MOTOR CONTROL
Tech Tips
VOLUME 3, ISSUE 20
February 2004
from HITACHI AMERICA, LTD.

BLDC Motor Drive Over Current Protection (I^2T Overload) - Part 4 (of 4 Parts)

Over Current Fault Protection Circuits

This TECH TIPS 20 is the 4th in our 4-Part (17, 18, 19 and **20**) series on ways to Detect and Protect BLDC Motor Drives from Over Current and Overload (power limit) Faults offers a detailed look at I^2T Overload Protection circuits (commonly used in Servo Drive applications) which folds back current as a function of Overload Current magnitude *and* Time.

If a Current Limit is set 10 (to 20)% above the Drive Continuous Working Rating, Cycle-by-Cycle Current Limiting (supplemented with Hiccup Delay) … see Tech Tips Issue 17 & 18 … is usually sufficient for both Catastrophic and Overload Protection; particularly if the Drive is further protected by a Locked-Rotor Timer … see Tech Tips Issue 17 & 18 … that shuts-down the system if the fault persists. However, in Servo and Traction Applications, drives are typically *designed* to handle Peak Output Power Levels 200 (to 300)% above their Continuous Power Rating *for limited time periods.* This Peak Power requirement, and the fact Servo Drives usually employ 4-Quadrant Torque Control, differentiates Servo Drives from their Continuous-Duty, 2-Quadrant cousins.

The time period a Servo Drive can operate in Overload varies *inversely* with Overload *magnitude* and *directly* with motor Thermal Time Constant and (in the limit) with the Thermal Time Constant of the power amplifier. To prevent Peak Demand abuse, an I^2T *current fold back* circuit is typically used (in-addition-to Cycle-by-Cycle and Hiccup Current Limit Protection) to *dynamically reduce* output power whenever Drives may be *expected* to operate in Overload for extended periods. The I^2T method protects a Servo Drive (i.e., motor and power amplifier) from damage while allowing the Drive to *normally* operate *up to* Thermal Limits. This type of Overload Protection might be based on a direct measurement of (internal) motor temperature, but wide temperature variations (within a given motor) make it difficult to accurately determine temperature. Consequently, a more practical Overload Protection method evolved based on monitoring the power flow between the Power Amplifier and Motor by keeping track of the *Magnitude* and *Duration* of Overload events. This method provides excellent results (and avoids direct temperature measurements) while using the motor feedback current *normally* required by the Servo Torque Control Loop.

Most of the energy (heat) dissipated in a motor is due to the [I^2rms **x** R] motor winding losses. This lost energy (i.e., energy *not* converted to Rotor Mechanical energy) can be estimated as the product of Motor Winding Loss and Time ($E = I^2rms \times R \times Time$) where Irms is the RMS Winding Current and R is the effective Motor Winding Resistance. The Continuous Power Rating of a motor is proportional to the maximum amount of power [$I^2rms \times R$] a motor *can* dissipate (at its rated RMS current) without exceeding motor temperature rating. The Continuous Power Rating is based on the principle of thermal equilibrium. That is (at the Continuous Current Rating), the flow of energy *into* the motor *must equal* the energy flow to the mechanical load (*plus* the energy lost as heat) in order for the Motor Temperature to achieve (and maintain) a constant, steady-state value.

However, under transient conditions, motors can tolerate a certain amount of energy-in-excess of this Continuous Limit. The amount of Overload Energy a motor can handle is dependent on the motor size, cooling methods and configuration. For a given motor (with winding resistance R), the energy dissipated *in-excess-of* the continuous limit is given by:

$$E_{transient} := Ip_{rms}^{2} \cdot R \cdot T - Ic_{rms}^{2} \cdot R \cdot T^{\blacksquare} \quad \text{(Joules)}$$

where Ip (rms) is the peak motor RMS current, Ic (rms) is the continuous RMS current rating of the motor and, T is the duration of the transient in seconds. The motor Continuous Current Rating (RMS Amperes), the motor Transient Peak Current Rating (RMS Amperes) and the maximum time duration of a Peak Current Transient (T in seconds) for a particular motor, can either be measured or obtained from the motor manufacturer. Given these numbers, the above equation can be simplified (by eliminating the motor resistance) to obtain:

$$E_{limit} := Ip_{rms}^{2} \cdot T - Ic_{rms}^{2} \cdot T^{\blacksquare} \quad \text{(AmpereSquared-Seconds)}$$

where E(limit) = E(transient)/R which has units of AmpereSquared-Seconds. This energy is referred to as “I-Squared-T” (or, simply I^2T) and is a *measure* of the energy in an Overload Transient. The comparable measure of Motor Overload Capability is calculated as the square of Motor Peak Current Rating (Amperes Squared) times the Rated Peak Current Time (Seconds).

