Order by AN1937/D
(Motorola Order Number)
Rev. 0, 9/02

3-Phase Switched Reluctance Motor Control with Encoder Using DSP56F80x

Design of a Motor Control Application Based on the Motorola Software Development Kit

Peter Balazovic, Radim Visinka

Contents

1. Introduction

This Application Note describes the design of an advanced 3-Phase Switched Reluctance (SR) motor drive. It is based on Motorola's DSP56F80x family for dedicated motor control devices. The software design takes advantage of the SDK (Software Development Kit) developed by Motorola.

SR motors are gaining wider popularity among variable speed drives. This is due to their simple low-cost construction characterized by an absence of magnets and rotor winding, high level of performance over a wide range of speeds, and fault-tolerant power stage design. for numerous applications, availability and the moderate cost of the necessary electronic components make SR drives a viable alternative to other commonly used motors like AC, BLDC, PM Synchronous or universal motors.

The concept of this application is an advance speed closed loop SR drive with encoder position sensor. An inner current loop with PI controller is included. The encoder position sensor provides an accurate measurement of the actual rotor position necessary for proper commutation. This application serves as an example of an advanced SR motor control. The entire system is designed using a Motorola DSP with SDK support. It also illustrates the usage of dedicated motor control libraries that are included in the SDK. The application helps start the development of the advanced SR drive dedicated to the targeted application.

This Application Note includes a description of Motorola DSP features, basic SR motor theory, system design concept, hardware implementation, and software design including the use of the software visualization tool.

2. Motorola DSP, Advantages and Features

The Motorola DSP56F80x family is well suited for digital motor control, combining a DSP's computational ability with an MCU's controller features on a single chip. These DSPs offer many dedicated peripherals like a Pulse Width Modulation (PWM) unit, Analog-to-Digital Converter (ADC), timers, communications peripherals (SCI, SPI, CAN), on-board Flash and RAM. Generally, all family members are well-suited for Switched Reluctance motor control.

One typical member of the family, the DSP56F805, provides the following peripheral blocks:

- Two Pulse Width Modulator modules (PWMA & PWMB), each with six PWM outputs, three Current Sense inputs, and four Fault inputs; fault tolerant design with deadtime insertion; supports both Center- and Edge- aligned modes
- Twelve bit, Analog to Digital Converters (ADCs), supporting two simultaneous conversions with dual 4-pin multiplexed inputs; the ADC can be synchronized by PWM
- Two Quadrature Decoders (Quad Dec0 & Quad Dec1), each with four inputs, or two additional Quad Timers A & B
- Two dedicated General Purpose Quad Timers totaling 6 pins: Timer C with 2 pins and Timer D with 4 pins
- CAN 2.0 A/B Module with 2-pin ports used to transmit and receive
- Two Serial Communication Interfaces (SCI0 & SCI1), each with two pins, or four additional GPIO lines
- Serial Peripheral Interface (SPI), with configurable 4-pin port, or four additional GPIO lines
- Computer Operating Properly (COP) Watchdog Timer
- Two dedicated external interrupt pins
- Fourteen dedicated General Purpose I/O (GPIO) pins, 18 multiplexed GPIO pins
- External reset pin for hardware reset
- JTAG/On-Chip Emulation (OnCE)
- Software-programmable, Phase Lock Loop-based frequency synthesizer for the DSP core clock

Table 2-1. Memory Configuration

	DSP56F801	DSP56F803	DSP56F805	DSP56F807
Program Flash	8188 x 16-bit	32252 x 16-bit	32252 x 16-bit	61436 x 16-bit
Data Flash	2K x 16-bit	4K x 16-bit	4K x 16-bit	8K x 16-bit
Program RAM	1K x 16-bit	512 x 16-bit	512 x 16-bit	2K x 16-bit
Data RAM	1K x 16-bit	2K x 16-bit	2K x 16-bit	4K x 16-bit
Boot Flash	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit

The most interesting peripherals, from the switched reluctance motor control point of view, are the fast Analog-to-Digital Converter (ADC) and the Pulse-Width-Modulation (PWM) on-chip modules. They offer extensive freedom of configuration, enabling efficient control of SR motors.

The **PWM module** incorporates a PWM generator, enabling the generation of control signals for the motor power stage. The module has the following features:

- Three complementary PWM signal pairs, or six independent PWM signals
- Complementary channel operation
- Deadtime insertion
- Separate top and bottom pulse width correction via current status inputs or software
- Separate top and bottom polarity control
- Edge-aligned or center-aligned PWM signals
- 15 bits of resolution
- Half-cycle reload capability
- Integral reload rates from one to 16
- Individual software-controlled PWM output
- Programmable fault protection
- Polarity control
- 20mA current sink capability on PWM pins
- Write-protectable registers

The SR motor control application utilizes the PWM module set in independent PWM mode, permitting fully independent generation of control signals for all switches of the power stage. In addition to the PWM generators, the PWM outputs can be controlled separately by software, allowing the setting of the control signal to logical 0 or 1. Thus, the state of the control signals can be changed instantly at a given rotor position (phase commutation) without changing the contents of the PWM value registers. This change can be made asynchronously with the PWM duty cycle update.

The **Analog-to-Digital Converter** (ADC) consists of a digital control module and two analog sample and hold (S/H) circuits. It has the following features:

- 12-bit resolution
- Maximum ADC clock frequency is 5MHz with 200ns period
- Single conversion time of 8.5 ADC clock cycles (8.5 x 200 ns = 1.7µs)
- Additional conversion time of 6 ADC clock cycles (6 x 200 ns = 1.2µs)
- Eight conversions in 26.5 ADC clock cycles (26.5 x 200 ns = 5.3µs) using simultaneous mode
- ADC can be synchronized to the PWM via the sync signal
- Simultaneous or sequential sampling
- Internal multiplexer to select two of eight inputs
- Ability to sequentially scan and store up to eight measurements
- Ability to simultaneously sample and hold two inputs
- Optional interrupts at end of scan at zero crossing or if an out-of-range limit is exceeded
- Optional sample correction by subtracting a pre-programmed offset value
- Signed or unsigned result
- Single ended or differential inputs

The application utilizes the ADC on-chip module in simultaneous mode and sequential scan. The sampling is synchronized with the PWM pulses for precise sampling and reconstruction of phase currents. Such a configuration allows instant conversion of the desired analog values of all phase currents, voltages and temperatures.

3. Target Motor Theory

3.1 Switched Reluctance Motor

A Switched Reluctance (SR) motor is a rotating electric machine where both stator and rotor have salient poles. The stator winding is comprised of a set of coils, each of which is wound on one pole. The rotor is created from lamination in order to minimize the eddy-current losses.

SR motors differ in the number of phases wound on the stator. Each of them has a certain number of suitable combinations of stator and rotor poles. **Figure 3-1** illustrates a typical 3-Phase SR motor with a 6/4 (stator/rotor) pole configuration.

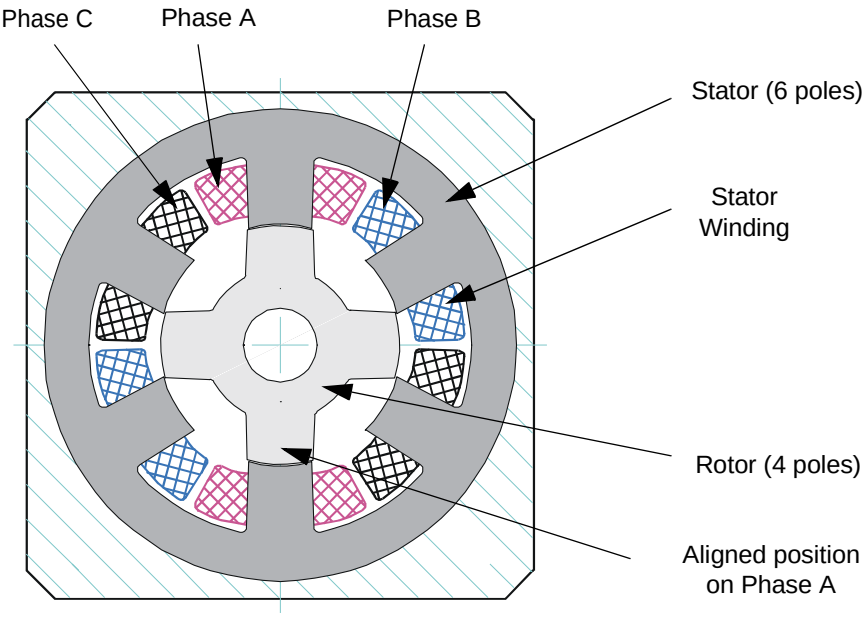

Figure 3-1. 3-Phase 6/4 SR Motor

The motor is excited by a sequence of current pulses applied at each phase. The individual phases are consequently excited, forcing the motor to rotate. The current pulses need to be applied to the respective phase at the exact rotor position relative to the excited phase. When any pair of rotor poles is exactly in line with the stator poles of the selected phase, the phase is said to be in an aligned position, i.e., the rotor is in the position of maximal stator inductance (see **Figure 3-1**). If the interpolar axis of the rotor is in-line with the stator poles of the selected phase, the phase is said to be in an unaligned position - the rotor is in a position of minimal stator inductance. The inductance profile of SR motors is triangular shaped, with maximum inductance when it is in an aligned position and minimum inductance when unaligned. **Figure 3-2** illustrates the idealized triangular-like inductance profile of all three phases of an SR motor with phase A highlighted. The individual Phases A, B, and C are shifted electrically by 120^{o} relative to each other. The interval, when the respective phase is powered, is called the dwell angle - θ_{dwell}. It is defined by the turn-on θ_{on} and the turn-off θ_{off} angle.

When the voltage is applied to the stator phase, the motor creates torque in the direction of increasing inductance. When the phase is energized in its minimum inductance position, the rotor moves to the forthcoming position of maximal inductance. The movement is defined by the magnetization characteristics of the motor. A typical current profile for a constant phase voltage is shown in **Figure 3-2**. For a constant phase voltage the phase current has its maximum in the position when the inductance starts to increase. This corresponds to the position when the rotor and the stator poles start to overlap. When the phase is turned off, the phase current falls to zero. The phase current present in the region of decreasing inductance generates negative torque. The torque generated by the motor is controlled by the applied phase voltage and by the appropriate definition of switching turn-on and turn-off angles. For more details, see **[3]**.

As is apparent from the description, the SR motor requires position feedback for motor phase commutation. In many cases, this requirement is addressed by using position sensors, like encoders, Hall sensors, etc. The result is that the implementation of mechanical sensors increases costs and decreases system reliability. Traditionally, developers of motion control products have attempted to lower system costs by reducing the number of sensors. A variety of algorithms for sensorless control have been developed, most of which involve evaluation of the variation of magnetic circuit parameters that are dependent on the rotor position **[2]**, **[5]**.

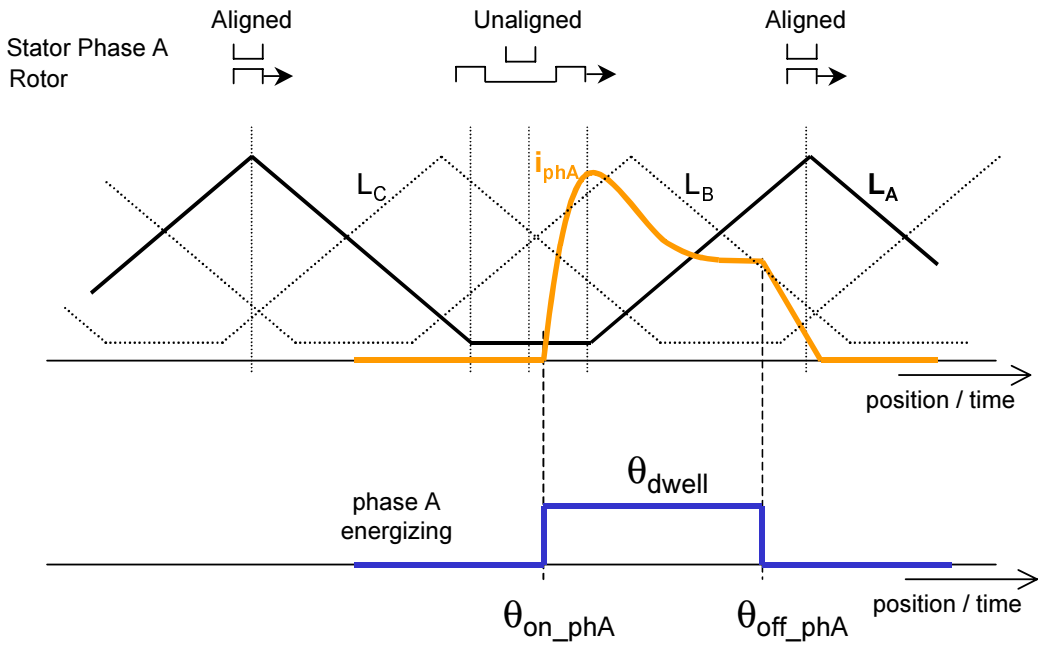

Figure 3-2. Phase Energizing

The motor itself is a low cost machine of simple construction. High-speed operation is possible, thus the motor is suitable for high speed applications, like vacuum cleaners, fans, white goods, etc. As discussed above, the disadvantage of the SR motor is the need for shaft-position information for the proper switching of individual phases. Also, the motor structure causes noise and torque ripple. The greater the number of poles, the smoother the torque ripple, but motor construction and control electronics become more expensive. Torque ripple can also be reduced by advanced control techniques, such as phase current profiling.

3.2 Mathematical Description of an SR Motor

An SR motor is a highly non-linear system, so a non-linear theory describing the behavior of the motor was developed. Based on this theory, a mathematical model can be created. On one hand it enables the simulation of SR motor systems and on the other hand, it makes the development and implementation of sophisticated algorithms for controlling the SR motor easier.

The electromagnetic circuit of the SR motor is characterized by non-linear magnetization. **Figure 3-3** illustrates a magnetization characteristic for a specific SR motor **[1]**. It is a function between the magnetic flux ψ, the phase current *i* and the motor position θ. The influence of the phase current is mostly apparent in the aligned position, where saturation effects can be observed.

The magnetization characteristic curve defines the non-linearity of the motor. The torque generated by the motor phase is a function of the magnetic flux, therefore the phase torque is not constant for a constant phase current for different motor positions. This creates torque ripple and noise in the SR motor.

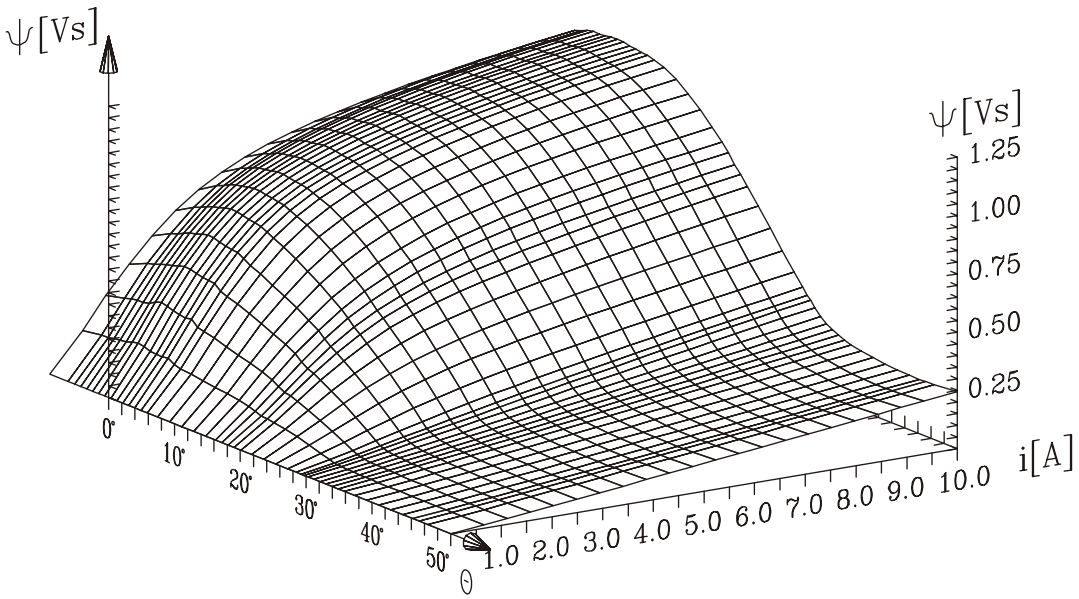

Figure 3-3. Magnetization Characteristics of the SR Motor

A mathematical model of an SR motor can be developed. The model is based on the electrical diagram of the motor, incorporating the phase resistance and phase inductance **[1]**. The diagram for one phase is illustrated in **Figure 3-4**.

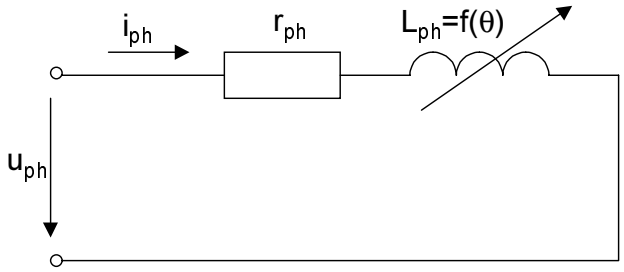

Figure 3-4. Electrical Diagram of One SR Motor Phase

According to **Figure 3-4**, any voltage applied to a phase of the SR motor can be described as a sum of voltage drops in the phase resistance and induced voltages on the phase inductance:

$$u_{ph}(t) = r_{ph} \cdot i_{ph}(t) + u_{Lph}(t) \qquad \text{(EQ 3-1.)}$$

where:

- u_{ph} is the voltage applied to a phase
- r_{ph} is the phase resistance
- i_{ph} is the phase current
- u_{Lph} is the induced voltages over the phase inductance.

The equation **(EQ 3-1.)** supposes that all phases are independent and have no mutual influence.

