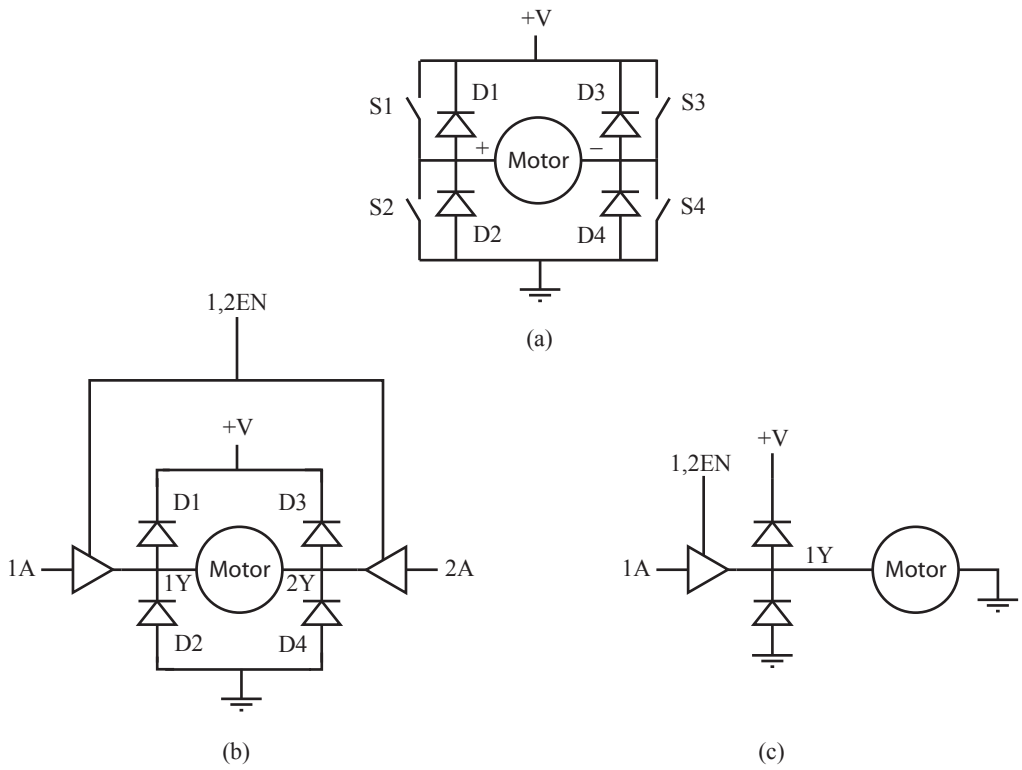


Figure 1: (a) An H-bridge constructed of discrete switches and flyback diodes. (b) The L293D H-bridge chip provides two full H-bridges (or four half H-bridges) with flyback diodes built in. This figure shows a motor being driven bidirectionally using a full H-bridge (two half H-bridges) with the inputs 1A and 2A. The outputs 1Y and 2Y are connected to the motor. The outputs are disabled (high impedance) if the input 1,2EN is low. (c) One half H-bridge being used to drive a motor unidirectionally.

1 Driving a DC Motor (or Any Inductive Load) with an H-bridge and PWM

1.1 The H-bridge

Figure 1(a) shows an H-bridge current amplifier for driving an inductive load, like a DC motor. It consists of four switches, typically implemented with bipolar junction transistors or MOSFETs, and four flyback diodes. An H-bridge can be used to run a DC motor bidirectionally, depending on which switches are closed:

Closed switches	Voltage across motor
S1, S4	positive
S2, S3	negative
S1, S3	zero (braking)
S2, S4	zero (braking)
none or one	open circuit (free-wheeling)

The switch settings not covered in the table (S1 and S2 closed, or S3 and S4 closed, or any set of three or four switches closed) all result in a short circuit and should obviously be avoided!

While you can build your own H-bridge out of discrete components, it is usually easier to buy one packaged in an integrated circuit. Apart from reducing your component count, these ICs also make it impossible for you to accidentally cause a short circuit. An example H-bridge IC is the L293D (see the Texas Instruments data sheet, google "Texas Instruments L293D"). This chip consists of two (full) H-bridges, each one of which is capable of providing 600 mA continuous or 1.2 A peak. It uses two voltage supplies: one high current supply, used to drive the motor, and another logic voltage supply, typically the same voltage used for your microcontroller. The L293D has a single ground for both power and logic.

Let's consider a single H-bridge of the L293D, circuits 1 and 2 (Figure 1(b)). The four digital inputs, controlling switches S1 to S4, are replaced by three digital inputs: one enable input (1,2EN) and two control inputs (1A and 2A). If the enable input is low, the H-bridge is disabled: all switches are open. If the enable input is high, then 1A and 2A control the operation. If 1A is high, then S1 is closed and S2 is open, so the output 1Y voltage is a high (power) voltage. If 1A is low, then S1 is open, S2 is closed, and the output 1Y is low. Similarly if 2A is high, then S3 is closed, S4 is open, and the output 2Y is high, and if 2A is low, then S3 is open, S4 is closed, and the output 2Y is low. There is no way to create a short circuit. Our previous table is replaced by the equivalent truth table

1,2EN	1A	2A	Voltage across motor
H	H	L	positive
H	L	H	negative
H	H	H	zero (braking)
H	L	L	zero (braking)
L	X	X	disabled, open circuit

where 'X' indicates "don't care."

From now on we will use the simplified diagram in Figure 1(b). When input 1A is logic high, output 1Y is power high, and when input 1A is logic low, output 1Y is power low. Power high voltage is somewhat less than +V, and power low voltage is somewhat more than 0 V, due to the voltage drops across the output transistors. The value V_{out} is used to represent the output voltage swing between power low and power high.

An important thing to note about our L293D is that flyback diodes are built in; we don't have to provide them externally. This is different from other H-bridge chips with similar names, such as the L293 or L293B. Those chips have the advantage of higher maximum current, but you have to provide four flyback diodes. These flyback diodes must be capable of carrying the maximum current that can flow through the motor, and they must be fast switching from nonconducting to conducting. Schottky diodes are common due to their fast switching and low forward bias voltage, resulting in less power lost to heat.

1.2 Control with PWM

(Note: the equations below are approximate only. Good for a first approximation, though.)

To control the speed of our motor, we use PWM to alternate between positive and negative voltage across the motor, creating an effective "average" voltage. For example, suppose 1A and 2A are always opposite each other; when one is high, the other is low. Suppose the duty cycle of 1A is 75%, or 0.75, and therefore the duty cycle of 2A is 25%. Then the motor is powered 75% of the time by V_{out} and 25% of the time by $-V_{out}$, giving an average of 0.5 V_{out} across the motor. More generally,

$$V_{ave} = (2*DC - 1)*V_{out},$$

where DC refers to the duty cycle of 1A and V_{out} refers to the power voltage. The average voltage varies linearly with the duty cycle, and it is zero at 50% duty cycle and $-V_{out}$ at 0% duty cycle.