Implementation

I^2T protection continuously monitors Motor Winding Current and evaluates Transient Overload Energy in real-time while continuously comparing that result to a predetermined I^2T set point. The predetermined Set Point is calculated via the I^2T equation using *known* Continuous Current, Peak Current and Peak-Current-Time parameters for your specific motor. If (and, when) the *measured* I^2T *exceeds* the I^2T set point, the protection circuit intercedes to limit the power amplifier output current *before* the Transient Energy exceeds motor limits. I^2T Overload Protection can be implemented digitally (MicroController) or, with analog circuitry. We will discuss the more common MicroController version in detail, and also provide insight for those interested in an analog implementation.

System Considerations

In general purpose BLDC Drives, designers need to configure the Drive (and, I^2T algorithm) to handle different (*specific*) motors. To accommodate a range of Motors, assume the power amplifier is *thermally* designed for a (maximum) Continuous Current Rating, a Peak Overload Current Rating and a Peak-Current-Time Rating similar to that required by the motor, but larger. Where, “larger” guarantees a power amplifier is capable of driving a range of motor sizes. For example, suppose the power amplifier has the *thermal* capacity to support a Continuous Rating of 10 amperes, a Peak Rating of 25 amperes, and can maintain the Peak Current for 2.5 seconds. Further, suppose the output of this power amplifier can be linearly adjustable from 20% to 100% of Peak Output, i.e., adjustable over a 5:1 range. That is, [(Max) / (5)] or, [(Max) x (0.2)]:

$Ic_{max} := 10$ (Amperes, RMS) $Ic_{min} := 10\cdot.2$ $Ic_{min} = 2$ (Amperes, RMS)

$Ip_{max} := 25$ (Amperes, RMS) $Ip_{min} := 25\cdot.2$ $Ip_{min} = 5$ (Amperes, RMS)

The corresponding Peak and Minimum I^2T (while safely reducing T to 2.381 seconds eliminates decimal points) are thus:

$I2T_{max} := \left(Ip_{max}^{\;2} - Ic_{max}^{\;2}\right)\cdot 2.381$ $I2T_{max} = 1250$ (AmpereSquared-Seconds)

$I2T_{min} := \left(Ip_{min}^{\;2} - Ic_{min}^{\;2}\right)\cdot 2.381$ $I2T_{min} = 50$ (AmpereSquared-Seconds)

Let's apply these I^2T (Max / Min) parameters using a MicroController (programmable) Drive architecture (Fault Processor) with an on-chip, multi-channel, A/D converter having 0 to 5 volts input range. A partial block diagram of this hypothetical Drive Control is shown in Figure 1. Suppose this Drive Control has 3 identical potentiometer circuits sensed (measured) by ANalog inputs AN1, AN2, AN3 … each having an identical, adjustable range of 1 to 5 volts. Now, assume:

AN1 represents the (adjustment range of) Continuous Motor Current
AN2 represents the (adjustment range of) Peak Motor Current
AN3 represents the (adjustment range of) I^2T Set Point.

Finally, assume MicroController firmware interprets the AN1, AN2 and AN3 inputs as:

$I_c := \left[\dfrac{\left(Ic_{max} - Ic_{min}\right)}{(5-1)}\right]\cdot V_{AN1}$ Or $I_c := 2\cdot V_{AN1}$ (Amperes)

$I_p := \left[\dfrac{\left(Ip_{max} - Ip_{min}\right)}{(5-1)}\right]\cdot V_{AN2}$ Or $I_p := 5\cdot V_{AN2}$ (Amperes)

and, finally:

$$\left(\frac{I2T}{50}\right)^{\frac{1}{2}} := \left[\frac{\left(\dfrac{I2T_{max}}{50}\right)^{\frac{1}{2}} - \left(\dfrac{I2T_{min}}{50}\right)^{\frac{1}{2}}}{(5-1)}\right]\cdot V_{AN3}$$

Or,

$I2T := \left(V_{AN3}\right)^2\cdot 50$ (AmpereSquared-Seconds)