The induced voltage u_{Lph} is defined by the magnetic flux linkage Ψ_{ph}, that is a function of the phase current i_{ph} and the rotor position θ_{ph}. So the induced voltage can be expressed as:

$$u_{Lph}(t) = \frac{d\Psi_{ph}(i_{ph}, \theta_{ph})}{dt} = \frac{\partial\Psi_{ph}(i_{ph}, \theta_{ph})}{\partial i_{ph}} \cdot \frac{di_{ph}}{dt} + \frac{\partial\Psi_{ph}(i_{ph}, \theta_{ph})}{\partial\theta_{ph}} \cdot \frac{d\theta_{ph}}{dt} \qquad \text{(EQ 3-2.)}$$

Then the phase voltage can be expressed as:

$$u_{ph}(t) = r_{ph} \cdot i_{ph}(t) + \frac{d\Psi_{ph}(i_{ph}, \theta_{ph})}{dt} \qquad \text{(EQ 3-3.)}$$

or:

$$u_{ph}(t) = r_{ph} \cdot i_{ph}(t) + \frac{\partial\Psi_{ph}(i_{ph}, \theta_{ph})}{\partial i_{ph}} \cdot \frac{di_{ph}}{dt} + \frac{\partial\Psi_{ph}(i_{ph}, \theta_{ph})}{\partial\theta_{ph}} \cdot \omega \qquad \text{(EQ 3-4.)}$$

where:

- ω is the angular speed of the motor.

The torque M_{ph} generated by one phase can be expressed as:

$$M_{ph} = \int_{0}^{I_{ph}} \frac{\partial \Psi_{ph}(i_{ph}, \theta_{ph})}{\partial \theta_{ph}} di_{ph} \quad \text{(EQ 3-5.)}$$

The mathematical model of an SR motor is then represented by a system of equations, describing the conversion of electromechanical energy.

For 3-Phase SR motors the equation **(EQ 3-4.)** can be expanded as follows:

$$u_a(t) = r_a \cdot i_a(t) + \frac{\partial \Psi_a(i_a, \theta_a)}{\partial i_a} \cdot \frac{di_a}{dt} + \frac{\partial \Psi_a(i_a, \theta_a)}{\partial \theta_a} \cdot \omega \quad \text{(EQ 3-6.)}$$

$$u_b(t) = r_b \cdot i_b(t) + \frac{\partial \Psi_b(i_b, \theta_b)}{\partial i_b} \cdot \frac{di_b}{dt} + \frac{\partial \Psi_b(i_b, \theta_b)}{\partial \theta_b} \cdot \omega \quad \text{(EQ 3-7.)}$$

$$u_c(t) = r_c \cdot i_c(t) + \frac{\partial \Psi_c(i_c, \theta_c)}{\partial i_c} \cdot \frac{di_c}{dt} + \frac{\partial \Psi_c(i_c, \theta_c)}{\partial \theta_c} \cdot \omega \quad \text{(EQ 3-8.)}$$

where *a, b, c* index the individual phases.

As stated in the above equations, the mutual effect between individual phases is not considered.

3.3 Digital Control of an SR Motor

The SR motor is driven by voltage strokes coupled with the given rotor position. The profile of the phase current together with the magnetization characteristics define the generated torque and thus the speed of the motor. Due to this fact, the motor requires electronic control for operation. Several power stage topologies are being implemented, according to the number of motor phases and the desired control algorithm. The particular structure of the SR power stage structure defines the freedom of control for an individual phase.

A power stage with two independent power switches per motor phase is the most used topology. Such a power stage for 3-Phase SR motors is illustrated in **Figure 3-5**. It permits control of the individual phases fully independent of each other and thus permits the widest freedom of control. Other power stage topologies share some of the power devices for several phases, thus saving on power stage cost, but with these the phases cannot be controlled fully independently. Note that this particular topology of SR power stage is fault tolerant, in contrast to power stages of AC induction motors, because it eliminates the possibility of a rail-to-rail short circuit.

During normal operation, the electromagnetic flux in an SR motor is not constant and must be built for every stroke. In the motoring period, these strokes correspond to the rotor position when the rotor poles are approaching the corresponding stator pole of the excited phase. In the case of Phase A, shown in **Figure 3-1**, the stroke can be established by activating the switches Q1 and Q2. At low-speed operation the Pulse Width Modulation (PWM), applied to the corresponding switches, modulates the voltage level.

Two basic switching techniques can be applied:

- Soft switching - where one transistor is left turned on during the whole commutation period and PWM is applied to the other one
- Hard switching - where PWM is applied to both transistors simultaneously

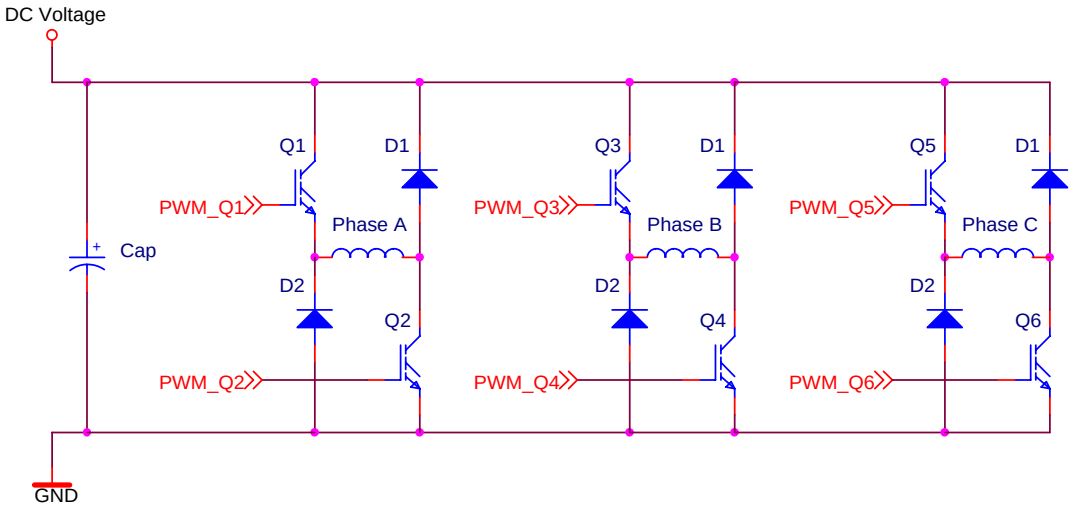

Figure 3-5. 3-Phase SR Power Stage

Figure 3-6 illustrates both soft and hard switching PWM techniques. The control signals for the upper and the lower switches of the above-described power stage define the phase voltage and thus the phase current. The soft switching technique generates lower current ripple compared to the hard switching technique. Also, it produces lower acoustic noise and less EMI. Therefore, soft switching techniques are often preferred for motoring operation. For more details, see **[3]**.

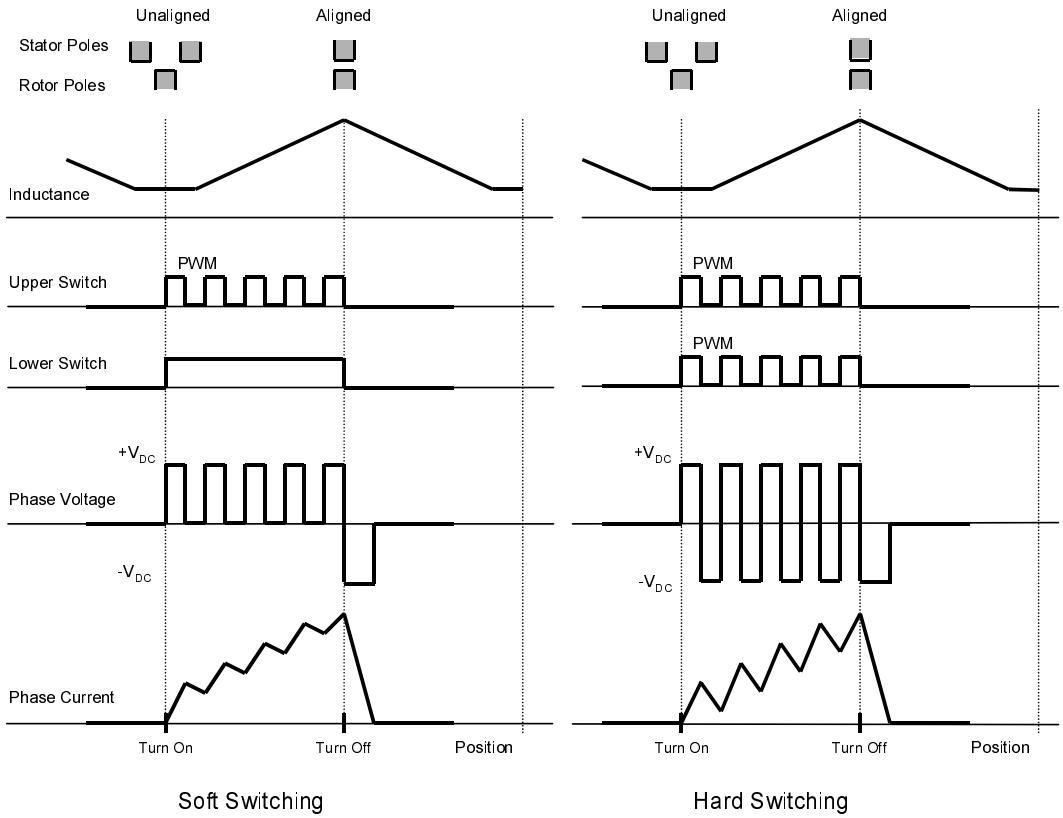

Figure 3-6. Soft Switching and Hard Switching

3.4 Voltage and Current Control for SR Motors

A number of control techniques for SR motors exist. They differ in the structure of the control algorithm and in position evaluation. Two basic techniques for controlling SR motors can be distinguished, according to the motor variables that are being controlled:

- Voltage control - where phase voltage is a controlled variable
- Current control - where phase current is a controlled variable

3.4.1 Voltage Control of an SR Motor

In voltage control techniques, the voltage applied to the motor phases is constant during the complete sampling period of the speed control loop. The commutation of the phases is linked to the position of the rotor.

The voltage applied to the phase is directly controlled by a speed controller. The speed controller processes the speed error, the difference between the desired speed and the actual speed, and generates the desired phase voltage. The phase voltage is defined by a PWM duty cycle implemented at the DC-Bus voltage of the SR inverter. The phase voltage is constant during a complete dwell angle. The technique is illustrated in **Figure 3-7**. The current and the voltage profiles can be seen in **Figure 3-8**. The phase current is at its peak at the position when the inductance starts to increase (stator and rotor poles start to overlap) due to the change in the inductance profile.

.

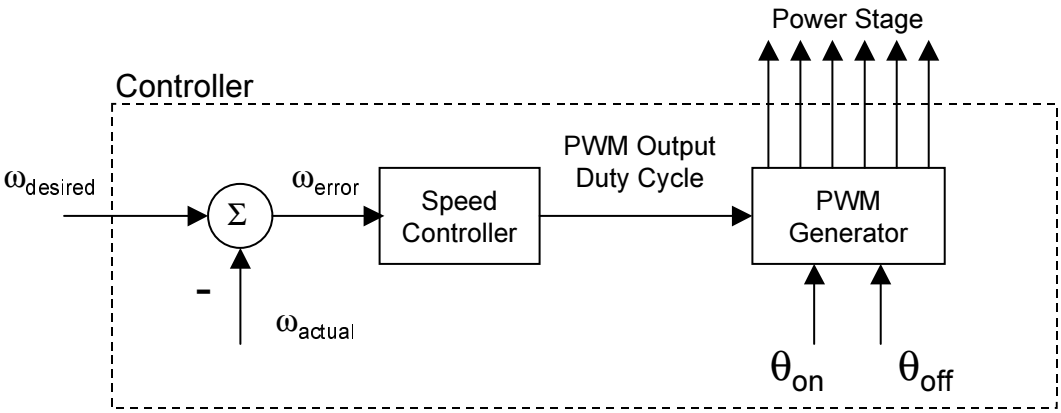

Figure 3-7. Voltage Control Technique

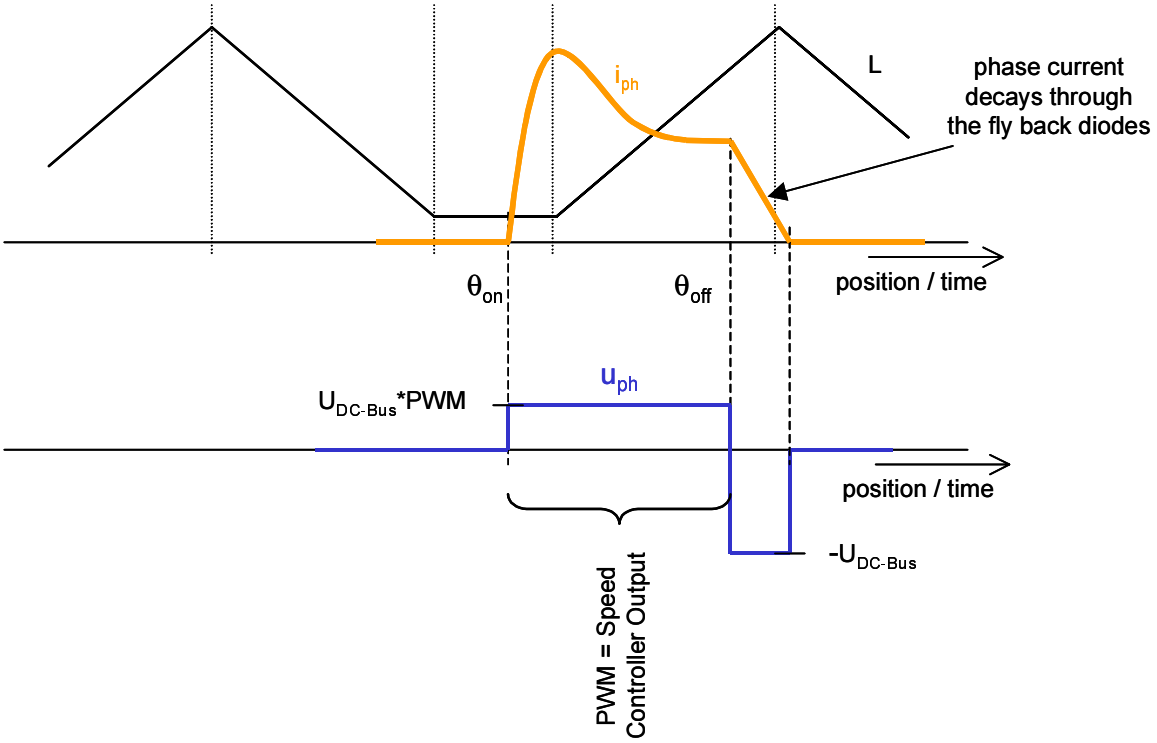

Figure 3-8. Voltage Control Technique - Voltage and Current Profiles

3.4.2 Current Control of an SR Motor

In current control techniques the voltage applied to the motor phases is modulated to reach the desired current at the powered phase. For most applications, the desired current is constant during the complete sampling period of the speed control loop. The commutation of the phases is linked to the position of the rotor.

The voltage applied to the phase is controlled by a current controller with an external speed control loop. The speed controller processes the speed error, the difference between the desired speed and the actual speed, and generates the desired phase current. The current controller evaluates the difference between actual and desired phase current and calculates the appropriate PWM duty cycle. The phase voltage is defined by a PWM duty cycle implemented at the DC-Bus voltage of the SR inverter. Thus, the phase voltage is modulated at the rate of the current control loop. This technique is illustrated in **Figure 3-9**.

The processing of the current controller needs to be linked to the commutation of the phases. When the phase is turned on (commutated), a duty cycle of 100% is applied to the phase. The increasing actual phase current is regularly compared to the desired current. As soon as the actual current slightly exceeds the desired current, the current controller is turned on. Current controller controls the output of the duty cycle until the phase is turned off (following commutation). The procedure is repeated for each commutation cycle of the motor. The current and the voltage profiles can be seen in **Figure 3-10**. In ideal cases the phase current is controlled to follow the desired current.

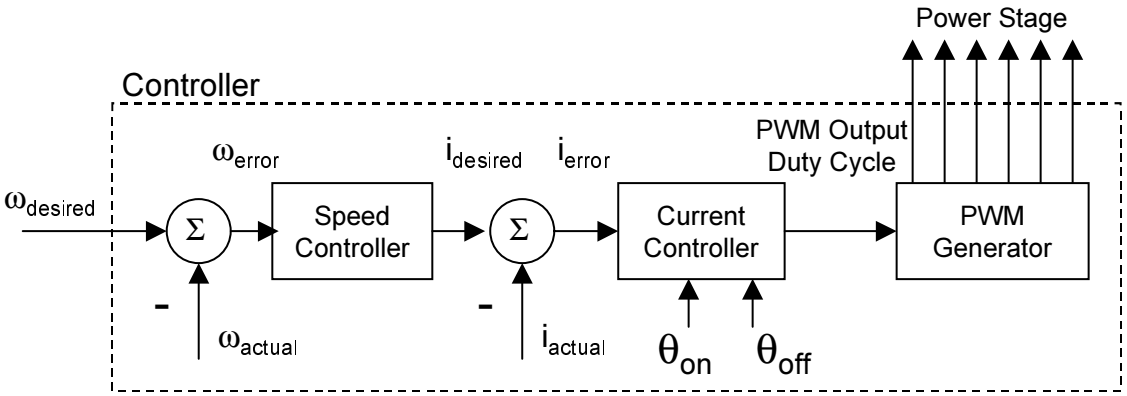

Figure 3-9. Current Control Technique

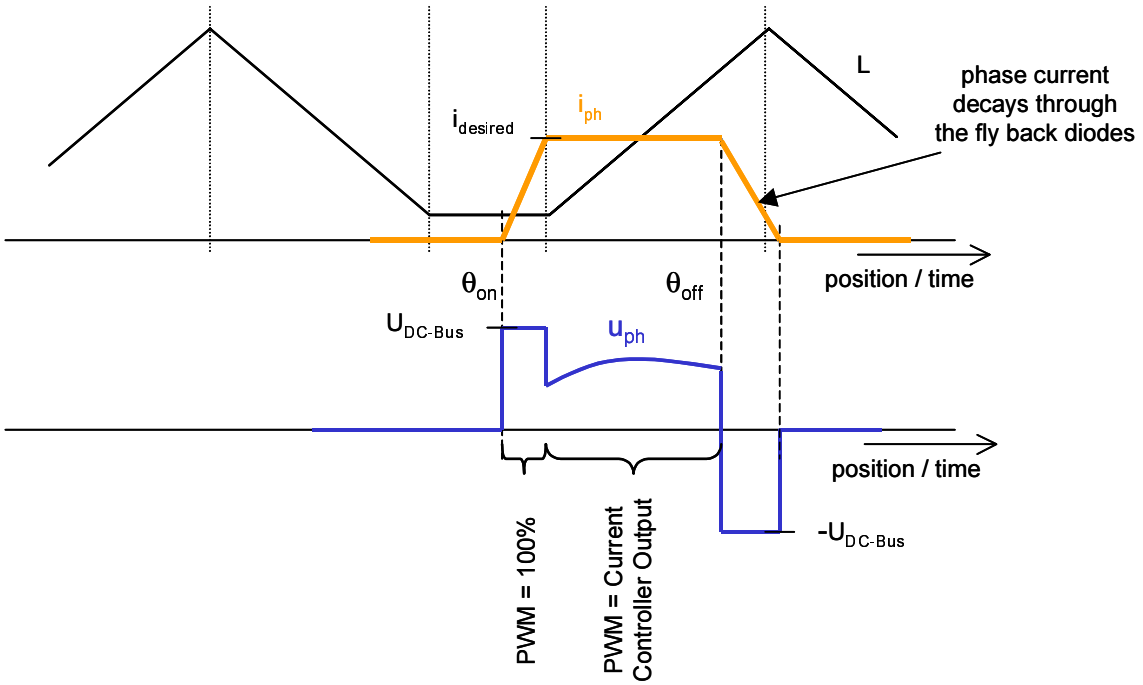

Figure 3-10. Current Control Technique - Voltage and Current Profiles

4. Switched Reluctance Motor Control Techniques with Encoder Position Sensor

A single-chip control system provides sufficient computational power for advanced algorithms permitting efficient motor control over wide speed ranges. There are several ways to control an SR motor. This control technique presents the current control method with shaft position information.

