

3-Phase Switched Reluctance (SR) Sensorless Motor Control Using a 56F80x, 56F8100 or 56F8300 Device

Design of a Motor Control Application

Radim Visinka

Note: The PC master software referenced in this document is also known as Free Master software.

1. Introduction

This Application Note describes the design of a sensorless 3-Phase Switched Reluctance (SR) motor drive. It is based on Freescale's 56F80x / 56F8300 dedicated motor control devices. The software design takes advantage of Processor Expert™ (PE).

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. 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 for numerous applications.

This application involves a sensorless speed closed-loop SR drive with an inner current loop using flux linkage position estimation. The change in phase resistance during motor operation due to its temperature dependency creates errors in the position estimation and significantly affects the performance of the drive. Therefore, a novel algorithm for on-the-fly estimation of phase resistance is included. This application demonstrates the sensorless SR motor drive and serves as an example of a system design using a

Contents

1. Introduction	1
2. Advantages and Features of Freescale's Hybrid Controller.....	2
2.1 56F805, 56800 Core Family.....	2
2.2 56F8346, 56800E Core Family	3
2.3 Peripheral Description.....	4
3. Target Motor Theory	5
3.1 Switched Reluctance Motor	5
3.2 Mathematical Description of an SR Motor	7
3.3 Digital Control of an SR Motor.....	10
3.4 Voltage and Current Control of SR Motors.....	12
4. Techniques for Sensorless Control of SR Motors	16
4.1 Sensorless Position Estimation using Flux Linkage Estimation.....	16
4.2 Flux Linkage Calculation in a Discrete Time Domain	18
4.3 Sensorless On-the-Fly Resistance Estimation	19
5. System Design	21
5.1 System Outline	21
5.2 Application Description	24
6. Hardware Implementation	36
6.1 Hardware Setup	36
6.2 Motor-Brake Specifications	38
7. Software Design	40
7.1 Data Flow	40
7.2 State Diagram.....	45
7.3 Software Design	46
8. Implementation Notes	51
8.1 Scaling of Quantities	51
8.2 Velocity Calculation.....	55
9. Processor Expert (PE) Implementation	57
9.1 Beans and Library Functions.....	57
9.2 Initialization of Beans	57
9.3 Interrupts	57
9.4 PC Master Software	58
10. Hybrid Controller Use	61
11. References	61

Freescale hybrid controller with PE support. It also illustrates the use of dedicated motor control libraries included in PE. The application helps start the development of the sensorless SR drive dedicated to the targeted application.

This application note includes a description of the Freescale hybrid controller's features, basic SR motor theory, system design concept, hardware implementation, and software design including the use of the PC master software visualization tool.

2. Advantages and Features of Freescale's Hybrid Controller

The Freescale 56F80x (56800 core) and 56F8300 (56800E core) families are ideal for digital motor control, combining a DSP's computational ability with an MCU's controller features on a single chip. These hybrid controllers offer many dedicated peripherals, including 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 appropriate for Switched Reluctance motor control.

The following sections use a specific device to describe the family's features.

2.1 56F805, 56800 Core Family

The 56F805 provides the following peripheral blocks:

- Two Pulse Width Modulator modules (PWMA and PWMB), each with six PWM outputs, three Current Sense inputs, and four Fault inputs; fault-tolerant design with dead time 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 the PWM
- Two Quadrature Decoders (Quad Dec0 and Quad Dec1), each with four inputs, or two additional Quad Timers A & B
- Two dedicated general purpose Quad Timers, totaling six pins: Timer C with two pins and Timer D with four pins
- CAN 2.0 B-compatible unit with 2-pin ports used to transmit and receive
- Two Serial Communication Interfaces (SCI0 and SCI1), each with two pins, or four additional GPIO lines
- A Serial Peripheral Interface (SPI), with a 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
- An external reset pin for hardware reset
- JTAG / On-Chip Emulation (OnCE)
- A software-programmable, Phase Lock Loop-based frequency synthesizer for the hybrid controller core clock

Table 2-1 Memory Configuration for 56F805 Devices

	56F801	56F803	56F805	56F807
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

2.2 56F8346, 56800E Core Family

The 56F8346 provides the following peripheral blocks:

- Two Pulse Width Modulator modules (PWMA and PWMB), each with six PWM outputs, three Current Sense inputs, and three Fault inputs for PWMA/PWMB; fault-tolerant design with dead time insertion, supporting both center-aligned and edge-aligned modes
- Two 12-bit Analog-to-Digital Converters (ADCs), supporting two simultaneous conversions with dual 4-pin multiplexed inputs; the ADC can be synchronized by PWM modules
- Two Quadrature Decoders (Quad Dec0 and Quad Dec1), each with four inputs, or two additional Quad Timers, A and B
- Two dedicated general purpose Quad Timers, totaling 3 pins: Timer C with one pin and Timer D with two pins
- CAN 2.0 B-compatible unit with 2-pin ports used to transmit and receive
- Two Serial Communication Interfaces (SCI0 and 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
- 61 multiplexed General Purpose I/O (GPIO) pins
- External reset pin for hardware reset
- JTAG / On-Chip Emulation (OnCE)
- Software-programmable, Phase Lock Loop-based frequency synthesizer for the hybrid controller core clock
- Temperature Sensor system

Table 2-2 Memory Configuration for 56F8300 Devices

	56F8322	56F8323	56F8345	56F8346	56F8347
Program Flash	16K x 16-bit	16K x 16-bit	64K x 16-bit	64K x 16-bit	64 x 16-bit
Data Flash	4K x 16-bit	4K x 16-bit	4K x 16-bit	4K x 16-bit	4K x 16-bit
Program RAM	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit
Data RAM	4K x 16-bit	4K x 16-bit	4K x 16-bit	4K x 16-bit	2K x 16-bit
Boot Flash	4K x 16-bit	4K x 16-bit	4K x 16-bit	4K x 16-bit	4K x 16-bit

	56F8355	56F8356	56F8357	56F8365	56F8366	56F8367
Program Flash	128K x 16-bit	128K x 16-bit	128K x 16-bit	256K x 16-bit	128K x 16-bit	128K x 16-bit
Data Flash	4K x 16-bit	4K x 16-bit	4K x 16-bit	16K x 16-bit	4K x 16-bit	4K x 16-bit
Program RAM	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit	2K x 16-bit
Data RAM	8K x 16-bit	8K x 16-bit	8K x 16-bit	16K x 16-bit	4K x 16-bit	8K x 16-bit
Boot Flash	4K x 16-bit	8K x 16-bit	8K x 16-bit	16K x 16-bit	8K x 16-bit	8K x 16-bit

2.3 Peripheral Description

The most interesting peripherals for switched reluctance motor control are the fast Analog-to-Digital Converter (ADC) and the Pulse Width Modulation (PWM) on-chip modules. They offer freedom of configuration, enabling efficient sensorless 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
- Dead time insertion
- Separate top and bottom pulse width correction via current status inputs or software
- Separate top and bottom polarity control
- Edge- or center-aligned PWM signals
- 15 bits of resolution
- Integral reload rates from one to 16 with a half-cycle reload capability
- 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 the 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 of 5MHz with a 200ns period
- Single conversion time of 8.5 ADC clock cycles ($8.5 \times 200\text{ns} = 1.7\mu\text{s}$)
- Additional conversion time of 6 ADC clock cycles ($6 \times 200\text{ns} = 1.2\mu\text{s}$)
- Eight conversions in 26.5 ADC clock cycles ($26.5 \times 200\text{ns} = 5.3\mu\text{s}$) 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 six stator / four rotor pole configuration.

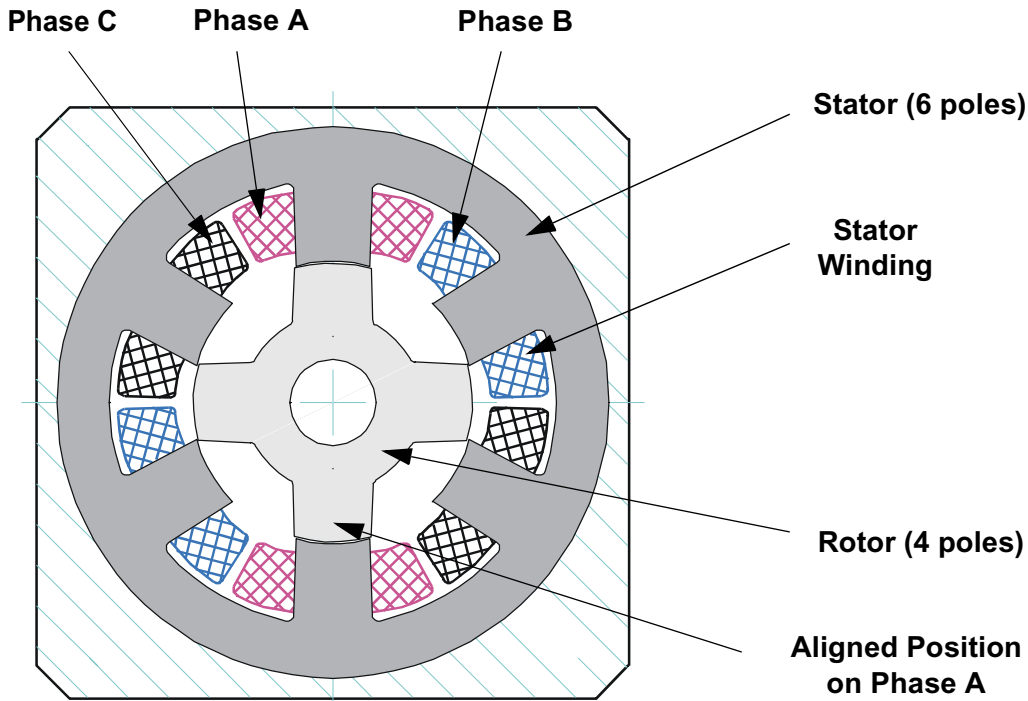


Figure 3-1 3-Phase 6 / 4 SR Motor

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 must 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 maximum 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; i.e., the rotor is in a position of minimal stator inductance. The inductance profile of SR motors is triangular, with maximum inductance when it is in an aligned position and minimum inductance when unaligned. [Figure 3-2](#) illustrates the idealized triangular 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° relative to each other. When the respective phase is powered, the interval is called the dwell angle, (θ_{dwell}). It is defined by the turn-on (θ_{on}) and the turn-off (θ_{off}) angles.

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 maximum 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 value in the position when the inductance begins to increase. This corresponds to the position where 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 [5], [References](#).

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, such as encoders or 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.

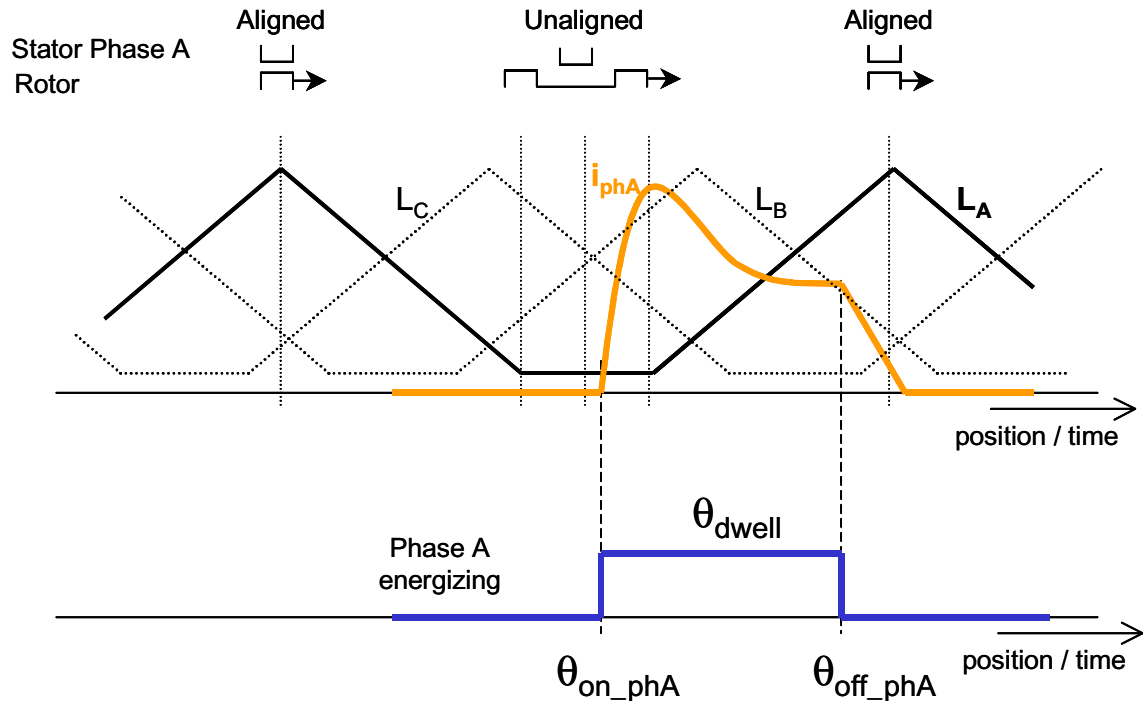


Figure 3-2 Phase Energizing

The motor itself is a low-cost, simply-constructed machine. Since high-speed operation is possible, the motor is suitable for high-speed applications, such as vacuum cleaners, fans, white goods, etc. As discussed previously, 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. A mathematical model can be created based on this theory. 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 SR motor's electromagnetic circuit is characterized by non-linear magnetization. [Figure 3-3](#) illustrates a magnetization characteristic for a specific SR motor; see [1], [References](#). 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 most 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.

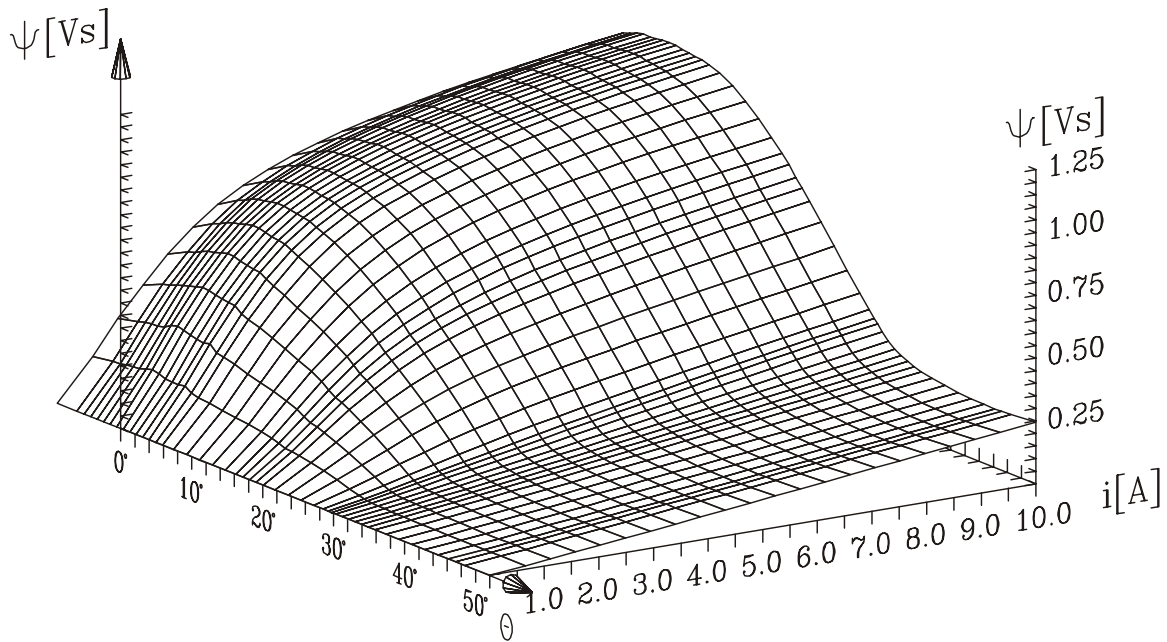


Figure 3-3 Magnetization Characteristics of the SR Motor

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

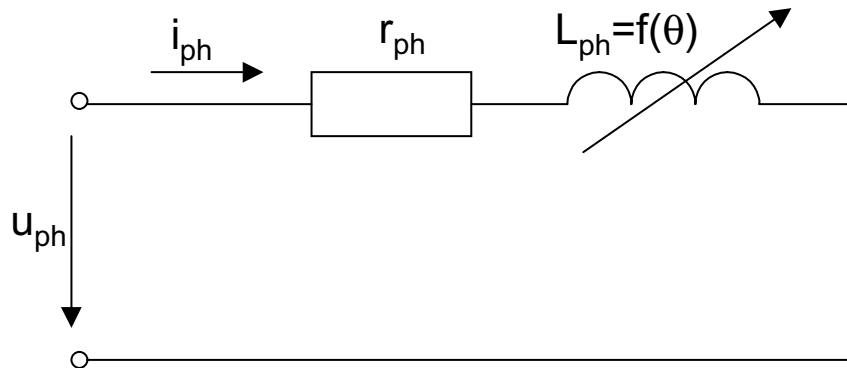


Figure 3-4 Electrical Diagram of One SR Motor Phase

According to the diagram, 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:

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

Where:

u_{ph}	=	The applied phase voltage
r_{ph}	=	The phase resistance
i_{ph}	=	The phase current
u_{Lph}	=	The induced voltage on the phase inductance

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 rotor position, θ_{ph} , so the induced voltage can be expressed as:

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

The phase voltage can then be expressed as:

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

or:

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

Where:

ω	=	The electrical 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, [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 and c index the individual phases

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, 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 enables fully independent control of the individual phases 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 fully controlled 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 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 entire commutation period and PWM is applied to the other transistor
- **Hard switching**, where PWM is applied simultaneously to both transistors

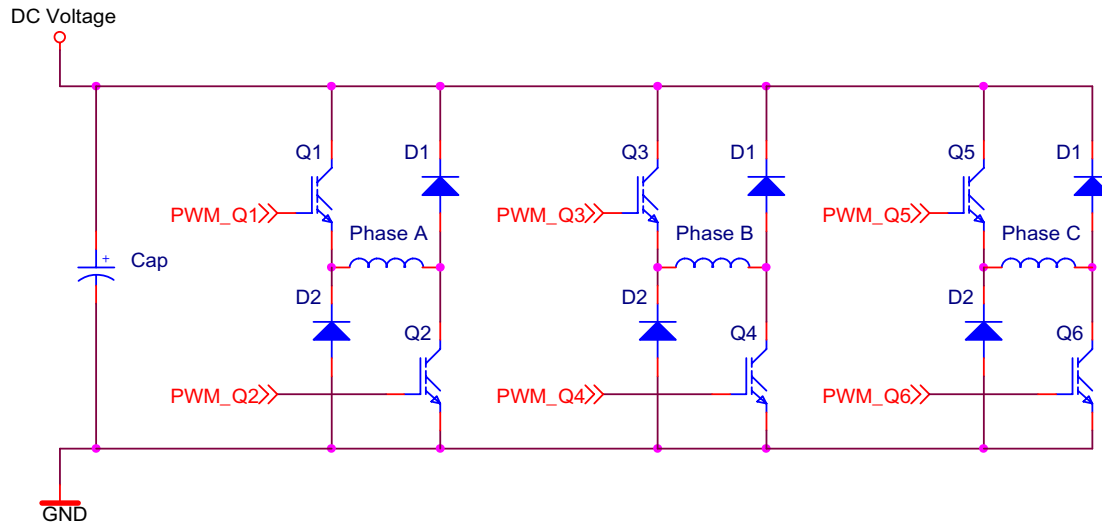


Figure 3-5 3-Phase SR Power Stage

Figure 3-6 illustrates both soft and hard switching PWM techniques. The control signals for the upper and the lower switches of the previously 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 [5], [References](#).

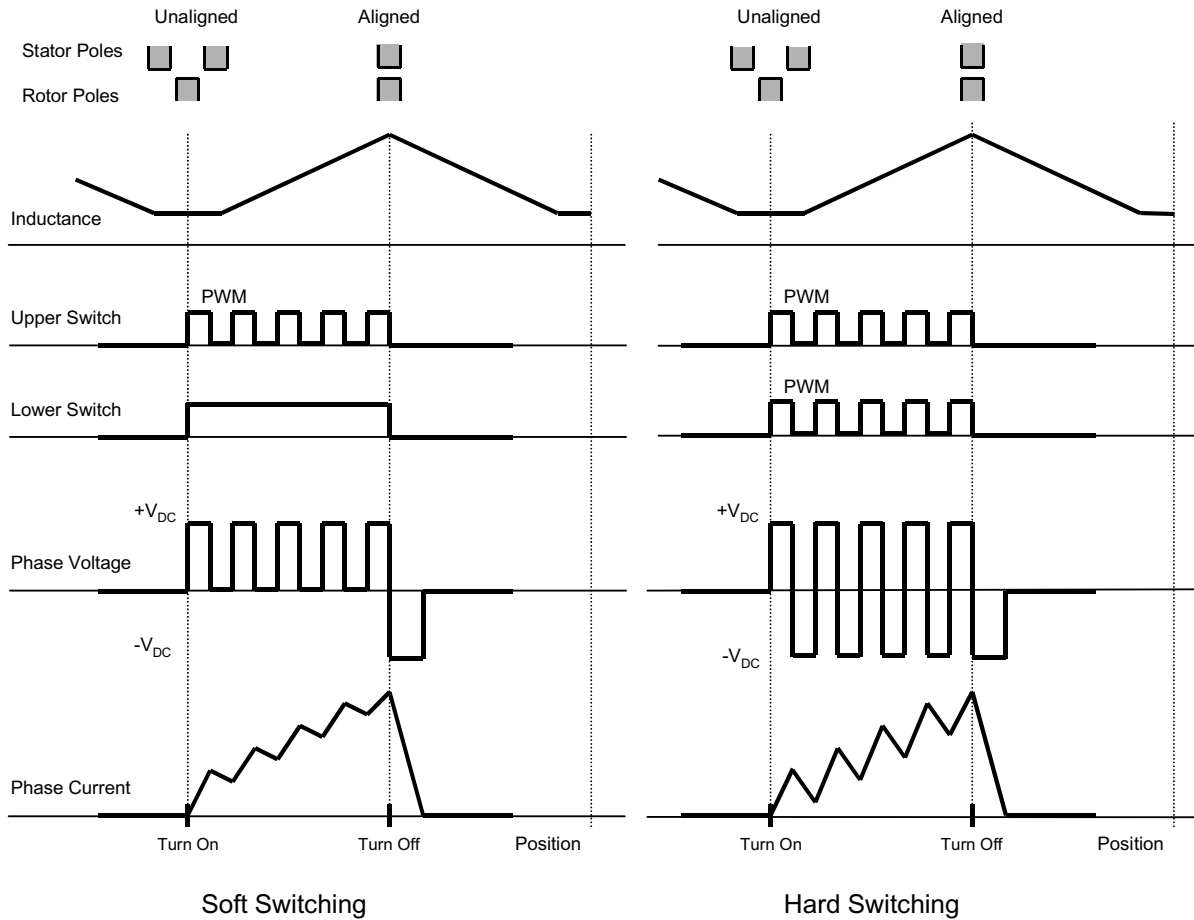


Figure 3-6 Soft Switching and Hard Switching

3.4 Voltage and Current Control of SR Motors

There are a number of control techniques for SR motors, which 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 DCBus 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 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.

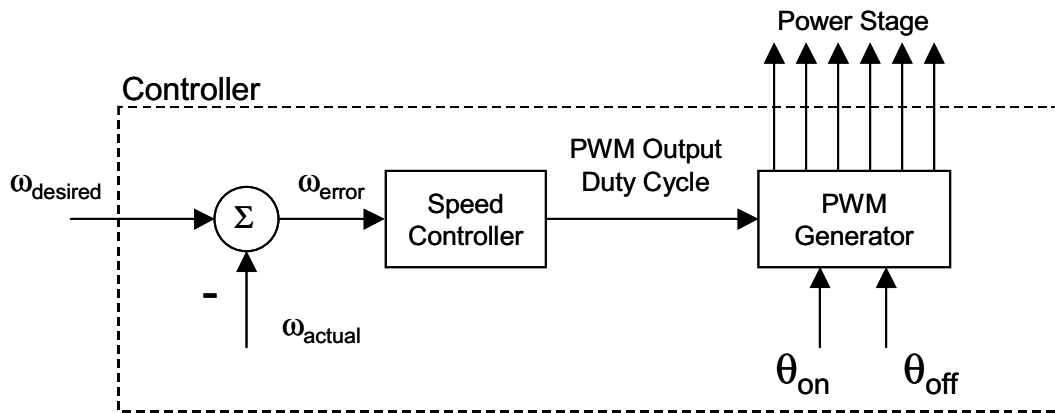


Figure 3-7 Voltage Control Technique

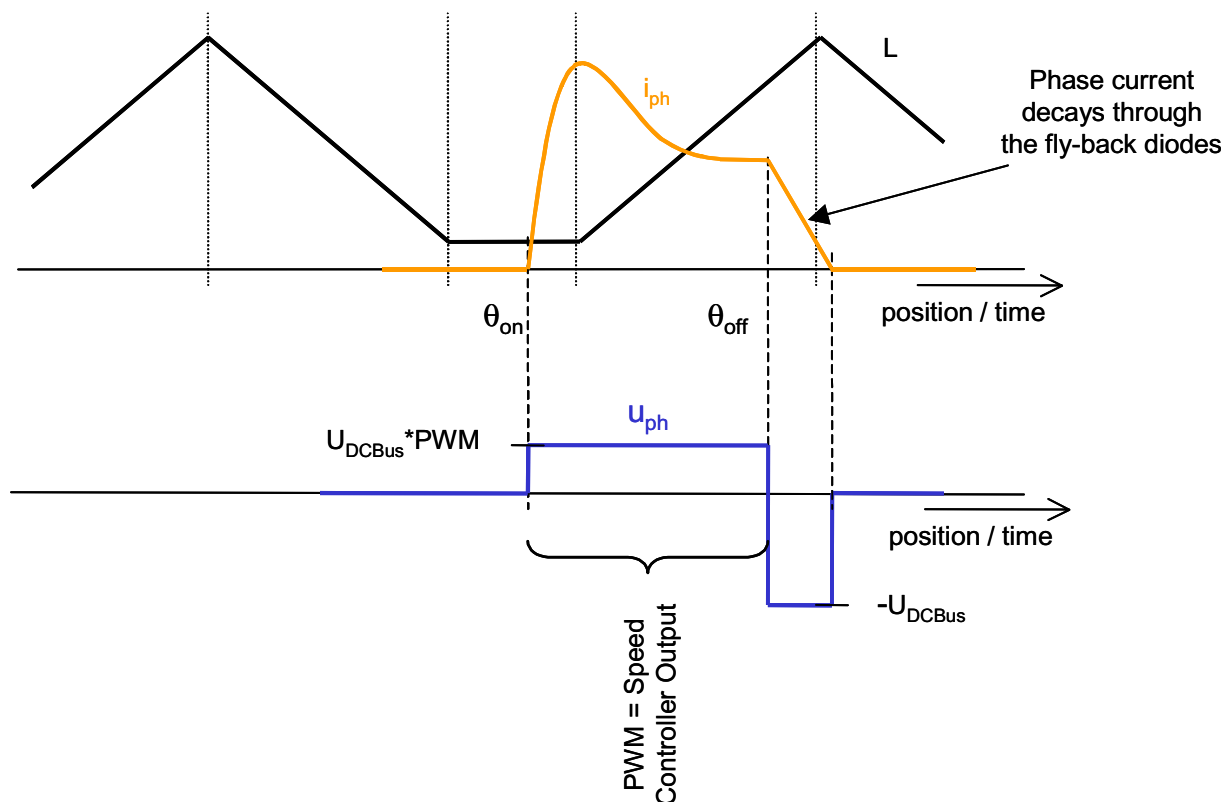


Figure 3-8 Voltage Control Technique—Voltage and Current Profiles

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 DCBus 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 must 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. The 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 voltage profiles can be seen in [Figure 3-10](#). Ideally, the phase current is controlled to follow the desired current.

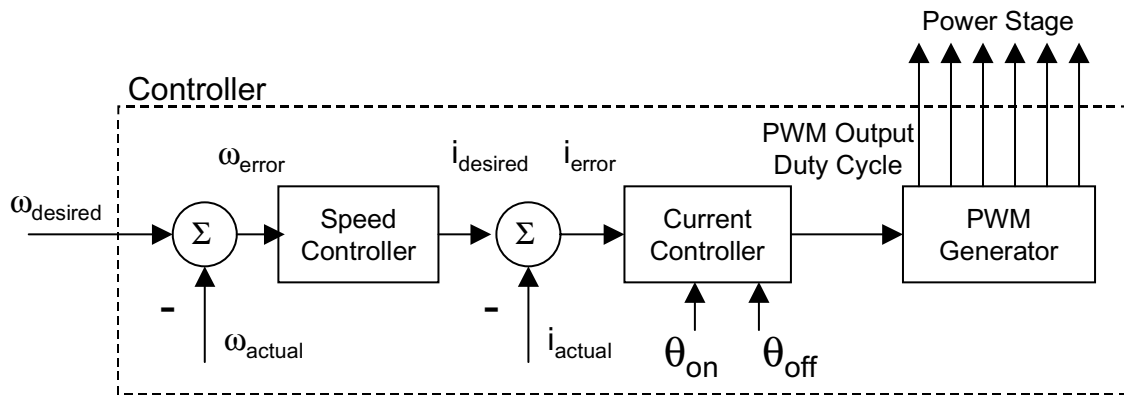


Figure 3-9 Current Control Technique

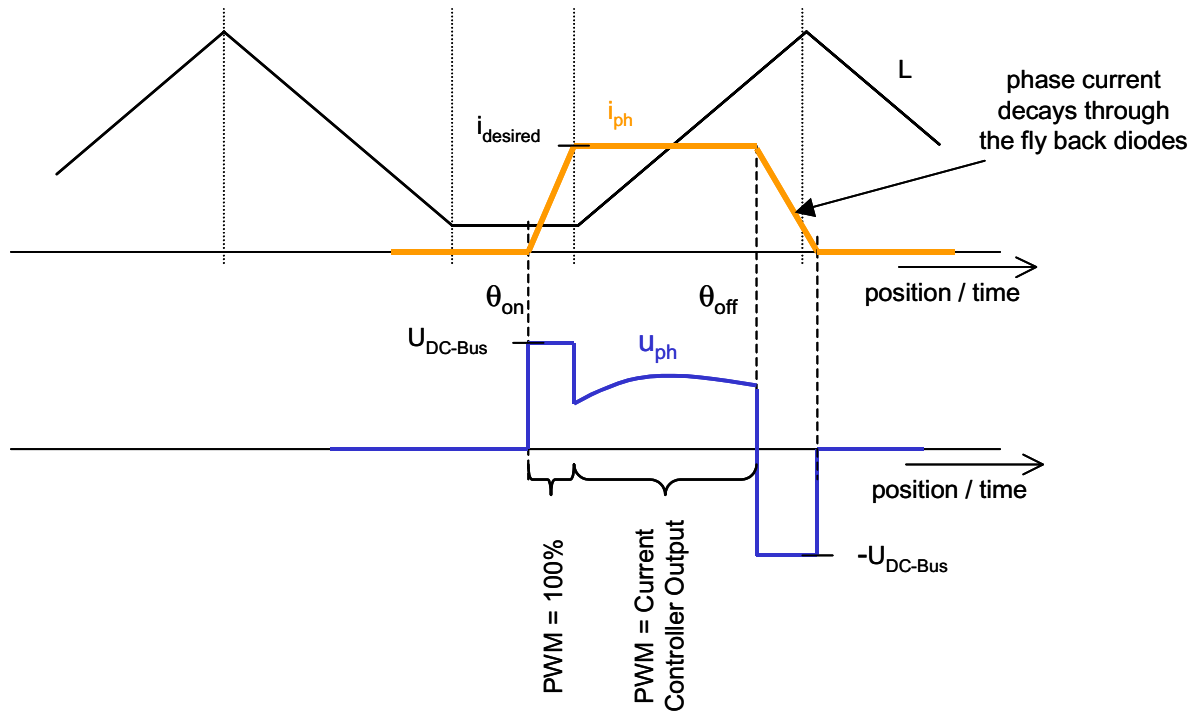


Figure 3-10 Current Control Technique—Voltage and Current Profiles

3.4.3 Calculation of the Turn-On Angle

The individual phases of an SR motor must be turned on at such a position that the phase current is able to rise to the desired level. The basic condition specifies that the phase current must achieve at least the desired level at a position where the stator and the rotor phases start to overlap. After the overlap position, the phase current begins to decrease due to the positive change in the inductance, so if the phase is turned on late, the phase current is not able to reach the desired level for the commutation stroke.

The turn-on position must be determined according to the applied phase voltage, the actual motor speed and the inductance profile of the motor. The phase is turned on at the position of minimal inductance, so the inductance can be considered a constant until the position where the stator and rotor poles start to overlap.

