & L6203/L298

Microstepping drives can be implemented using the L6506 controller and bridge IC's like the L6201, L6202, L6203 and L298. The main difference between the standard half step application and a microstepping application is that the two references of the L6506 are set by D/A converter outputs. Figure 13 shows a microstepping application using the L6506 and the L6203. Outputs Px 1 through Px 4 from the microprocessor set the phase for the L6506/L6203 combination and output Px 5 and PX 6 are used to enable the bridges. Again, the D/A function could be implemented using the PWM outputs of the microprocessor as was done in the example above or it could be implemented using and integrated D/A. The same logic configuration can be used with the L6201, L6202 or L298.

When using the typical connection between the L6506 and the L6203 (as shown in Figure 13a), the PWM signal is applied to one of the phase inputs (the phase that is normally high) and you get the slow decay mode of operation.

To implement the fast decay mode of operation, the PWM signal needs to be applied to the ENABLE inputs of the L6203s. This can be accomplished by rearranging the connections from the microprocessor. Inputs IN 1 and IN 2 of the L6203s are disconnected from the L6506 and connected directly to the Px 1 through Px 4 outputs of the microprocessor, which will continue to provide the phase information as before. The two PWM current control loops in the L6506 are then used to control the ENABLE inputs of the two L6203, as shown in Figure 13b. Px 5 and Px 6 are now connected to the inputs of the L6506 so that each bridge can be disabled to get zero current. Finally the Power On Reset (POR) is connected to the RESET input of the L6506 to disable the bridge during power up.