4.1 Encoder Sensor

Whenever mechanical rotary motions have to be monitored, the encoder is the most important interface between the mechanics and the control unit. Encoders transform rotary or linear movement into a sequence of electrical pulses. A rotary encoder can differentiate a number of discrete positions per revolution. The number of segments determines the resolution of the movement and hence the accuracy of the position and this number is called its points per revolution. The speed of an encoder is in counts-per-second.

Although there are various kinds of digital encoders, the most common one is the optical encoder. Rotary and linear optical encoders are used frequently for motion and position sensing. A disc or a plate containing opaque and transparent segments passes between a light source (such an LED) and detector to interrupt a light beam. The electronic signals that are generated are then fed into the DSP controller where position and velocity information is calculated based upon the signals received. Many incremental encoders also have a feature called the index pulse. In rotary encoders an index pulse occurs once per encoder revolution. It is used to establish an absolute mechanical reference position within one encoder count of the 360° encoder rotation. The index signal can be used to do several tasks in the system. It can be used to reset or preset the position counter and/or generate an interrupt signal to the system controller.

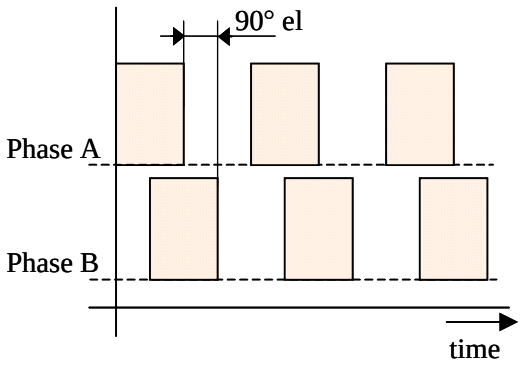

Figure 4-1. Quadrature Encoder Signals

Quadrature encoders are a particular kind of incremental encoder with at least two output signals, generally called Phase A and Phase B. As seen in **Figure 4-1**, channel B is offset 90 degrees from channel A. The addition of a second channel provides direction information in the feedback signal. This signal, leading or lagging by 90 electrical degrees, guarantees the exact determination of the direction of rotation at all times. The ability to detect direction is critical if encoder rotation stops on a pulse edge. Without the ability to decode direction, the counter may count each transition through the rising edge of the signal and lose position. Another benefit of the quadrature signal scheme is the ability to electronically multiply the counts during one encoder cycle. In the times-1 mode, all counts are generated on the rising edges of channel A. In the times-2 mode, both the rising and falling edges of channel A are used to generate counts. In the times-4 mode, the rising and falling edges of channel A and channel B are used to generate counts. This increases the resolution by a factor of four. For encoders with sine wave output, the channels may be interpolated for very high resolution.

4.2 Commutation Angle Calculation

In an SR motor the switched-on and switched-off angles are complex functions of many parameters and are variable for optimum operation. Their fine tuning is necessary to maintain optimum performance at different motor speed and load conditions. The control of firing angle can be accomplished a number of ways and strongly depends on position sensor. If the position information is precisely acquired, it is possible to suitably utilize a sophisticated algorithm.

This control technique varies the firing angle continuously with the fixed dwell angle. The switched-on angle is calculated in such a way that the excitation current should reach the maximum defined value at the beginning of the stator and rotor tooth overlap. The phase current is built up in corresponding windings of the stator since the inductance is at a minimum level in an unaligned position and there is adequate time to increase it to the desired value before the motoring torque is being produced. The conduction angle remains fixed through the entire run of the application to ensure the phase current is decreased before reaching the braking region (following the aligned position). The calculation neglects the stator winding resistance, which simplifies the equation. The resistance neglect can be recognized only at large values of resistance R, which is the case of very small switched reluctance machines.

Figure 4-2 explains the proposed algorithm for advance angle calculation. The computation method is derived from **(EQ 3-6.)** - **(EQ 3-8.)** and is rearranged into the following expression as:

$$u_{ph} = r_{ph} i_{ph} + L_{ph} \cdot \frac{di_{ph}}{dt} + i_{ph} \cdot \frac{dL_{ph}}{d\theta} \cdot \omega \qquad \text{(EQ 4-1.)}$$

where:

- u_{ph} is the voltage applied to a phase
- r_{ph} is the phase resistance
- i_{ph} is the phase current
- L_{ph} is the phase inductance
- θ is the rotor position

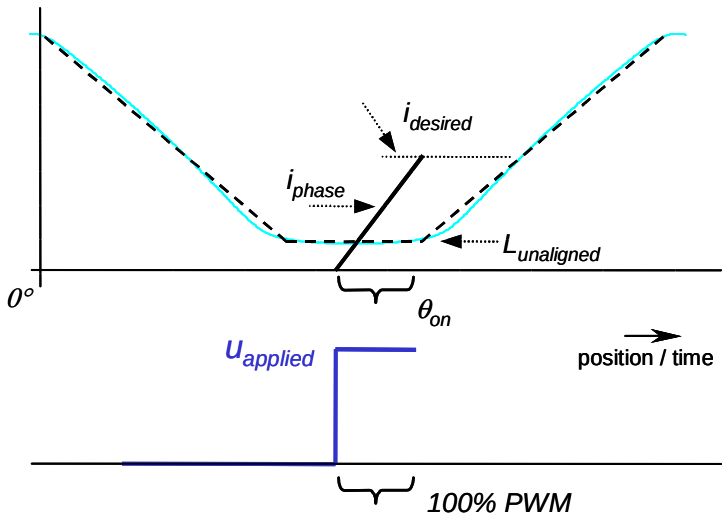

Figure 4-2. Commutation Angle Calculation

The unaligned phase inductance is considered as constant near the turn-on instant. If voltage drop across phase resistance is neglected, then the following expression is given as **(EQ 4-2.)** using a first order approximation:

$$\theta_{on} = L_{unaligned} \cdot \frac{i_{desired}}{u_{ph}} \cdot \omega_{actual} \qquad \text{(EQ 4-2.)}$$

where:

θ_{on} is the advanced angle

$i_{desired}$ is the desired current to be achieved

$L_{unaligned}$ is the unaligned inductance

u_{phase} is the applied phase voltage

ω_{actual} is the actual rotor speed

4.3 Commutation Strategy

In general, the commutation strategy determines the performance of the SR motor. The commutation method uses rotor position feedback to derive the commutating signals for the inverter switches. The controlled parameters are the applied phase voltage and the turn-on angle θ_{on}. The dwell angle is fixed prior to motor starts. The number of commutations per mechanical revolution is proportional to the number of rotor poles and number of stator phases **(EQ 4-3.)**. It arises from the mechanical construction of the SR motor. The number of motor commutations is calculated as follows:

$$NumOfCommut = N_r \cdot m \qquad \text{(EQ 4-3.)}$$

where:

NumOfCommut is the number of commutations per one mechanical revolution

N_r is the number of rotor poles

m is the number of stator phases

An SR motor is usually described in terms of low-speed and high-speed regions. The low-speed operating region is graphically depicted in **Figure 4-3**. In this low-speed operating area, the phase current can be arbitrarily controlled to any desired value. Increasing the rotor speed makes it difficult to control the phase current. There is an influence of back-EMF effect combined with a diminishing amount of time to perform the commutation.

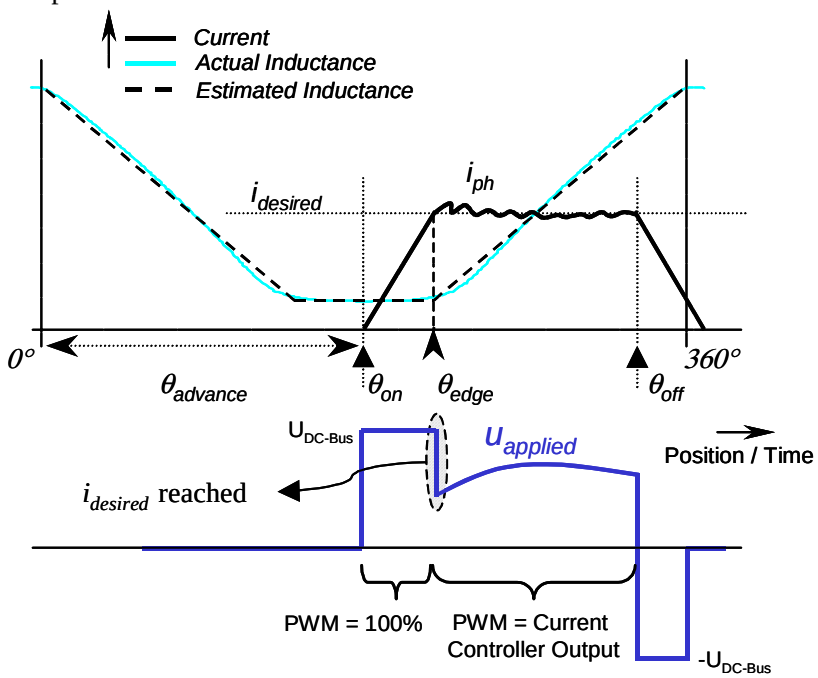

Figure 4-3. Commutation Strategy

The commutation itself can be performed in a number of ways. The presented control technique utilizes the encoder sensor information to initiate the commutation routine, which ensures turn-off of the previous stator phase, and consecutively the next stator phase is turned on depending on the direction of the rotor rotation. The appropriate firing angle, θ_{on}, is calculated through advance angle calculation (see **Section 4.2**). The commutation software algorithm determines the necessary advance angle, $\theta_{advance}$, for turning on the correct stator phase. The full DC-Bus voltage is applied after switching on the correct phase in the $\theta_{advance}$ instant. If the actual value of phase current exceeds the desired current value then the current controller with sufficient controller initialization is started to maintain the actual value of the phase current within the requested magnitude. This is achieved by chopping the DC-Bus voltage.

The simplest scheme is to leave the lower transistor on during current regulation and to switch the upper one on and off at a high fixed PWM frequency with a varying duty cycle. This strategy is often called soft switching (see **Figure 3-6**). The current waveform during soft switching is similar to that shown in **Figure 4-3**.

4.4 The Current Controller

Basically, there are three different modes of operation, namely, voltage control, current control, and single-pulse control. The current control method is normally used to control the torque efficiently, while single-pulse mode is entered for high-speed operation. The main difficulty when designing switched reluctance motor current controllers is that the winding back electromotive force (back-EMF) and electrical time constant vary significantly within one electrical cycle and with the motor speed and phase current level. The voltage equation of the SRM is given by **(EQ 3-4.)**. This equation indicates a nonlinear model which is dependent on position, current and speed. The electrical time constant of a phase winding and the back-EMF vary greatly with current and rotor position.

As **Figure 4-3** implies, the current controller is switched on when the desired stator phase current is reached. At this point, the slope of increasing inductance (inductance derivation over position) is considered as a constant value, and the phase current is preserved at a defined target value; then **(EQ 3-4.)** can be rearranged as follows:

$$u_{phase\_applied}(t) = r_{ph} \cdot i_{ph}(t) + i_{ph}(t) \cdot \frac{dL_{ph}(\theta_{ph})}{d\theta_{ph}} \cdot \omega \qquad \text{(EQ 4-4.)}$$

The applied phase voltage is roughly maintained near the value of **(EQ 4-4.)**, where i_{ph} is the desired phase current, ω is the actual angular speed of the rotor. Derivation over the position of the corresponding phase inductance is determined from motor parameter measurement. Knowing these parameters, the initial current controller is set up using **(EQ 4-4.)** in the time instance (red point - see **Figure 4-4**) when the controller is switched on.

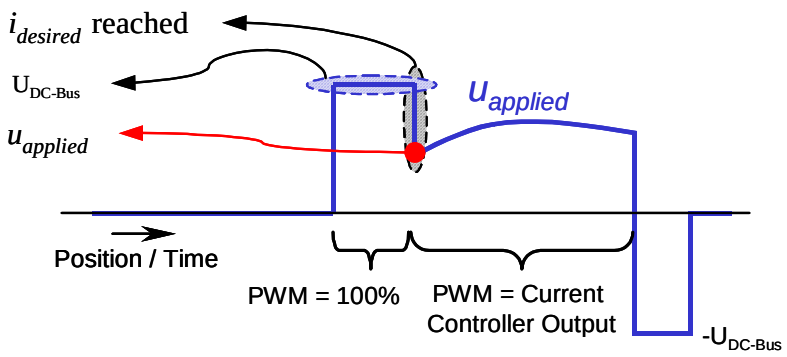

Figure 4-4. Phase Voltage Generation

5. System Design

5.1 System Outline

This system is designed to drive a 3-Phase SR motor. The application meets the following performance specifications:

- Speed control of an SR motor with Encoder position sensor with an inner current closed loop
- Targeted for DSP56F803EVM, DSP56F805EVM, DSP56F807EVM
- Running on a 3-Phase SR HV Motor Control Development Platform at a variable line voltage of between 115V AC and 230V AC (voltage range -15% ... +10%)
- The control technique incorporates
  - current SRM control with speed closed loop
  - motor starts from any motor position with rotor alignment
  - one direction of rotation
  - motoring mode
  - minimal speed 600 rpm
  - maximal speed 2600 rpm at input power line 230V AC
  - maximal speed 1600 rpm at input power line 115V AC
- Encoder position reference for commutation
- Manual Interface (Start/Stop switch, Up/Down push button control, LED indicator)
- PC master software control interface (motor Start/Stop, speed set-up)
- Power stage identification
- DC-Bus over-voltage, DC-Bus under-voltage, DC-Bus over-current and over-heating fault protection
- PC master software Monitor
  - graphical control page (required speed, actual motor speed, operational mode PC/manual, start/stop status, drive fault status, DC-Bus voltage level, identified power stage boards, system status)
  - speed scope (observes actual and desired speeds)
  - current controller (observes actual and desired phase current, applied phase voltage)

5.2 Application Description

For the drive, a standard system concept was chosen (see **Figure 5-1**). The system incorporates the following hardware parts:

- A 3-Phase SR high-voltage development platform (power stage with optoisolation board, motor brake)
- Feedback sensors: DC-Bus voltage, current Phase A, current Phase B, current Phase C, temperature
- A DSP56F80x controller

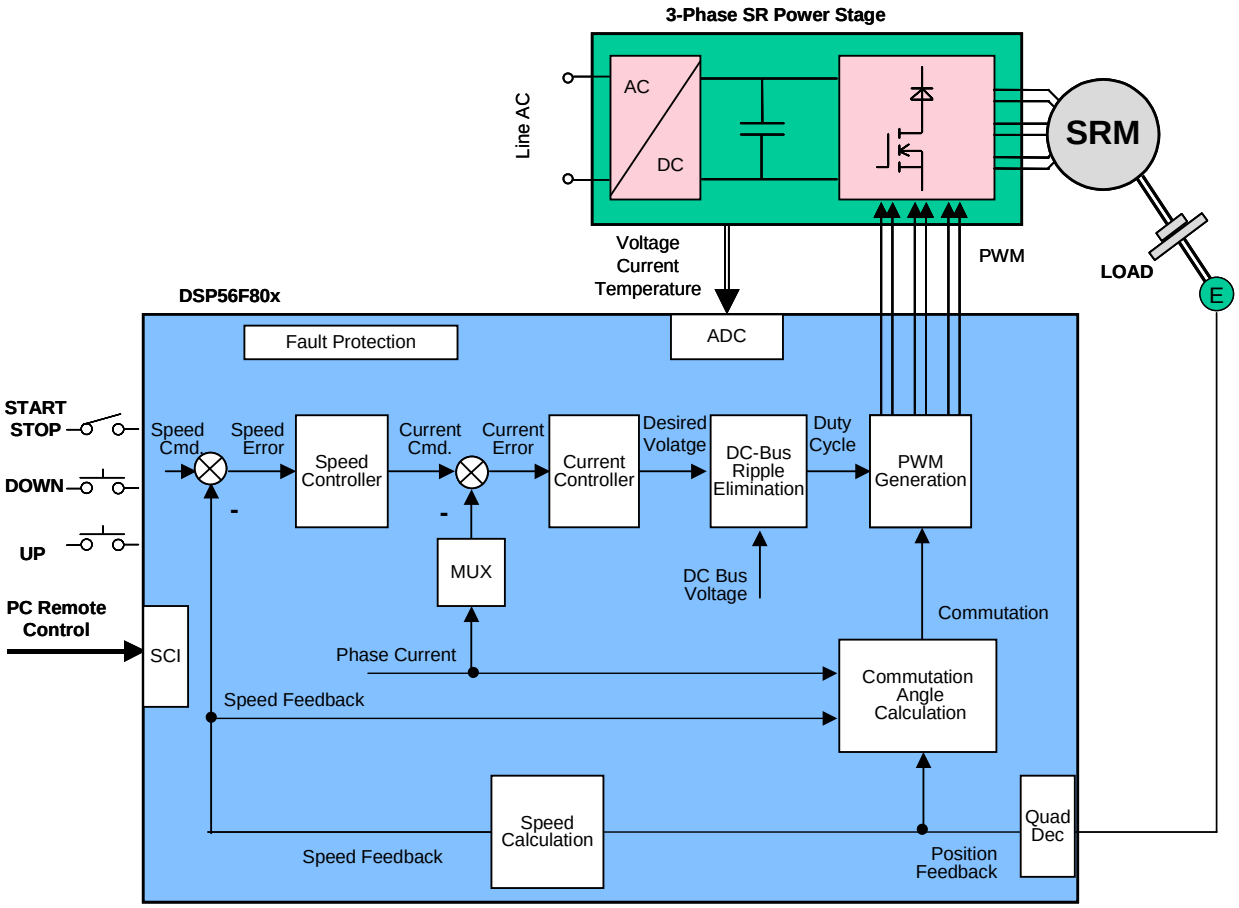

Figure 5-1. System Concept

The DSP runs the main control algorithm. It generates 3-Phase PWM output signals for the SR motor power stage according to the user interface input and feedback signals.