BLDC Motor Drive Over Current Protection (I^2T Overload) - Part 4

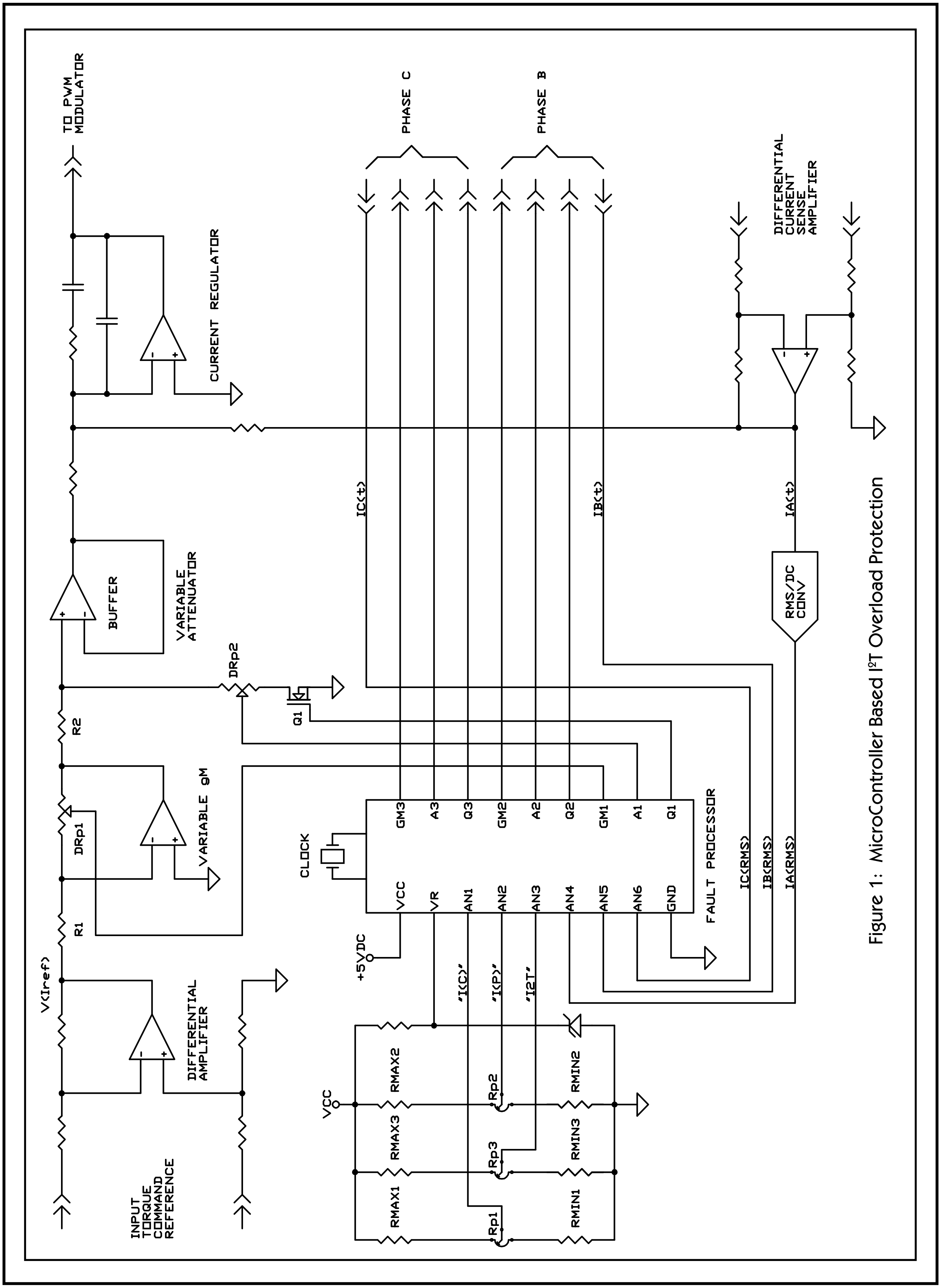

Figure 1: MicroController Based I^2T Overload Protection

That is, the desired Continuous Current is found by *reading* ADC input AN1 and scaling by a factor of 2, the Peak Current is found by *reading* ADC input AN2 and scaling by a factor of 5, and the I^2T Set Point is found by *reading* ADC input AN3, squaring that value and multiplying by 50. Once the 3 program variables (AN1, AN2, AN3) are initialized, they're then used as simple scaling constants by MicroController firmware. These scaling inputs allow the User to configure the Power Amplifier for any BLDC motor with a Continuous Current within the 2 to 10 Ampere range, a Peak Current within the 5 to 25 Ampere range, and an I^2T limit in the range of 50 to 1250 AmpereSquared-Seconds. As an example, assume the motor has a 5 Ampere Continuous Current Rating, a Peak Rating of 15 Amperes and a Peak-Current-Time rating of 0.5 Seconds:

$Ic_m := 5$ $Ip_m := 15$ (Amperes, RMS) $T_m := .5$ (Seconds)

The system can be programmed to operate this motor by adjusting VAN1, VAN2 and VAN3 to:

$V_{AN1} := \frac{Ic_m}{2}$ $V_{AN1} = 2.5$ (Volts)

$V_{AN2} := \frac{Ip_m}{5}$ $V_{AN2} = 3$ (Volts)

$V_{AN3} := \left[\frac{\left[\left(Ip_m^2 - Ic_m^2\right)\cdot T_m\right]}{50}\right]^{\frac{1}{2}}$ $V_{AN3} = 1.414$ (Volts)

This should (presumably!) be done with the Drive in a Stopped condition. If the MicroController includes a Parameter Load Switch, then new settings can immediately be loaded without powering down. Otherwise, power must be Cycled-OFF/ON (MicroController RESET) to read the programmed values. NOTE: The programmed information is not lost, assuming the potentiometer wiper positions are not changed while the power is OFF.

After the MicroController reads the new (VAN1, VAN2, VAN3) data, the Drive is ready to run.

The I^2T algorithm does not use the Peak Current Rating programming information. However, in a Programmable Drive the Drive *transconductance* (i.e., the ratio of Peak Output Current to Peak Command Current) needs to be adjustable. Let's assume the Input Current Command is (represented by) a +/-10 Volt signal that swings 0 to 10 Volts for Full Output Current in the Forward direction, and 0 to -10 volts for the Opposite direction. To be consistent with our above scaling we need to adjust *transconductance* from:

5/10 = 0.5 Amps/Volt
… to …
25/10 = 2.5 Amps/Volt

over the Power Amplifier operating range. For simplicity, let's also assume this can be created with a digital potentiometer (DRp1 … see Fig 1) located in the feedback loop of a programmable amplifier (placed in the input signal path of the Power Amplifier current regulator). The Drive *transconductance* can now be *controlled* by MicroController firmware based on Ip data input at the A/D converter input

AN2. For our example motor (defined above) the current loop *transconductance* needs to be programmed for:

$$G_m := \frac{Ip_m}{10} \qquad G_m = 1.5 \quad \text{(Amps / Volts)}$$

which is accomplished by setting:

DRp1 = [Gm/Gm(max)] **x** R1

Where:

Gm(max) = Ip(max)/10 … or, 2.5Amps/Volt

… and …

R1 is the input resistance (see Fig 1) of the programmable gain amplifier.

Thus, a full-scale command (reference) of +10 Volts can now produce 15 Peak Amperes at the amplifier output.

Programmable Attenuator

We also need to define an adjustable attenuator (or, limiter) that can restrict the Output Current when an Overload is detected. We can do this with a fixed resistor and another digital potentiometer (DRp2), connected as a voltage divider, and located in the Current Loop Command reference signal [V(Iref)] path as shown in Fig 1. In this case, the digital potentiometer is switched to ground by a small-signal FET (Q1) to invoke current limiting. The Gate of Q1 is driven by a MicroController output pin: driven (of course) by the output of the run time I^2T algorithm. The value of the digital potentiometer needs to be set only once, i.e., during the power up initialization routine. The required attenuation is proportional to the ratio of the programmed values of Ic and Ip defined earlier. The limiter must dynamically reduce the full-scale value of V(Iref) (which normally corresponds to the peak [Ip] output current) down to the continuous current limit as defined by Ic. Continuing with our above example, the required attenuation for the 5-ampere continuous, 15-ampere peak, motor is:

$$A := \frac{Ic_m}{Ip_m} \qquad A = 0.333 \qquad \text{(Volts / Volt)}$$

The value of the digital potentiometer (DRp2) is then found from:

$$DRp_2 := \left(\frac{A}{1-A}\right) \cdot R_2 \qquad \text{(Ohms)}$$

With R2 fixed at value of: $R_2 := 10000$ (Ohms)

The digital potentiometer will be automatically adjusted to:

$$Dp_2 := \left(\frac{A}{1-A}\right) \cdot R_2 \qquad Dp_2 = 5 \cdot 10^3 \qquad \text{(Ohms)}$$

Which reduces the full-scale input current command reference:

$V_{Iref} := 10$ (Volts)

to a maximum value of:

$V_{Iref} := V_{Iref} A \qquad V_{Iref} = 3.333$ (Volts)

reducing the maximum commanded current from 15 amperes at V(Iref) equals 10 volts to:
Gm **x** 3.333 = 5 amps (i.e., the programmed continuous current) whenever the I^2T algorithm detects an overload condition.