**Figure 13a. Microstepping using L6506 and L6203 (Slow Decay)**

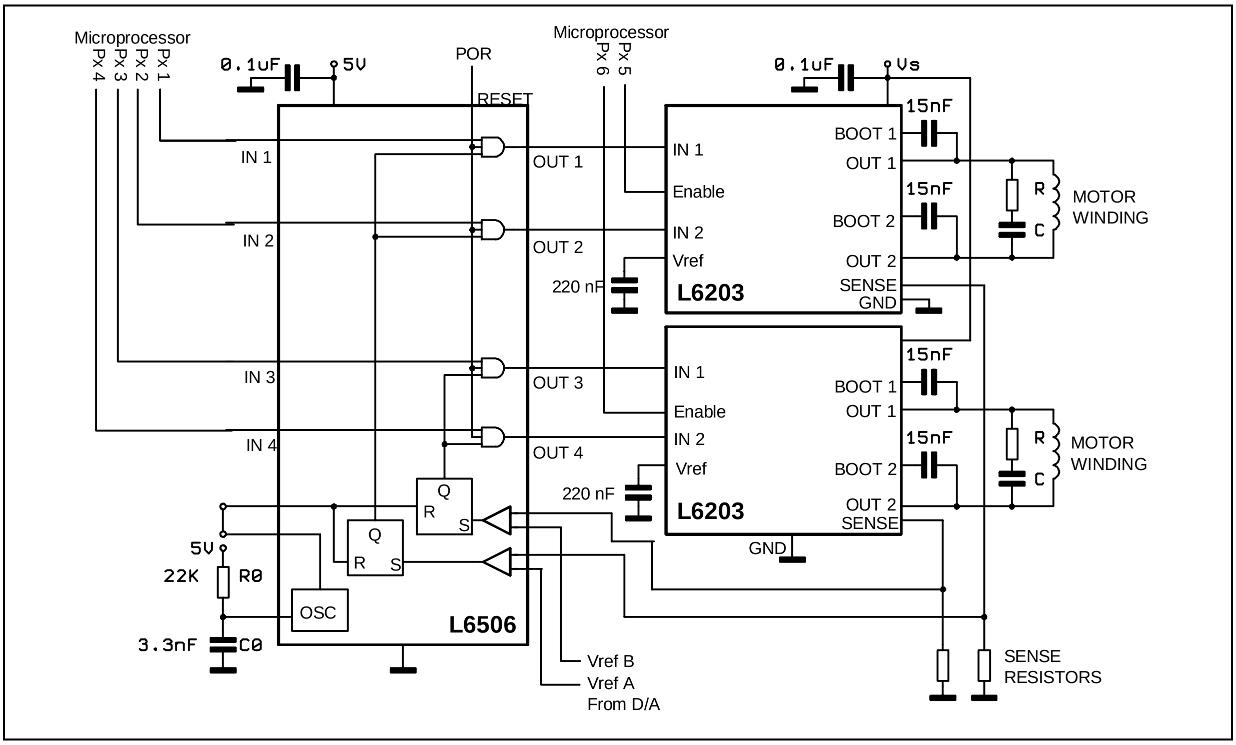
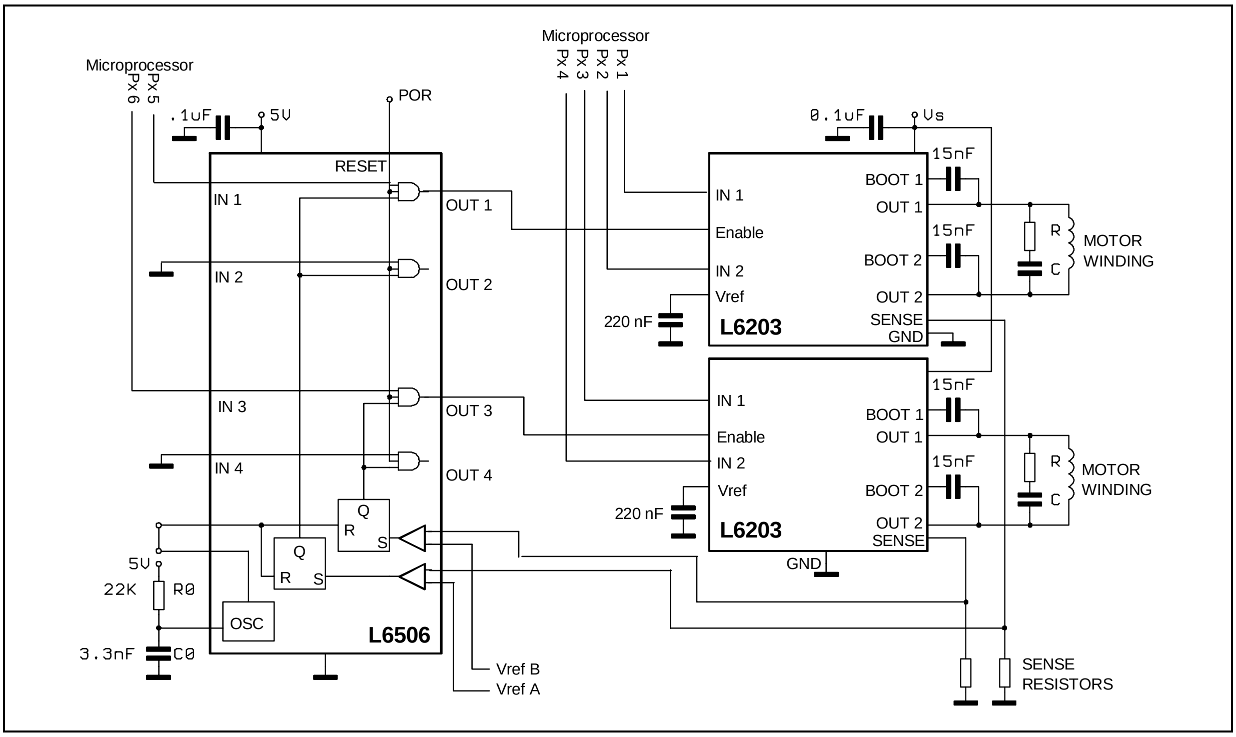