The drive can be controlled in two different ways (or operational modes):

- In Manual operational mode, the required speed is set by a Start/Stop switch and Up and Down push buttons.
- In PC master software operational mode, the required speed is set by the PC master software

After RESET, the drive is initialized and it automatically enters MANUAL operational mode. Note, PC master software can only take over control when the motor is stopped. When the Start command is detected (using the Start/Stop switch or the PC master software button “Start”) and while no fault is pending, the start-up sequence with the rotor alignment is performed and the motor is started.

Rotor position is evaluated using an encoder position sensor. The commutation angle is calculated according to the desired speed, the desired current and the actual DC-Bus voltage. When the actual position of the rotor is equal to the reference position, the commutation of the phases in the desired direction of rotation is done; the actual phase is turned off and the following phase is turned on.

The actual motor speed is derived from the position information, so an additional velocity sensor is unneeded. The reference speed is calculated according to the control signals (start/stop switch, up/down push buttons) and PC master software commands (when controlled by the PC master software). The acceleration/deceleration ramp is implemented. The comparison between the reference speed and the measured speed gives a speed error. Based on the speed error, the speed controller generates the desired phase current. When the phase is commutated, it is turned on with a duty cycle of 100%. Then, during each PWM cycle, the actual phase current is compared with the desired current. As soon as the actual current exceeds the desired current, the current controller is turned on. The current controller controls the output duty cycle until the phase is turned off (following commutation). Finally, the 3-Phase PWM control signals are generated. The procedure is repeated for each commutation cycle of the motor.

DC-Bus voltage, DC-Bus current, and power stage temperature are measured during the control process. The measurements are used for DC-Bus over-voltage, DC-Bus under-voltage, DC-Bus over-current and over-temperature protection of the drive. DC-Bus under-voltage and over-temperature protection are performed by software, while DC-Bus over-current and the DC-Bus over-voltage fault signals utilize the Fault inputs of the DSP on-chip PWM module. The line voltage is measured during initialization of the application. According to the detected level, the 115VAC or 230VAC mains are recognized. If the line voltage is detected outside -15% ... +10% of the nominal voltage, the fault "Out of the Mains Limit" disables drive operation. If any of the above mentioned faults occur, the motor control PWM outputs are disabled in order to protect the drive. The fault status can only be exited when the fault conditions have disappeared and the Start/Stop switch is moved to the STOP position. The fault state is indicated by the on-board LED.

The SR power stage uses a unique configuration of power devices, different than AC or BLDC configuration. SR software would cause the destruction of AC or BLDC power stages due to the simultaneous switching of the power devices. Since the application software could be accidentally loaded into an AC or BLDC drive, the software incorporates a protection feature to prevent this. Each power stage contains a simple module which generates a logic signal sequence that is unique for that type of power stage. During the initialization of the chip, this sequence is read and evaluated according to the decoding table. If the correct SR power stage is not identified, the fault, "Wrong Power Stage", disables drive operation.

5.2.1 Initialization and Start-Up

Before the motor can be started, rotor alignment and initialization of the control algorithms must be performed (see **Figure 5-2**) since the absolute position is not known.

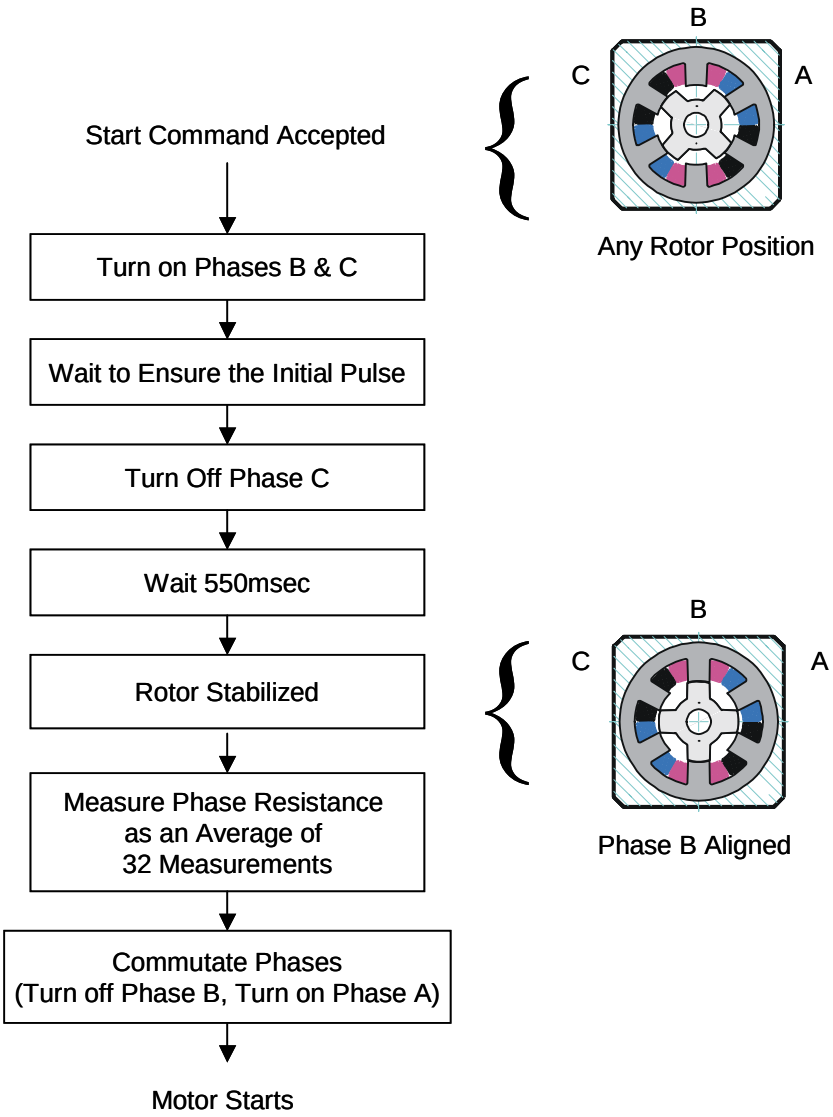

Figure 5-2. Start-Up Sequence

First, the rotor needs to be aligned to a known position to be able to start the motor in the desired direction of rotation. This is done in the following steps:

1. Two phases are turned on simultaneously (Phases B & C)
2. After 50msec one phase is turned off (Phase C), the other phase stays powered (Phase B)
3. After an additional 550 msec, the rotor is stabilized enough in the aligned position with respect to the powered phase (Phase B).

Step 1 provides the initial impulse to the rotor. If Phase B is exactly in an unaligned position and thus does not generate any torque, Phase C provides the initial movement. Then, Phase C is disconnected and Phase B stays powered (Step 2). The stabilization pulse to Phase B must be long enough to stabilize the rotor in the aligned position with respect to that phase.

In total the stabilization takes 1 sec. After this time, the rotor is stable enough to reliably start the motor in the desired direction of rotation.

5.2.2 Position and Speed Sensing

The position information is used to generate accurate switching instants of the power converter, ensuring drive stability and fast dynamic response. Velocity feedback is derived from the position information, so that an additional velocity sensor is unneeded. All members of the Motorola DSP 56F80x family, except 56F801 have an on-chip quadrature decoder module connected to a quadrature timer. This peripheral is commonly used for position and speed sensing.

The quadrature decoder position counter counts up/down each edge of Phase A and Phase B signals according to their order (see **Figure 5-3**). The Phase A and Phase B inputs of the DSP controller are routed through a switch matrix to a general purpose timer module and quadrature decoder module as well (see **Figure 5-4**). The timer module can use all four available inputs as normal timer input capture channels. This does not preclude the use of the quadrature decoder module. Both timer and decoder take advantage of the digital filter incorporated in the quadrature decoder module.

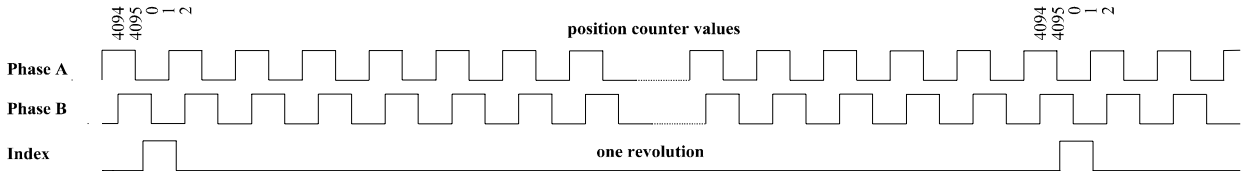

Figure 5-3. Quadrature Encoded Signals

The presented application uses the quad decoder module approach for speed measurement using a 16-bit position difference counter. The counter acts as a differentiator whose count value is proportional to the change in position since the last time the position counter was read. The speed can be computed by calculating the change in the position counter per unit time, or by reading the position difference counter register (POSD) and calculating speed. The second method is employed in this application for rotor speed measurement and also as a feedback signal to the speed controller. The position difference register (POSD) is regularly scanned at the pre-defined time period and consecutively this value is used to compute the actual rotor speed.

In addition, quadrature decoder module 0 shares pins with quadrature timer module A. If the shared pins are not configured as timer outputs, then the pins are available for use as inputs to the quad decoder modules. The quad timer module contains four identical counter/timer groups. Due to the wide variability of quad timer modules, it is possible to use this module to decode quadrature encoder signals and to sense position and speed as well. The presented application uses the configuration arranged for position sensing and commutation instance determination. The quad timer A0 and the quad timer A1 decode the primary and secondary external inputs as quad-encoded signals generated by the rotary sensor to monitor movement of the motor shaft. Quad signal decoding provides both count and direction information. The timer A0 is programmed to count up to a programmed value that corresponds to one electric revolution and then immediately to re-initialize after the terminal count value is reached. This timer A0 is assigned as a master and broadcast compares signals to quad timer A1. The timer A1 is configured to be re-initialized to a predetermined value when a master timer's compare event occurs. This counter continues repeatedly counting past the compare value. When the count matches the compare value, an interrupt is enabled and the compare register 2 value is used for commutation instances generation.

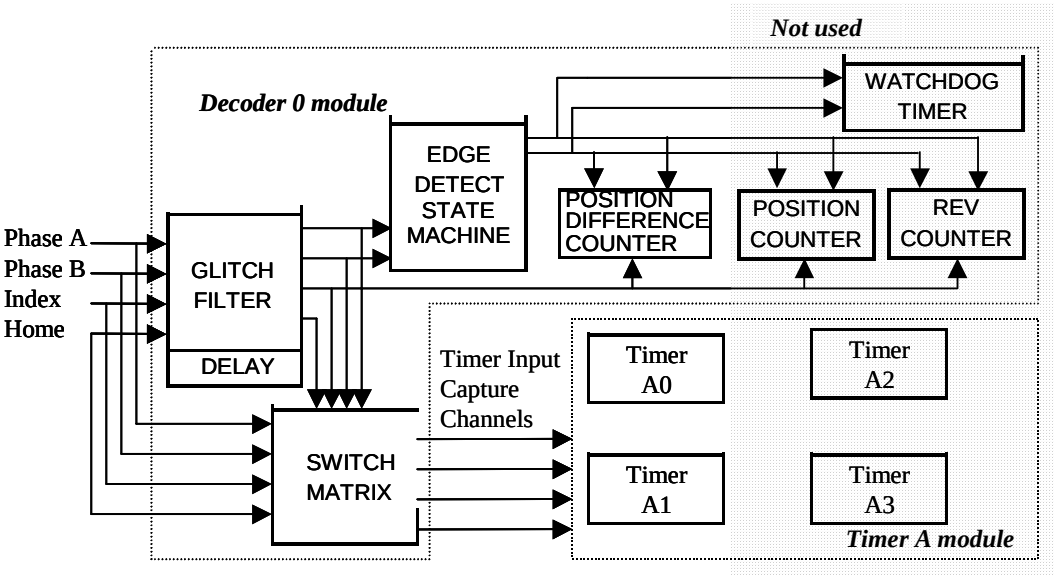

Figure 5-4. Decoder and Timer Arrangement

5.2.3 Commutation Algorithm

The SR motor commutation strategy uses rotor position feedback to drive the commutating signals for the inverter switches. The core of the control algorithm includes the calculation of the commutation angle, and phases commutation. The calculation of the commutation angle is performed according to **(EQ 4-2.)**. It is calculated regularly during motor operation.

The commutation algorithm is described in **Figure 5-5**. After the finish of the start-up routine, which includes the alignment procedure and initialization of the necessary commutation variables, the rotor is sufficiently stabilized and is ready for run mode. This is the point from which the commutation routine has to start. The first procedure of the commutation routine is to turn on the corresponding phase. Choosing the correct phase to switch on depends on the defined rotation of the rotor. The turn on angle is at the unaligned position, and the current rises linearly until the poles begin to overlap.

In a regular switched reluctance motor, the angle of rising inductance is half of the pole-pitch. The pole pitch is the angle of rotation between two successive aligned positions. Ideally, the flux should be zero throughout the period of falling inductance, because current flowing in that period produces a negative (or braking) torque. To avoid this, the dwell angle θ_{dwell} can be restricted. In practice, a dwell angle of 120 electrical degrees is usually used, because the gain in torque-impulse during the increasing inductance exceeds the small braking torque impulse. This condition occurs when the current has a tail extending beyond the aligned position. The torque is negative during this tail period, but it is small. The turn-off angle θ_{off} instant is determined.

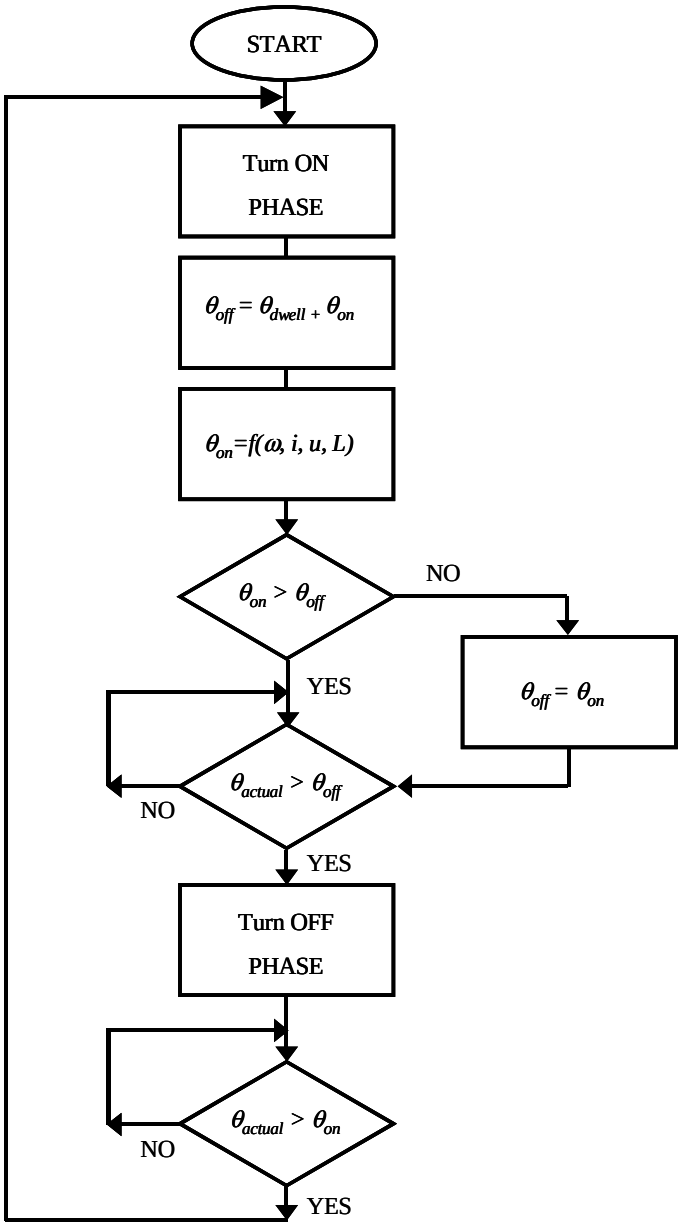

Figure 5-5. Commutation Algorithm Flowchart

The next step of the proposed commutation algorithm is to calculate the advance turn-on angle. The entire calculation explanation is presented in **Section 4.2**. The firing angle θ_{on} is set up for the next commutation instant. The presented commutation algorithm does not allow parallel current conduction of two phases at the same time. The angle comparison of turn-on θ_{on} and turn-off θ_{off} assures that the current phase is turned off before the following phase is turned on. In the case of a 120 electrical degree dwell angle, the switching on and switching off are performed simultaneously. If the conduction (dwell) angle is restricted, the turning off overtakes turning on, as is clear in **Figure 5-5**. The comparison $\theta_{actual} > \theta_{off}$ block waits for an appropriate position to commutate off the corresponding stator phase, and in the next comparison $\theta_{actual} > \theta_{on}$ block the algorithm remains the same until the proper position occurs to switch on the following stator phase. The algorithm loop is closed and ready for other commutation occurrences.