For constant inductance, the phase current may be considered as rising in a linear fashion. The time then required to achieve the desired current is determined from [EQ. 3-3](#) as:

$$\Delta t = \frac{L_u \cdot i_{desired}}{u_{phase} \cdot \gamma} \quad \text{EQ. 3-9}$$

Where:

Δt	=	The required time to achieve the desired current
$i_{desired}$	=	The desired current to be achieved
L_u	=	The unaligned inductance
u_{DC_Bus}	=	The DCBus voltage
γ	=	The PWM duty cycle

The electrical angle corresponding to the time required to reach the desired current can be determined as:

$$\Delta\vartheta = \omega_{actual} \cdot \Delta t \quad \text{EQ. 3-10}$$

Where:

$$\omega_{actual} = \text{The actual speed}$$

4. Techniques for Sensorless Control of SR Motors

4.1 Sensorless Position Estimation using Flux Linkage Estimation

The flux linkage estimation method belongs among the most popular sensorless SR position estimation techniques. A number of methods that use the flux linkage calculation have been developed; see [4] and [6], [References](#). These methods calculate the actual phase flux linkage and use its relation to the reference flux linkage for position estimation.

The method implemented in this application is based on the comparison of the estimated flux linkage and the reference flux linkage, defined for the turn-off (commutation) position. When the estimated flux linkage reaches the desired reference flux linkage, it indicates that the commutation position was reached. The actual phase is turned off and the following phase is turned on.

The **reference flux linkage** is derived from the magnetization characteristic as a function of phase current for the desired commutation position; see [Figure 4-1](#).

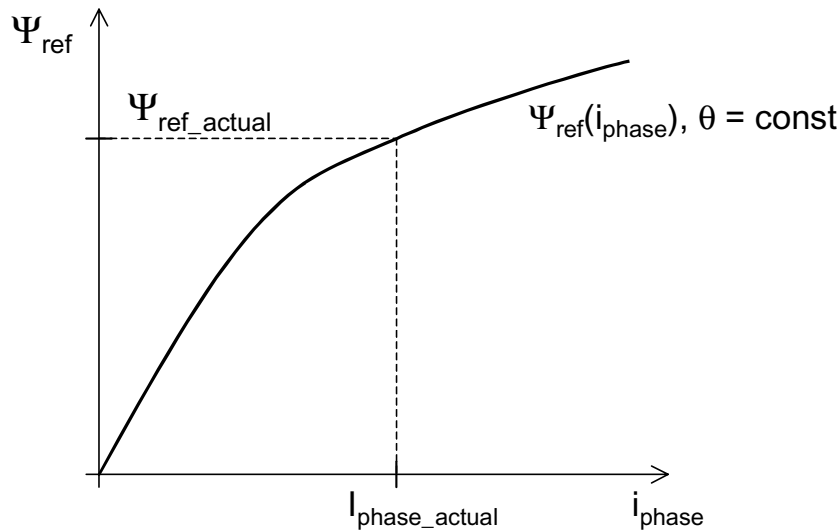


Figure 4-1 Reference Magnetization Curve for Constant Position

In order to simplify the determination of the reference flux linkage, it can be assumed that the flux linkage rises in a linear fashion in the interval between the unaligned and the aligned positions for a constant current. This assumption can be considered in the region of the expected commutation, so the reference flux linkage can then be derived from the flux linkage in the aligned position as:

$$\Psi_{\theta_{off}}(i_{ph}) = k(\theta_{off}) \cdot \Psi_{\theta_{Aligned}}(i_{ph}) \quad \text{EQ. 4-1}$$

Where:

$k(\theta_{off})$ is a linear function corresponding to the commutation angle. It can reach a value in the interval $\langle 0, 1 \rangle$, (0 corresponds to the unaligned position, 1 corresponds to the aligned position).

The reference magnetization curve, $\Psi(i_{ph})$, for the aligned position, $\theta_{Aligned}$, is stored in controller memory.

The **estimated flux linkage**, Ψ_{ph} , of the turned-on phase is calculated using the following equation:

$$\Psi_{ph} = \int_{t_{on}}^t (u_{ph} - R \cdot i_{ph}) dt \quad \text{EQ. 4-2}$$

Where:

- u_{ph} = The voltage applied to the motor phase (coil) winding
- i_{ph} = The actual phase current
- R = The phase resistance

Flux linkage estimation starts when the phase is turned on. The simultaneously sampled phase current and phase voltage are measured periodically at predetermined intervals and the flux linkage is estimated. Each time the flux linkage is calculated, it is compared with the reference level taken from the reference magnetization curve as a function of the actual phase current. When the estimated flux linkage exceeds the reference flux linkage, it indicates that the switching position has been reached and the commutation can be performed. The method is illustrated in **Figure 4-2**.

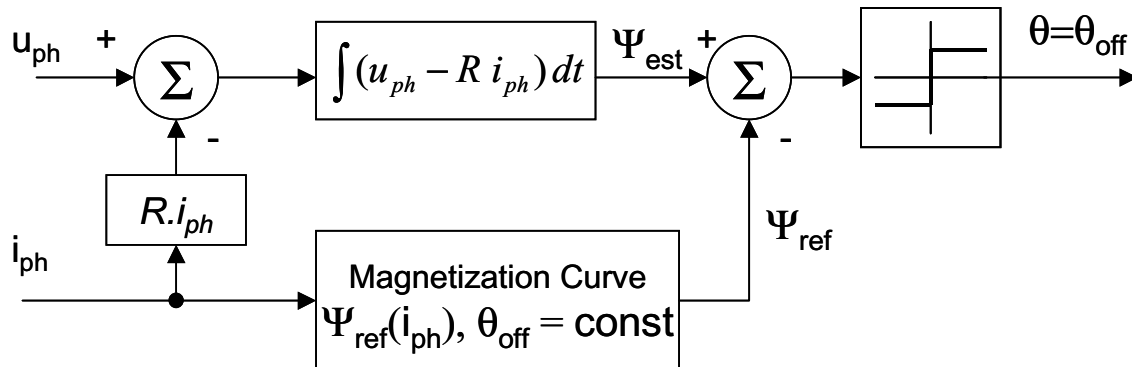


Figure 4-2 Position Estimation using One Reference Flux Linkage Function

The advantage of flux linkage estimation methods is that they are usable over wide speed ranges, from start up to high speeds. The position can accurately be estimated if the phase resistance is determined correctly; four-quadrant operation is possible.

The main disadvantage of all these methods is that the estimation of flux linkage is based on a precise knowledge of phase resistance. Phase resistance varies significantly with temperature, which yields to unwanted integration errors, especially at low speed. The integration error creates a significant position estimation error. Note that powerful hybrid controller-based controllers (like the 56F80x devices) can easily perform all of the sensorless flux linkage algorithm's needed calculations.

4.2 Flux Linkage Calculation in a Discrete Time Domain

The introduced algorithm for the flux linkage estimation can be used for both analog and digital controllers. Digital control is preferred today for reasons of cost, flexibility and performance. For digital systems, the flux linkage calculation based on [EQ. 4-2](#) must be converted at the discrete time domain.

Flux linkage estimation is performed regularly at the sampling frequency of the measurements of phase voltage and phase current. [EQ. 4-2](#) can be converted to:

$$\Psi_N = \sum_{k=1}^N [u_k - i_k r_k] \cdot T, \quad \text{EQ. 4-3}$$

Where:

T	=	The sampling period
u_k	=	The sampled phase voltage
R	=	The phase resistance
i_k	=	The sampled phase current
r_k	=	The sampled phase resistance
Ψ_N	=	The calculated flux linkage at sample N

Flux linkage, Ψ_N , is calculated regularly at each sampling cycle from the beginning of the commutation stroke, t_1 . The sampling period T is constant. [EQ. 4-3](#) can be transformed to the following form:

$$\Psi_N = [u_N - i_N r_k] \cdot T + \Psi_{N-1}, \quad \text{EQ. 4-4}$$

Where:

Ψ_{N-1}	=	The calculated flux linkage for the previous measuring cycle (N-1)
--------------	---	--

In order to decrease the computational requirements, [EQ. 4-4](#) can be transferred to:

$$\frac{\Psi_N}{T} = [u_N - i_N r_k] + \frac{\Psi_{N-1}}{T} \quad \text{EQ. 4-5}$$

So, the flux linkage divided by the sampling period is calculated rather than the pure flux linkage. Because the sampling period is kept constant, the division can be considered a scaling factor. For proper functionality of the position estimation algorithm, the reference flux linkage must be scaled in the same way.

4.3 Sensorless On-the-Fly Resistance Estimation

The resistance of the phase winding is one of the most decisive factors in the magnetic flux linkage estimation [EQ. 4-2](#). During motor operation, the variation of the resistance can exceed 30% of the nominal value because the phase resistance depends strongly on temperature. The effect of the phase resistance drift is more significant at low- and middle-speed ranges, where the voltage drop on the winding is comparable to the phase supply voltage, u_{ph} . This variation causes an inaccurate estimation of the flux linkage, so it generates position estimation errors and, based on such magnetic flux estimations, the sensorless techniques do not give satisfactory results. Therefore, in the case of an accurate and robust sensorless control algorithm, the actual value of the winding resistance must be accurately measured or estimated during motor operation.

In order to improve the behavior of the sensorless flux linkage estimation algorithm, an on-the-fly phase resistance estimator has been invented. The resistance estimation algorithm was patented as No. 6,366,865 at the US Patent Office; see [3], [References](#).

The development of the phase resistance estimation was based on the flux linkage estimator [EQ. 4-2](#), which calculates flux linkage, Ψ_{Est} , at time t , using the following formula:

$$\Psi_{Est} = \int_{t1}^t (u_{ph} - R^* \cdot i_{ph}) dt \quad \text{EQ. 4-6}$$

Where:

- u_{ph} = The voltage applied to the motor phase (coil) winding
- i_{ph} = The phase current
- R^* = The assumed phase winding resistance
- $t1$ = The time when the motor phase winding starts to be energized

The assumed phase winding resistance, R^* , is the sum of the actual phase winding resistance, R , and the resistance error, ΔR . The resistance error can be caused by temperature drift, an inaccurately obtained value, etc.

$$R^* = R + \Delta R \quad \text{EQ. 4-7}$$

[Figure 4-3](#) illustrates the flux linkage waveforms calculated by the flux linkage estimator during a typical working cycle of one phase of an SR motor. Unlike the sensorless flux linkage estimation method, where the flux linkage is calculated up to the phase commutation angle θ_{off} , the flux linkage is calculated during the entire time in which the current is flowing through the phase. The phase current and the shape of the flux linkage are defined by the control strategy, rotor position, and magnetization characteristic. SR motors are driven in a way that the motor phases are energized sequentially and the phase current therefore rises from zero, at the beginning of the cycle where the phase is turned on ($t1 \approx \theta_{on}$), up to θ_{off} , where the phase is disconnected and then falls down to zero again at the end of the cycle ($t2$). As shown, the flux linkage rises during the interval between the turn-on ($t1$) and the turn-off angles of the phase. When the phase is turned off, flux linkage decreases until the phase current disappears. If all the parameters in [EQ. 4-6](#) are obtained correctly, and the resistance error ΔR is zero, then the flux linkage is equal to zero at $t2$, seen in [Figure 4-3](#).

$$\Psi_{t2} = 0 \quad \text{EQ. 4-8}$$

For the influence of the resistance error, assume that:

- The phase voltage and the phase current were measured correctly and the measurement error can be ignored
- The resistance error ΔR is not equal to zero, but it affects the estimation of the flux linkage

Because the flux estimation is the result of an integration (see [Figure 4-3](#)), the total flux estimation error at the end of the working cycle (t_2) can be quite significant.

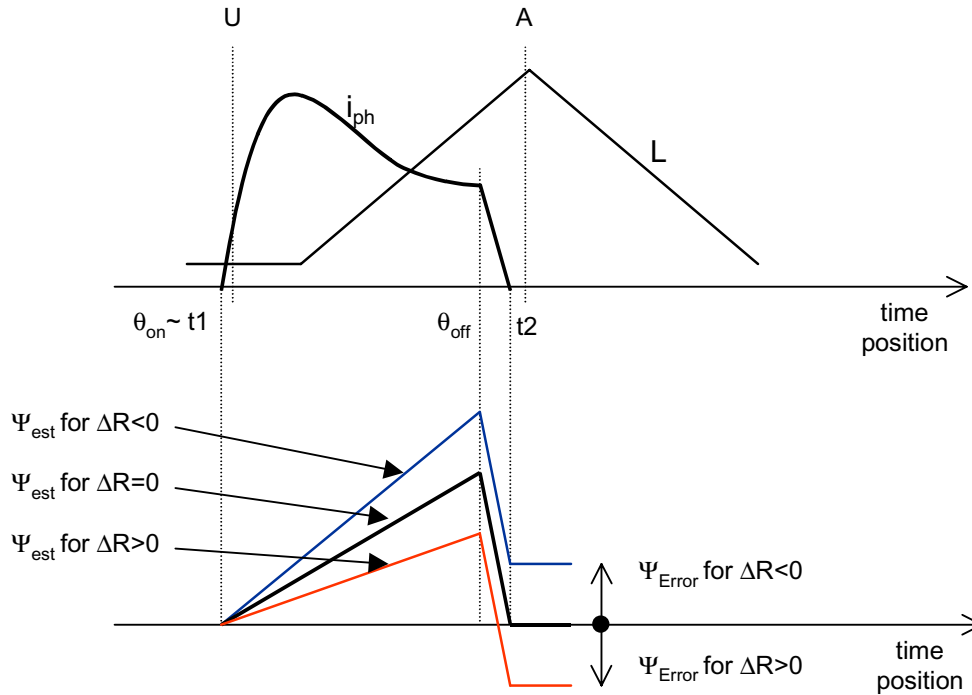


Figure 4-3 Flux Linkage and Phase Current

The basis of the **resistance estimation algorithm** is that if the phase current is zero, then the magnetic flux must be zero as well. Resistance error leads to flux estimation error; see [Figure 4-3](#). Thus, it enables calculation of the flux estimation error at the point in time (t_2) when the phase current falls to zero.

$$\Psi_{phEstim(t_2)} = \int_{t_1}^{t_2} (u_{ph} - R \cdot i_{ph} - \Delta R \cdot i_{ph}) dt = \Psi_{ph(t_2)} + \Psi_{Error(t_2)} \quad \text{EQ. 4-9}$$

Because the flux linkage at time t_2 is equal to zero (see [EQ. 4-8](#)), the estimation error is equal to:

$$\Psi_{phEstim(t_2)} = \Psi_{Error(t_2)} = - \int_{t_1}^{t_2} \Delta R \cdot i_{ph} dt \quad \text{EQ. 4-10}$$

Based on [EQ. 4-10](#), it is apparent that if the flux linkage estimation error is positive, the resistance error is negative, and if the flux linkage estimation error is negative, the resistance error is positive.

$$\Psi_{Error(t2)} > 0 \quad \Rightarrow \quad \Delta R < 0 \quad \text{EQ. 4-11}$$

$$\Psi_{Error(t2)} < 0 \quad \Rightarrow \quad \Delta R > 0 \quad \text{EQ. 4-12}$$

Assume that the rate of change of the phase resistance is small during one commutation of the SR motor (which is valid for temperature drift):

$$\frac{\Delta R}{t2 - t1} \cong 0 \quad \text{EQ. 4-13}$$

Using the previous assumption, [EQ. 4-10](#) can be rewritten as the following:

$$\Psi_{EstErr(t2)} = -\Delta R \int_{t1}^{t2} i_{ph} dt \quad \text{EQ. 4-14}$$

The resistance error can then be expressed as:

$$\Delta R = -\frac{\Psi_{EstErr(t2)}}{\int_{t1}^{t2} i_{ph} dt} \quad \text{EQ. 4-15}$$

[EQ. 4-15](#) illustrates that the resistance error can be expressed as the ratio between the calculated flux linkage error at time t2, where the phase current decreases to zero, and the integral of the phase current, both of which are calculated over the complete phase current pulse.

More details of this algorithm can be found in [2], [References](#).