**Figure 13b. Microstepping using L6506 and L6203 (Fast Decay)**

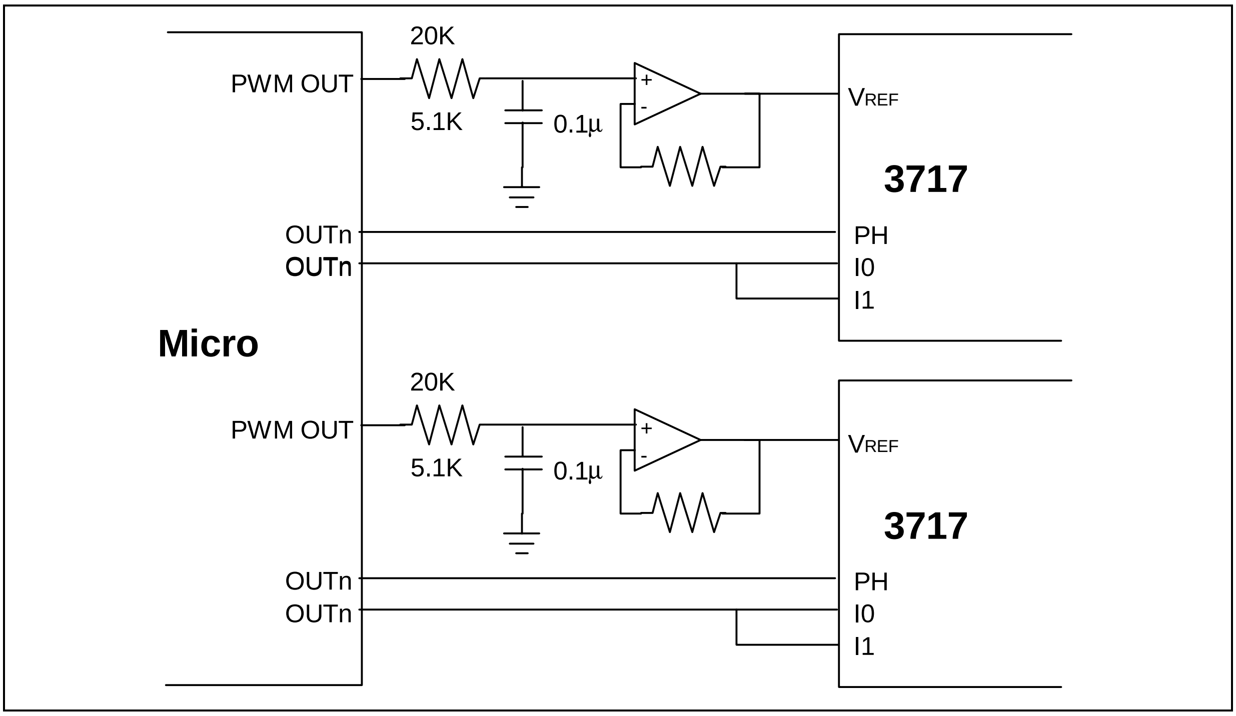


### PBL3717, TEA3717, TEA3718 and L6219

Devices like the PBL3717, TEA3717 TEA3718 and L6219 can also be used to implement microstepping. The main limitation in these devices is that, due to their internal connections, they can only implement the slow decay mode. The microstepping application is the same as the typical application for the device except that D/A converters must control the reference pins. With these devices since the reference is designed to operate from 5V and includes an internal voltage divider, a low impedance output must be used to drive the reference. If the PWM from the microprocessor is used for the D/A function, then only an RC filter is used without the second resistor for the divider. The resulting signal must them be buffered by an amplifier before driving the reference input.

The connections between the microprocessor and the PBL3717 family of devices are shown in Figure 14. For the best resolution it is suggested to set the I0 and I1 inputs to select the maximum current level. One should also be aware that the specifications of the L6219 have a minimum input reference voltage level.  This level must be respected and will then determine the minimum current that can be achieved in a microstepping circuit.

**Figure 14. Microstepping connection using PBL3717**



## CONCLUSION

Although they were not designed specifically to implement microstepping, many of the integrated motor control/ drive circuits can be used to implement microstepping stepper motor drives. The limits imposed by a peak detecting current control technique and the selected decay mode will directly affect the performance of the motor drive. Specifically it's ability to follow the desired current waveform. So long as these limits allow the designer to achieve the desired resolution in the microstepping application the devices provide a cost effective implementation.