5.2.4 Current Controller Implementation

The current controller utilization flowchart reveals the algorithm process of the controller switching. If the appropriate stator phase is turned on, the DC-Bus voltage is applied to the corresponding rotor phase. The phase current rises almost linearly until a predefined target value is attained. At this point, by the processing of the proposed algorithm the current controller is switched on and maintains the actual current flowing within the desired value.

Before the current controller is switched on, the necessary initialization is required. It is mainly concerned with the integral portion in the k-1 step of the current PI controller. This part of the controller structure is preset according to equation (EQ 4-4.). The following commutation instance turns the controller flag off so the corresponding rotor phase is fully voltage loaded until reaching the desired value of phase current. **Figure 5-6** clarifies the entire controller usage algorithm.

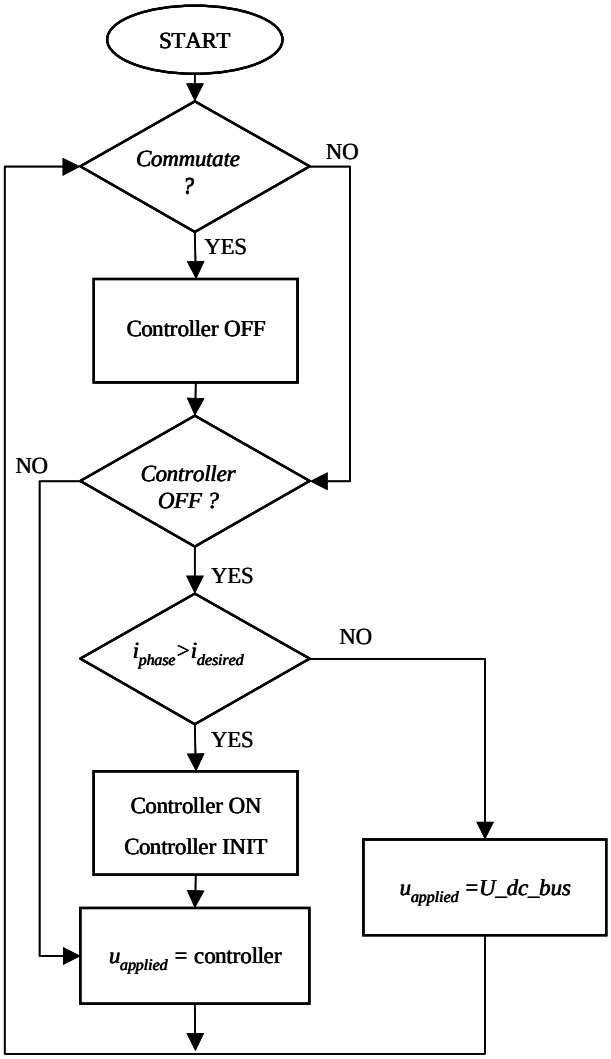

Figure 5-6. Controller Utilization

5.2.5 Current and Voltage Measurement

Precise measurement of phase current and DC-Bus voltage is a key factor for current control implementation.

5.2.5.1 Current Sensing

Current measurement needs to be investigated according to the current sensors used and the influence of the noise on the measurement.

The quality of current measurement depends heavily on the type of current sensors used. The most useful are Hall effect sensors. Unfortunately, these sensors are expensive and thus not suitable for most cost-sensitive applications. Therefore, current shunt resistors inserted into the current path of the phase are often implemented (see **Figure 5-7**). The phase current is sensed as a voltage drop across the sense resistor.

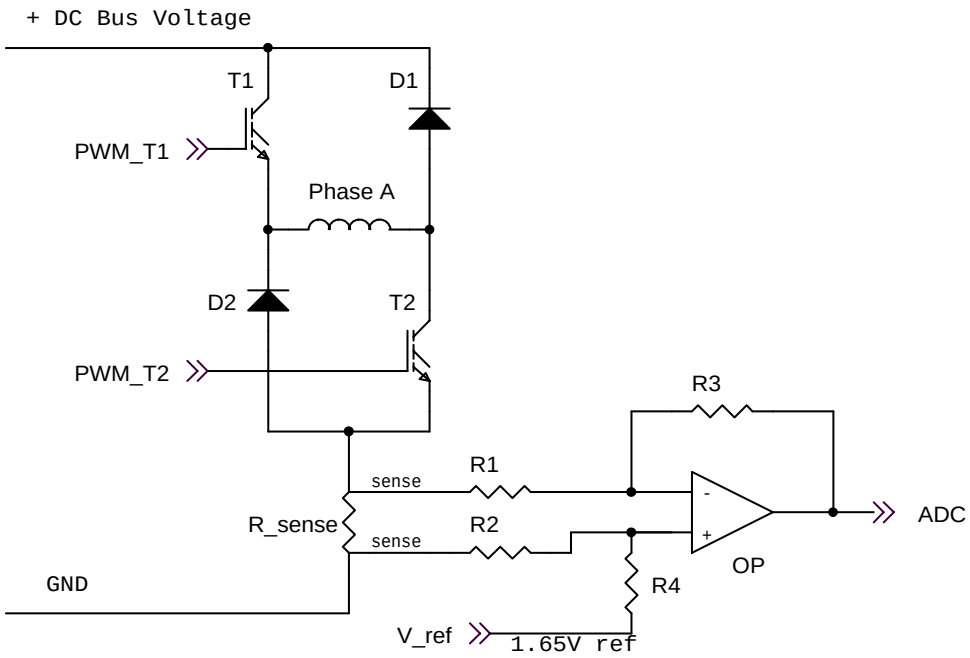

Figure 5-7. Shunt Resistors Current Sensors

When the power switches' soft switching is used (the lower switch is left ON during a complete commutation period, while the upper switch is modulated by the PWM), the current is not visible on the shunt resistor all the time. The soft switching phase current, measured at the shunt resistor, is shown in **Figure 5-8**. The phase current is visible only when both switches are turned on (the phase current flows through switches and the sensing resistor) or when both switches are turned off (phase current flows through the freewheeling diodes and the sensing resistor). When both switches of the phase are turned on, the measured current is negative, so it needs to be inverted. The diagram shows that for a reliable current shape reconstruction, the sensing needs to be synchronized with the PWM frequency at the center of the PWM pulse and both positive and the negative voltage drop polarities should be measured. The zero current may be set to half of the ADC range, so both the positive and the negative voltage drops on the phase current shunt resistors can be measured. The voltage drop is then amplified according to the ADC range. Proceeding like this, the current can be read with accuracy and credibility.

Figure 5-9 illustrates the actual phase currents of a 3-Phase motor, measured on the shunt resistors as described above.

The previously specified current sensing method is described from the DSP processor point of view. It seems the measured phase current is negative, which is caused by inverting differential amplifier. Actually, the measured phase current flowing through shunt resistor is sensed and consecutively inverted by a differential amplifier.

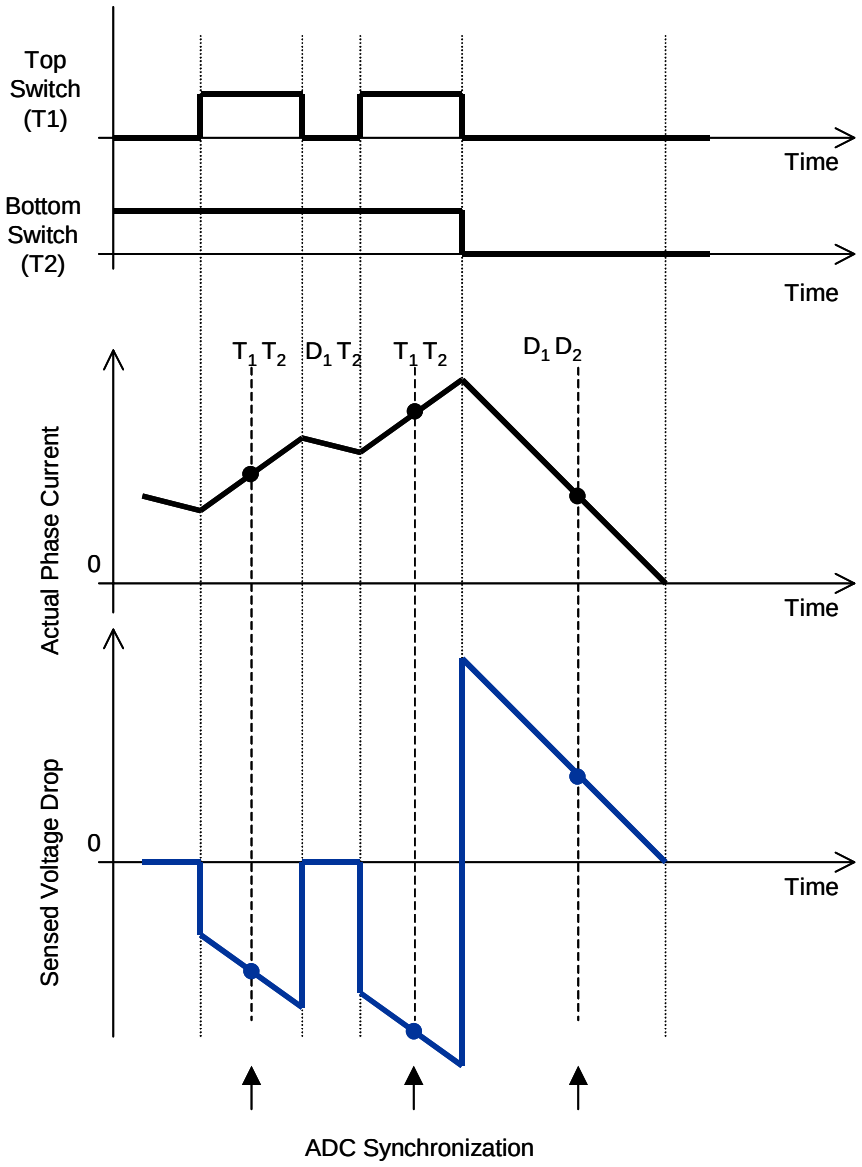

Figure 5-8. Soft Switching Current Sensed on ADC

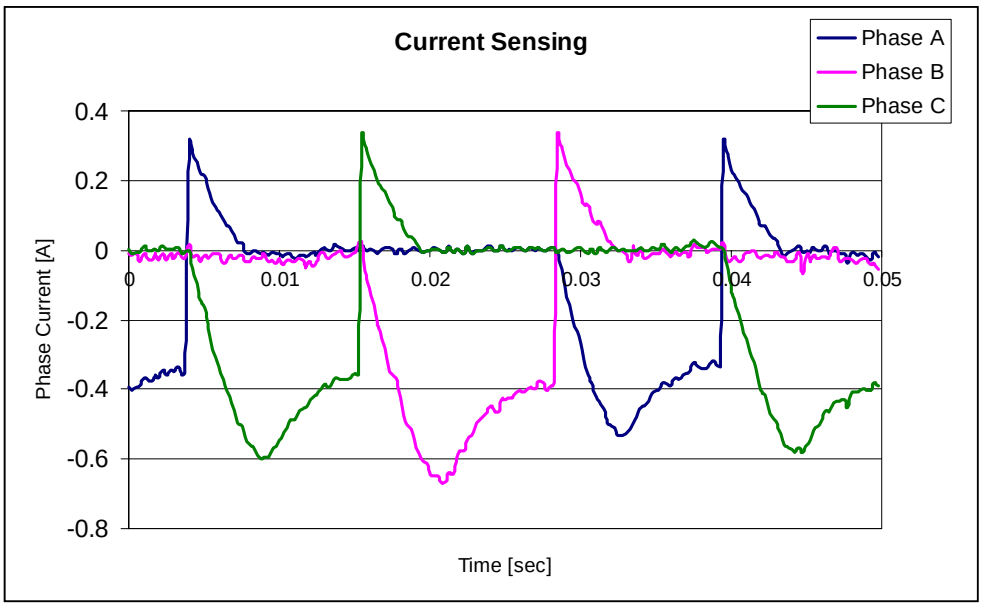

Figure 5-9. Phase Current Measured at Current Shunt Resistors

The low cost shunt resistor sensors create one serious issue. Due to the low-voltage drop sensed across the shunt current resistors, the measured signals are susceptible to noise.

A technique for noise elimination has been developed and successfully implemented. The technique is based on the assumption that the same noise is induced simultaneously on all measured signals. The method supposes the measurement of two signals simultaneously, one known signal (a reference) and one signal to be measured. Then the reference signal consists of a known signal and noise, while the measured signal consists of an actual signal and the same noise.

$$MeasuredSignal = ActualSignal + Noise \quad \text{(EQ 5-1.)}$$

$$ReferenceSignal = KnownSignal + Noise \quad \text{(EQ 5-2.)}$$

If the noise is the same, it can be eliminated by subtraction of the reference signal from the measured signal. As described above, the necessary condition is the simultaneous sampling of both signals, ensuring that the noise on both signals is identical.

$$ActualSignal = MeasuredSignal - ReferenceSignal + KnownSignal \quad \text{(EQ 5-3.)}$$

This technique has been implemented for phase current sensing. The SR motor is controlled in a way in which the phases are commutated sequentially, which means that when the working phase is turned off, the following phase, in the direction of rotation, is turned on. Thus one phase of the motor is never powered during a complete commutation interval. This phase is considered as a reference. Because the reference phase is not powered, the reference phase current should be equal to zero. The measured value of the reference current can be then considered as noise for a given commutation interval. The actual phase current is equal to the difference between the measured current and the reference current:

$$I_{ph} = I_{sensed} - I_{reference} \quad \text{(EQ 5-4.)}$$

The reference signal needs to be commutated together with the commutation of the phases. **Table 5-1.** defines the active, discharge and reference phases for the commutation sequence C - B - A - C. It is derived from **Figure 5-9**.

Table 5-1. Commutation Sequence of the Reference Phase

Step	Active Phase	Discharge Phase	Reference Phase
1	C	A	B
2	B	C	A
3	A	B	C
1	C	A	B

The efficiency of the current sensing noise reduction technique is illustrated in **Figure 5-10**. The figures illustrate the phase current as it is measured (the active phase current is inverted compared to **Figure 5-9**), and the same current with the implemented noise reduction technique. As can be seen, the implemented technique improves current sensing significantly. It eliminates not only the noise on the current sensors, but also the noise induced on the sensing cables and the noise of the ADC reference power supply.

5.2.5.2 Voltage Sensing

The DC-Bus voltage sensor is represented by a simple voltage divider. DC-Bus voltage does not change rapidly. It is nearly constant with the ripple given by the power supply structure. If a bridge rectifier for rectification of AC line voltage is used, the ripple frequency is two times the AC line frequency. The ripple amplitude should not exceed 10% of the nominal DC-Bus value, if the power stage is designed correctly.

The measured DC-Bus voltage needs to be filtered in order to eliminate noise. One of the most useful techniques is at moving average filter, that calculates an average value from the last N samples:

$$u_{DCBus} = \sum_{n=1}^{-N} u_{DCBus}(n) \qquad \text{(EQ 5-1.)}$$

In order to increase the precision of the voltage sensing, the voltage drop on the power switches and on the diodes of the power stage can be incorporated into the determination of the actual voltage present in the motor phase.

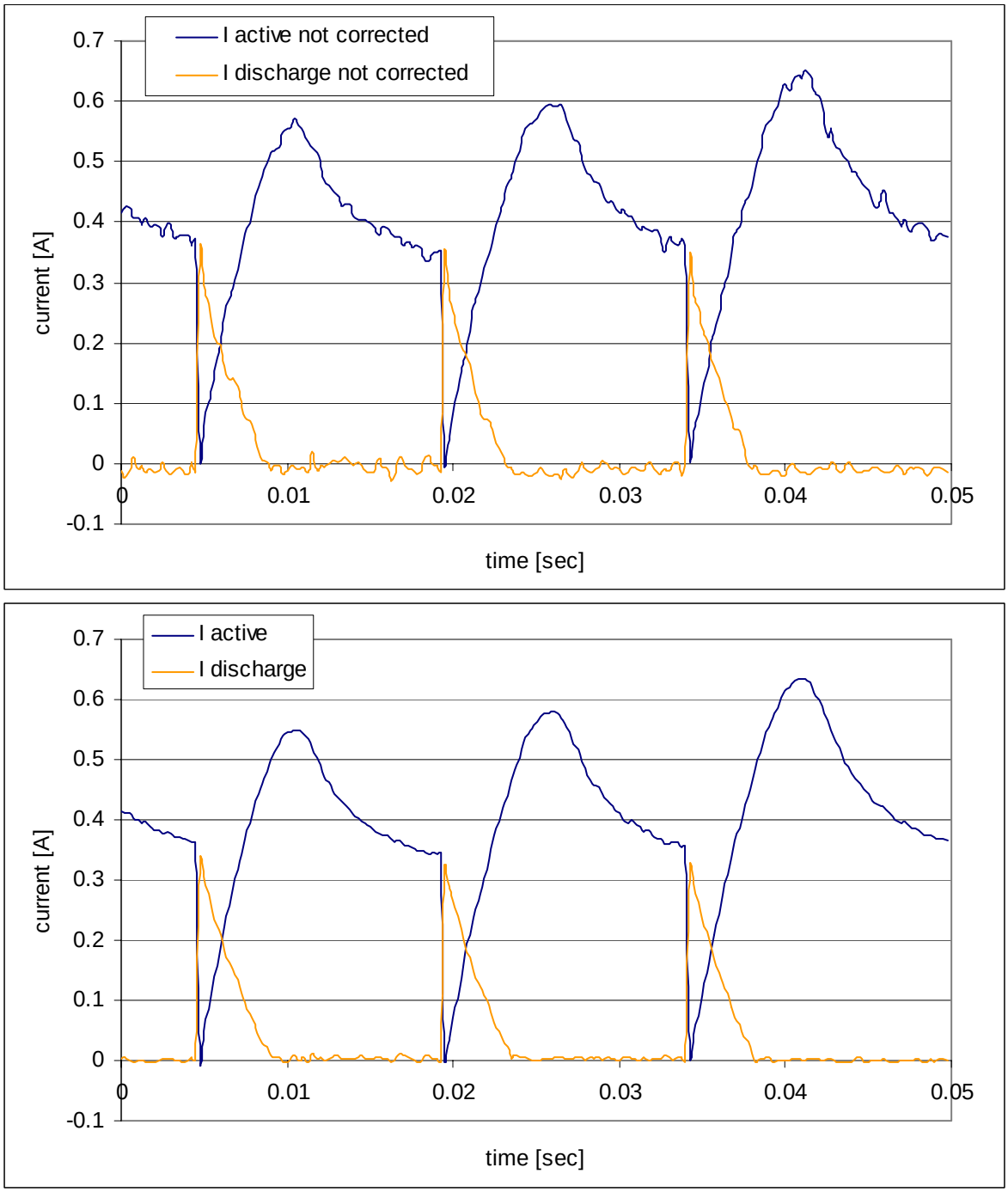

Figure 5-10. Measured 3-Phase Currents without Noise Correction and with Noise Correction Implemented

5.2.6 Power Module Temperature Sensing

The measured power module temperature is used for thermal protection The hardware realization is shown in **Figure 5-11**. The circuit consists of four diodes connected in series, a bias resistor, and a noise suppression capacitor. The four diodes have a combined temperature coefficient of 8.8 mV/°C. The resulting signal, *Temp_sense*, is fed back to an A/D input where software can be used to set safe operating limits. In the presented application, the temperature in degrees Celsius is calculated according to the conversion equation:

$$temp = \frac{Temp\_sense - b}{a} \qquad \text{(EQ 5-2.)}$$

where:

temp	is the power module temperature in degrees Celsius
Temp_sense	is the voltage drop on the diodes which is measured by ADC
a	is the diode-dependent conversion constant (a = -0.0073738)
b	is the diode-dependent conversion constant (b = 2.4596)

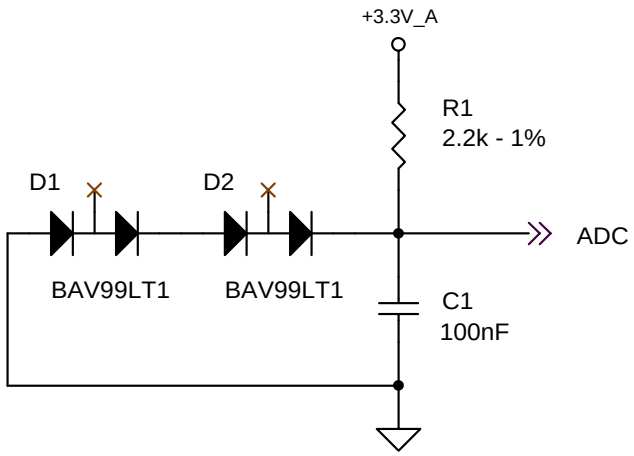

Figure 5-11. Temperature Sensing Topology

6. Hardware Implementation

6.1 Hardware Setup

As already stated, the application runs on Motorola motor control DSPs using the DSP EVM boards and a dedicated 3-Phase SR high-voltage platform.