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:

- Sensorless speed control of an SR motor using a flux linkage estimation technique with an inner-current closed loop
- Targeted for 56F80xEVM or 56F83xxEVM plus LMDC
- 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% to +10%)

- The control technique incorporates:
 - Current SRM control with a speed-closed loop
 - Phase resistance measurement during start-up
 - Phase resistance estimation at low speeds
 - Motor starts from any position with rotor alignment
 - Rotation of one direction
 - Motoring mode
 - Minimum speed of 600rpm
 - Maximum speed of 2600rpm at input power line 230V AC
 - Maximum speed of 1600rpm at input power line 115V AC
- Encoder position reference for evaluation of position estimation, visualized by PC master software (not used for SR control technique)
- Manual interface
 - RUN / STOP switch
 - UP / DOWN push button control
 - LED indicator
- PC master software control interface
 - Motor start / stop
 - Speed set-up
- PC master software monitor
 - Graphical control page
 - Required speed
 - Actual motor speed
 - Manual or PC operating mode
 - Start / stop status
 - Drive fault status
 - DCBus voltage level
 - Identified power stage boards
 - System status
 - Speed scope observes:
 - Actual and desired speeds
 - Desired current
 - Start-up recorder observes:
 - Start-up phase current
 - Flux linkage
 - Output duty cycle
 - Encoder position reference with fine resolution

- Flux linkage recorder observes:
 - Phase current
 - Estimated flux linkage
 - Reference flux linkage
 - Encoder position reference with fine resolution
- Current controller recorder observes:
 - Actual and desired phase current
 - Output duty cycle
 - Encoder position reference with fine resolution
- Fault protection from:
 - DCBus overvoltage
 - DCBus undervoltage
 - DCBus overcurrent
 - Overheating

5.2 Application Description

5.2.1 Application Concept

A standard system concept was chosen for the drive; see [Figure 5-1](#). The system incorporates the following hardware parts:

- 3-Phase SR high-voltage development platform (power stage with optoisolation board, SR motor with attached brake)
- Feedback sensors for:
 - DCBus voltage
 - DCBus current
 - Phase currents
 - Temperature
- 56F80x or 56F8300 controller

The hybrid controller 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 operating modes:

- In **Manual** operating mode, the required speed is set by a START / STOP switch and UP and DOWN push buttons
- In **PC master software** operating mode, the required speed is set by PC master software

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.

DCBus voltage, DCBus current, and power stage temperature are measured during the control process. The measurements are used to protect the drive from DCBus overvoltage, DCBus undervoltage, DCBus overcurrent and overtemperature. DCBus undervoltage and overtemperature protection are performed by software, while DCBus overcurrent and the DCBus overvoltage fault signals utilize the fault inputs of the hybrid controller's on-chip PWM module. 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 the -15% to +10% of the nominal voltage, the fault "*Out of the Mains Limit*" disables drive operation. If any of the 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 RUN / STOP switch is moved to the STOP position. The fault state is indicated by the on-board LED.

5.2.2 Initialization and Start-Up

Rotor alignment and initialization of the control algorithms must be performed before the motor can be started; see [Figure 5-2](#). Initialization of the control algorithm includes the measurement of the actual start-up phase resistance.

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

1. Phases B & C are turned on simultaneously
2. After 50ms, Phase C is turned off; Phase B stays powered
3. After an additional 550ms, 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 torque, Phase C provides the initial movement. Phase C is then 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.

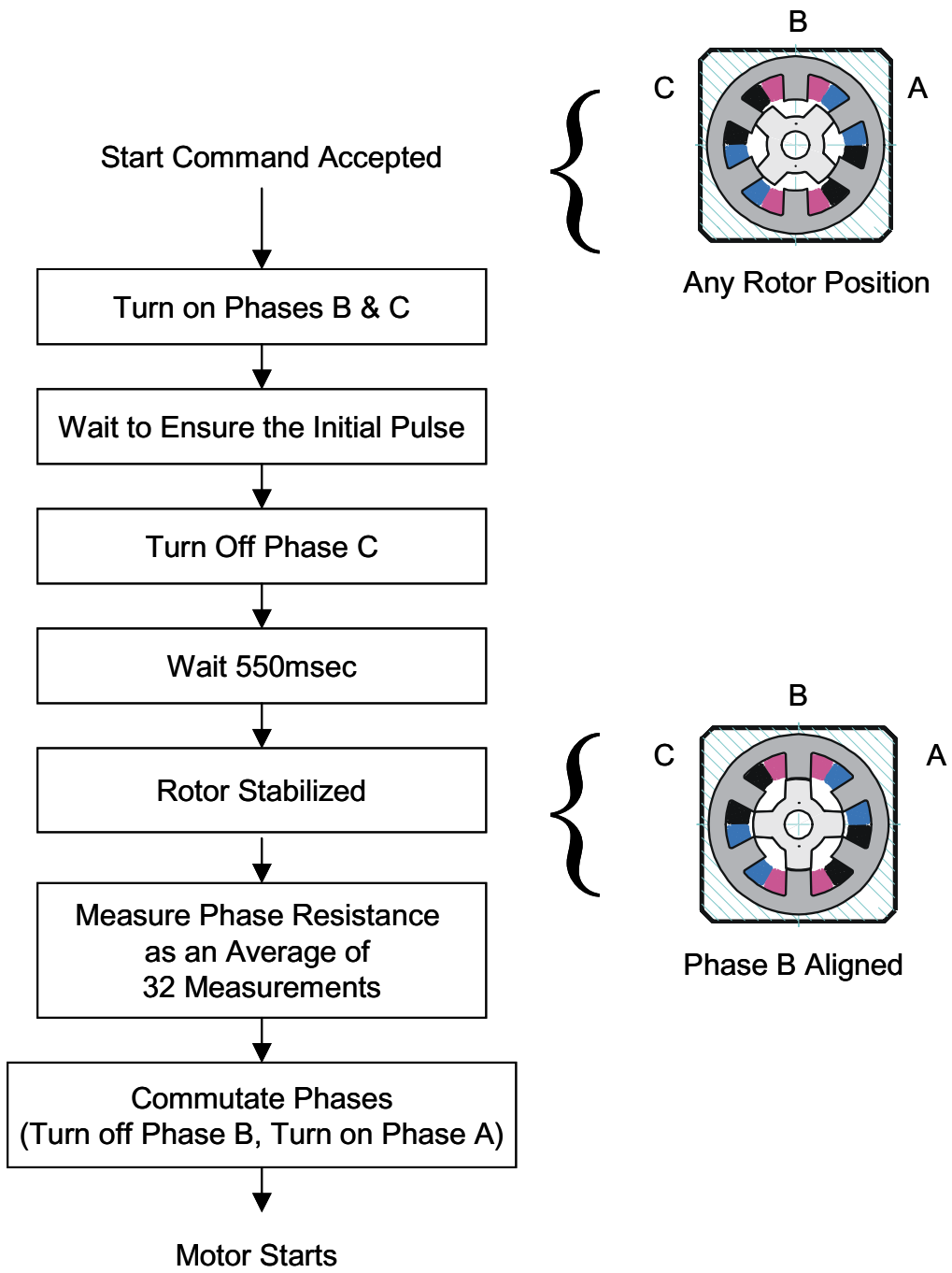


Figure 5-2 Start-Up Sequence

When the rotor is stabilized at the known position, measurement of the phase resistance of the powered phase can be performed. Phase resistance is calculated from the measured phase current, i_{ph} , DCBus voltage, U_{DC-Bus} , and the applied PWM duty cycle, γ . It is assumed that the resistance of all three phases is identical. Phase resistance, R_0 , is calculated as:

$$R_0 = \frac{\sum(\gamma \cdot U_{DCBus})}{\sum i_{ph}} \quad \text{EQ. 5-1}$$

In total, stabilization and the resistance measurement take 1 second. The rotor is sufficiently stable to reliably start the motor in the desired direction of rotation. When the phase resistance has been measured, the motor can be started by commutation of the phases (turning off the stabilization of Phase B and applying power to start Phase A).

This sequence is followed for every start-up of the motor because neither the initial rotor position nor the actual phase resistance is known.

5.2.3 Commutation Algorithm and Resistance Estimation

The core of the control algorithm includes the calculation of the commutation angle, the flux linkage, the reference flux, the commutation of phases and an estimation of the phase resistance.

Calculation of the commutation angle is performed regularly during motor operation according to [EQ. 3-9](#) and [EQ. 3-10](#).

Flux linkage is estimated during a complete current stroke of the powered phase, from the moment the phase is turned on until the moment the phase current disappears. It serves for both position estimation (determination of the commutation instance) and for resistance estimation. Commutation of the motor phases is based on a comparison of the actual estimated flux linkage and the reference flux linkage for the required commutation angle; see [Section 4.1](#). Phase resistance is estimated according to the flux linkage error, which is captured the moment the phase current disappears; see [Section 4.3](#). A detailed block diagram of the control algorithm is shown in [Figure 5-3](#).

The control process starts at the moment the given phase is turned on. It can be either during start-up or after the rotor is aligned and commutated.

When the phase is turned on (θ_{on}), the phase current and the phase voltage are measured simultaneously at the center of the PWM pulses. The phase current, i_{ph} , is measured directly using the phase current sensing circuitry with software noise elimination implemented, while phase voltage, u_{ph} , is calculated according to the measured DC Bus voltage and the actual PWM duty cycle, γ :

$$u_{ph} = \gamma \cdot U_{DCBus} \quad \text{EQ. 5-2}$$

The measured phase current and DCBus voltage are used for calculating the actual flux linkage, Ψ_{actual} , as shown in [EQ. 4-5](#).

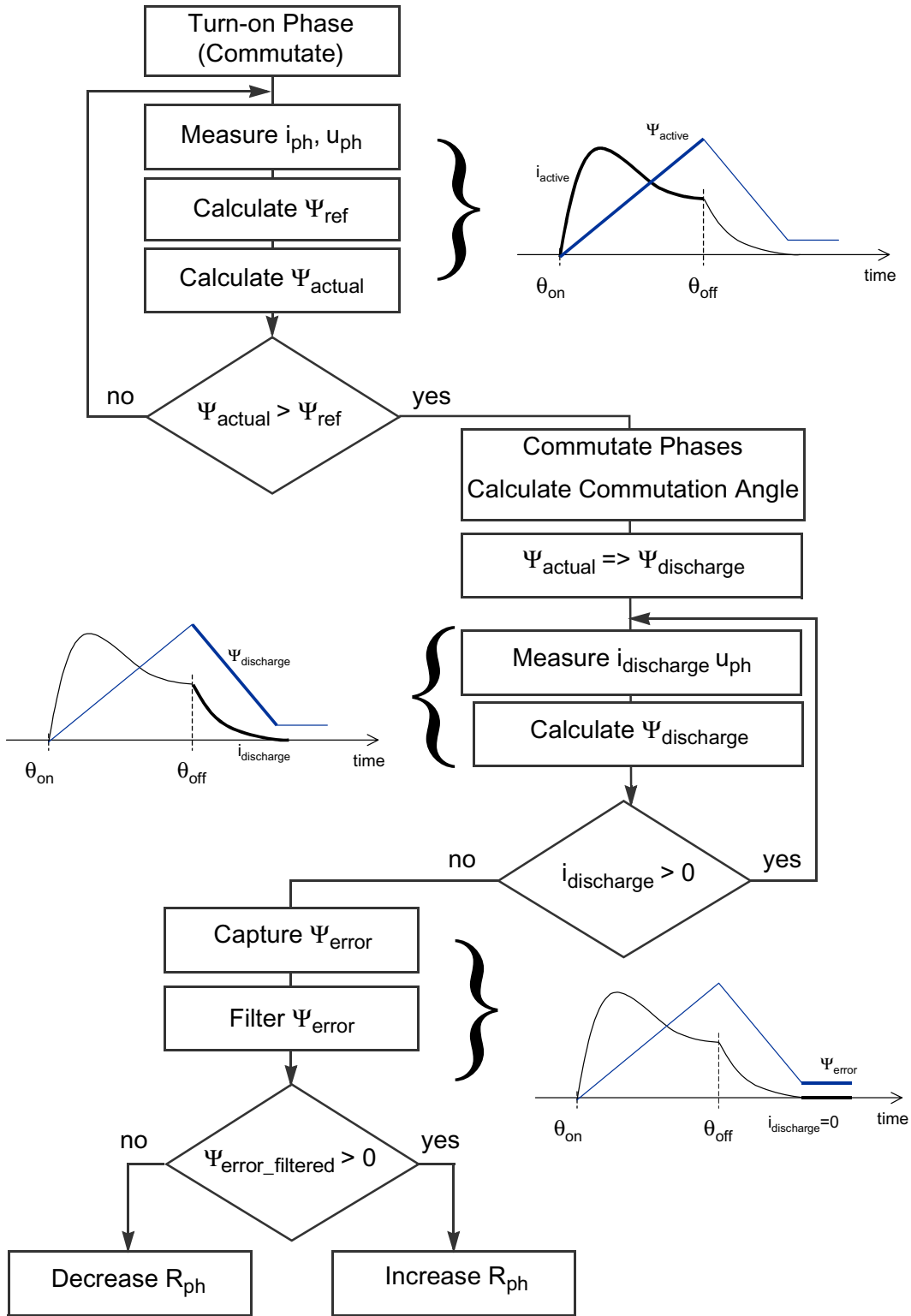


Figure 5-3 Control Flow Diagram

The reference flux linkage, Ψ_{ref} , for a given commutation angle, θ_{off} , is a function of the phase current i_{ph} , $\Psi_{ref} = f(i_{ph}, \theta_{off})$. The reference flux linkage characteristic for the aligned position must be derived from the motor magnetization characteristic. Such a characteristic for the motor tested is shown in **Figure 5-4**. Compare it with **Figure 4-1**, which illustrates the general magnetization curve. As demonstrated, the measured characteristic is linear—this application works in the linear part of the magnetization characteristic. For other positions, the reference flux linkage is calculated according to **EQ. 4-1**.

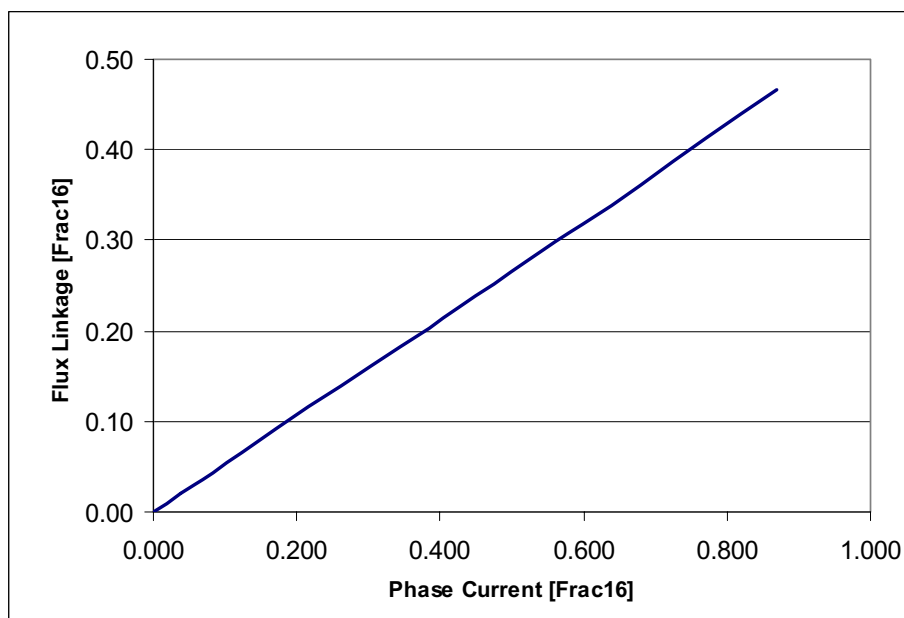


Figure 5-4 Flux Linkage as a Function of Phase Current for the Aligned Position

The estimated flux linkage, Ψ_{actual} , is compared with the reference flux linkage, Ψ_{ref} . If the estimated value is lower than the reference value, the estimation continues regularly at the sampling frequency. When the estimated value reaches the reference value, this indicates that the desired position, θ_{off} , is achieved. At that moment, commutation of the phases is performed; the powered phase is turned off and the following phase, in the direction of the rotation, is turned on. The flux linkage calculation for determining the following commutation event starts again at an initial value of zero.

When the phase is turned off, the phase current starts to decrease; the phase is discharged. The flux linkage, $\Psi_{discharge}$, continues to be calculated regularly at the rate of the sampling period (PWM frequency) during the phase current discharge. The discharge phase current, $i_{discharge}$, is monitored. As soon as the phase current approaches zero, the flux linkage error, Ψ_{Error} , is captured. The flux linkage error corresponds to the phase resistance error used for the flux linkage calculation.

The flux linkage error is then filtered through several samples in order to eliminate calculation, measurement, and noise error.

The filtered value is used for evaluation of phase resistance according to **EQ. 4-11** and **EQ. 4-12**. If the filtered flux linkage error is greater than zero, the estimated phase resistance is increased by a small amount (0.1%). In the opposite case, the estimated phase resistance is decreased by a small amount (0.1%). The corrected resistance value is then used during the next flux linkage estimation process. In this way, phase resistance is tracked throughout operation.

5.2.4 Current and Voltage Measurement

Precise phase current and DCBus voltage measurement is a key factor in the implementation of sensorless flux linkage estimation. Any inaccuracy in the measurement leads to a flux linkage estimation error and thus to position estimation error and resistance estimation error.

5.2.4.1 Current Sensing

Current measurement must be investigated according to the current sensors used and the influence of 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 are not suitable for most cost-sensitive applications. Therefore, current shunt resistors inserted into the phase's current path are often used; see [Figure 5-5](#). The phase current is sensed as a voltage drop across the sense resistor.