To create two control signals, 1A and 2A, that are always the opposite of each other, we could use a single PWM output and an external inverter chip. To avoid this extra component, though, we could instead use PWM on 1A only, and use a digital output for signal 2A, creating a "direction" bit that is changed only infrequently. If 2A is low, for example, and the duty cycle of 1A is 75%, then 75% of the time the motor would have V_{out} across it, and 25% of the time it would have zero volts across it. With this way of driving the motor, our formula for V_{ave} is

$$V_{ave} = V_{out}*(DC - \text{sign}(2A)),$$

where sign(2A) is 1 if 2A is high and 0 if 2A is low. Stated another way, if our desired V_{ave}, call it V_c for commanded voltage, satisfies $V_{out} \geq V_c \geq 0$, then

$$DC = V_c/V_{out}, \qquad 2A = \text{low},$$

and if $-V_{out} \leq V_c$‘0, then

$$DC = 1 + V_c/V_{out}, \qquad 2A = \text{high}.$$

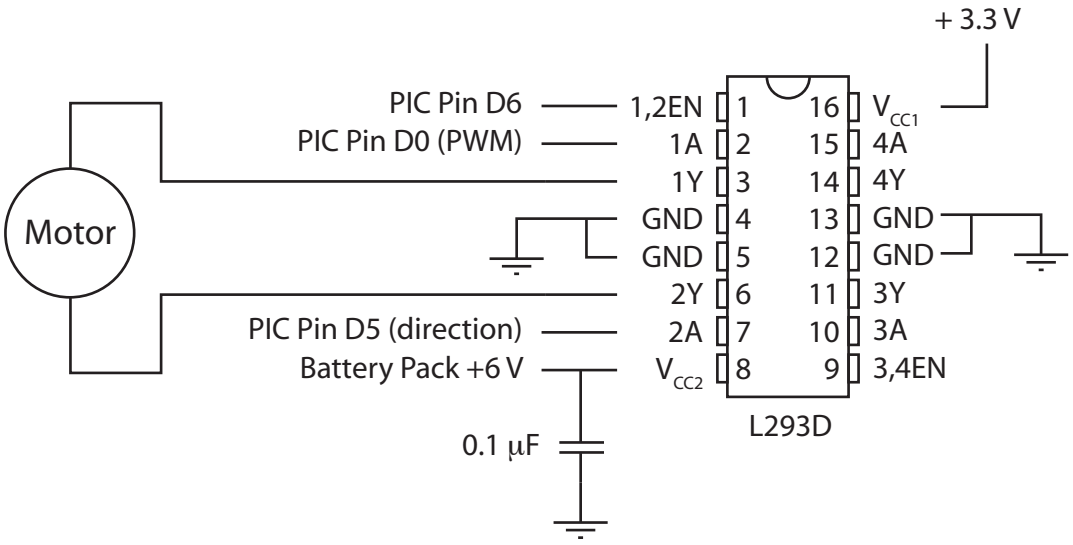


Figure 2: A sample H-bridge circuit.

So far we have been assuming that switching between positive and negative voltages with a constant duty cycle is equivalent to applying a constant average voltage. Let's look at that more closely. Assume that a current I_0 is flowing left to right through the motor (a positive current), and that 1A is logic high and 2A is logic low. Therefore the voltage across the motor is $V = \mathrm{V_{out}}$. Now we switch so that both 1A and 2A are logic low, and the outputs 1Y and 2Y attempt to enforce zero voltage across the motor. The motor, being inductive, still demands current, but 1Y's output transistors may not be able to source current when the input 1A is low. In that case, current flows from GND, through the flyback diode D2, out 1Y, through the motor and into 2Y.

Assuming zero voltage across the motor (ignoring transistor and diode voltage drops), and ignoring back-emf (it won't matter here), the motor equation

$$V = IR + L\frac{dI}{dt} + k_e\omega$$

becomes

$$0 = I_0 R + L\frac{dI}{dt}$$

or

$$\frac{dI}{dt} = -I_0\frac{R}{L},$$

with solution

$$I(t) = I_0\, e^{-\frac{R}{L}t} = I_0\, e^{-\frac{t}{\tau_e}}.$$

The time constant of this first-order decay of current is the motor's electrical time constant, $\tau_e = L/R$. Assuming "typical" values of $L = 1$ mH and $R = 10\ \Omega$ for a small motor, the time constant is 0.1 ms. Thus the current will decay to about 37% of its original value in 0.1 ms. On the other hand, the mechanical time constant τ_m is typically significantly larger (e.g., two orders of magnitude or more), particularly with a load attached to motor. Let's assume $\tau_m = 10$ ms. Then if the PWM frequency is 10 kHz, and input 1A has a 50% duty cycle, then we have a braking phase of duration $0.5/10^4$ s $= 50\ \mu$s during each PWM cycle, meaning that the speed of the shaft will drop to about

$$e^{-5\times10^{-5}/\tau_m} = 99.5\%$$

of its original value. Not much change.

Thus we should choose the PWM frequency sufficiently high so as to avoid observable variation in the motor's speed during a PWM cycle. If we choose the frequency too high, however, the transistors will spend a significant fraction of time switching from saturated to off, and vice-versa. In this linear regime, there is a significant voltage drop across the transistors. This will cause the transistors to heat up, possibly to the point of failure. Common PWM frequencies are 10 kHz to 40 kHz.

Figure 2 shows an L293D H-bridge set to receive PWM control from a PIC32.

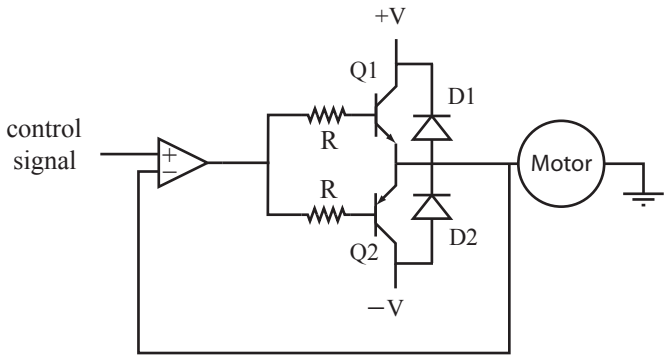


Figure 3: A simple linear push-pull amplifier.

1.3 Other Practical Considerations

Motors are noisy devices, creating both electromagnetic interference (EMI) and voltage spikes on the power lines. These effects can disrupt the functioning of your microcontroller, cause erroneous readings on your sensor inputs, etc. EMI shielding is beyond our scope here, but it is easy to use *optoisolation* to separate noisy power and clean logic voltage supplies.

An optoisolator consists of an LED and a phototransistor. When the LED turns on, the phototransistor is activated, allowing current to flow from its collector to its emitter. Thus a digital on/off signal can be passed between the logic circuit and the power circuit using only an optical connection, eliminating an electrical connection. In our case, the PWM signals would be applied to the LEDs, converted by the phototransistors to high and low signals to be passed to the 1A and 2A inputs of the H-bridge. Optoisolators can be bought in packages with multiple optoisolators. Each LED-phototransistor pair uses four pins: two for the internal LED and two for the collector and emitter of the phototransistor. Thus you can get a 16-pin DIP chip with four optoisolators, for example.