## REFERENCES

[1] A NEW FULLY INTEGRATED STEPPER MOTOR DRIVER IC, Domenico Arrigo, Thomas L. Hopkins, Angelo Genova, Vincenzo Marano, and Aldo Novelli, Proceedings of PCIM 2001, Septermber 2001, Intertech Communication

[2] L6208 Data Sheet

[3] STEPPER MOTOR DRIVES, COMMON PROBLEMS AND SOLUTIONS, AN460, T. Hopkins, STMicroelectronics

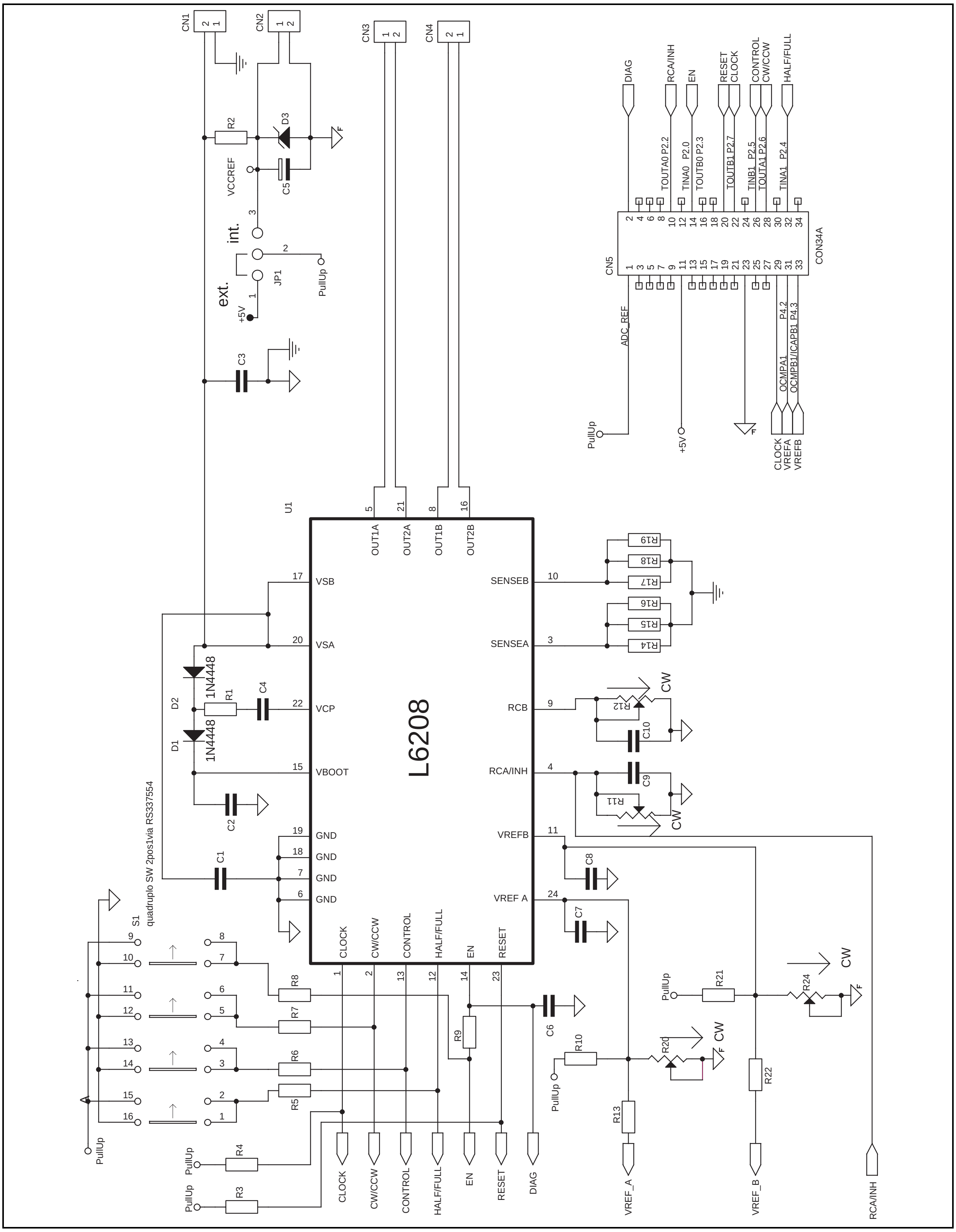
[4] L297 Data Sheet

[5] THE L297 STEPPER MOTOR CONTROLLER, AN470, STMicroelectronics

[6] STEPPER MOTOR DRIVING, AN235, H. Sax, STMicroelectronics

## APPENDIX A

### Figure 15. Scheme of the EVAL6208N



## APPENDIX B

```
set_micro_cvi_08:
                                        nop
                                        ei

                                        ld             micro_dir,#01000000b

                                        spp            #2
                                        ld             P2C2R, #00001100b
                                        ld             P2C1R, #11111111b
                                        ld             P2C0R, #00000000b

                                        spp            #P3C_PG
                                        or             P3C2R,#00001000b
                                        and            P3C1R,#11110111b
                                        or             P3C0R,#00001100b

                                        spp            #P4C_PG
                                        and            P4C2R, #11111011b
                                        or             P4C1R, #00000100b
                                        or             P4C0R, #00001100b

; to block pin 4.5 (ST)
                                        or             P4C2R, #00100000b
                                        and            P4C1R, #11011111b
                                        or             P4C0R, #00101100b

; Port 4 was defined already in TIMER.ASM
                                        spp            #P6C_PG
                                        ld             P6C2R, #00001000b
                                        ld             P6C1R, #00000100b
                                        ld             P6C0R, #00001000b

step_routine:
                                        ld             STEP_TOLD,STEP_T
                                        call      MFT0_step_time

; p2.7 is for clock do not affect the output
                                        spp            #21
                                        and            P2DR_P,#11101111b
                                        or             P2DR_P,#10000111b

                                        spp     #T0D_PG
                                        ld             T_PRSR,#0

                                        cp             micro_dir,#00000000b
                                        jpeq      start_CCW_timer

                                        call      start_MFtimer0m_CW
                                        jp             micro_cvi_08
```

```
start_CCW_timer:
                                        call    start_MFtimer0m_CCW

micro_cvi_08:
; Check - STOP ?
                                        ld           CSR_TMP,CSR
                                        and     CSR_TMP,#00000001b
                                        cp           CSR_TMP,#00000000b
                                        jpeq    STOP_cvi_8M

; Check - DIRECTION CHANGE
                                        ld           CSR_TMP,CSR
                                        and     CSR_TMP,#01000000b

                                        cp           CSR_TMP,micro_dir
                                        jpne    dir_changed

; Derection Setting
                                        cp           CSR_TMP,#01000000b