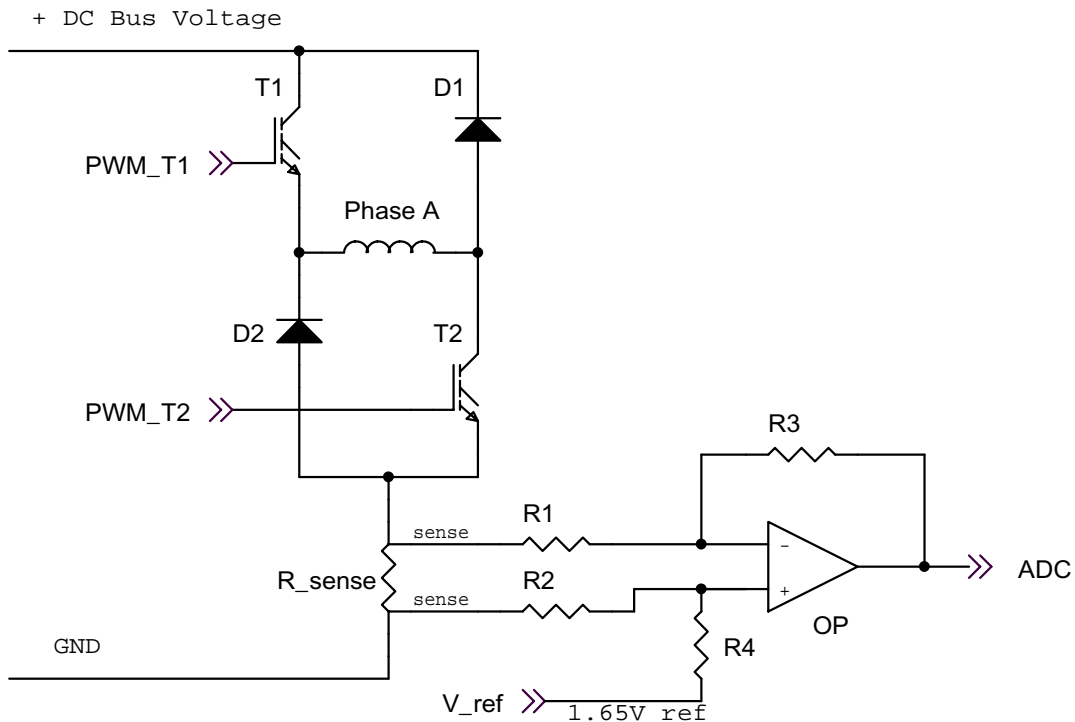


Figure 5-5 Shunt Resistors Current Sensors

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 5-6](#). 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 must be inverted. The diagram shows that for a reliable current shape reconstruction, the sensing must 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. Following this process allows the current to be read with accuracy and credibility.

Figure 5-7 illustrates the actual phase currents of a 3-phase motor, measured on the shunt resistors as described previously.

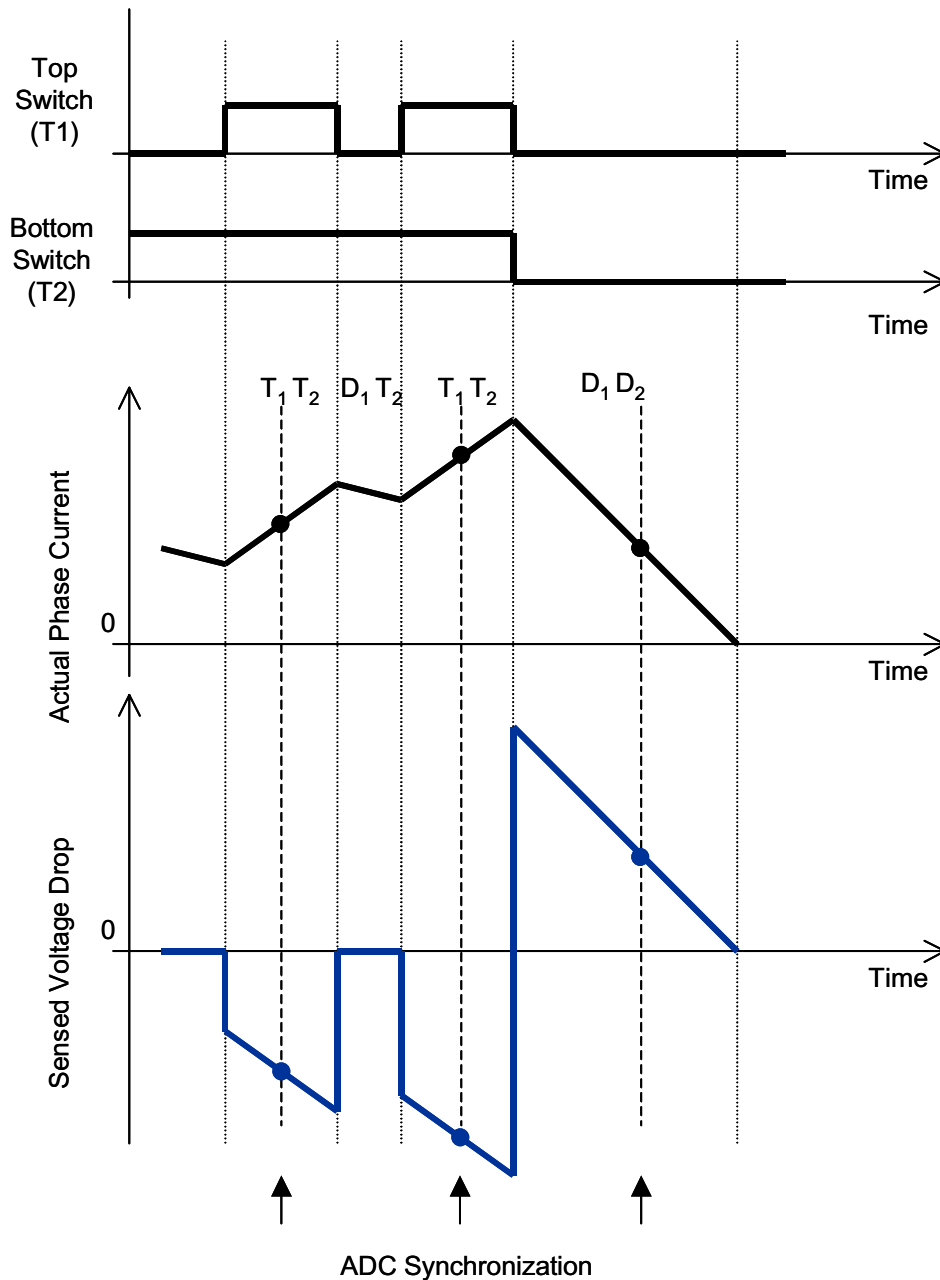


Figure 5-6 Soft Switching Current on Shunt Resistors

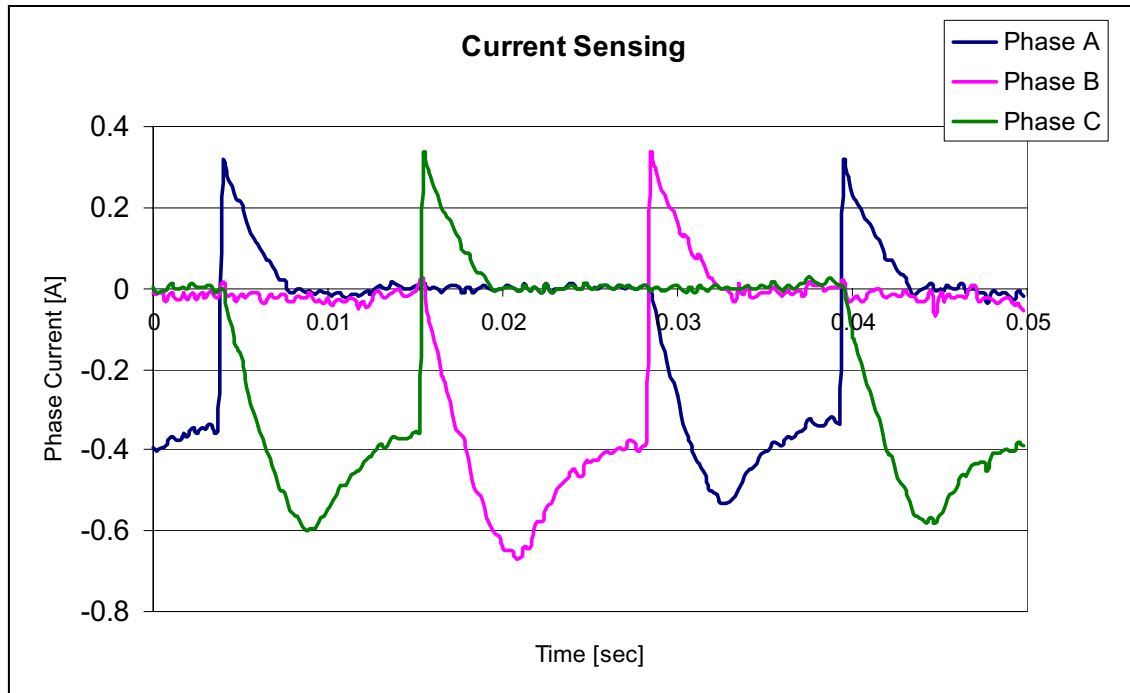


Figure 5-7 Phase Current Measured at Current Shunt Resistors

There is one serious disadvantage to use of low cost shunt resistor sensors. Due to the low-voltage drop sensed across the shunt current resistors, the measured signals are susceptible to noise.

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

$$\text{MeasuredSignal} = \text{ActualSignal} + \text{Noise} \quad \text{EQ. 5-3}$$

$$\text{ReferenceSignal} = \text{KnownSignal} + \text{Noise} \quad \text{EQ. 5-4}$$

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.

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

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 as the working phase is turned off, and 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:

$$I_{ph} = I_{measured} - I_{reference} \quad \text{EQ. 5-6}$$

The reference signal must 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 5-7**.

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 5-8**. The figures illustrate the phase current as it is measured (the active phase current is inverted compared to **Figure 5-7**), and the same current with the implemented noise reduction technique. As demonstrated, 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. Thus, position estimation and resistance evaluation are also improved.

5.2.4.2 Voltage Sensing

The DCBus voltage sensor is represented by a simple voltage divider. DCBus 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. If the power stage is designed correctly, the ripple amplitude should not exceed 10% of the nominal DCBus value.

The measured DCBus voltage must be filtered in order to eliminate noise. One of the most useful techniques is a moving average filter that calculates an average value from the last N samples:

$$u_{DCBus} = \sum_{n=1}^{-N} u_{DCBus}(n) \quad \text{EQ. 5-7}$$

In order to increase the precision of 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.

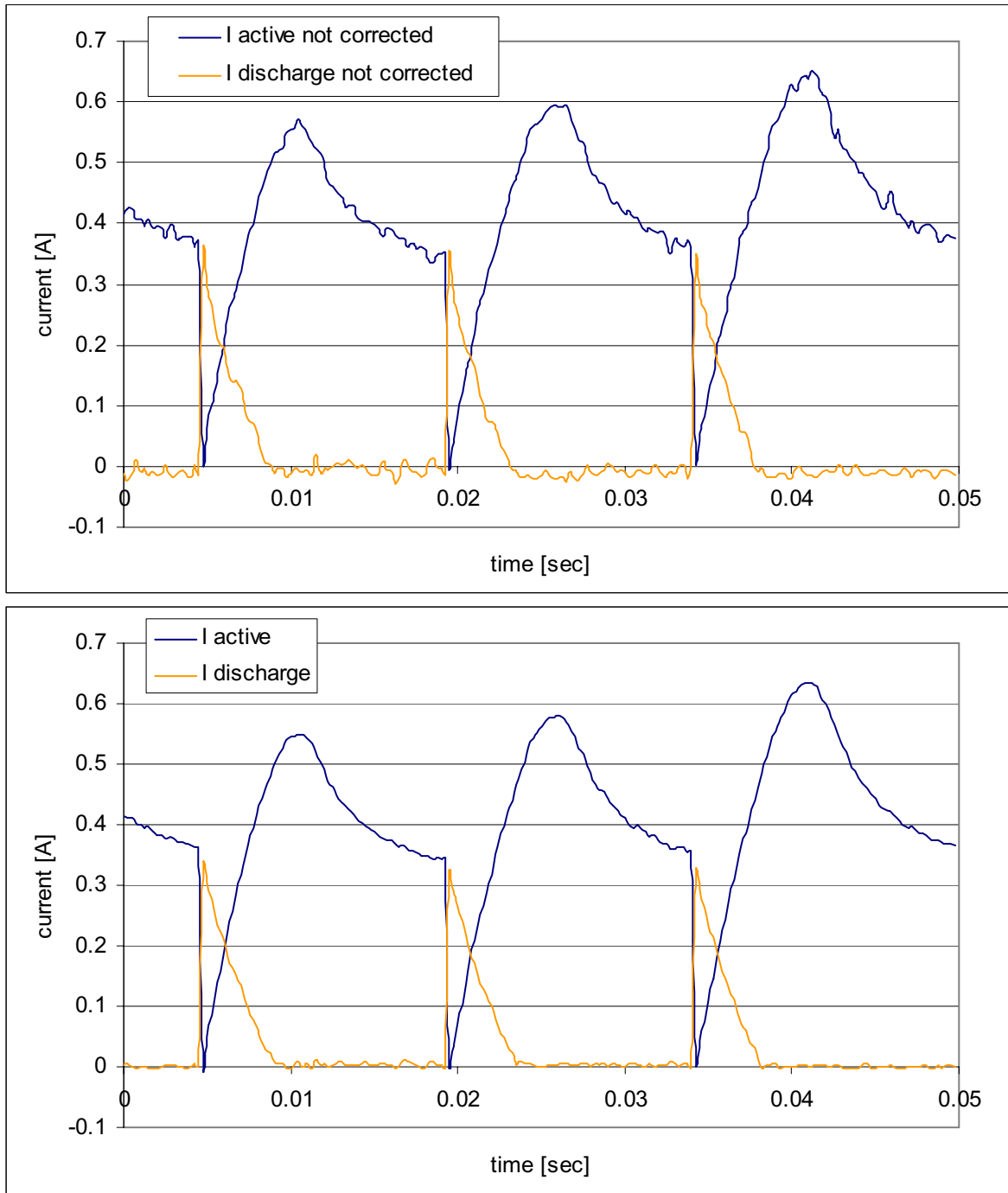


Figure 5-8 Measured 3-Phase Currents without Noise Correction and with Noise Correction Implemented

5.2.5 Power Module Temperature Sensing

The power module's temperature is measured and used for thermal protection. The hardware realization is shown in [Figure 5-9](#). 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 this application, the temperature in degrees Celsius is calculated according to the conversion equation:

$$\text{temp} = \frac{\text{Temp_sense} - b}{a} \quad \text{EQ. 5-8}$$

Where:

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

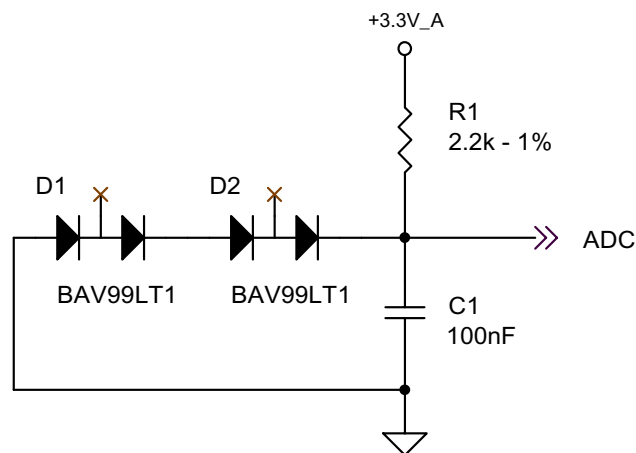


Figure 5-9 Temperature Sensor Topology

6. Hardware Implementation

This section explains the hardware implementation for targeting a 56F83xxEVM.

6.1 Hardware Setup

As previously stated, the application runs on Freescale's motor control hybrid controllers using the hybrid controller EVM boards and a dedicated 3-Phase SR high-voltage platform.

The application can be controlled by Freescale's 56F83xx motor control hybrid controller.

[Figure 6-1](#) illustrates how application hardware is set up. The system hardware set up for a particular hybrid controller varies only by the EVM used. Application software is identical for all hybrid controllers. The EVM and chip differences are handled by PE's off-chip drivers for the particular hybrid controller EVM.

Details about the application's hardware set up can be found in the **Targeting 56F8300 Demonstration Board** manual.