RMS/DC Converter

The RMS (Root-Mean-Squared) value of the current feedback waveform can be measured with a discrete (or, integrated) Analog RMS-to-DC converter preceding an I^2T circuit or, in Drives based on a MicroController, by computing the RMS value of the instantaneous feedback current within the run time algorithm. In our mixed analog/digital system, let's assume that an Analog RMS to DC converter provides this function and, is located between the output of the current sense amplifier and the MicroController (AN4 analog input) containing the I^2T algorithm.

The I^2T Algorithm

A flowchart of a typical I^2T algorithm is illustrated in Figs 2 and 3. Figure 2 is an initialization routine executed each time the Drive System is powered-up. During this period, all system variables effecting drive operation that are controlled or effected by the MicroController (including the I^2T algorithm) are initialized. This includes initializing any programmable gains, configuring on-chip and off-chip peripherals such as programmable timers, the A/D converter, configuring the I/O ports, retrieving run-time variables from nonvolatile memory and so on. With respect to the I^2T algorithm, the Drive's continuous current and I^2T Limit Set Points must be defined and, the I^2T tracking variable (a variable [memory] location in Random Access Memory or, one of the MicroController CPU Registers) used to continuously monitor the transient overload (I^2T energy) in the run time algorithm … will be set to zero. Finally, the I^2T Limit Data is divided by the fixed sample rate while the Continuous Current is pre-squared before saving for use by the runtime algorithm. These final 2 steps eliminate some multiplications during the run-time routine.

The Continuous Current and I^2T Set Points (required by the I^2T algorithm) may be read from an external programming device like a voltage divider (or, potentiometer) connected to an A/D converter input (described earlier) or, be retrieved from a previously-programmed nonvolatile memory device. In dedicated applications, the Continuous and Peak Current Ratings may simply be pre-programmed constants stored in MicroController program memory.

The run time portion of the routine is shown in Fig 2. This routine is Interrupt Driven by a timer which is typically an On-Chip MicroController peripheral. Given that Thermal Time Constants of the Power Amplifier and Motor vary from a fraction of a second to a few seconds, a reasonable sample rate for the I^2T algorithm is about 1,000 times/second: thus, the period of the interrupt timer should be set to 1 millisecond. The I^2T algorithm can be implemented on each motor phase (this requires 3 current