                                        jpeq    cw_cvi_08M_set

                                        cp           CSR_TMP,#00000000b
                                        jpeq    ccw_cvi_08M_set

cw_cvi_08M_set:
                                        ld           micro_dir,#01000000b
                                        spp     #21
                                        or           P2DR_P,#01000111b
                                        jp           CVI_KEEP_CHECK_08M

ccw_cvi_08M_set:
                                        ld           micro_dir,#00000000b
                                        spp     #21
                                        and          P2DR_P,#10111111b


CVI_KEEP_CHECK_08M:
; Check - SPEED
                                        cp           SPEED_KEY,#SW0
                                        jpeq    inc_speed_cvi_8M
                                        cp           SPEED_KEY,#SW5
                                        jpeq    dec_speed_cvi_8M
                                        ld           SPEED_KEY,#0

; Check - CLOCK FREQUENCY
                                        cp           STEP_T,STEP_TOLD
                                        jpne    STEP_TIME_CVI_8M

                                        jp           micro_cvi_08
```

```
STOP_cvi_8M:
                                        call      stop_MFtimer0m_CW
                                        call      MFtimer0_init
                                        spp     #T0D_PG
                                        ld             T_PRSR,#0

                                        call      EFT0_init
                                        call      EFT1_init

                                        ld             COUNT4,#0
      spp                              #21
                                        and            P2DR_P,#01111110b

                                        ld             CSR_TMP,CSR
                                        and       CSR_TMP,#00000001b
                                        cp             CSR_TMP,#00000001b
                                        jpeq      micro_cvi_08
                                        jp             STOP_cvi_8M

inc_speed_cvi_8M:
                                        pushu   PPR
                                        spp     #T0D_PG
                                        subw      T_REG0R,#150
                                        ld             SPEED_KEY,#0
                                        ld             OLD_KEY,#0
                                        popu      PPR
                                        jp             micro_cvi_08

dec_speed_cvi_8M:
                                        pushu   PPR
                                        spp     #T0D_PG
                                        addw      T_REG0R,#150
                                        ld             SPEED_KEY,#0
                                        ld             OLD_KEY,#0
                                        popu      PPR
                                        jp             micro_cvi_08


STEP_TIME_CVI_8M:
                                        ld             STEP_TOLD,STEP_T
                                        call      MFT0_step_time

                                        jp             micro_cvi_08


dir_changed:
                                        spp       #21
                                        and            P2DR_P,#01111101b
                                        and        P2DR_P,#01111110b
                                        ld         COUNT4,#0

                                        cp         CSR_TMP,#01000000b
                                        jpeq    cw_cvi_08M