Another issue arises when the motor and load attain significant kinetic energy. (This won't be the case in ME 333.) When we brake the load, the energy must go somewhere. Some of it is lost to friction, and some of it is lost to I^2R winding heating. Remaining energy is dumped back into the power supply, essentially trying to "charge it up," whether it wants to be charged or not. Current passing through the flyback diodes D1 and D3, for example, are essentially charging the power supply and increasing the voltage. Some power supplies can handle this better than others; power supplies with large output capacitors may fare better than switching supplies, for example. You can also address the problem by putting a high-capacitance, high-voltage capacitor across the power supply. This capacitor acts as a repository for the kinetic energy converted to electrical energy. If the capacitor voltage gets too high, a switch can allow the back-current to be redirected to a "regen" power resistor, which is designed to dump electrical energy as heat. "Regen" is short for "regenerative" or "regeneration."

1.4 Comparison to a Linear Push-Pull Amp

Another common method for driving a load is the linear push-pull amp, shown schematically in Figure 3. In this amplifier, the NPN transistor Q1 "pushes" current from +V, through the motor, to GND, while the PNP transistor Q2 "pulls" current from GND, through the motor, to $-$V. The control signal is a low-current analog signal. It is fed into an op-amp which is configured as a voltage follower. Since there is feedback from the output of the op-amp to the inverting input, the op-amp does whatever it can to make sure that the signals at the inverting and non-inverting inputs are equal. Since the inverting input is connected to the motor, the voltage across the motor should be equal to the control signal voltage. The circuit simply boosts the current available to drive the motor beyond the current available from the control signal (i.e., it has a high impedance input and a low impedance output).

As an example, if +V = 10 V, and the control signal is at 5 V, then 5 V should be across the motor. To double-check that our circuit works as we expect, we calculate the current that would flow through the motor (for example, when it is stalled); this is the current I_e that must be provided by the emitter of Q1. If the transistor is capable of providing that much current, we then check if the op-amp is capable of providing the base current $I_b = I_e/(\beta + 1)$, where β is the transistor gain. If so, we are in good shape. The voltage at the base of Q1 is a diode drop (or more) higher than the voltage across the motor, and the voltage at the op-amp output is that base voltage plus I_bR.

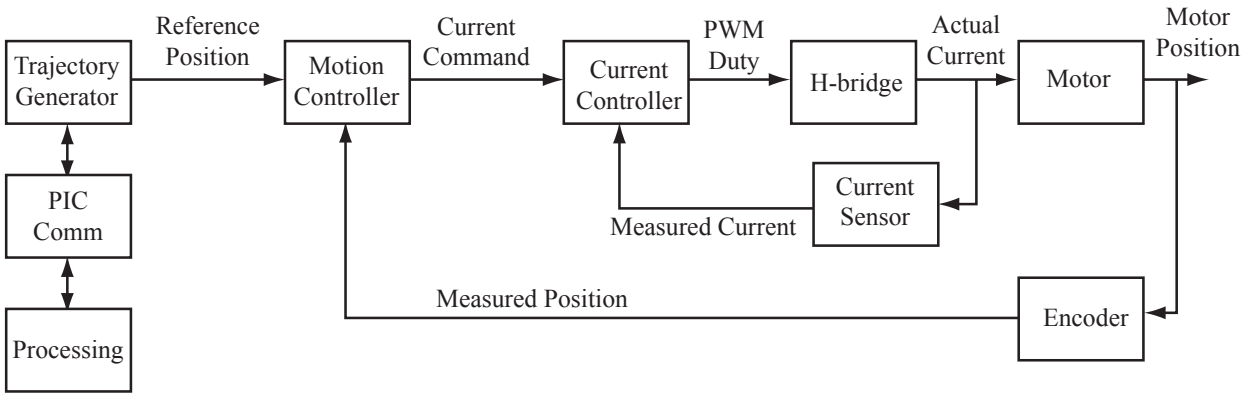


Figure 4: A block diagram for motion control.

An example application would be controlling a motor speed based on a knob (potentiometer). The potentiometer could be connected to +V and −V, with the wiper voltage serving as the control signal.

If the op-amp by itself can provide enough current, we can connect the op-amp output directly to the motor and flyback diodes, eliminating the resistors and transistors. Power op-amps are available, but they tend to be expensive relative to using output transistors to boost current.

We could instead eliminate the op-amp by connecting the control signal directly to the base resistors of the transistors. The drawback is that neither transistor would be activated for control signals between -0.7 and 0.7 V, or whatever the base-emitter voltage is when the transistors are activated. We have a "deadband" or "crossover distortion" from the control signal to the motor voltage.

Comparing an H-bridge driver to a linear push-pull amp, we see the following advantages of the H-bridge:

- Only a unipolar power supply is required, as opposed to a bipolar power supply (plus and minus voltage in addition to ground).
- The H-bridge is driven by a PWM pulse train, which is easily generated by a microcontroller, as opposed to an analog signal.
- Transistors spend most of their time in the off or saturated mode, with relatively small voltage drop across them, so that relatively little power is dissipated as heat by the transistors. This is in contrast to the linear amp, where the output transistors usually operate in the linear regime with significant voltage drops across them.

Linear amps are sometimes used in motor control when analog control signals and a bipolar power supply are available, and power dissipation and heat are not a concern. They are also preferred for better sound quality in speaker applications. There are many improvements to and variations on the basic circuit in Figure 3, and audio applications have raised amplifier circuit design to an art form. You can use a commercial audio amplifier to drive a DC motor, but you would have to remove a high-pass filter. Since we can't hear sound below 20 Hz, and low-frequency currents simply heat up the speaker coils without producing audible sound, audio amplifiers typically cut off frequencies below 10 Hz.

2 Motion Control of a DC Motor

An example block diagram for control of a DC motor is shown in Figure 4. A trajectory generator uses human input and creates a reference position as a function of time. To drive the motor to follow this reference trajectory, we use two nested control loops: an outer motion control loop and an inner current control loop. These two loops are roughly motivated by the two time scales of the system: the mechanical time constant and the electrical time constant.