BLDC Motor Drive Over Current Protection (I^2T Overload) - Part 4

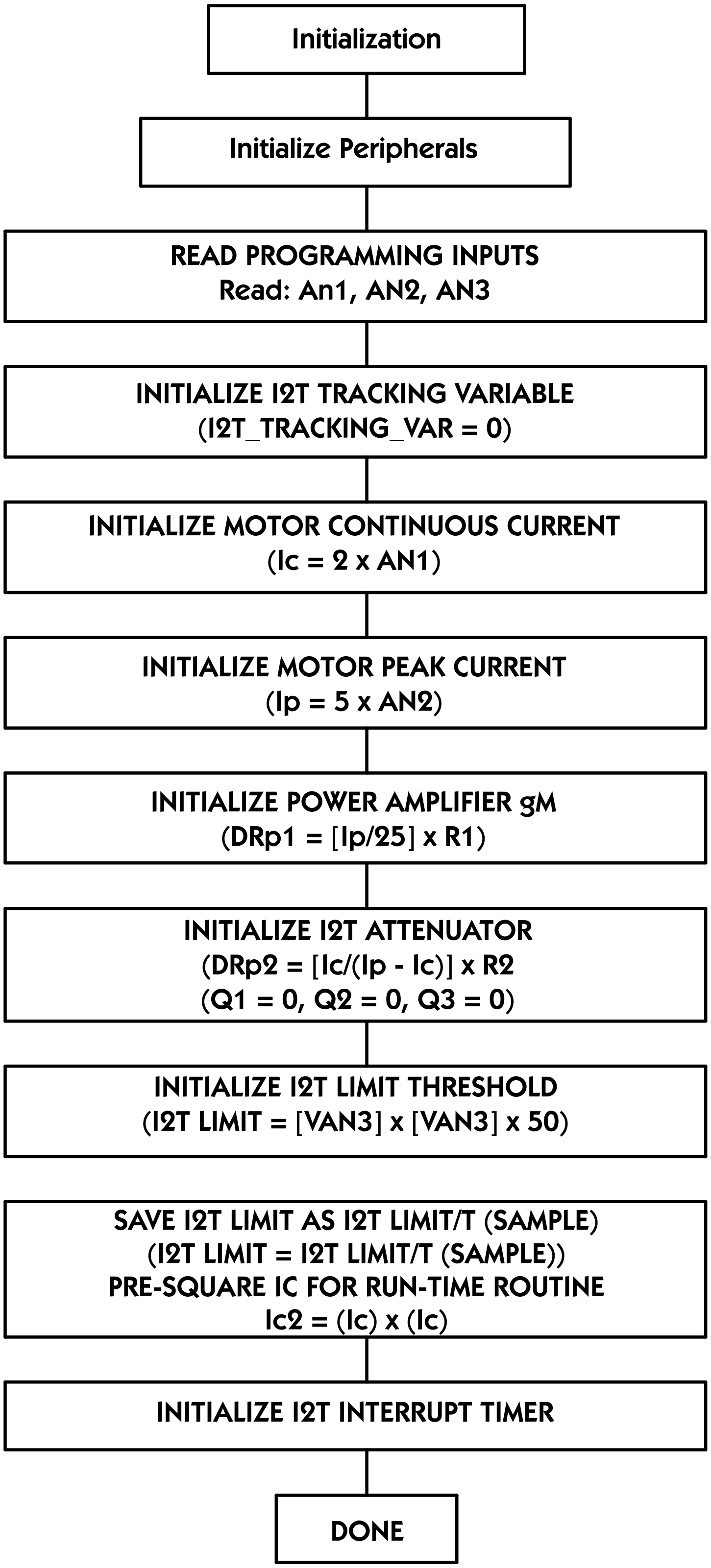

Figure 2: Power-up Initalization

BLDC Motor Drive Over Current Protection (I^2T Overload) - Part 4

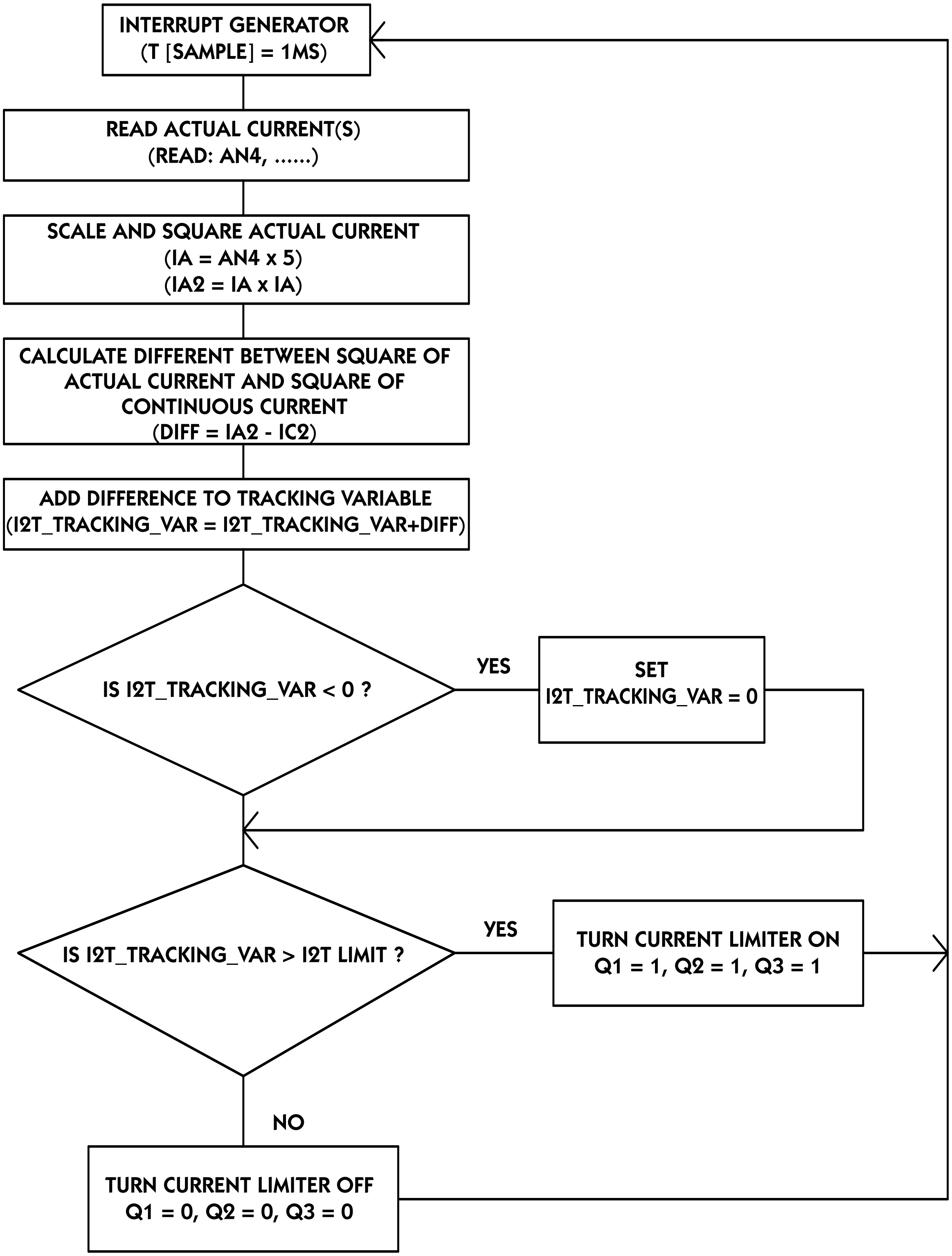

Figure 3: I^2T Run Time Routine (One Phase)

feedback signals and 3 passes through the I^2T algorithm) or, on a composite signal such as the Vector Sum of the 3 Motor Phases and/or the DC Bus Current, as long as the appropriate scaling constants are used to derive the per-phase winding current. The flowchart and block diagram show the implementation of only 1-Phase of a large Power Inverter, which actually consists of 3 individual H-Bridge Power Blocks, each generating 1-of-3 sinusoidal-weighted PWM Output Phases. In this case, *each* phase is *independently* controlled using AC winding current to close the feedback loops: for simplicity, only 1-Phase is defined. At each interrupt, the Output Current is sampled via the MicroController A/D converter, scaled to match the Peak Command Current, then squared. The I^2T algorithm then calculates the difference between the square of the sample and the square of the Continuous Current Limit (Ic) defined during initialization. NOTE: The square of Ic is performed in the initialization routine. If this difference is now multiplied by the sample time (1 millisecond) the result is the incremental increase (or, decrease) in transient energy. However, in order to save a multiplication in the run-time program, the I^2T Limit Set Point was divided by our fixed sample rate before saving that value during initialization. The difference ($Ia^2 - Ic^2$) instead of ($Ia^2 - Ic^2$) **x** T(sample) is