The application can be controlled by the following Motorola motor control DSPs:

- DSP56F803
- DSP56F805
- DSP56F807

Application hardware setup is shown in **Figure 6-1**. The system hardware setup for a particular DSP varies only by EVM Board used. The application software is identical for all DSPs. The EVM and chip differences are handled by the SDK off-chip drivers for the particular DSP EVM board.

Detailed application HW setup can be found in the document **Targeting_DSP5680x_Platform** that is part of the SDK documentation **[12]**.

Dedicated user's manuals describe the individual boards in detail. The User's Manuals incorporate a schematic of the board, description of individual function blocks and a bill of materials for the board. The individual boards can be ordered from Motorola as standard products. Descriptions of all the mentioned boards and documents can be found at: *http://www.motorola.com/*

All system parts are supplied and documented according to the following references:

- U1 - Controller Board for DSP56F803:
  - — supplied as: DSP56803EVM
  - — described in: **DSP56F803EVMUM/D DSP Evaluation Module Hardware User's Manual -** see **[9]**

  or U1 - Controller Board for DSP56F805:
  - — supplied as: DSP56805EVM
  - — described in: **DSP56F805EVMUM/D DSP Evaluation Module Hardware User's Manual -** see **[10]**

  or U1 - Controller Board for DSP56F807:
  - — supplied as: DSP56807EVM described in: **DSP56F807EVMUM/D DSP Evaluation Module Hardware User's Manual -** see **[11]**
- U2 - 3-Phase SR High-Voltage Power Stage
  - — supplied as a kit with an Optoisolation Board as: ECOPTHIVSR
  - — described in: **MEMC3PSRHVPSUM/D Motorola Embedded Motion Control 3-Phase SR High-Voltage Power Stage User's Manual -** see **[14]**
- U3 - Optoisolation Board (not supposed to be in the final application)
  - — supplied in 3-ph. SR High Voltage Power Stage as: ECOPTHIVSR
  - — or supplied separately as: ECOPT - optoisolation board
  - — described in: **MEMCOBUM/D Optoisolation Board User's Manual** - see **[13]**
- MB1 Motor-Brake SR40V + SG40N
  - — supplied as: ECMTRHIVSR

Warning: The use of optoisolation (optocouplers and optoisolation amplifiers) is strongly recommended during development to avoid electric shock or any damage to the development equipment.

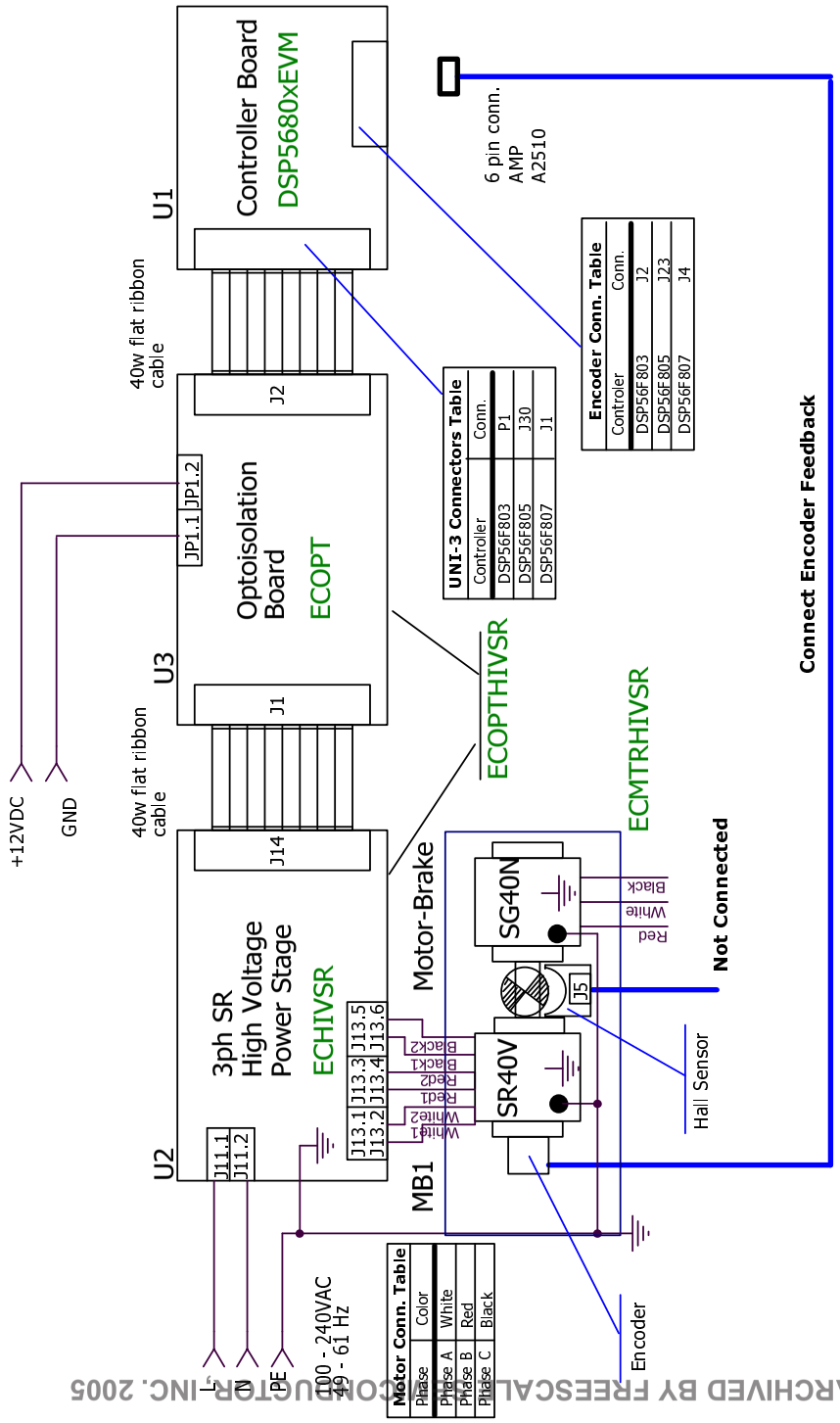

Figure 6-1. 3-Phase SR High Voltage Platform Configuration

6.2 Motor-Brake Specifications

The SR Motor Brake set incorporates a 3-Phase SR Motor and attached BLDC motor brake. The detailed specifications are listed in **Table 6-1.**

The SR motor has six stator poles and four rotor poles. This combination yields 12 strokes (or pulses) per single mechanical revolution. The SR motor is characterized by a dedicated inductance profile. The motor inductance profile as a function of mechanical position is shown in **Figure 6-2**. The mechanical angle 90^{o}_{mech} corresponds to one electrical period of the stroke. The presented profile was used for the determination of the advanced commutation angle.

On the motor brake shaft, a position encoder and position Hall sensor are attached. They allow position sensing if it is required by the control algorithm. The introduced drive uses the Encoder for the position determination

Table 6-1. Motor - Brake Specifications

Set Manufacturer	EM Brno, Czech Republic	
Motor Specification:	Motor Type:	SR40V (3-Phase SR Motor)
	Stator / Rotor Poles:	6/4
	Speed Range:	< 5000 rpm
	Nominal Voltage:	3 x 300V
	Nominal Current:	1.2A
Brake Specification:	Brake Type	SG40N 3-Phase BLDC Motor
	Nominal Voltage:	3 x 27V
	Nominal Current:	2.6 A
Position Encoder	Type	Baumer Electric BHK 16.05A 1024-12-5
	Pulses per Revolution	1024

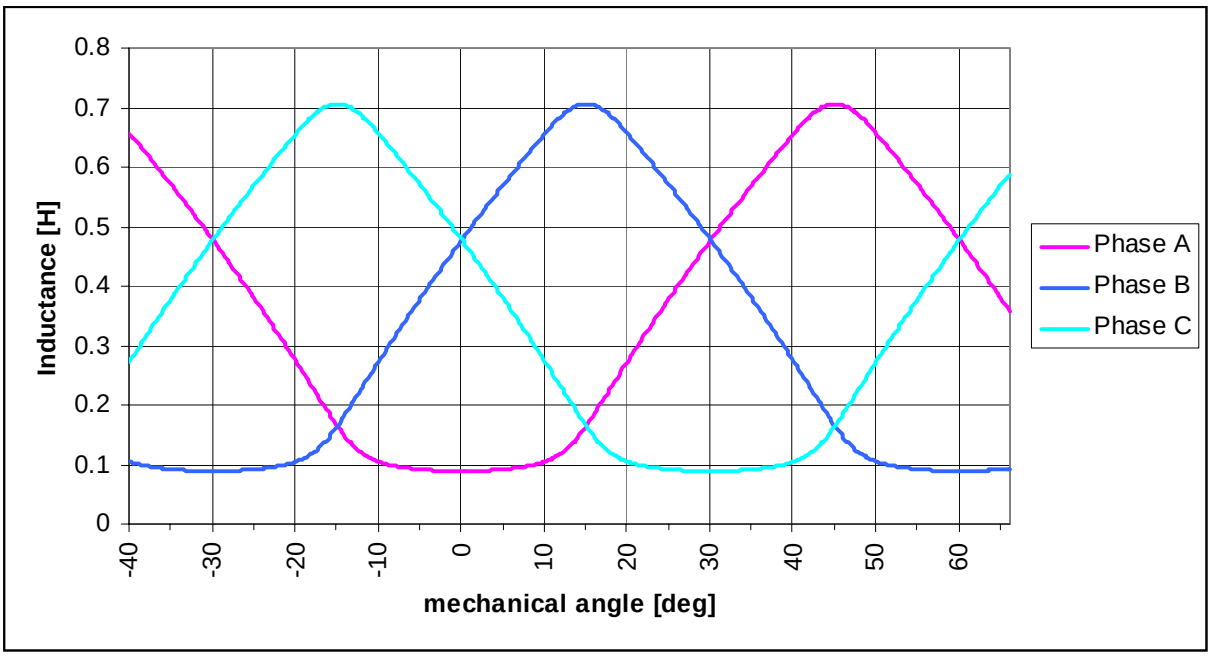

Figure 6-2. Inductance Characteristic

7. Software Design

This section describes the design of the software blocks of the drive. The software will be described in terms of:

- Control algorithm data flow
- State diagram
- Software implementation

7.1 Data Flow

The control algorithm of a closed loop SR drive is described in **Figure 7-1** and **Figure 7-2**. It is based on the system description.

The individual processes are described in detail in the following sections.

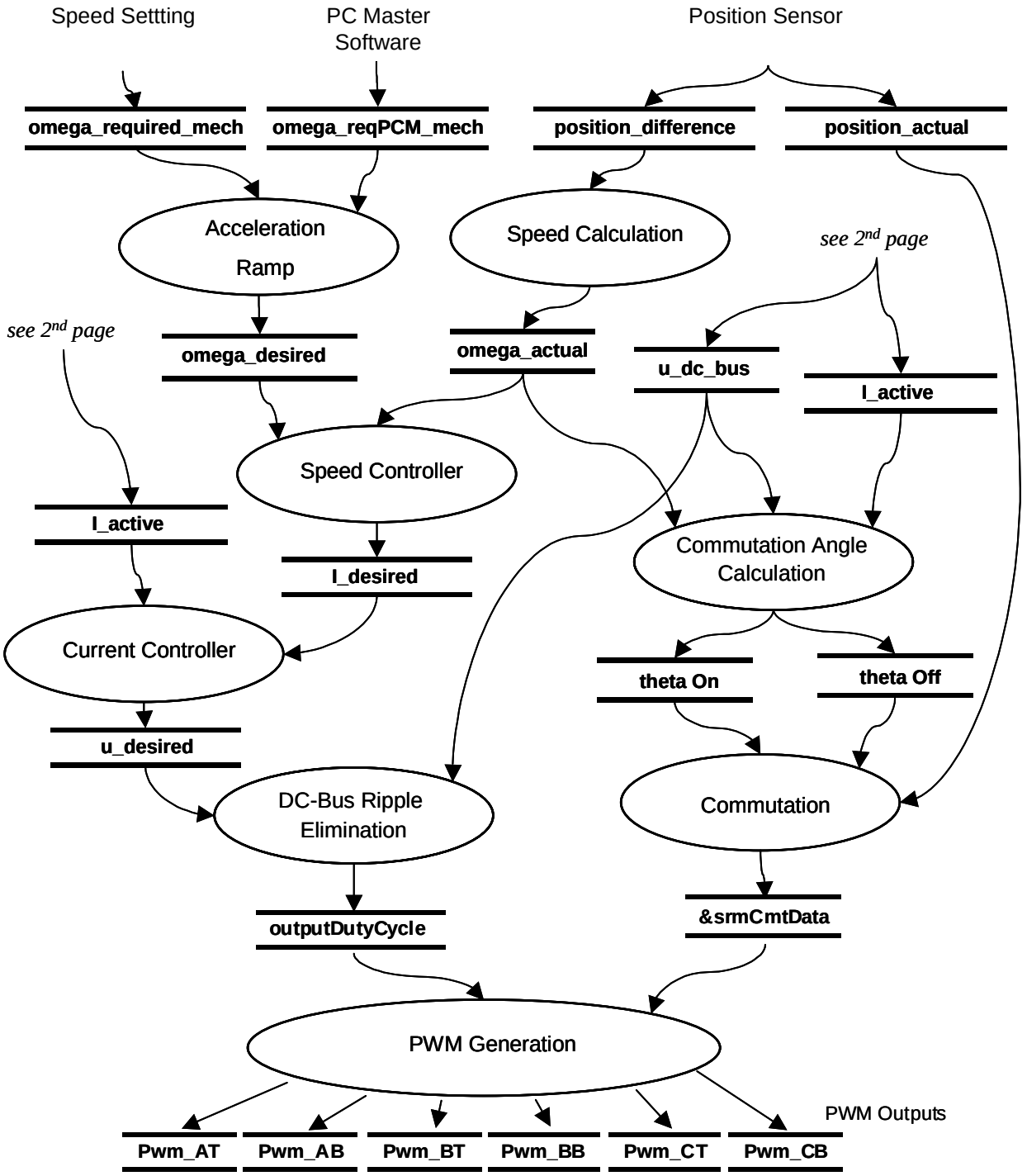

Figure 7-1. System Data Flow I - SR Motor Control

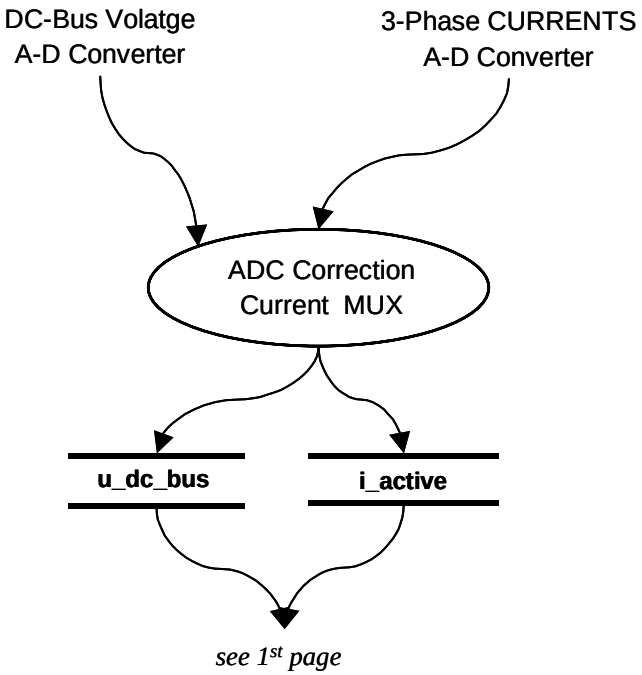

Figure 7-2. System Data Flow II - AD Converter

7.1.1 Acceleration Ramp

This process calculates the desired speed based on the required speed according to the acceleration / deceleration ramp. The required speed is set either manually, using the push buttons (when in manual operational mode), or by PC master software (when in PC master software operational mode).