- **Outer motion control loop.** This outer loop runs at a lower frequency, typically a few hundred Hz to a few kHz. The motion controller takes as input the desired position and/or velocity, as well as the motor's current position, as measured by an encoder or potentiometer, and possibly the motor's current velocity, as measured by

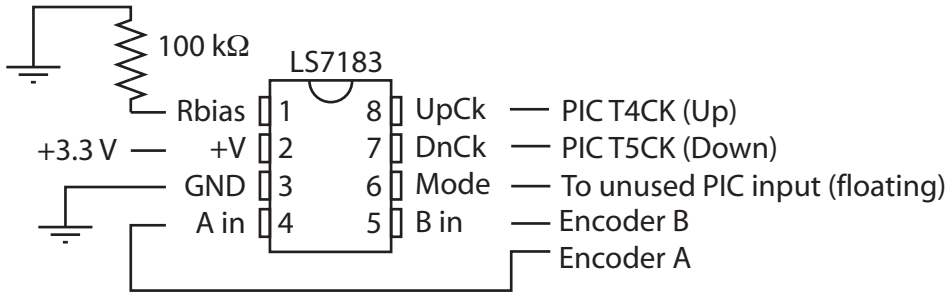


Figure 5: Connecting the PIC32 to the LS7183 encoder interface chip. The Mode pin determines whether we use x1, x2, or x4 decoding. Attaching Mode to GND uses x1 decoding; attaching Mode to 3.3 V chooses x2 decoding; and leaving Mode floating (by attaching to a floating input on the PIC) chooses x4 decoding. The choice of the Rbias resistor determines the duration of the up and down pulses, with larger resistances giving longer pulses. A 100 kΩ resistor gives a duration on the order of 1-2 microseconds.

a tachometer. The output of the controller is a commanded current I_c. The current is directly proportional to the torque. Thus the motion control loop treats the mechanical system as if it has direct control of motor torque.

- **Inner current control loop.** This inner loop typically runs at a higher frequency, from a few kHz to tens of kHz, but no higher than the PWM frequency. The purpose of the current controller is to deliver the current requested by the motion controller. To do this, it monitors the actual current flowing through the motor and outputs a commanded average voltage V_c (i.e., the PWM duty cycle) to compensate error.

Traditionally a mechanical engineer might design the motion control loop, and an electrical engineer might design the current control loop. But you are mechatronics engineers, so you will do both.

2.1 Reading an Encoder and Motion Control

We will use a US Digital LS7183 chip and Timers 4 and 5 on the PIC32 to keep track of the encoder. The A and B pulse trains are converted to up pulses and down pulses. These pulses increment Timer 4 (T4CK pin) and Timer 5 (T5CK pin), respectively. To determine the angle of the encoder, simply subtract the down count from the up count, taking care to account for rollover of the 16-bit counters. A sample circuit is shown in Figure 5.

An estimate of velocity can be obtained from position data by finite differencing. This may lead to a very noisy signal, however, particularly if the encoder's resolution is not high. Low-pass filtering or other fitting of the data is likely a good idea.

Let θ and $\dot{\theta}$ be the actual position and velocity, and θ_d and $\dot{\theta}_d$ be the desired position and velocity. Define the error $e = \theta_d - \theta$, error rate of change $\dot{e} = \dot{\theta}_d - \dot{\theta}$, and error integral $e_i = \Delta t \sum_k e(k)$, where Δt is the controller time step. Then a reasonable choice of controller would be a PID (proportional-integral-derivative) controller,

$$I_{c,fb} = k_p e + k_i e_i + k_d \dot{e},$$

where $I_{c,fb}$ is the commanded current. You should start with $k_d = k_i = 0$ and find a value of k_p that gives a decent response. This acts as a virtual spring that tries to pull the motor to the desired angle. Then add in k_d, which acts as a virtual damper between the desired velocity and the actual velocity. If you cannot get good tracking with just k_p and k_d, adding k_i will not likely help. You can add in nonzero k_i at the end to try to reduce steady-state error.

A PID controller is called a *feedback* controller, since it uses sensor feedback. You could instead try a *feedforward* controller. A feedforward controller uses only a model of the system and the desired trajectory to choose a commanded current. For example, you could choose your current command to be

$$I_{c,ff} = \frac{1}{k_t}(J\ddot{\theta}_d + mgr\sin\theta + b_0\,\mathrm{sgn}(\dot{\theta}) + b_1\dot{\theta}),$$

where k_t is the motor torque constant, J is the motor and load inertia, $\ddot{\theta}_d$ can be obtained by finite differencing the desired trajectory, mg is the weight of the load, r is the distance of the load center of mass from the motor axis, θ is the angle of the load from vertical, b_0 is Coulomb friction torque, and b_1 is a viscous friction coefficient. See Figure 6.

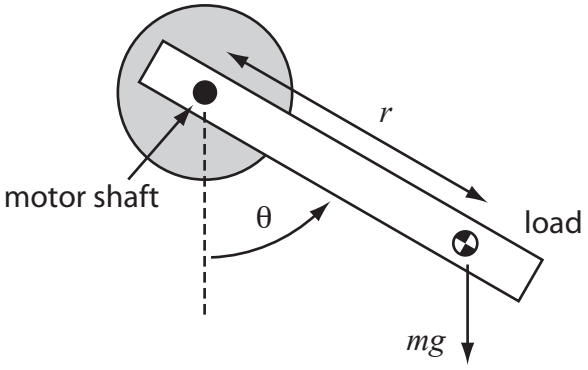


Figure 6: An unbalanced load in gravity.

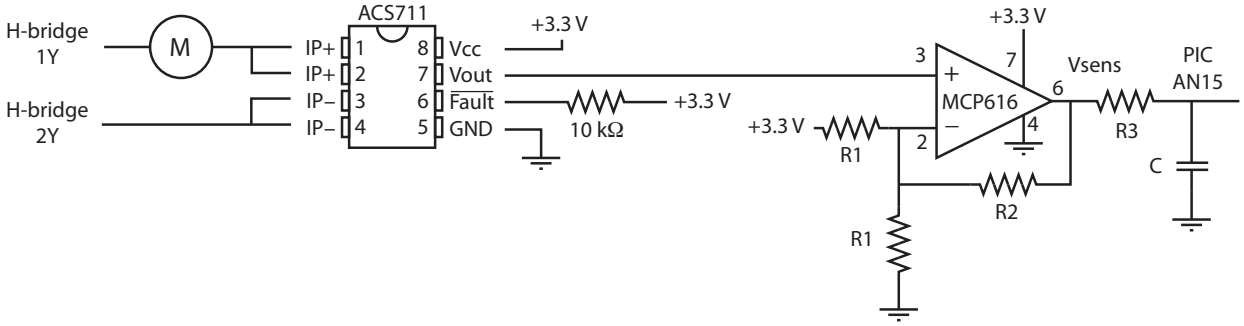


Figure 7: The current sensing circuit. $\overline{\text{Fault}}$ is driven low if the IP current exceeds 25 A.

Feedforward control alone will never yield acceptable performance, as no model will be sufficiently accurate. However, feedback control can only respond to errors. Why wait for effects you can model to manifest themselves as error? If you have a good model of the mechanical properties of your system and you use a commanded current $I_c = I_{c,fb} + I_{c,ff}$, you should be able to get better tracking control than you can by either controller alone.

2.2 Current Sensing and Current Control

2.2.1 Current Sensor

We will use an Allegro ACS711 Hall effect current sensor, coupled with a Microchip MCP616 op-amp, to measure the current flowing through the motor (Figure 7). The ACS711 is in series with the motor. Current flowing through the ACS711 creates a small magnetic field picked up by a linear Hall effect sensor, creating a change in output voltage Vout proportional to the current. At zero current, the output reads Vcc/2, or 1.65 V, and the output varies from this baseline amount by 55 mV/A. Because this voltage swing is small, we need to amplify it with an op-amp before reading it into an analog input.