now added to the current value of the Tracking Variable. Thus, this Tracking Variable acts as an Accumulator that tracks Drive Overload History. The incremental value added to the Tracking Variable (at each sample) can be positive or negative. Thus, the Tracking Variable grows and falls. However, within the body of the algorithm, it's minimum value is clamped to zero.

Each time the Tracking Variable is updated, it is compared with the I^2T set point defined by Rp3, and divided by the sample time during initialization. If the (accumulated) Tracking Variable value is greater than the I^2T set point divided by T(sample), then the MicroController invokes current limiting by turning on Q1 to engage the programmable attenuator DRp2. Power Amplifier Output Current is then forced to a level no greater than the Continuous Current Limit. If the I^2T Tracking Variable is less than the I^2T Set Point, then no action is taken, i.e., the Power Amplifier is allowed to continue "normal" operation.

Operation of the I^2T algorithm is predictable. Using our previous example with Ip = 15 A, Ic = 5A and T = 0.5 seconds, note that, if the Power Amplifier input reference is held at 10 volts indefinitely, the output current will rise to 15 Amperes Peak (a 300% overload) and the Tracking Variable will begin to rise at a rate of ($15^2 – 5^2$) **x** T(sample) = 0.2 units (per millisecond) and, will reach the programmed I^2T limit after 100/(0.2) = 500 samples or 500 **x** T(sample) = 0.5 seconds as expected. At this point, the current reference will be attenuated from 10 to 3.333 volts and, the Output Current will drop to 5 Amperes when the attenuator (the I^2T Current Limit) is activated.