                                        cp         CSR_TMP,#00000000b
                                        jpeq    ccw_cvi_08M

;=====================================================================
cw_cvi_08M:
                                        ld         micro_dir,#01000000b

                                        call    stop_MFtimer0m_CCW
                                        call    CW_micro_timer_setup

                                        spp        #2
                                        ld         P2C2R, #00001100b
                                        ld         P2C1R, #11111111b
                                        ld         P2C0R, #00000000b

                                        spp        #P3C_PG
                                        or         P3C2R,#00001000b
                                        and        P3C1R,#11110111b
                                        or         P3C0R,#00001100b

                                        spp        #P4C_PG
                                        and        P4C2R, #11111011b
                                        or         P4C1R, #00000100b
                                        or         P4C0R, #00001100b

; to block pin 4.5 (ST)
                                        or         P4C2R, #00100000b
                                        and        P4C1R, #11011111b
                                        or         P4C0R, #00101100b

; Port 4 was defined already in TIMER.ASM
                                        spp        #P6C_PG
                                        ld         P6C2R, #00001000b
                                        ld         P6C1R, #00000100b
                                        ld         P6C0R, #00001000b

                                        jp         step_routine

;===============================================================
ccw_cvi_08M:
                                        ld         micro_dir,#00000000b

                                        call    stop_MFtimer0m_CW
                                        call    CCW_micro_timer_setup
```

```
                                   spp          #2
                                   ld           P2C2R, #00001100b
                                   ld           P2C1R, #11111111b
                                   ld           P2C0R, #00000000b

                                   spp          #P3C_PG
                                   or           P3C2R,#00001000b
                                   and          P3C1R,#11110111b
                                   or           P3C0R,#00001100b

                                   spp          #P4C_PG
                                   and          P4C2R, #11111011b
                                   or           P4C1R, #00000100b
                                   or           P4C0R, #00001100b

; to block pin 4.5 (ST)
                                   or           P4C2R, #00100000b
                                   and          P4C1R, #11011111b
                                   or           P4C0R, #00101100b

; Port 4 was defined already in TIMER.ASM
                                   spp          #P6C_PG
                                   ld           P6C2R, #00001000b
                                   ld           P6C1R, #00000100b
                                   ld           P6C0R, #00001000b


                                   jp           step_routine

    ret
; ===================================================
```

Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

STMicroelectronics GROUP OF COMPANIES
Australia - Brazil - Canada - China - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan -Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States.
**http://www.st.com**