We use the MCP616 op-amp to amplify the signal. This op-amp has two key characteristics: (1) it can be powered by a small voltage, in this case 3.3 V, and (2) the output of the op-amp can go "rail to rail"—it can swing all the way from the negative voltage supply to the positive voltage supply. In the circuit shown, we can use ideal op-amp rules with negative feedback to show that the resistors R1, R2, and R3 implement the formula

$$\text{Vsens} = 1.65\ \text{V} + \left(2 * \frac{\text{R2}}{\text{R1}} + 1\right) \Delta\text{V},$$

where ΔV is the ACS711's output voltage perturbation, i.e., $\Delta\text{V} = \text{Vout} - 1.65$ V. A reasonable choice is R1 = 10 kΩ and R2 = 100 kΩ. This gives an amplifier with a perturbation gain of 21, and therefore sensitivity of 21*55 mV/A = 1.155 V/A. Since you will not have currents of more than an amp, the voltage swing is well within the safe range for

Figure 8: The Copley Controls Accelus amplifier.

the PIC's 0 to 3.3 V ADC inputs. You could choose an even higher gain, as you are unlikely to see currents of more than 0.6 A.

The relation between motor current and sensor voltage should be linear, but you should calibrate your current sensor by doing experiments with known currents. You should experimentally determine (1) the sensor voltage corresponding to zero current and (2) the actual ratio from the current to the voltage perturbation. You can calibrate the current sensor by replacing the motor with a pure resistive load (a high-power, low resistance resistor) and using a known voltage across the resistor. You can also double-check your answer using your multimeter as a current meter.

Finally, to filter out the current ripple due to PWM switching, we add an RC LPF using R3 and C. Reasonable choices are R3 = 10 kΩ and C = 0.01 μF, giving a cutoff frequency of approximately 1600 Hz. This is well below our typical PWM frequency of 20 kHz, but not so low as to make our current controller sluggish.

2.2.2 Current Control

The output of the current controller is V_c, the commanded average voltage (to be converted to a PWM duty cycle). The simplest current controller would be

$$V_c = k_V I_c.$$

This would be a good choice if your load were only a resistance. Even if not, if you do not have a mechanical model of your system, achieving a particular current may not matter anyway. You can just tune your motion control PID gains, use $k_V = 1$, and not worry about what the actual current is.

On the other hand, if your battery pack voltage changes (due to discharging, or changing batteries, or changing from a 6 V to a 12 V battery pack), the change in behavior of your overall controller will be much larger if you do not measure the actual current in your current controller. More sophisticated current controller choices might be a mixed model-based and I feedback controller

$$V_c = I_c R + k_e \dot{\theta} + k_{I,i}\ e_{I,i}$$

or a PI feedback controller

$$V_c = k_{I,p}\ e_I + k_{I,i}\ e_{I,i},$$

where e_I is the error between the commanded current I_c and the measured current, $e_{I,i}$ is the integral of current error, R is the motor resistance, k_e is the motor electrical constant, $k_{I,p}$ is a proportional current control gain, and $k_{I,i}$ is an integral current control gain. A good current controller would closely track the commanded current.

2.3 An Industrial Example: The Copley Controls Accelus Amplifier

Copley Controls, http://www.copleycontrols.com, is a well known manufacturer of amplifiers for brushed and brushless motors for industrial applications and robotics. One of their models is the Accelus, pictured in Figure 8. The Accelus supports a number of different operating modes, for example control of motor current or velocity to be proportional to an analog voltage input or the duty cycle of a PWM input. A microcontroller on the Accelus interprets the analog input or PWM duty cycle and implements a controller similar to that in Figure 4. (Note: the duty cycle of a PWM input can be determined using the Input Capture peripheral on the PIC32, which we haven't discussed.)

The mode most relevant to us is the Programmed Position mode. In this mode, the user specifies a few parameters to describe a desired rest-to-rest motion. The controller's job is to drive the motor to track this trajectory.

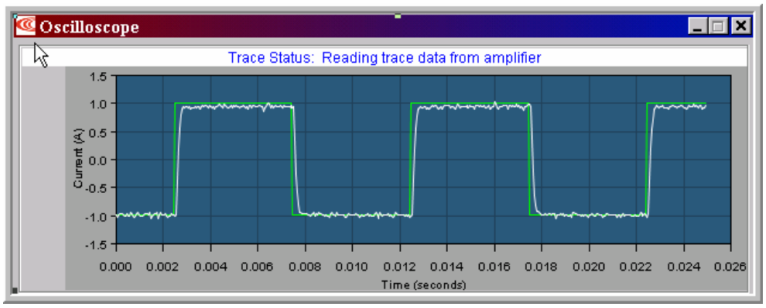


Figure 9: A plot of the reference square wave current and the actual measured current during PI current controller tuning.

When the amplifier is first paired with a motor, some initialization steps must be performed. A GUI interface on your PC, provided by Copley, communicates with the microcontroller on the Accelus using RS-232.

1. **Enter motor parameters.** From the motor's data sheet, enter the inertia, peak torque, continuous torque, maximum speed, torque constant, resistance, and inductance. These values are used for initial guesses at control gains for motion and current control. Also enter the number of lines per revolution of the encoder.

2. **Tune the current control loop.** Set a limit on the integrated current to avoid overheating the motor. This limit is based on the integral $\int I^2 dt$, which is related to how much energy the motor coils have dissipated recently. (When this limit is exceeded, the motor current is limited to the continuous operating current until the history of currents indicates that the motor has cooled.) Also, tune the values of P and I control gains for the PI current controller. This tuning is assisted by plots of reference and actual currents as a function of time. See Figure 9. The current control loop executes at 20 kHz, which is also the PWM frequency (i.e., the PWM duty cycle is updated every cycle).

3. **Tune the motion control loop with the load attached.** Attach the load to the motor and tune PID feedback control gains, a feedforward acceleration term, and a feedforward velocity term to achieve good tracking of sample reference trajectories. This process is assisted by plots of reference and actual positions and velocities as a function of time. The motion control loop executes at 4 kHz.

Once the initial setup procedures have been completed, the Accelus microcontroller saves all the motor parameters and control gains to nonvolatile flash memory. These tuned parameters then survive power cycling and are available the next time you power up the amplifier.

Now the amplifier is ready for use. The user specifies a desired trajectory using any of a number of interfaces (RS-232, CAN, etc.), and the amplifier uses the saved parameters to drive the motor to track the trajectory.

ME 333 Introduction to Mechatronics
Final Project: Brushed DC Motor Control

In-class milestone: Tues March 8 (demo only)
Final demo and competition: Wed March 16, 7-9 PM, LR5
Code and written portions must be submitted electronically before 7 PM, March 16

We will provide you with code that you will use and must not change, to keep the playing field even for everyone. This code is represented in the block diagram in Figure 4. You will add to this code your own ideas on system modeling, filtering, control, etc.