A dedicated User Manual describes the EVM in detail and includes a schematic of the board, description of individual function blocks, and a bill of materials for the EVM. The individual boards can be ordered from Freescale as standard products. Descriptions of all boards and documents can be found at:

www.freescale.com

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

- U1 - Controller Board for 56F8300
 - Supplied as MC56F83xxEVM
 - Described in the **56F83xxEVMUM Evaluation Module Hardware User's Manual** for the specific device being implemented
- U2 - Legacy Motor Daughter Card (LMDC)
 - Supplies limited; please contact your Freescale representative
- U3 - 3-Phase SR High-Voltage Power Stage
 - Supplied as a kit with an Optoisolation Board as Freescale Part #ECOPHIVSR
 - Described in **Freescale's Embedded Motion Control 3-Phase SR High-Voltage Power Stage User's Manual**
- U4 - Optoisolation Board
 - Supplied in 3-phase SR High-Voltage Power Stage as Freescale Part #ECOPHIVSR
 - Or
 - Supplied separately as Freescale Part #ECOPT
 - Described in **Optoisolation Board User's Manual**
- MB1 Motor-Brake SR40V + SG40N

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

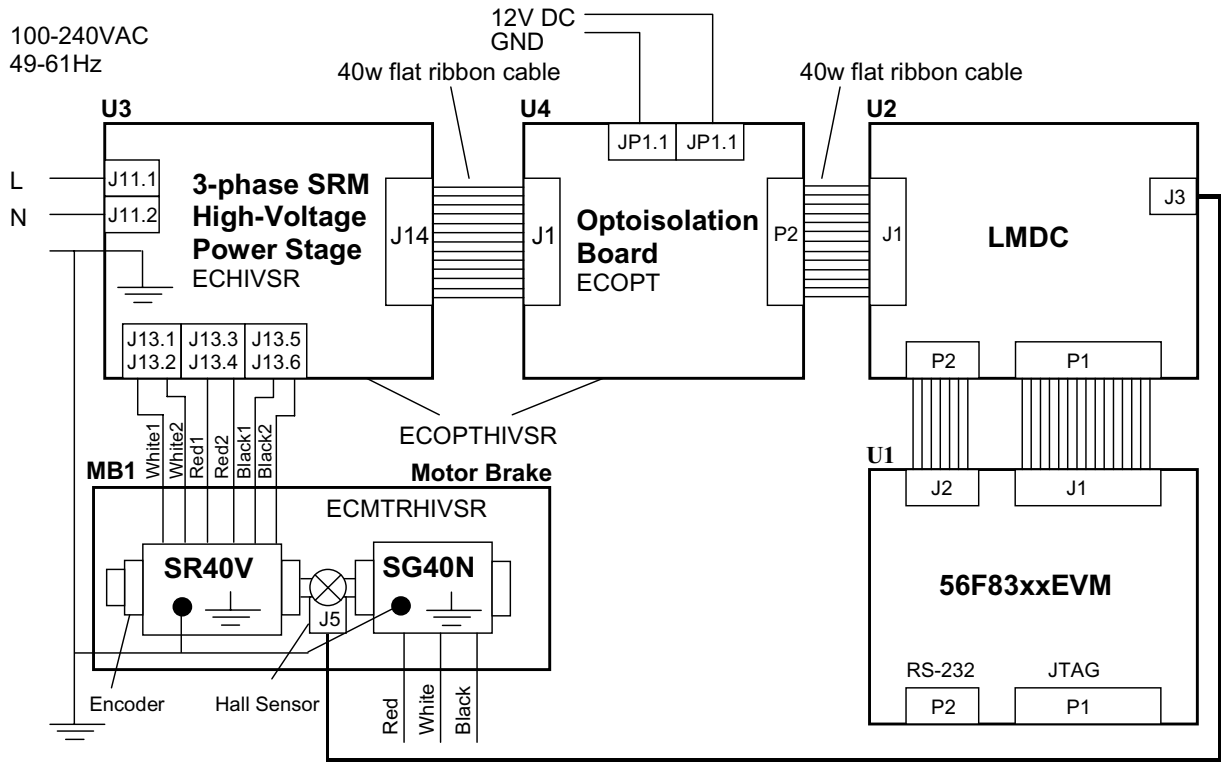


Figure 6-1 3-Phase SR High-Voltage Platform Configuration

6.2 Motor-Brake Specifications

The SR Motor Brake set incorporates a 3-Phase SR Motor and attached BLDC motor brake. Detailed specifications are shown in [Table 6-1](#).

Table 6-1 Motor - Brake Specifications

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

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°_{mech} , corresponds to one electrical period of the stroke. The profile presented was used for the determination of the reference flux linkage using the simulations.

On the motor brake shaft, a position encoder and position Hall sensor are attached. They allow position sensing if required by the control algorithm. The sensorless drive introduced does not use these sensors for the control algorithm. The encoder signals are only used for the evaluation of the sensorless technique.

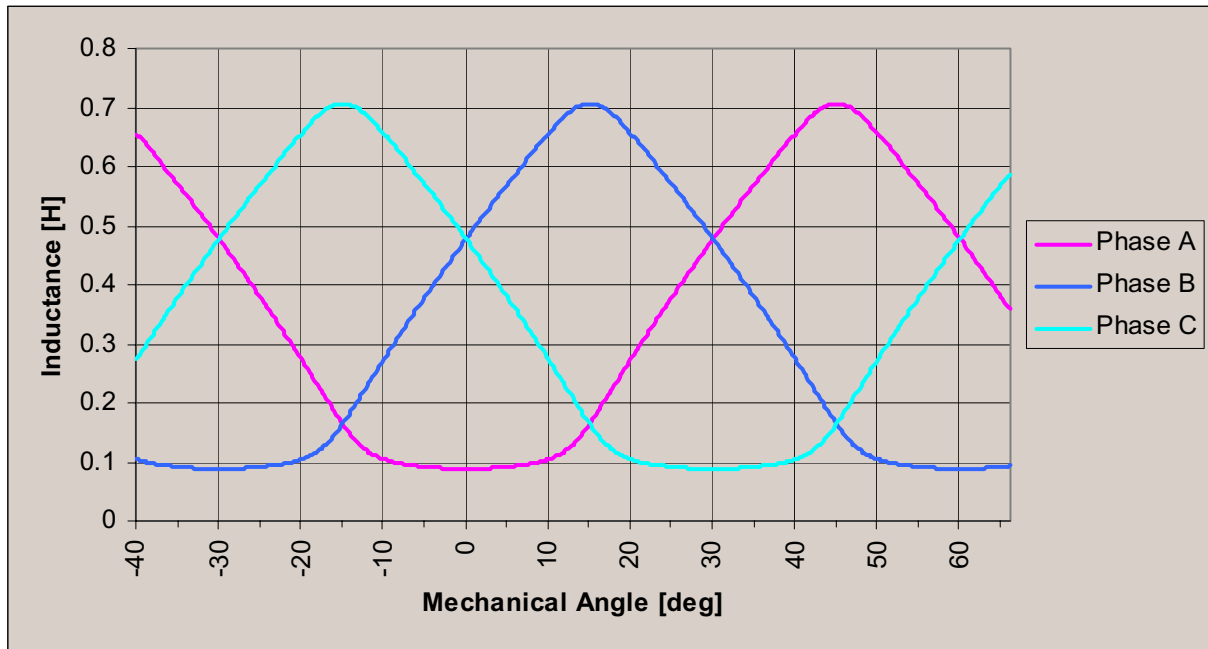


Figure 6-2 Inductance Characteristic

7. Software Design

This section explains the software design for targeting a 56F83xxEVM and 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.

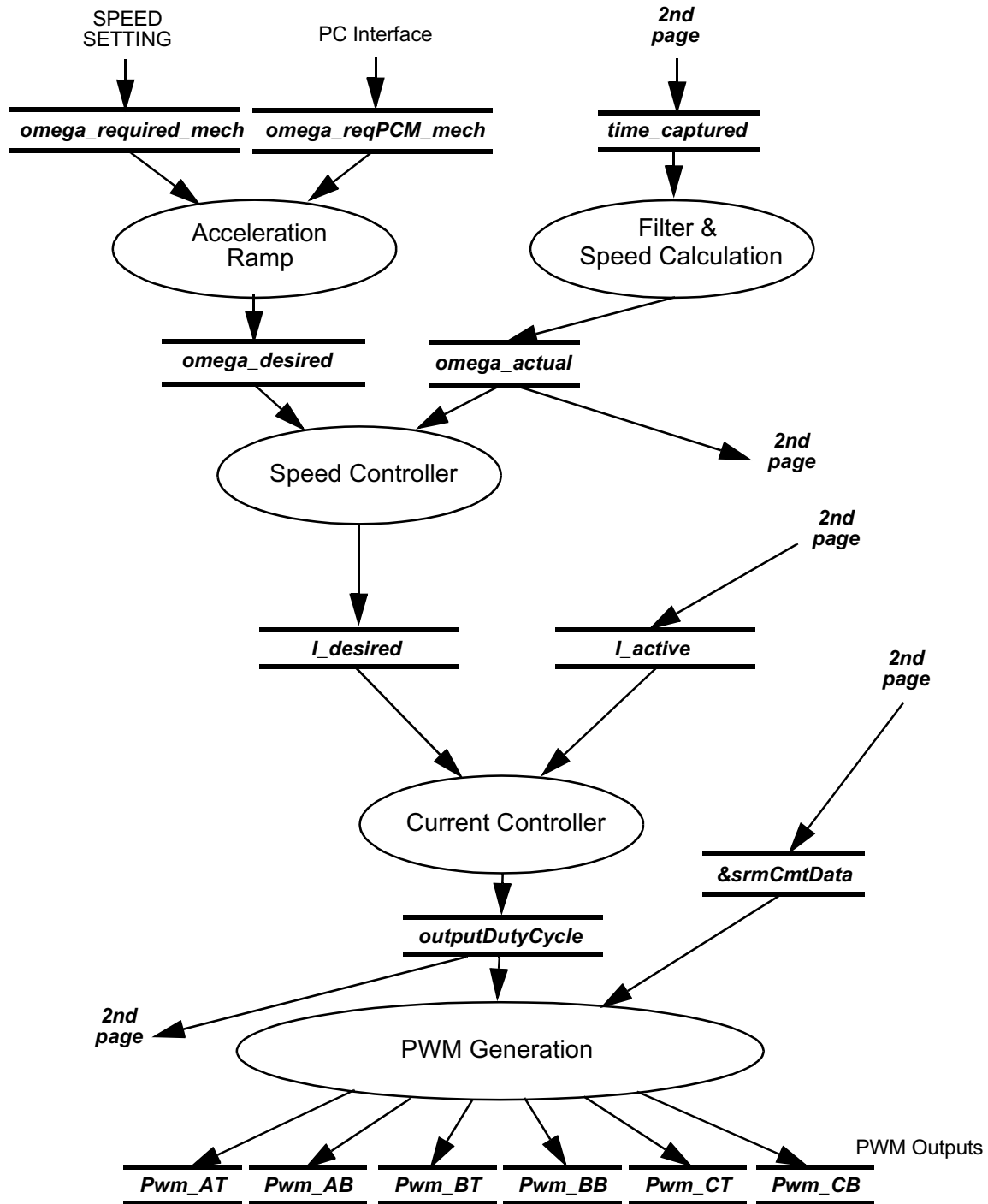


Figure 7-1 System Data Flow I - Speed & Current Control

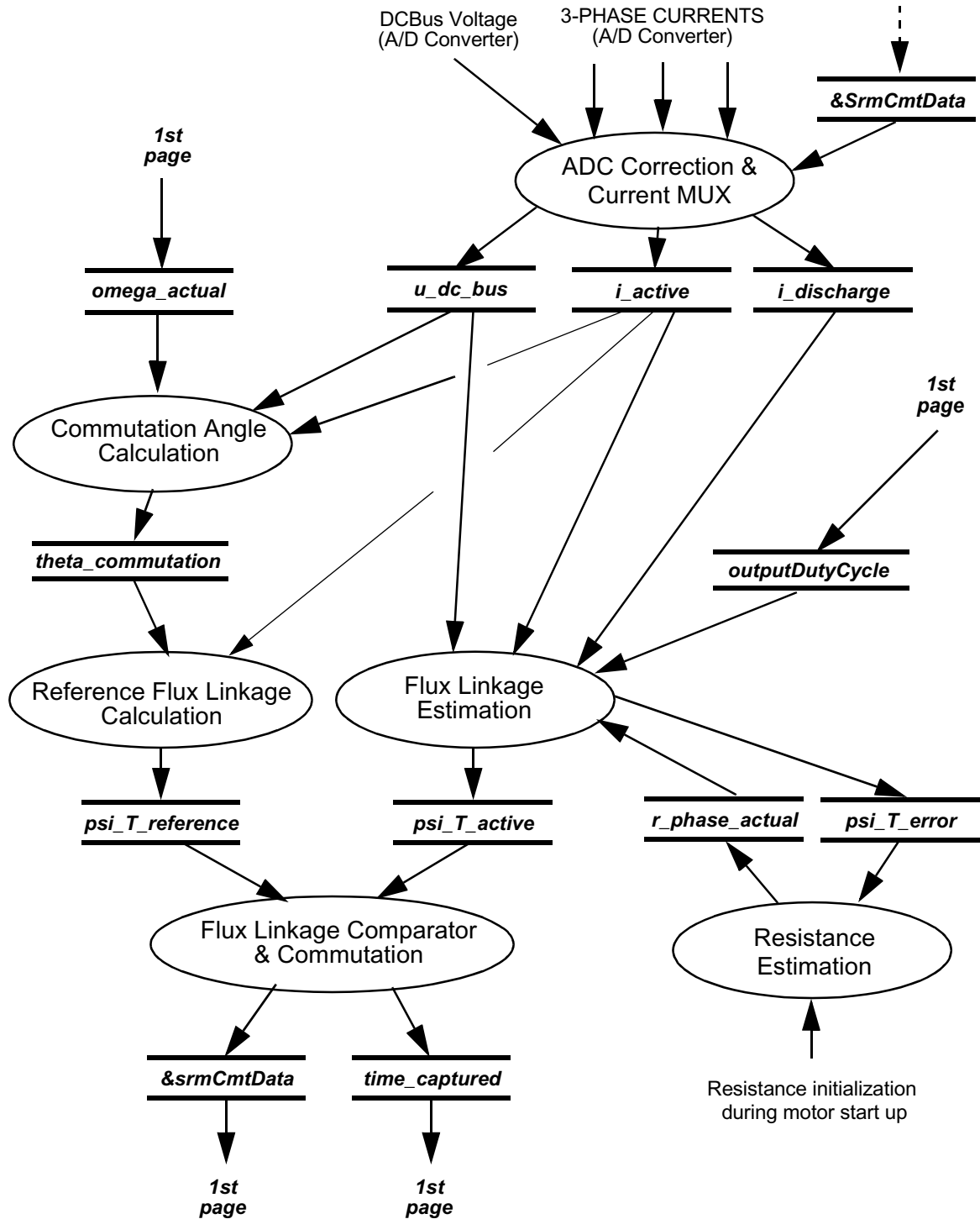


Figure 7-2 System Data Flow II - Commutation

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 operating mode), or by PC master software (when in PC master software operating mode).

7.1.2 Filter and Speed Calculation

The process calculates the motor's actual speed. The calculation is based on the evaluation of the time between the commutation instances.

Each time the commutation is performed, the actual time is captured. The process reads the time between the sequential commutation events and calculates the actual motor speed accordingly.

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) 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, which is the difference between the actual phase current and desired phase current. A PI controller is implemented. The current controller constants are tuned experimentally according to the type of motor used.

7.1.5 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 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.

7.1.6 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 DCBus voltage and temperature are filtered using a moving average filter; see [Section 5.2.4](#) for a detailed description.

7.1.7 Flux Linkage Estimation

This process calculates the actual flux linkage. The calculation of the active flux linkage is started with each commutation of phases. Flux linkage error is captured at the end of the current pulse and is further used for phase resistance estimation; see [Section 5.2.3](#) and [Section 4.2](#).

7.1.8 Commutation Angle Calculation

This process calculates the commutation angle according to the actual speed, the DCBus voltage and desired current (see [Section 5.2.3](#)).

7.1.9 Reference Flux Linkage Calculation

This process calculates the reference flux linkage according to the stored characteristic $\Psi(i_{phase})$ of the aligned position. The process requires the commutation angle and the actual phase current for determination of the reference flux linkage; see [Section 5.2.3](#).

7.1.10 Flux Linkage Comparator & Commutation

This process compares the reference flux linkage and the active flux linkage to determine commutation events. When the actual flux linkage exceeds the reference, a commutation is performed; see [Section 5.2.3](#). Also, the actual time is captured to be used for actual speed calculation.