7.1.2 Speed Calculation

The process calculates the actual speed of the motor. The calculation is based on the evaluation of the position information. The on-chip Quadrature Decoder provides information on position difference through a 16-bit counter. When the position register is read, the position difference of the counter's contents are copied into the position difference hold register (POSDH) and position difference counter is cleared.

The register is regularly read and the captured value is used for speed calculation. The speed is computed by reading the position difference counter register per pre-defined time sample.

A software moving average filter applied to the speed measurement is incorporated into the process for greater noise immunity. The actual motor speed is calculated as the average value of the last four measurements.

7.1.3 Speed Controller

This process calculates the desired phase current according to the speed error. Speed error is the difference between the actual speed and desired speed. A Proportional-Integrational (PI) type of controller is implemented. The constants of the speed controller are tuned experimentally according to the load profile and the speed limits.

7.1.4 Current Controller

This process calculates the duty cycle of the PWM based on phase current error. Phase current error is the difference between the actual phase current and desired phase current. A PI type of controller is implemented. The current controller constants are tuned experimentally according to the type of used motor used.

7.1.5 DC-Bus Ripple Elimination

This process provides the elimination of the voltage ripple on the DC-Bus. It compensates an amplitude of the desired phase voltage generated by the PI current controller. The output of the calculation is the duty cycle of the PWM that is applied to corresponding stator phase.

7.1.6 PWM Generation

This process sets the on-chip PWM module for generation of the control pulses for the 3-Phase SR motor power stage. Generation of these pulses is based on the software control register that is formulated by the process of the Commutation Calculation and is based on the required duty cycle generated by the Speed Controller process. The calculated software control word is loaded into the proper PWM register and the PWM duty cycle is updated according to the required duty cycle. The PWM Generation process is accessed regularly at a rate given by the PWM frequency. It is frequent enough to ensure the precise generation of commutation pulses.

7.1.7 ADC Correction and Current MUX

This process takes care of the Analog-to-Digital Converter. The sampling of the ADC is synchronized to the PWM pulses. The process selects the proper ADC channels to be converted and reads and processes the results of the ADC conversion.

The active and discharge phase currents are selected and corrected using the measured reference noise signal. The DC-Bus voltage and temperature are filtered using a moving average filter. See **Section 5.2.5 Current and Voltage Measurement** for a detailed description.

7.1.8 Commutation Angle Calculation

This process calls the commutation angle calculation routine which calculates the advanced angle according to the actual speed, the DC-Bus voltage and the desired current (see **Section 4.2**).

The algorithm **3-Phase SR Motor Commutation Angle Calculation** `srmcac` generates the required advance angle of commutation according to the principle described in **Section 4.2**. Before the calculation routine call, the scaling constant must be properly determined.

```
/* scaling constant */
scale_const = FRAC16((L_UN*I_MAX*OMEGA_MAX*4)/(U_MAX*60));
```

The following functions of the algorithm need to be called in order to calculate the commutation angle:

```
/* routine call */
adv_angle = srmcacAngleCalc(i_ph,u_ph,w_actual,scale_const);
/* u_ph      => voltage across phase winding */
/* i_ph      => phase current    */
/* w_actual  => actual speed */
```

These functions are called in the Process Commutation. A detailed description of the algorithm can be found in the SDK algorithm documentation.

7.1.9 Commutation

This process provides the comutation of the motor phases.

The DSP on-chip PWM module is used in a mode for generation of independent output signals that can be controlled either by software or by the PWM module.

The commutation technique distinguishes the three following cases:

- When the PWM output needs to be modulated, the PWM generator controls the channel directly
- When the PWM output needs to be switched to an inactive state (0), the software output control of the corresponding PWM channel is handed over and the channel is turned off manually
- When the PWM output needs to be switched to the active state (1), the software output control of the corresponding PWM channel is handed over and the channel is turned on manually

The on-chip PWM module enables control of the outputs from the PWM module either by the PWM generator, or by using the software. Setting the output control enable bit, OUTCTLx, enables software to drive the PWM outputs instead of the PWM generator. In independent mode, with OUTCTLx = 1, the output bit OUTx controls the PWMx channel. Setting or clearing the OUTx bit activates or deactivates the PWMx output. The OUTCTLx and OUTx bits are in the PWM output control register.

This control technique requires the preparation of the output control register. For the calculation of the OUTCTLx and OUTx bits in the PWM output control register, a dedicated commutation algorithm, **3-Phase SR Motor Commutation Handler for H/W Configuration 2-Switches-per-Phase**, `srmcmt3ph2spp,` was developed. The algorithm generates an output control word according to the desired action and the desired direction of rotation. For example, when Phase A needs to be turned off, the algorithm sets the corresponding OUTCTLx bits to enable the output control of the required PWMs and clears the OUTx bits to turn off the PWMs. The other output control register bits are not affected. A detailed description of the algorithm can be found in the SDK motor control library.

7.2 State Diagram

The processes described above are implemented in a single state machine, as illustrated in **Figure 7-3**. The state machine provides a transition amongst the application states INIT, STOP, RUN, FAULT. The following variables are used to invoke the transition between the individual states:

- switchState (Stop, Run): state of the Start/Stop switch
- appFault (NO_FAULT, any fault): fault occurrence
- appOpMode (change from Manual to PC and vice versa): change operational mode

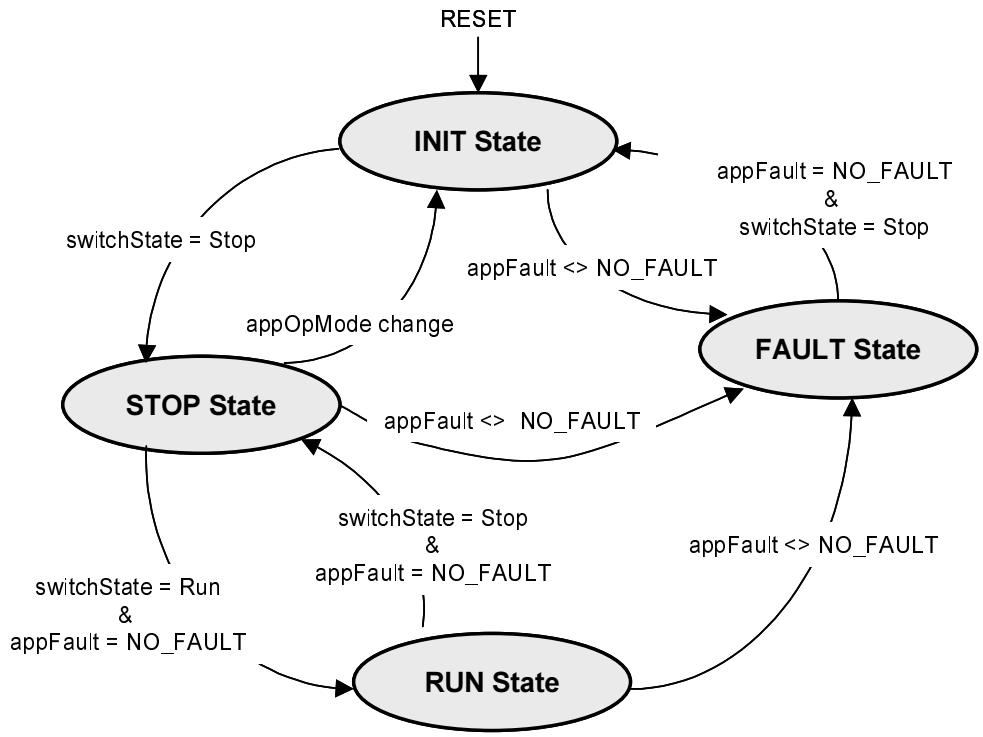

Figure 7-3. Application State Diagram

7.2.1 Application State - INIT

After RESET the application enters the INIT state. In this state, the drive is disabled and the motor cannot be started.

If any fault is detected, the application transits to the FAULT state (protection against faults). If no fault is present, and the Start/Stop switch is detected in the STOP position, the application transits to the STOP state (protection against start after reset if the Start/Stop switch is accidentally in START position).

7.2.2 Application State - STOP

The STOP state can be entered either from the INIT state or from the RUN state. In the STOP state, the drive is enabled and the application waits for the START command.

When the application is in the STOP state, the operational mode can be changed, either from MANUAL mode to PC master software mode or vice versa. When the operational mode is changed, the application always transits to the INIT state.

If any fault in the STOP state is detected, the application enters the FAULT state (fault protection). If no fault is present and the start command is accepted, the application transits to the RUN state and the motor is started.

7.2.3 Application State - RUN

The RUN state can be entered from the STOP state. In the RUN state the drive is enabled and the motor is running.

If any fault in the RUN state is detected, the application enters the FAULT state (fault protection). If no fault is present and the STOP command is accepted the application transits to the STOP state and the motor is stopped.

7.2.4 Application State - FAULT

The STOP state can be entered from any state. In the FAULT state, the drive is disabled and the application waits for the faults to be cleared.

When it is detected that the fault has been eliminated, and the fault clear command is accepted (the Start/Stop switch is moved to the STOP position), then the application transits to the INIT state.

7.3 Software Design

The general software diagram incorporates: (1) the Main routine entered from Reset, and (2) the Interrupt Service Routines (ISR). The diagram is illustrated in **Figure 7-4**.

After Reset, the Main routine provides board identification, initialization of the DSP, initialization of the application, and then it enters an infinite background loop. The background loop contains Fault Detection, Application State Machine, and a scheduler routine.

The scheduler routine provides the timing sequence for two tasks called Timeout 1 and Timeout 2. The Timeout 1 and Timeout 2 flags are periodically set to predetermined intervals by the ADC Conversion Completed ISR. The scheduler utilizes these flags and calls the required routines:

- The routine in Timeout 1 provides a user interface, calculates the required speed, the start-up routines and the speed ramp (acceleration/deceleration).
- The routine in Timeout 2 calculates the Speed Controller.

The Timeout 1 and Timeout 2 tasks are performed in the run state, instead of interrupt routines, in order to reduce time and avoid software bottlenecks.

The following interrupt service routines are utilized:

- ADC Conversion Completed ISR - services ADC and provides all the control tasks linked to the event; the ADC is synchronized with the PWM pulses.
- Fault ISR - services faults invoked by external hardware faults.
- SCI ISR - services PC master software communication.
- Push Button Up ISR - services the Up Push Button.
- Push Button Down ISR - services the Down Push Button.
- Timer A1 Compare ISR - services Commutation Callback.

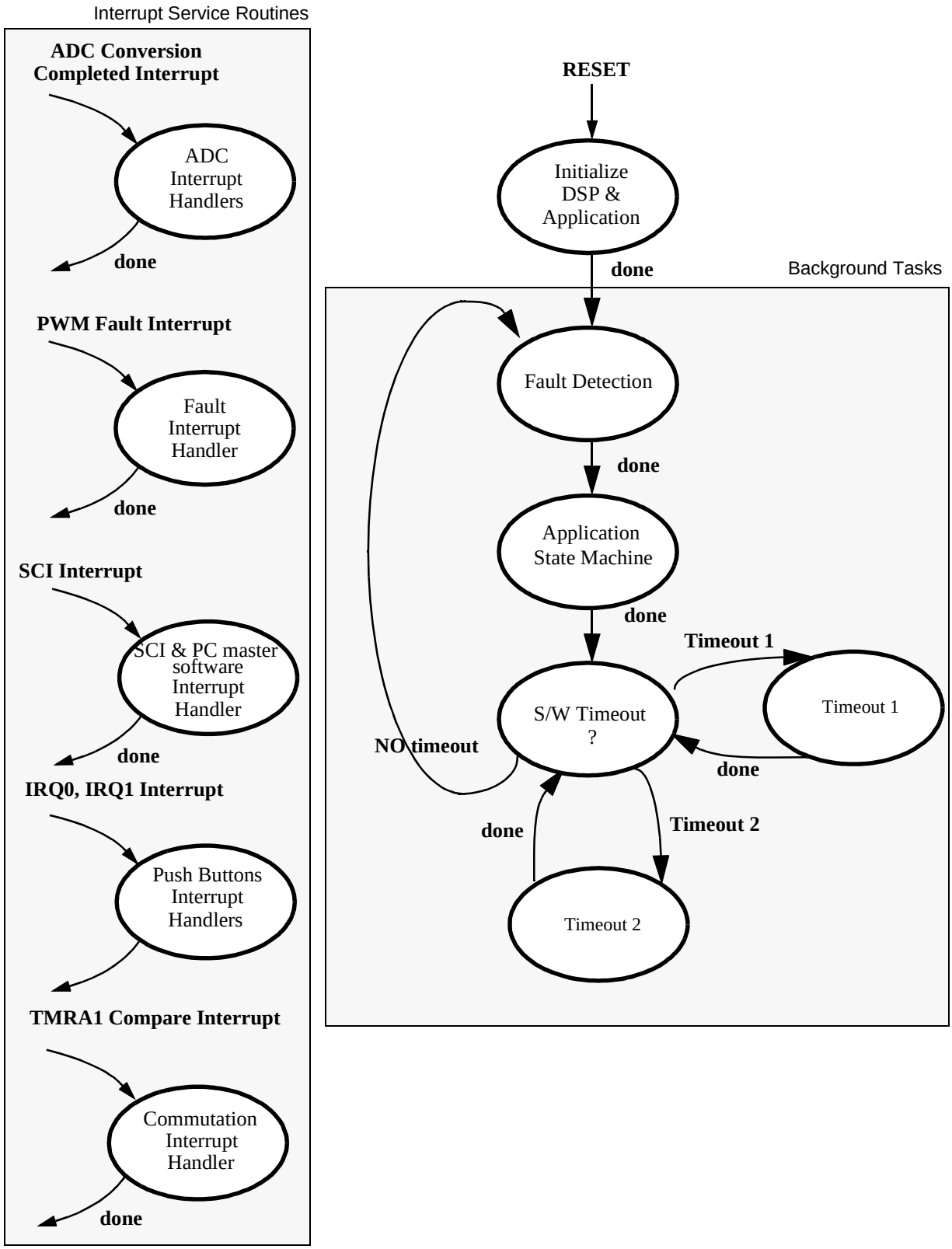

Figure 7-4. Software Design - General Overview

7.3.1 Initialization

After Reset, the initialization of the DSP is performed. At the beginning of the initialization, interrupts are disabled; at the end of initialization they are enabled.

DSP Initialization:

- Disable Interrupts
- Identify power stage board
  - identify SR High-Voltage H/W set
- Initialize ADC on-chip module
  - ADC triggered simultaneously
  - associate interrupt with ADC conversion completed event
  - 1st sample of ADC_A: Current Phase A
  - 2nd sample of ADC_A: DC-Bus Voltage
  - 3rd sample of ADC_A: Temperature
  - 1st sample of ADC_B: Current Phase B
  - 2nd sample of ADC_B: Current Phase C
  - 3rd sample of ADC_B: void
- Initialize Quadrature Timer A0 on-chip module (position measurement)
  - set Quad count mode
  - count repeatedly up to 1024
- Initialize Quadrature Timer A1 on-chip module (commutation callback)
  - set Quad count mode
  - count repeatedly, the binary roll over
- Initialize Quadrature Decoder on-chip module
  - sets digital filter for input signals
  - connects Quadrature Decoder signals to the Quadrature TimerA1
- Initialize PWM on-chip module:
  - center aligned independent PWM mode, positive polarity
  - set PWM modulus for PWM frequency 16kHz
  - set PWM interrupt reload each PWM pulse
  - set FAULT2 (DC-Bus over-current fault) in manual mode, interrupt enabled
  - set FAULT1 (DC-Bus over-voltage fault) in manual mode, interrupt enabled
  - associate interrupt with PWM Fault events
- Initialize brake driver
- Initialize LED driver
- Initialize push buttons
  - push buttons on interrupts IRQ0, IRQ1
- Initialize switch driver
  - switch driver used for DSP56F805EVM and DSP56F807EVM

Application initialization:

- Set individual parameters of the application to their initial values
- Start ADC conversion
- Measure offset of individual current sensors
- Measure DC-Bus voltage and temperature
- Calculate application parameters according to DC-Bus voltage
- Initialize Quadrature Timer C2 Driver (ADC-PWM Synchronization)
  - set ADC synchronization delay to 0
  - enable Quadrature Timer C2 to be started on first SYNC
- Initialize ADC Driver
  - set ADC synchronization ON
  - enable 8-sample conversion
- Initialize all variables for motor start-up
- Set ADC according to start-up phase
- Enable interrupts

7.3.2 Fault Detection

The Fault Detection routine checks application faults. If a fault occurs, it disables the PWM outputs and sets the application FAULT status. Note that in the case of DC-Bus over-current and DC-Bus over-voltage faults, PWM outputs are disabled directly via internal PWM module fault protection (see **Section 7.3.7 Fault ISR**).

7.3.3 Application State Machine

The Application State Machine provides transition between the individual states of the application: INIT, STOP, RUN, and FAULT. For reference, see **Section 7.2 State Diagram**.

7.3.4 Scheduler Timeout 1

This routine is accessed from the main scheduler at a period of Timeout 1 (10 msec). The following tasks are then performed:

- Push button filter - debounces push button switching noise
- Start/Stop switch filter - debounces Start/Stop switch noise
- According to the operational mode, desired speed is calculated
  - In manual mode according to the push buttons
  - in PC master software control mode, according to the PC master software command
- Start-up routine is performed if required and start-up switching pattern is generated. For a detailed description refer to **Section 5.2.1 Initialization and Start-Up**.
- Speed command is calculated using the acceleration / deceleration ramp using the desired speed setup
- LED is controlled according to the state of the drive. It can indicate a STOP state, RUN state or FAULT state.

7.3.5 Scheduler Timeout 2

This state is accessible from the main scheduler in period of Timeout 2 (2.5 msec). The following tasks are then performed:

- Speed controller calculates the desired phase current according to the actual and the desired speed. The speed controller constants are determined experimentally and set during the initialization of the chip.

7.3.6 ADC Conversion Completed ISR

The ADC Conversion Completed ISR is the most critical and the routine most demanding of the processor's time. Most of the application control processes need to be linked with this ISR.