Processing Code We are giving you code to communicate with the PIC, to send it commands and to receive results for plotting. This code gives you three basic operating modes:

1. **Current Control Tuning**. When this command is sent to the PIC, it sends with it a set of parameters that are communicated to the Current Controller. The current controller, with these parameters, then attempts to track a 100 Hz square wave current command for a few cycles. The results are sent back to Processing for plotting.

2. **Motion Control**. When this command is sent to the PIC, it sends with it any parameters (gains) for the Motion Controller, as well as parameters describing the reference trajectory to be followed. The PIC then zeros the encoder counts, enables the H-bridge, executes the trajectory-following control (trajectories always begin at zero position and velocity), and sends back the results to be plotted by Processing. It also sends back a total "score" indicating how well the motor tracked the desired trajectory.

3. **User Defined**. When this command is sent to the PIC, it sends any parameters specified in the GUI, and executes a function that the user has defined in the PIC software. Typical usage would be a calibration routine that you could use to estimate the inertia of the load, determine the torque/current needed to hold an unbalanced load stationary at different angles, etc. At the end of the routine, you can return information to be displayed by Processing.

Each one of these modes has a specific start time (when the user enters the command in Processing and the parameters are sent to the PIC) and a specific stop time (when the PIC returns results). After any of these modes is finished, the motor is disabled, by disabling the H-bridge. This allows you to reposition the motor between runs.

Information sent back and forth from Processing to the PIC includes:

- **From Processing to the PIC**: The GUI sends information on the specified operating mode, the current control parameters (gains), the motion control parameters (gains), the motion control specification, and any other user-defined data.

- **From the PIC to Processing**: The PIC returns the reference and actual position trajectory arrays, reference and actual motor current data arrays (primarily used in Current Control Tuning mode), any user-specified data arrays, an optional user-defined message, and a "score" characterizing how well the motor tracked the desired trajectory (only useful in the Motion Control mode). Any of the arrays can be plotted in the GUI to provide the user feedback.

PIC Communication Code The PIC communication code interfaces with the Processing code. When it receives a command from Processing, it calls one of its three routines (current control tuning, motion control, or user defined) with the appropriate parameters. When the routine finishes, the communication code sends the results back to Processing. (Each routine must end by disabling the H-bridge.)

Trajectory Generator The trajectory generator takes an n-vector of accelerations `float acc[]`, in encoder counts per (sample time)^2, and an $n-1$ vector of times `int dt[]`, in sample times. The reference trajectory is stored in a position array. The trajectory is generated by integrating `acc[0]` for time `dt[0]` to get positions in encoder counts, and encoder counts per sample time, respectively. The final time `dt[n-1]` (that goes with `acc[n-1]`) is determined as the time needed to bring the motor back to zero velocity. If the total time of the motion is too long, or if the trajectory never brings the motor back to rest, an error is returned.

Motion Controller The outer-loop motion controller is a timed ISR that operates at 200 Hz, or a sample time of 0.005 s. At each sample time, the controller reads in the new reference position from the trajectory. It also checks the motor's position, in counts, from the encoder. It then calculates a new current (or, equivalently, torque, by the torque constant) to request from the current controller. When the trajectory ends, the H-bridge is disabled.

The code we give you will read the encoder and the trajectory from the trajectory generator. It will also keep a running "score" of how well your controller works. You will need to write your own controller. This could include a feedforward model of the load, motor model, PID control, velocity and integration filters, etc.

If the PIC is operating in the Current Control Tuning mode, then the motion controller will alternate between requesting a small positive current and a small negative current (100 Hz square wave). This is useful for tuning the current control gains.

While many motor controllers run at higher frequencies (e.g., 1 kHz), we choose 200 Hz because of the relatively low resolution of our encoder. No need to read the encoder so often if the count does not change quickly.

Current Controller The inner-loop current controller executes at 10 kHz. The job of the current controller is to make the actual current through the motor match the requested current by the motion controller. It does this by reading in the motor current from the ADC, then updating the PWM signal. You will need to decide how to do this update.

Milestone, March 8

Nothing is to be turned in on March 8. Instead, you will demonstrate that you have basic command of the control process by doing a demo in class. This demo will count as 20% of your final project grade. The total value of the final project is approximately the same as two normal assignments.

1. **Demonstrate current control.** Show that your current controller can give a decent approximation to the 100 Hz square wave reference signal.

2. **Demonstrate simple motion control.** Show that your motor approximately follows a simple bang-bang acceleration trajectory. Nothing fancy is needed here (a simple proportional controller on the encoder error suffices for this demo). We just want to make sure that you are reading your encoder properly and are able to drive the motor.

Final Demo, March 16

Seeding

Each round of the demo will proceed in the following steps:

1. We tell you the load to apply to the motor. The load will always include the bar, but may include different numbers of weights on each end of the bar/arm. After you have set up the arm, you may take a minute or two to perform any calibration, modeling, or tuning steps.

2. You position your arm so that it is vertical in gravity, with the center of mass (if it is offset from the motor axis) directly below the motor axis. We then tell you the motion vector describing the desired motion of the arm. You type in the motion vector, execute it, and report the score.

Based on the results of this seeding round, we will have a motor control tournament.

Tournament

Students will be placed into brackets based (approximately) on the results of the seeding. We will then have a single-elimination tournament with students competing head-to-head, following the procedure outlined above. Lowest score wins. Bonus project points will be awarded to students advancing to the quarterfinals, semifinals, and finals.

Final Report

Submit your well-documented code and a separate brief writeup explaining your solution. This writeup should be approximately one page, and should describe

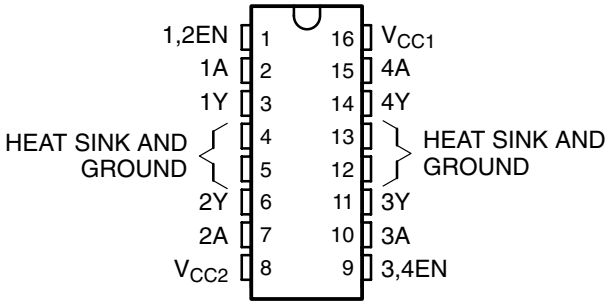
- your motion control algorithm,
- your current control algorithm,
- any modeling/calibration/filtering you performed manually or implemented in code, and
- any other noteworthy aspects of your project, if any.