The hybrid controller's 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 following cases:

- When the PWM output must be modulated, the PWM generator controls the channel directly
- When the PWM output must 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 must 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 Hardware 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 must 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 PE documentation.

7.1.11 Resistance Estimation

This process evaluates the flux linkage estimation error at the end of the phase current stroke and estimates the actual phase resistance; see [Section 5.2.3](#).

7.2 State Diagram

The previously described processes are implemented in a single state machine, as illustrated in [Figure 7-3](#). The state machine provides a transition among the application states INIT, STOP, RUN, and FAULT. The following variables are used to invoke the transition between the individual states:

- *switchState* (Stop, Run): state of the RUN / STOP switch
- *appFault* (NO_FAULT, any fault): a fault has occurred
- *appOpMode* (change from Manual to PC operating mode and vice versa): change operating mode

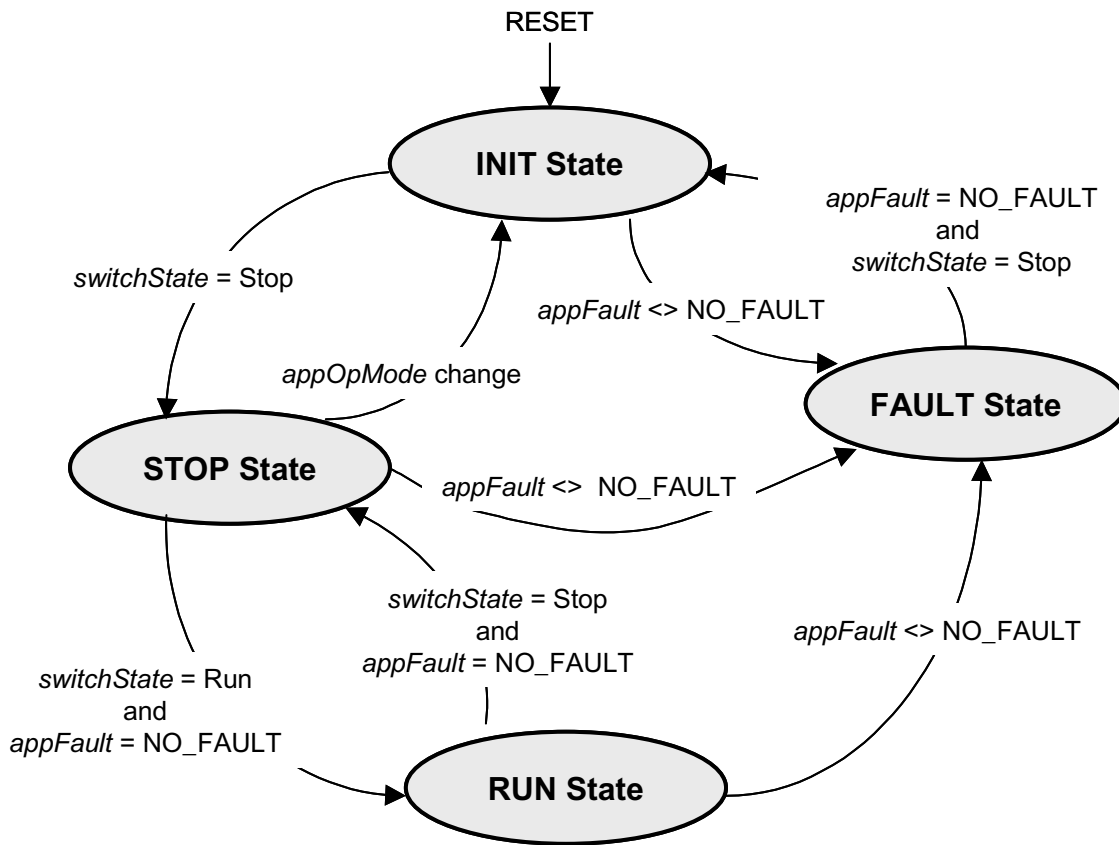


Figure 7-3 Application State Diagram

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 RUN / STOP switch is detected in the STOP position, the application transits to the STOP state, providing protection against a start after reset if the RUN / STOP switch is accidentally in the 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 operating mode can be changed, either from Manual mode to PC master software mode, or vice versa. When the operating 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, providing 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, providing 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 RUN / STOP switch is moved to the STOP position, and the application then 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 hybrid controller, and initialization of the application, then 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 and Resistance Estimator

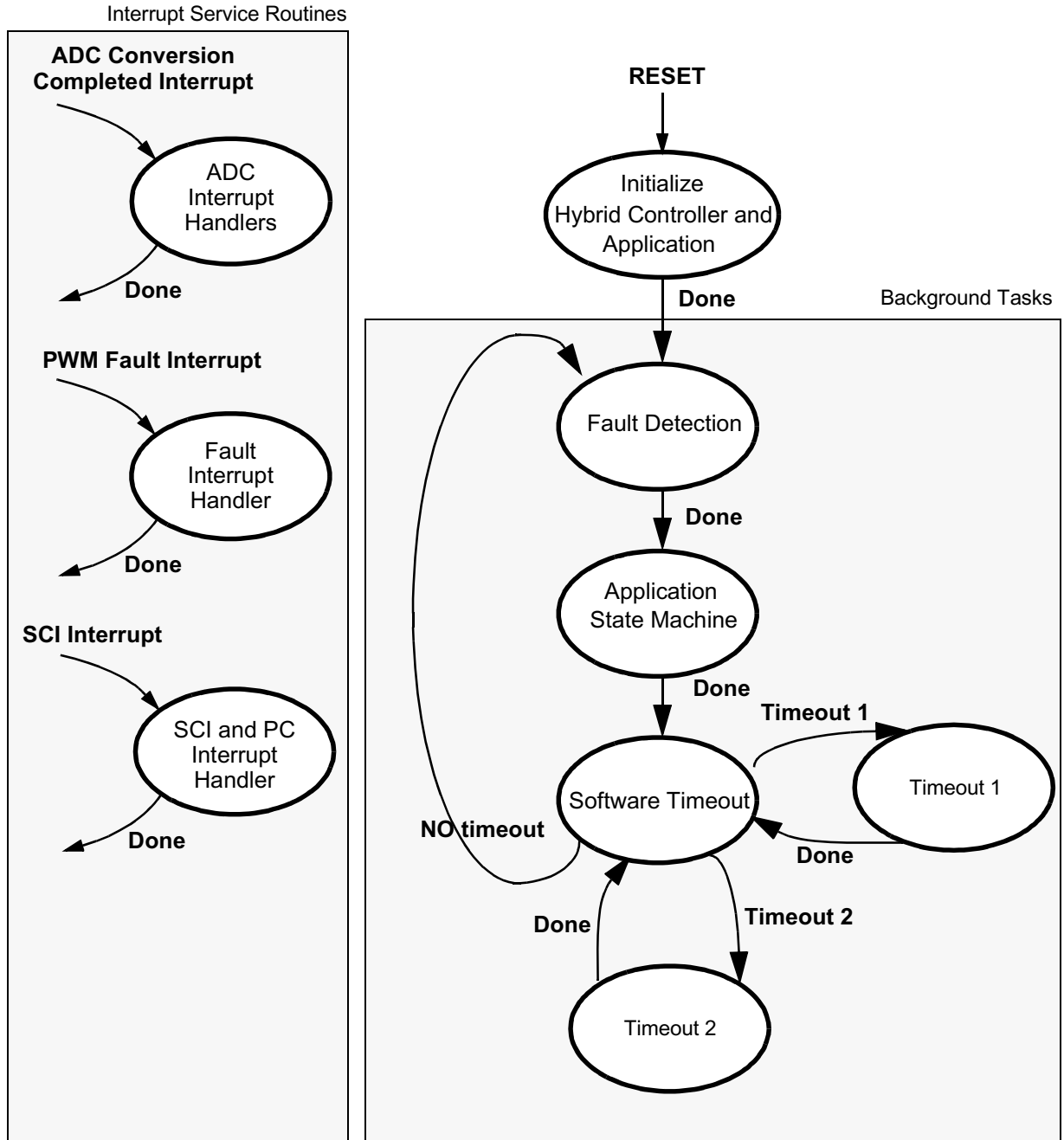


Figure 7-4 Software Design - General Overview

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

The following interrupt service routines are utilized:

- ADC Conversion Completed ISR services the ADC and provides all control tasks linked to the event; the ADC is synchronized with PWM pulses
- Fault ISR services faults invoked by external hardware fault
- SCI ISR services PC master software communication

7.3.1 Initialization

The initialization of the hybrid controller is performed after reset. At the beginning of initialization, interrupts are disabled; at the end of initialization, they are enabled.

Hybrid controller initialization:

- Disables interrupts
- Initializes PWM on-chip module:
 - Center-aligned independent PWM mode, positive polarity
 - Sets PWM modulus for PWM, frequency 16kHz
 - Sets PWM interrupt reload each PWM pulse
 - Sets FAULT2 (DCBus overcurrent fault) in Manual mode, interrupt enabled
 - Sets FAULT1 (DCBus overvoltage fault) in Manual mode, interrupt enabled
 - Associates interrupt with PWM Fault events
- Initializes ADC on-chip module
 - ADC triggered simultaneously
 - Associates interrupt with ADC conversion completed event
 - 1st sample of ADC_B (0-3): Current Phase A
 - 2nd sample of ADC_B (0-3): DCBus Voltage
 - 3rd sample of ADC_B (0-3): Temperature
 - 1st sample of ADC_B (4-7): Current Phase B
 - 2nd sample of ADC_B (4-7): Current Phase C
 - 3rd sample of ADC_B (4-7): void
- Initializes Quad Timer B0 on-chip module (speed measurement)
 - Counts up
 - Prescaler set to 128
- Initializes Quad Timer B1 on-chip module (position reference for visualization using PC master software)
 - Counts Quadrature Decoder input
 - Counts repeatedly up to 255
- Initializes Quadrature Decoder on-chip module (position reference for PC master software)
 - Sets digital filter for input signals
 - Connects Quadrature Decoder signals to Quad Timer B1
- Initializes LED driver (PWMB module is used for LED outputs)
- Initializes push buttons

Application initialization:

- Sets individual application parameters to their initial values
- Starts ADC conversion
- Measures offset of individual current sensors
- Measures DCBus voltage and temperature
- Calculates application parameters according to DCBus voltage
- Initializes Quad Timer C3 driver (ADC-PWM Synchronization)
 - Sets ADC synchronization delay to 0
 - EnablesQuad Timer C3 to be started on first SYNC
- ADC driver initialization
 - Sets ADC synchronization to ON
 - Enables 8-sample conversion
- Initializes all variables for motor start-up
- Sets ADC according to start-up phase
- Enablesinterrupts

7.3.2 Fault Detection

Fault Detection routinely checks for application faults. If a fault occurs, it disables the PWM outputs and sets the application FAULT status.

Note: If overcurrent and overvoltage faults occur, PWM outputs are directly disabled via internal PWM module fault protection; see [Section 7.3.7](#).

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 more information, see [Section 7.2](#).

7.3.4 Scheduler Timeout 1

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

- Push button filter debounces push button switching noise
- RUN / STOP switch filter debounces RUN / STOP switch noise
- Desired speed is calculated according to operating mode
 - In the Manual mode, according to the push buttons
 - In the PC master software control mode, according to the PC master command
- Start-up routine is performed if required and start-up switching pattern is generated. For a detailed description see [Section 5.2.2](#).
- Speed command is calculated using the acceleration / deceleration ramp with the desired speed set up
- LED is controlled according to the state of the drive and can indicate a STOP state, RUN state, or FAULT state

7.3.5 Scheduler Timeout 2

This state is accessed from the main scheduler at a period of Timeout 2 (2.5ms). 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.
- Resistance estimator estimates the phase resistance according to the flux linkage estimation error

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 must 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 DCBus voltage, and the temperature all at once. When the conversion is finalized, the ADC Completed ISR is called.

The routine provides the following services and calculations:

- Reads the time for speed calculation reference
- Reads the ADC conversion results:
 - Phase currents
 - DCBus voltage
 - Temperature
- Calculates the ADC offsets for the phase currents
- Calculates the reference and the actual flux linkage and determines commutation
- Current controller calculates the output duty cycle according to the desired and the actual phase currents
- Provides commutation when required
- Provides speed measurement
- Records selected recorder variables (PC master software)
- Loads PWM registers
- Calculates the references for software 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. If a DCBus overcurrent or a DCBus overvoltage fault is detected, the external hardware circuit generates a fault signal, detected on the Fault input pin of the hybrid controller. 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.

8. Implementation Notes

This section explains implementation notes for targeting a 56F83xxEVM.

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 the 1.[N-1] format (1 sign bit, N-1 fractional bits). Signed fractional numbers (SF) lie in the following range:

$$-1.0 \leq SF \leq +1.0 \cdot 2^{-[N-1]} \quad \text{EQ. 8-1}$$

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 $1.0 \cdot 2^{-15}$, and the most positive long-word is \$7FFFFFFF or $1.0 \cdot 2^{-31}$.

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:

<i>Fractional Value</i>	=	The fractional representation of the real value [Frac16]
<i>Real Value</i>	=	The real value of the quantity [V, A, rpm, etc.]
<i>Real quantity range</i>	=	The maximum range of the quantity, defined in the application [V, A, rpm, etc.]

8.1.1 Voltage Scaling

The application voltages are scaled to the maximum measured voltage. For DCBus voltage, the scaling equation is:

$$u_{dc_bus} = \frac{V_{DC_BUS}}{V_{MAX}} \quad \text{EQ. 8-3}$$

Where:

u_{dc_bus}	=	The scaled variable of the DCBus voltage [Frac16]
V_{DC_BUS}	=	The measured DCBus voltage [V]
V_{MAX}	=	The maximum measurable DCBus voltage [V]

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

The other application voltage variables are scaled in the same way (active phase voltage, u_{active} , discharge phase voltage, $u_{discharge}$, DCBus undervoltage limit, start-up voltage).

8.1.2 Phase Current Scaling

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

$$i_active = \frac{i_active}{i_phase_max} \quad \text{EQ. 8-4}$$

Where:

- i_active = The scaled variable of the active phase current [Frac16]
- i_active = The measured active phase current [A]
- i_phase_max = The maximum measurable phase current [A]

In the application, $i_phase_max = 5.86\text{A}$ 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 Phase Resistance Scaling

There is no general way to scale the resistance. In this application, the phase resistance was scaled according to the scaling of the measured voltage and the phase current in order to decrease the calculation requirements. The scaling equation for the actual phase resistance is:

$$r_phase_actual = \frac{R_phase_actual}{\frac{u_MAX}{i_phase_max}} \quad \text{EQ. 8-5}$$

Where:

- r_phase_actual = The scaled variable of the actual phase resistance [Frac16]
- R_phase_actual = The measured actual phase resistance [Ω]
- u_MAX = The maximum measurable DCBus voltage [V]
- i_phase_max = The maximum measurable phase current [A]

In the application, $u_MAX/i_phase_max = 407\text{V}/5.86\text{A} = 69.4\Omega$.

The other application resistance variables are scaled the same way (resistance sample, r_phase_sample).

8.1.4 Phase Inductance Scaling

There is no general way to scale the inductance. In order to decrease the calculation requirements, the phase inductance in the application was scaled according to the scaling of the measured voltage and the phase current. The scaling equation for unaligned phase inductance is:

$$L_unaligned = \frac{L_{unaligned}}{\frac{u_{MAX}}{i_{phase_max}}} \quad \text{EQ. 8-6}$$

Where:

$L_unaligned$	=	The scaled variable of the unaligned phase inductance [Frac16]
$L_{unaligned}$	=	The unaligned phase inductance [H]
u_{MAX}	=	The maximum measurable DCBus voltage [V]
i_{phase_max}	=	The maximum measurable phase current [A]

In the application, $u_{MAX}/i_{phase_max} = 407\text{V}/5.86\text{A} = 69.4\text{V/A}$.

8.1.5 Flux Linkage Scaling

The application phase linkage is calculated as a flux linkage divided by a sampling period T [EQ. 4-5](#). The 16-bit phase flux increments ($u_k - r_k * i_k$) are summed to the 32-bit flux linkage sum variable (Ψ_N/T). The integration output can overflow if more than 65,536 samples are calculated. The sampling period T is defined by the PWM frequency of 16kHz. In the application: $T = 1/16000 = 62.5 * 10^{-6}$ seconds.

The 32-bit flux linkage, $psi_T_active_sum$, is further scaled to the 16-bit variable, psi_T_active .

$$psi_T_active = psi_T_active_sum \cdot 256 \quad \text{EQ. 8-7}$$

Where:

psi_T_active	=	The scaled variable of the active flux linkage [Frac16]
$psi_T_active_sum$	=	The scaled variable of the active flux linkage sum [Frac32]

The other application 16-bit flux linkage variables are scaled in the same way (flux linkage error, psi_T_error , reference flux linkage, $psi_T_reference$, delta flux linkage, psi_T_delta).