The Analog-to-Digital converter is initiated synchronously with a PWM reload pulse (center of the PWM pulse). It scans all three phase currents, the DC-Bus voltage and the temperature at once. When the conversion is finalized, the ADC Completed ISR is called.

The routine provides the following services and calculations:

- Reads the ADC conversion results (phase currents, noise, DC-Bus voltage, temperature)
- Calculates the ADC offsets for phase currents
- Current controller calculates the desired phase voltage according to the desired and the actual phase current
- Provides commutation when required
- Records selected recorder variables (PC master software)
- Loads PWM registers
- Calculates the references for software timers Timer1 and Timer2
- Enables the next ADC synchronization trigger

7.3.7 Fault ISR

The PWM Fault ISR is the highest priority interrupt implemented in the software. In the case of a DC-Bus over-current or a DC-Bus over-voltage fault detection, the external hardware circuit generates a fault signal, that is detected on the Fault input pin of the DSP. The signal disables the motor control PWM outputs in order to protect the power stage and generates a Fault interrupt, where the fault condition is handled. The routine records the corresponding fault source to the fault status register.

7.3.8 SCI ISR

This interrupt handler provides SCI communication and PC master software service routines. These routines are fully independent of the motor control tasks.

7.3.9 Push Button UP/Down ISR

The Push Button Interrupt Handlers take care of the push buttons service. The Up Button Interrupt Handler sets the Up Button flag, the Down Button Interrupt Handler sets the Down Button flag. The desired speed is incremented/decremented according to the debounced Up/Down button flag.

7.3.10 TMRA1 Compare ISR

The compare interrupt handler takes care of commutation call. This callback routine sets-on the `commutate` flag to indicate that the commutation is required. The commutation flag is regularly checked in the ADC conversion-completed routine and upon a successful compare, the commutation routine is called to perform commutation itself.

8. Implementation Notes

8.1 Scaling of Quantities

The SR motor control application uses a fractional representation for all real quantities except time. The N-bit signed fractional format is represented using 1.[N-1] format (1 sign bit, N-1 fractional bits). Signed fractional numbers (SF) lie in the following range:

$$-1.0 \le SF \le +1.0 - 2^{-[N-1]} \quad \text{(EQ 8-1.)}$$

For words and long-word signed fractions, the most negative number that can be represented is -1.0, whose internal representation is $8000 and $80000000, respectively. The most positive word is $7FFF or $1.0 - 2^{-15}$, and the most positive long-word is $7FFFFFFF or $1.0 - 2^{-31}$.

The following equation shows the relationship between the real and the fractional representations:

$$\text{Fractional Value} = \frac{\text{Real Value}}{\text{Real quantity range}} \quad \text{(EQ 8-2.)}$$

where:

Fractional Value is the fractional representation of the real value [Frac16]

Real Value is the real value of the quantity [V, A, rpm, etc.]

Real quantity range is the maximal range of the quantity, defined in the application [V, A, rpm, etc.]

8.1.1 Voltage Scaling

The application voltages are scaled to the maximal measured voltage. For DC-Bus voltage the scaling equation is the following:

$$u\_dc\_bus = \frac{V_{DC\_BUS}}{V_{MAX}} \quad \text{(EQ 8-3.)}$$

Where:

u_dc_bus is the scaled variable of the DC-Bus voltage [Frac16]

V_{DC_BUS} is the measured DC-Bus voltage [V]

V_{MAX} is the maximal measurable DC-Bus voltage [V]

In the application, V_{MAX} = 407V for the high voltage platform.

The other application voltage variables are scaled in the same way (active phase voltage `u_active`, discharge phase voltage `u_discharge`, DC-Bus under-voltage limit, start-up voltage).

8.1.2 Phase Current Scaling

The application phase currents are scaled to the maximal measured phase current. For the active phase current the scaling equation is the following:

$$i\_active = \frac{i_{active}}{i_{phase\_max}} \quad \text{(EQ 8-4.)}$$

Where:

i_active is the scaled variable of the active phase current [Frac16]

i_{active} is the measured active phase current [A]

i_{phase_max} is the maximal measurable phase current [A]

In the application, i_{phase_max} = 5.86A for the high-voltage platform.

The other application phase current variables are scaled in the same way (desired current `i_desired`, discharge current `i_discharge`, current offsets `i_phase_A_offset`, `i_phase_B_offset`, `i_phase_C_offset`).

8.1.3 Electrical Angle Scaling

The application electrical angle is scaled to the electrical angle in the aligned position (see **Figure 8-1**). For the electrical commutation angle the scaling equation is the following:

$$theta\_on\_el = \frac{\vartheta_{on\_el}}{\vartheta_{aligned\_el}} \quad \text{(EQ 8-5.)}$$

Where:

theta_on_el is the scaled variable of the electrical commutation angle [Frac16]

ϑ_{on_el} is the desired commutation angle [$^{o}_{el}$]

$\vartheta_{aligned\_el}$ is the electrical angle in aligned position [$^{o}_{el}$].

In the application, $\vartheta_{aligned\_el} = 360^{o}_{el}$

The other application electrical angle variables are scaled in the same way (angle where stator and rotor poles start to overlap `theta_edge`).

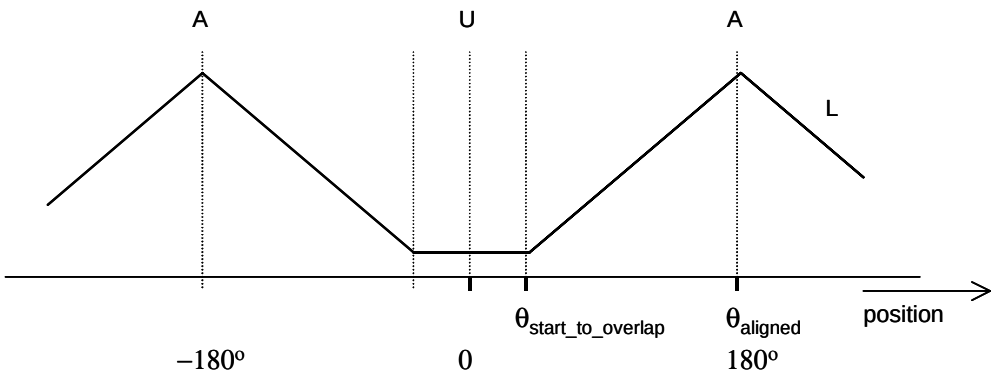

Figure 8-1. Electrical Angle Definition

8.1.4 Speed Scaling

Speed is scaled to the maximal speed of the drive. For the desired start-up speed, the scaling equation is the following:

$$omega\_desired\_startup = \frac{\omega_{start\_up}}{\omega_{MAX}} \quad \text{(EQ 8-6.)}$$

Where:

omega_desired_startup is the scaled variable of the desired start-up speed [Frac16]

$\omega_{start\text{-}up}$ is the desired start-up speed [rpm]

ω_{MAX} = maximal speed of the drive [rpm]

In the application, ω_{MAX} = 3000 rpm.

The other application speed variables are scaled in the same way (actual speed, `omega_actual_mech`, speed limits, `omega_reqMAX_mech & omega_reqMIN_mech`, push button speed increment, `omega_increment_pb`).

8.1.5 Duty Cycle Scaling

The duty cycle is scaled to the maximal duty cycle of the drive. For the output duty cycle the scaling equation is the following:

$$output\_duty\_cycle = \frac{duty\_cycle_{output}}{duty\_cycle_{MAX}} \quad \text{(EQ 8-7.)}$$

Where:

output_duty_cycle is the scaled variable of output duty cycle [Frac16]

$duty\_cucle_{output}$ is the desired output duty cycle [%]

$duty\_cycle_{MAX}$ is the max. applicable duty cycle [%]

In the application, $duty\_cycle_{MAX}$ = 100 %

The other application duty cycles are scaled in the same way (high and low duty cycle limits for speed controller, start up output duty cycle `outputDutyCycleStartup`).

8.2 Velocity Calculation

The actual speed of the motor is calculated from the time, *TimeCaptured*, captured by the on-chip Quadrature Timer between the two following edges of the position Hall sensors. The actual speed, *OmegaActual* is calculated according to the following equation:

$$OmegaActual = \frac{SpeedCalcConst}{TimeCaptured} \quad \text{(EQ 8-8.)}$$

where:

OmegaActual is the actual speed [rpm]

TimeCaptured is the time, in terms of number of timer pulses, captured between two edges of the position sensor [-]

SpeedCalcConst is a constant defining the relationship between the actual speed and number of captured pulses between the two edges of the position sensor

The constant *SpeedCalcConst* is calculated as:

$$SpeedCalcConst = 2^{15} \times \frac{SpeedMin}{SpeedMax} \quad \text{(EQ 8-9.)}$$

where:

SpeedMin is the minimal measured speed [rpm]

SpeedMax is the maximal measured speed [rpm]

Minimal measured speed, *SpeedMin*, is given by the configuration of the sensors and parameters of the DSP on-chip timer used for speed measurement. It is calculated as:

$$SpeedMin = \frac{\frac{1}{NoPulsesPerRev} \times 60}{\frac{2^{15}}{BusClockFreq} \times Presc} \quad \text{(EQ 8-10.)}$$

where:

NoPulsesPerRev is the number of sensed pulses of the position sensor per single revolution [-]

Presc is the prescaler of the Quadrature Timer used for speed measurements

BusClockFreq is the DSP Bus Clock Frequency [Hz]

Maximal measured speed, *SpeedMax*, is selected as:

$$SpeedMax = k \times SpeedMin \quad \text{(EQ 8-11.)}$$

where:

k is an integer constant greater than 1

Then the speed calculation constant is determined as:

$$SpeedCalcConst = BusClockFreq \times \frac{60}{NoPulsesPerRev \times Presc \times SpeedMax} \quad \text{(EQ 8-12.)}$$

In the application:

NoPulsesPerRev = 12 Hall sensor pulses per 1 revolution of the motor

Presc = 128

BusClockFreq = $36*10^6$ Hz

SpeedMax = 3000 rpm

Then, *SpeedCalcConst* = 468 [rev^{-1}]

9. SDK Implementation

The Motorola Embedded SDK is a collection of APIs, libraries, services, rules and guidelines. This software infrastructure is designed to let DSP5680x software developers create high-level, efficient and portable code. The application code is available in SDK. This chapter describes how the SR motor control application is written under SDK.

9.1 Drivers and Library Function

The SR motor control application uses the following drivers:

- ADC driver
- Quadrature Timer driver
- Quadrature Decoder driver
- PWM driver
- LED driver
- SCI driver
- PC master software driver
- Switch driver (only for DSP56F805EVM &DSP56F807EVM)
- Brake driver

All drivers are included in the *bsp.lib* library.

The SR motor control application uses the following library functions:

- srmcmt3ph2spp (SR motor commutation algorithm; *srm.lib* library)
- srmcacAngleCalc (SR motor commutation angle calculation algorithm; *srm.lib* library)
- controller (standard PI controller, *mcfunc.lib* library)
- switchcontrol (switch control, *mcfunc.lib* library)
- boardId (board identification, *bsp.lib* library)

9.2 Appconfig.h File

The purpose of the *appconfig.h* file is to provide a mechanism for overwriting default configuration settings which are defined in the *config.h* file.

There are two *appconfig.h* files. The first *appconfig.h* file is dedicated to External RAM (..*ConfigExtRam* directory) and the second one is dedicated to FLASH memory (..*ConfigFlash* directory). In the case of a SR motor control application, both files are identical with the following exceptions:

- The *appconfig.h* for ExtRAM target contains a PC master software recorder buffer of 25000 samples long, while *appconfig.h* for Flash target contains a PC master software recorder buffer of only 100 samples long. This is due to the limited RAM memory size.
- The *appconfig.h* for DSP56F805EVM and DSP56F807EVM contains the definition of a switch driver, while the *appconfig.h* for DSP56F803EVM does not.

The *appconfig.h* file can be divided into two sections. The first section defines which components of the SDK libraries are included in the application, the second part overwrites the standard settings of components during their initialization.

9.3 Initialization of Drivers

Each peripheral on the DSP chip or on the EVM board is accessible through a driver. The driver initialization of all used peripherals is described in this chapter. For a detailed description of the drivers see the document, **Embedded SDK Targeting Motorola DSP5680x Platform.**

The following steps are required to use a driver:

- Include the driver support in the *appconfig.h*
- Fill the configuration structure in the application code for the specific drivers (depends on driver type)
- Initialize the configuration setting in *appconfig.h* for the specific drivers (depends on driver type)
- Call the *open* (create) function

Access to individual driver functions is provided by the *ioctl* function call.

9.4 Interrupts

The SDK serves the interrupt routine calls and automatically clears interrupt flags. The user defines the callback functions called during interrupts. The callback functions are assigned during driver initialization - `open()`. Callback function assignment is defined as one item of the initialization structure, which is used as a parameter of the `open()` function. Some drivers define the callback function in the *appconfig.h* file.

9.5 PC Master Software

PC master software was designed to provide application debugging, diagnostic, and demonstration tools for development of algorithms and applications. It runs on a PC connected with the DSP EVM via an RS232 serial cable. A small program resident in the DSP communicates with PC master software to parse commands, return status information to the PC and process control information from the PC. PC master software, executing on a PC, uses part of Microsoft Internet Explorer as a user interface to the PC.

The PC master software is part of the Motorola Embedded SDK and may be selectively installed during SDK installation.

To enable PC master software operation on the DSP target board application, the following lines must be added to the *appconfig.h* file:

```
#define INCLUDE_SCI          /* SCI support */
#define INCLUDE_PCMASTER     /* PC master software support */
```

It automatically includes the SCI driver and installs all necessary services.

The default baud rate of the SCI communication is 9600Bd. It is set automatically by the PC master software driver and can be changed if needed.

A detailed description of PC master software is provided by the dedicated User's Manual **[17]**.

The 3-Phase SR Motor Control with Encoder utilizes PC master software for remote control from a PC. It enables the user to:

- Take over control of the PC master software
- Start/stop control
- Set the motor speed

Variables read by the PC master software and displayed to the user are:

- Required and actual motor speeds
- Application operational mode
- Start/stop status
- Drive fault status
- DC-Bus voltage
- Identified power stage boards
- Identified voltage level
- System status

Profiles of required and actual speeds together with the desired phase current can be seen in the Speed Scope window.

The courses of quickly changing variables, like the phase current profiles can be observed in the Recorder windows. The Recorder can **ONLY** be used when the application is running from **External RAM** due to the limited on-chip memory. The length of the recorded window may be set in "Recorder Properties" => bookmark "Main" => "Recorded Samples". The dedicated memory space is defined in the *appconfig.h* file of the ExtRAM target. Recorder samples are taken each 64.5 µsec, at the rate of the PWM frequency.

Current Controller Recorder captures:

- Desired phase current
- Active phase current
- Desired phase voltage

Current Controller Recorder may be initiated any time during the motor run.

10. DSP Usage

Table 10-1. shows how much memory is needed to run advanced 3-Phase SR drive with Encoder drive. The recorder buffer of PC master software was set to zero. A majority of the DSP memory is still available for other tasks.

Table 10-1. RAM and FLASH Memory Usage for SDK2.4 and CW4.1

Memory (in 16 bit Words)	**Available DSP56F803 DSP56F805**	**Available DSP56F807**	**Used Application + Stack**	**Used Application without PC master software, SCI**
Program FLASH	32K	60K	11590	7508
Data FLASH	4k	8K	338	310
Program RAM	512	2K	42	18
Data RAM	2K	4K	701 + 352 stack	381 + 352 stack

11. References

The following materials were used to produce this paper:

[1] Chalupa, L., *Pohon se spinanym reluktancnim motorem*, Master's Thesis, FEI-VUT BRNO, UPVE, 1994

[2] Gallegos-Lopez, G., A New Sensorless Low-Cost Method for Switched Reluctance Motor Drives, *University of Glasgow - SPEED Laboratory*, 1997

[3] Miller, T.J.E., *Switched Reluctance Motors and Their Control*, Magna Physics Publishing and Clarendon Press, ISBN 0-19-859387-2, 1993

[4] 3-Phase SR Motor Control with Hall Sensors using DSP56F80x, *AN1912/D*, Motorola, 2000

[5] 3-Phase SR Sensorless Motor Control using DSP56F80x, *AN1932/D*, Motorola, 2001

[6] CodeWarrior for Motorola DSP56800 Embedded Systems, CWDSP56800, Metrowerks, 2001

[7] DSP56F800 16-bit Digital Signal Processor, Family Manual, DSP56F800FM/D, Motorola, 2001

[8] DSP56F80x 16-bit Digital Signal Processor, User's Manual, DSP56F801-7UM/D, Motorola, 2001

[9] DSP56F803 Evaluation Module Hardware User's Manual, DSP56F803EVMUM/D, Motorola, 2001

[10] DSP56F805 Evaluation Module Hardware User's Manual, DSP56F805EVMUM/D, Motorola, 2001

[11] DSP56F807 Evaluation Module Hardware User's Manual, DSP56F807EVMUM/D, Motorola, 2001

[12] Embedded Software Development Kit for 56800/56800E, MSW3SDK000AA, available on Motorola SPS web page, Motorola, 2001

[13] Motorola Embedded Motion Optoisolation Board User's Manual, MEMCOBUM/D, Motorola, 2000

[14] Motorola Embedded Motion Control 3-Phase Switched Reluctance High-Voltage Power Stage User's Manual, MEMC3PSRHVPSUM/D, Motorola, 2000

[15] Motorola Embedded Motion Control 3-Phase Switched Reluctance Low-Voltage Power Stage User's Manual, MEMC3PSRLVPSUM/D, Motorola, 2000

[16] Motorola SPS web page: *http://e-www.motorola.com/*

[17] User's Manual for PC master software, included in the SDK documentation, Motorola, 2001

Freescale Semiconductor, Inc.
ARCHIVED BY FREESCALE SEMICONDUCTOR, INC. 2005

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 1–303–675–2140 or 1–800–441–2447

JAPAN: Motorola Japan Ltd.; SPS, Technical Information Center, 3–20–1, Minami–Azabu. Minato–ku, Tokyo 106–8573 Japan. 81–3–3440–3569

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong. 852–26668334

Technical Information Center: 1–800–521–6274

HOME PAGE: http://www.motorola.com/semiconductors/

MOTOROLA

AN1937/D