More Info

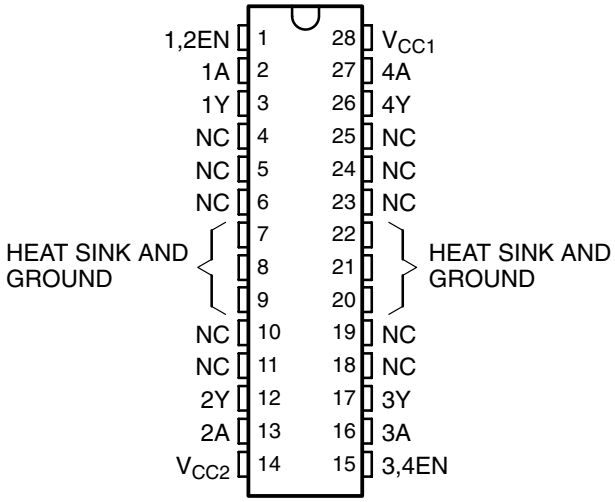
To find the full data sheets for the chips on the subsequent pages, you can google "Texas Instruments L293D," "LS7183," "ACS711," and "MCP616."

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functionally Similar to SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

L293 . . . N OR NE PACKAGE
L293D . . . NE PACKAGE
(TOP VIEW)



L293 . . . DWP PACKAGE
(TOP VIEW)



description/ordering information

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	HSOP (DWP)	Tube of 20	L293DWP	L293DWP
	PDIP (N)	Tube of 25	L293N	L293N
	PDIP (NE)	Tube of 25	L293NE	L293NE
		Tube of 25	L293DNE	L293DNE

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

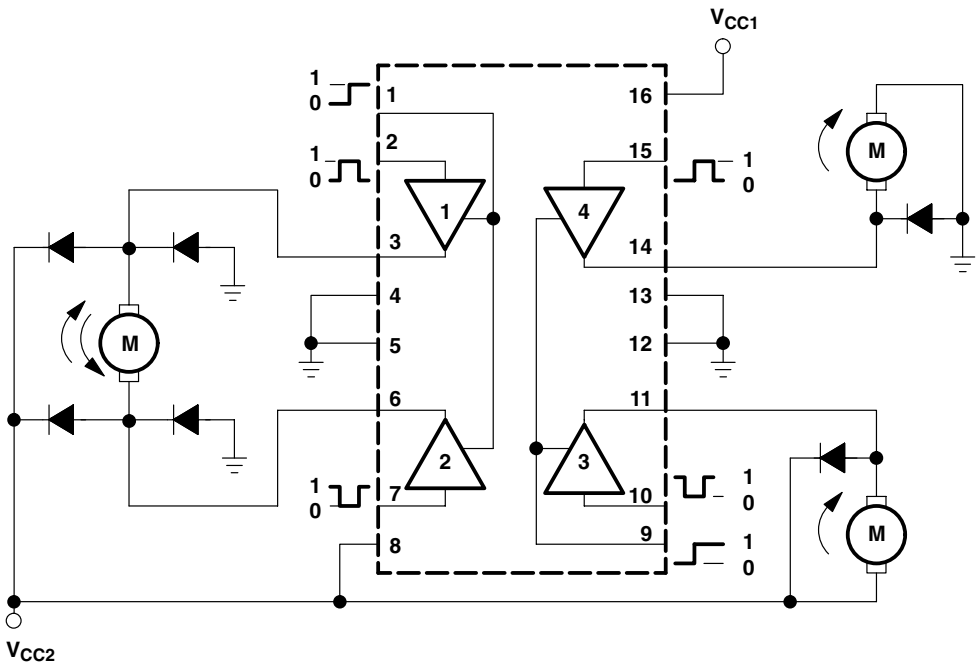
description/ordering information (continued)

On the L293, external high-speed output clamp diodes should be used for inductive transient suppression.

A V_{CC1} terminal, separate from V_{CC2}, is provided for the logic inputs to minimize device power dissipation.

The L293and L293D are characterized for operation from 0°C to 70°C.

block diagram



NOTE: Output diodes are internal in L293D.

FUNCTION TABLE
(each driver)

INPUTS†		OUTPUT
A	EN	Y
H	H	H
L	H	L
X	L	Z

H = high level, L = low level, X = irrelevant,
Z = high impedance (off)

† In the thermal shutdown mode, the output is in the high-impedance state, regardless of the input levels.

LS7183 / LS7184 Encoder to Counter Interface Chips

Description:

The **LS7183** and **LS7184** provide an interface between industry standard A and B quadrature incremental encoder outputs to standard up/down counters. The **LS7183** outputs can connect directly to the up and down clock inputs of counters such as 74193 or 40193. The **LS7184** outputs can connect directly to the Clock and Up/Dn inputs of counters such as 4516 or 74169.

The **LS7183** and **LS7184** are improved designs over the **LS7083** and **LS7084** products and should be considered first for all new product designs. The primary differences between the old and new LS chips are the addition of a X2 resolution multiplication, power supply operating range and improved output pulse timing characteristics.

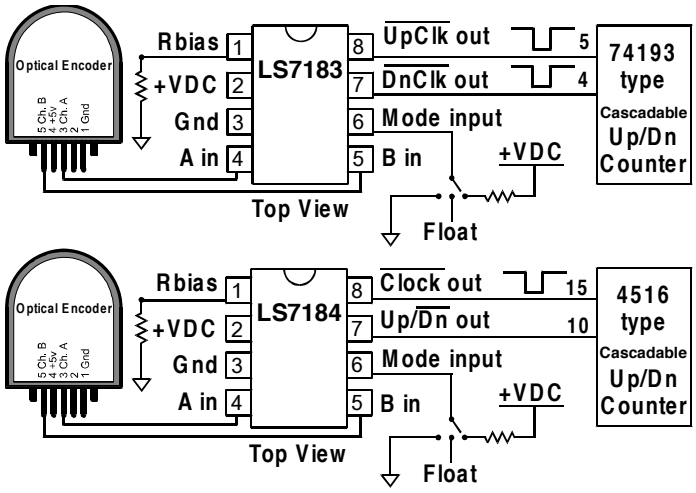
Please Note: Rbias values for output pulse width timing are not the same as the **LS7083** and **LS7084** values.

Features:

- X4, X2 or X1 resolution multiplication
- TTL and CMOS compatible
- Low power (micro-amps)
- 8-pin DIP or SOIC package
- No external clocks required
- Drive standard Up/Dn counters
- Monolithic CMOS
- Operates from 3V to 5V power supply

Absolute Maximum Ratings:

Parameter	Min.	Max.	Units
Operating Temperature	-20	85	°C
Storage Temperature	-55	150	°C
Voltage @ Any Input	-.3	VCC+.3	Volts
Supply Voltage (VCC)		7	Volts



Pin Descriptions:

Pin 1 (Rbias Input):
Input for external component connection. A resistor connected between this input and ground adjusts the output pulse width. See Rbias Resistor Value vs. Timing Table for further information.

Pins 4 & 5 (A & B Inputs):
Connect to the A and B quadrature outputs of the encoder. Both inputs have debounce filters. Minimum pulse width is set at 300ns. There is no maximum limit. Input current is less than 1μA. The A and B inputs can be swapped to reverse the direction of the external counters.