8.1.6 Electrical Angle Scaling

The application's 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:

$$theta_commutation_el = \frac{\vartheta_{commutation_el}}{180^\circ} \quad \text{EQ. 8-8}$$

Where:

$\theta_{commutation_el}$ = The scaled variable of the electrical commutation angle [Frac16]

$\vartheta_{commutation_el}$ = The desired commutation angle [$^{\circ}_{el}$]

In the application, $\vartheta_{aligned_el} = 180^{\circ}_{el}$

The other application electrical angle variables are scaled in the same way (delta theta required for phase current to reach the desired current, θ_{delta_el} , theta where stator and rotor poles start to overlap, $\theta_{start_to_overlap_el}$).

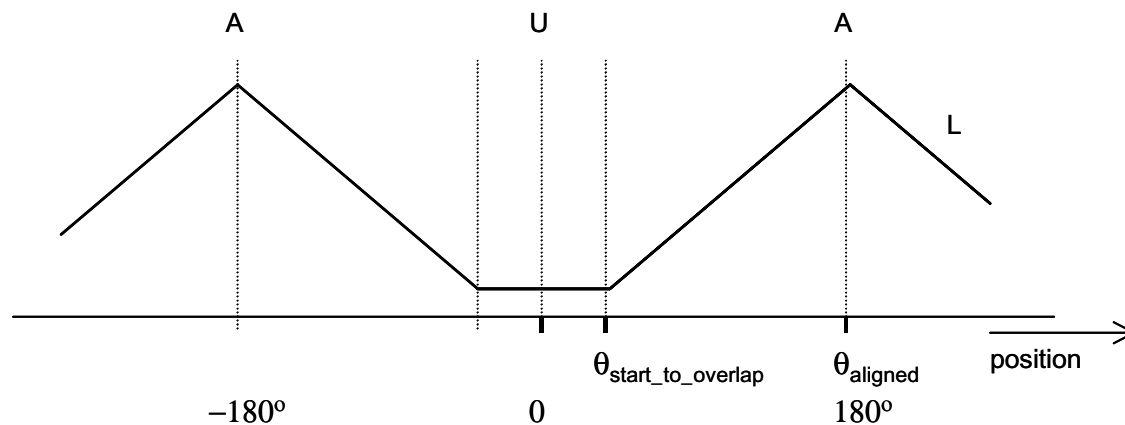


Figure 8-1 Electrical Angle Definition

8.1.7 Speed Scaling

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

$$\omega_{desired_startup} = \frac{\omega_{start_up}}{\omega_{MAX}} \quad \text{EQ. 8-9}$$

Where:

$\omega_{desired_startup}$ = The scaled variable of the desired start-up speed [Frac16]

ω_{start_up} = The desired start-up speed [rpm]

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

In the application, $\omega_{MAX} = 3000\text{rpm}$.

The other application speed variables are scaled in the same way (actual speed, ω_{actual_mech} , speed limits, ω_{reqMAX_mech} and ω_{reqMIN_mech} , push button speed increment, $\omega_{increment_pb}$).

8.1.8 Duty Cycle Scaling

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

$$output_duty_cycle = \frac{duty_cycle_{output}}{duty_cycle_{MAX}} \quad \text{EQ. 8-10}$$

Where:

- $output_duty_cycle$ = The scaled variable of output duty cycle [Frac16]
- $duty_cycle_{output}$ = The desired output duty cycle [%]
- $duty_cycle_{MAX}$ = The maximum 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 Quad Timer between the two following edges of the position Hall sensors. The actual speed, $OmegaActual$, is calculated according to the following equation:

$$OmegaActual = \frac{SpeedCalcConst}{TimeCaptured} \quad \text{EQ. 8-11}$$

Where:

- $OmegaActual$ = The actual speed [rpm]
- $TimeCaptured$ = The time, in terms of number of timer pulses, captured between two edges of the position sensor [-]
- $SpeedCalcConst$ = 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:

$$SpeedCalcConst = 2^{15} \times \frac{SpeedMin}{SpeedMax} \quad \text{EQ. 8-12}$$

Where:

- $SpeedMin$ = The minimum measured speed [rpm]
- $SpeedMax$ = The maximum measured speed [rpm]

Minimum measured speed, $SpeedMin$, is given by the configuration of the sensors and parameters of the hybrid controller on-chip timer used for speed measurement. It is calculated as:

$$SpeedMin = \frac{\frac{1}{NoPulsesPerRev} \times 60}{\frac{2^{15}}{BusClockFreq} \times Presc} \quad \text{EQ. 8-13}$$

Where:

- $NoPulsesPerRev$ = The number of sensed pulses of the position sensor per single revolution [-]
- $Presc$ = The prescaler of the Quad Timer used for speed measurements
- $BusClockFreq$ = The hybrid controller Bus Clock Frequency [Hz]

Maximum measured speed, $SpeedMax$, is selected as:

$$SpeedMax = k \times SpeedMin \quad \text{EQ. 8-14}$$

Where:

- k = An integer constant greater than 1

The speed calculation constant is then determined as:

$$SpeedCalcConst = BusClockFreq \times \frac{60}{NoPulsesPerRev \times Presc \times SpeedMax} \quad \text{EQ. 8-15}$$

In the application:

- $NoPulsesPerRev$ = 12 Hall sensor pulses per 1 revolution of the motor
- $Presc$ = 128
- $BusClockFreq$ = 30×10^6 Hz
- $SpeedMax$ = 3000rpm

In this case, $SpeedCalcConst = 390$ [rev^{-1}]

9. Processor Expert (PE) Implementation

PE is a collection of beans, APIs, libraries, services, rules and guidelines. This software infrastructure is designed to let 56F80x and 56F8300 software developers create high-level, efficient, and portable code. The application code is available in PE, and this chapter describes how the SR motor control application is written under PE.

9.1 Beans and Library Functions

The sensorless SR motor control application uses the following beans:

- ADC bean
- Quad Timer bean
- Quadrature Decoder bean
- PWM bean
- PC master software driver
- GPIO bean

The SR motor control application uses the following library functions:

- *srmcmt3ph2sppSoftSw* (SR motor commutation algorithm; *MC_SrmCommutation* bean)
- *srmcmt3ph2sppPhOff* (SR motor commutation algorithm; *MC_SrmCommutation* bean)
- *srmcmt3ph2sppInit* (SR motor init algorithm; *MC_SrmCommutation* bean)
- *controllerPItype1* (standard PI controller, *MC_PIController* bean)
- *phasefluxestCalc* (phase flux linkage estimation, *MC_PhaseFluxEst* bean)
- *phasefluxestInit* (phase flux linkage estimation, *MC_PhaseFluxEst* bean)
- *rampGetValue* (ramp generation, *MC_Ramp* bean)

9.2 Initialization of Beans

Each peripheral on the hybrid controller chip or on the EVM board is accessible through a bean. The bean initialization of all peripherals used is described in this section. For a more-detailed description of drivers, see the **Targeting 56F8300 Demonstration Board** manual.

To use a bean, following these steps:

- Add the required bean:
 - Right click *Beans* under the Processor Expert tab in project window
 - Select *Add Beans*, which will open the *Bean Selector* window of PE
 - Select the desired bean
- Configure the added bean
- Call the bean's *init* function, or use PE initialization, by selecting *Call init* in the CPU *init* code

Access to individual driver functions is provided from *PESL* support by the *ioctl* or *PESL* function call. To enable access to these functions, *PESL support* should be enabled in the *CPU bean* used.

9.3 Interrupts

When configuring a bean in PE, the user defines the callback functions called during interrupts.

9.4 PC Master Software

PC master software was designed to provide a debugging, diagnostic and demonstration tool for development of algorithms and applications. It consists of components running on PCs and components running on the target hybrid controller, connected by an RS-232 serial port. A small program is resident in the hybrid controller that communicates with the 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 Microsoft Internet Explorer as a user interface to the PC.

To enable the PC master software operation on the hybrid controller target board application, add the *PC_Master* bean to the application. The *PC_Master* bean is located under *CPU External Devices -> Display* in PE's *Bean Selector*.

The PC master bean automatically includes the SCI driver and installs all necessary services. This means there is no need to install the SCI driver, because the *PC_Master* bean encapsulates its own SCI driver.

The default baud rate of the SCI communication is 9600 and is set automatically by the PC master software driver.

A detailed PC master software description is provided in PE documentation.

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

- Take control over the PC master software
- Control the motor's start / stop
- Set motor speed

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

- Required and actual motor speeds
- Application operating mode
- Start / stop status
- Drive fault status
- Power stage boards identified
- Voltage level identified
- 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 or the flux linkage 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 PE's *PC_Master* bean of the *ExtRAM* target. Recorder samples are taken every 64.5 μ s, at the rate of the PWM frequency.

The following records can be captured:

The **Start-up Recorder** captures:

- Desired phase current
- Active phase current
- Reference flux linkage
- Active flux linkage
- Output duty cycle
- Encoder position reference

The Start-up Recorder is initiated only when the motor starts.

The **Flux Linkage Recorder** captures:

- Active phase current
- Discharge phase current
- Active flux linkage
- Discharge flux linkage
- Reference flux linkage
- Encoder position reference

The Flux Linkage Recorder may be initiated any time while the motor is running.

The **Current Controller Recorder** captures:

- Desired phase Ccurrent
- Active phase current
- Output duty cycle
- Encoder position reference

The Current Controller Recorder may be initiated any time while the motor is running.

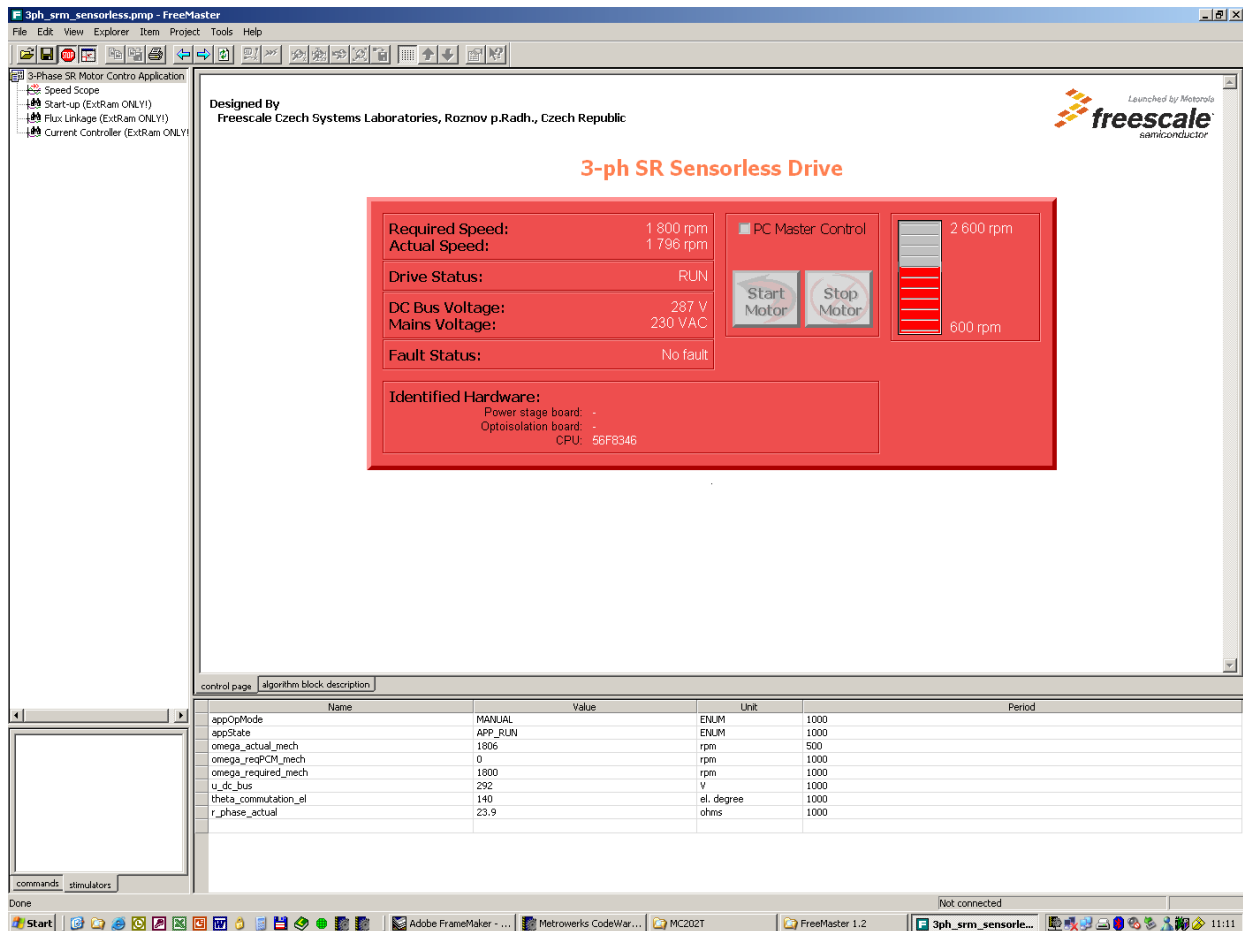


Figure 9-1 PC Control Window

10. Hybrid Controller Use

Table 10-1 shows how much memory is needed to run the 3-Phase SR drive sensorless drive based on the flux linkage estimation algorithm. The PC master software recorder buffer is set to 2K words and the bulk of the hybrid controller's memory is still available for other tasks.

Table 10-1 RAM and FLASH Memory Usage for PE 2.94 and CodeWarrior 6.1.2

Memory (in 16-bit Words)	Available for 56F8300 Hybrid Controllers	Used Application + Stack	Used Application without PC Master software, SCI
Program Flash	64K	8970	4420
Data Flash	4K	21	8
Program RAM	2K	0	0
Data RAM	4K	2600 + 512 stack	416 + 512 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] Chalupa, L., Visinka, R., On-Fly Phase Resistance Estimation of Switched Reluctance Motor for Sensorless based Control Techniques, *Conference Power Conversion and Intelligent Motion*, Nurnberg, PCIM, 2000
- [3] Freescale, *Apparatus and Method for Estimating the Coil Resistance in an Electric Motor*, Chalupa, L., Visinka, R., *US Patent*, 6,366,865, 2002-04-02
- [4] Gallegos-Lopez, G., A New Sensorless Low-Cost Method for Switched Reluctance Motor Drives, *University of Glasgow - SPEED Laboratory*, 1997
- [5] Miller, T.J.E., *Switched Reluctance Motors and Their Control*, Magna Physics Publishing and Clarendon Press, ISBN 0-19-859387-2, 1993
- [6] Lyons, J.P., MacMinn, S.R., Preston, Flux / Current Methods for SRM Rotor Position Estimation, *Proc. IEEE-IAS'91*, 1991
- [7] *3-Phase SR Motor Control with Hall Sensors using DSP56F80x*, AN1912, Freescale Semiconductor, Inc.
- [8] *CodeWarrior for Freescale DSP56800 Embedded Systems*, CWDSP56800, Metrowerks
- [9] *56800 Family Manual*, DSP56F800FM, Freescale Semiconductor, Inc.
- [10] *DSP56F800 User Manual*, DSP56F801-7UM, Freescale Semiconductor, Inc.
- [11] *56F8300 Peripheral User Manual*, MC56F8300UM, Freescale Semiconductor, Inc.
- [12] Targeting 56F8300 Demonstration Board, MC56F8300TUM, Freescale Semiconductor, Inc.
- [13] *56F805 Evaluation Module Hardware User's Manual*, DSP56F805EVMUM, Freescale Semiconductor, Inc.

- [14] *56F83xx Evaluation Module Hardware User's Manual* for the specific device being implemented, MC56F83xxEVMUM, Freescale Semiconductor, Inc.
- [15] *Freescale Embedded Motion Optoisolation Board User's Manual*, MEMCILOBUM, Freescale Semiconductor, Inc.
- [16] *Freescale Embedded Motion Control 3-Phase Switched Reluctance High-Voltage Power Stage User's Manual*, MEMC3PSRHVPSUM, Freescale Semiconductor, Inc.
- [17] *Freescale Embedded Motion Control 3-Phase Switched Reluctance Low-Voltage Power Stage User's Manual*, MEMC3PSRLVPSUM, Freescale Semiconductor, Inc.
- [18] User Manual for PC Master Software, included in Processor Expert documentation
- [19] Freescale SPS web page:
www.freescale.com

How to Reach Us:

Home Page:

www.freescale.com

E-mail:

support@freescale.com

USA/Europe or Locations Not Listed:

Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:

Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. This product incorporates SuperFlash® technology licensed from SST.

© Freescale Semiconductor, Inc. 2004. All rights reserved.