Note that the Tracking Variable will remain above the Trip Threshold since it can fall only if Iout (Power Amplifier Output Current) is driven below Ic. That is, the incremental value added to the Tracking Variable must swing negative (i.e., Ic must be greater than Iout) to allow the Tracking Variable to fall. This implies the User must reduce V(Iref) and the output power below the Drive Maximum Continuous Rating: this allows the system to return to normal operation. This is required since, the entire Excess Transient Energy available from the Drive has now been consumed and, Time is needed to remove the Excess Heat stored in the system. However, note that driving the system to Peak Output for indefinite periods of time is hardly a normal Drive sequence - which might occur if the Drive input signal connection was broken and the input signal was left floating. In normal operation, the Transient Energy Capability is used to quickly accelerate a load to some set point within rated conditions … nuisance tripping of the I^2T Limit will not normally occur, unless the motor was incorrectly sized with respect to load. You, the reader, can verify that the time allowed in Overload increases inversely with $Ip^2 - Ic^2$, rising, for this particular set of operating conditions, to:

3.2 seconds at Ip = 1.50 **x** Ic
7.1 seconds at Ip = 1.25 **x** Ic
19.1 seconds at Ip = 1.10 **x** Ic
Finally, to infinity at Ip = Ic = Rated Current.

Miscellaneous Implementation Notes

MicroController based implementations of I^2T algorithms appear to be most common - probably due to the reduced circuit complexity (derived from MicroController flexibility) and the innate ability to handle complex math. In larger, high performance Servo Drives, it is common to find a separate MicroController dedicated to fault processing algorithms (such as the I^2T algorithm), while another MicroController (or, analog circuit) implements Current Regulation and Pulse-Width Modulation.In larger drives (particularly sinusoidal PWM BLDC drives), Motor Line Current (to which Motor Continuous, Peak and Peak-Current-Time data are normally referenced) is directly measured and each phase operates with its own Pulse-Width-Modulator and Current Regulator. In this case, an I^2T algorithm can be directly implemented on each Phase … as depicted in our simple example circuit. However, when an Overload is detected in one Phase, ALL Phases should simultaneously invoke Current Limiting.

In smaller drives, operating with trapezoidal BEMF and linear trailing-edge PWM modulation, the Current Regulator and PWM Modulator operate on the vector sum of Motor Currents referred to a single current shunt located in the DC Bus Rail between the Bus Capacitor and the Power Inverter. In this type of system, we can either scale the composite current feedback signal to derive the line current quantities (for use by the I^2T algorithm) or, refer the Motor Thermal Data (e.g., the Continuous, Peak and Peak-Current-Time Rating used to calculate an I^2T Set Point) to Bus Current quantities.

Note that Motor Definition is completely arbitrary with regard to the Ip / Ic ratio and to T … with the exception that Ip and Ic must fall within Ip(max) to Ip(min) and Ic(max) to Ic(min) respectively, and the Motor Peak-Current-Time parameter is bounded by the limits of I^2T(min) to I^2T(max).

Analog I^2T circuits are based on general-purpose Analog Multipliers and RMS-to-DC Converters using the (so-called) implicit solution method in which the Output is averaged with a Low Pass Filter and fed-back to the input. Essentially, the Low Pass Filter associated with RMS-to-DC Converter is configured to model the Motor Thermal Time Constant, as well as being an integration time constant for the RMS-to-DC converters. The net result is, the converter performs a running RMS average that operates similar to the (zero-order) Hold and Accumulate function of the Tracking Variable described above in the digital version. However, an Analog version of an I^2T circuit is less configurable than a MicroController version. This probably accounts for the popularity of the Digital version in general-purpose, configurable drives. The tremendous flexibility of a MicroController is a critical factor since, once we decide to use a MicroController, we also tend to include many other functions in the firmware (and circuitry) as possible.

HITACHI
Inspire the Next

For additional information and/or support, contact Rick Tonis, Sales Manager, (248) 203-6770

HITACHI

Hitachi America, Ltd.
Power and Industrial Division
50 Prospect Ave. •Tarrytown, NY 10591
Tel: (914) 631-0600 • Fax: (914) 631-3672
E-mail: power.devices@hal.hitachi.com
www.hitachi.co.jp/pse

Printed in U.S.A.