Pin 6 (Mode Input):
Mode is a 3-state input to select resolution X1, X2 or X4. The input quadrature clock rate is multiplied by factors of 1, 2 or 4 in X1, X2 or X4 modes respectively in producing the output Up/Dn clocks. X1, X2 or X4 modes are selected by input logic levels as follows:

- Mode = 0 VDC = X1 Selection
- Mode = +VDC = X2 Selection
- Mode = Float = X4 Selection

In X4 mode, one pulse is generated for each A/B state change. In X1 mode, one pulse is generated per quadrature cycle. In X2, two pulses per quadrature cycle.

LS7183 Pin 7 (Down Clock Output):
Normally high, low-true. The low level pulse width is set by pin 1. Down counts are enabled only when B leads A.

LS7184 Pin 7 (Up/Down Clock Output):
This output steers the external counter up or down. High = Up (A leads B), Low = Down (B leads A).

LS7183 Pin 8 (Up Clock Output):
Normally high, low-true. The low level pulse width is set by pin 1. Up counts are enabled only when A leads B.

LS7184 Pin 8 (Clock Output):
Normally high, low-true. The low level pulse width is set by pin 1. The external counter should count on the rising (high-going) edge of this output.

Surface Mount Package:
The 8-pin SOIC package has the same pin-out as the DIP version shown above.

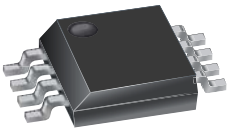
US DIGITAL
info@usdigital.com ▪ www.usdigital.com
Local: 360.260.2468 ▪ Sales: 800.736.0194
Support: 360.397.9999 ▪ Fax: 360.260.2469
1400 NE 136th Ave. ▪ Vancouver, Washington ▪ 98684 ▪ USA

Hall Effect Linear Current Sensor with Overcurrent Fault Output for <100 V Isolation Applications

Features and Benefits

- No external sense resistor required; single package solution
- 1.2 mΩ internal conductor resistance; reduced power loss
- Economical low- and high-side current sensing
- Output voltage proportional to AC or DC currents
- ±12.5 A and ±25 A full scale sensing ranges
- Overcurrent $\overline{\text{FAULT}}$ trips and latches at 100% of full-scale current
- Low-noise analog signal path
- 100 kHz bandwidth
- Small footprint, low-profile SOIC8 package
- 3.0 to 5.5 V, single supply operation
- Integrated electrostatic shield for output stability
- Factory-trimmed for accuracy
- Extremely stable output offset voltage
- Zero magnetic hysteresis
- Ratiometric output from supply voltage

Package: 8 Lead SOIC (suffix LC)



Approximate Scale 1:1 

Description

The Allegro® ACS711 provides economical and precise solutions for AC or DC current sensing in <100 V audio, communications systems, white goods, and automotive applications. The device package allows for easy implementation by the customer. Typical applications include circuit protection, current monitoring, and motor and inverter control.

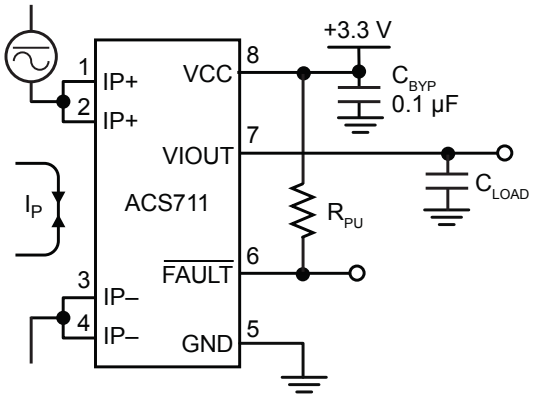
The device consists of a linear Hall sensor circuit with a copper conduction path located near the surface of the die. Applied current flowing through this copper conduction path generates a magnetic field which is sensed by the integrated Hall IC and converted into a proportional voltage. Device accuracy is optimized through the close proximity of the magnetic signal to the Hall transducer.

The output of the device has a positive slope proportional to the current flow from IP+ to IP– (pins 1 and 2, to pins 3 and 4). The internal resistance of this conductive path is 1.2 mΩ typical, providing a non-intrusive measurement interface that saves power in applications that require energy efficiency.

The ACS711 is optimized for low-side current sensing applications, although the terminals of the conductive path are electrically isolated from the sensor leads (pins 5 through 8), providing sufficient internal creepage and clearance dimensions for a low AC or DC working voltage applications. The thickness

Continued on the next page…

Typical Application



Application 1. The ACS711 outputs an analog signal, V_{IOUT}, that varies linearly with the bi-directional AC or DC primary current, I_P, within the range specified. The $\overline{\text{FAULT}}$ pin trips when I_P reaches ±100% of its full-scale current.

MCP616/7/8/9

2.3V to 5.5V Micropower Bi-CMOS Op Amps

Features

- Low Input Offset Voltage: ±150 µV (maximum)
- Low Noise: 2.2 μV_{P-P} (typical, 0.1 Hz to 10 Hz)
- Rail-to-Rail Output
- Low Input Offset Current: 0.3 nA (typical)
- Low Quiescent Current: 25 µA (maximum)
- Power Supply Voltage: 2.3V to 5.5V
- Unity Gain Stable
- Chip Select ($\overline{CS}$) Capability: MCP618
- Industrial Temperature Range: -40°C to +85°C
- No Phase Reversal
- Available in Single, Dual and Quad Packages

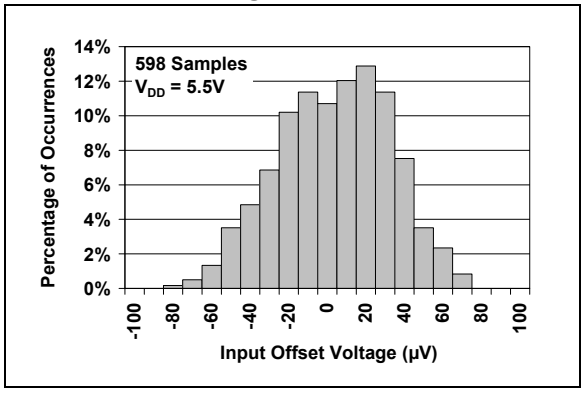
Typical Applications

- Battery Power Instruments
- Weight Scales
- Strain Gauges
- Medical Instruments
- Test Equipment

Design Aids

- SPICE Macro Models
- Microchip Advanced Part Selector (MAPS)
- Mindi™ Circuit Designer & Simulator
- Analog Demonstration and Evaluation Boards
- Application Notes

Input Offset Voltage



Description

The MCP616/7/8/9 family of operational amplifiers (op amps) from Microchip Technology Inc. are capable of precision, low-power, single-supply operation. These op amps are unity-gain stable, have low input offset voltage (±150 µV, maximum), rail-to-rail output swing and low input offset current (0.3 nA, typical). These features make this family of op amps well suited for battery-powered applications.

The single MCP616, the single MCP618 with Chip Select ($\overline{CS}$) and the dual MCP617 are all available in standard 8-lead PDIP, SOIC and MSOP packages. The quad MCP619 is offered in standard 14-lead PDIP, SOIC and TSSOP packages. All devices are fully specified from -40°C to +85°C, with power supplies from 2.3V to 5.5V.

Package Types