# A Complete Implementation Procedure for State Estimation in Induction Machines on the eZdsp F2812

Ali M. Bazzi and Philip T. Krein

Grainger Center for Electric Machinery and Electromechanics

Department of Electrical and Computer Engineering

University of Illinois at Urbana-Champaign, Urbana, Illinois, USA

## Motivation

In the last three decades, the trend in commercial induction motor drives has been to implement field-oriented or vector control (FOC) and direct torque control (DTC) for closed-loop control. While open-loop or V/Hz control is still used in a wide range of applications, e.g. air conditioning and pumping, more efficient and robust operation can be achieved with closed-loop drives. Closed-loop drives have superior dynamic performance, and allow for the implementation of energy-saving techniques. Most closed-loop drives require feedback of variables that are either unavailable or expensive to measure. Reliability of the drive is also an important factor when considering feedback: sensors add to the possible points of failure; therefore there has been significant research on “sensorless” control. In reality, it is impossible to achieve a completely “sensorless” closed-loop drive, i.e. having no voltage, current, or speed information. Engineers try to avoid the cost and failures of speed encoders, which initiated research for several speed-sensorless control schemes. Another important variable in vector control is the machine magnetic flux, but its measurement is complex. When closed-loop torque-control is desired, knowledge of the machine electromechanical torque is required, but torque sensors are expensive. Therefore flux, speed, and torque estimators or observers are used to replace expensive and less-reliable sensors.

Literature on machines, drives, power electronics, industry applications, and energy conversion is rich with flux, torque, and speed estimation techniques, but lacks comprehensive implementation procedures. These techniques provide a suitable replacement for the sensors on a more reliable platform: digital circuits rather than mechanical transducers. Estimators in motor drives can be categorized into three main groups: back electro-motive force (EMF) methods, model reference adaptive systems (MRAS), and observer-based approaches such as Kalman filters, Luenberger observers, sliding-mode observers, and nonlinear observers. Such estimators differ in terms of estimation errors, dependence on motor parameters, and settling time.

This article presents a high-level procedure for implementing estimators on a digital signal processor (DSP) in induction machine applications. The procedure provides basic building blocks for digital control and estimation in motor drives. The simplest form of back-EMF estimation is used here to demonstrate the procedure, but the choice of a specific estimator is up to the designer.

## A Typical Closed-Loop Induction Motor Drive

The block diagram of a typical induction motor drive is shown in Fig. 1 An induction machine is fed by a three-phase inverter from a dc bus. To achieve the desired torque-speed response, the control and estimation algorithms use information from sensors. The speed measurement, labeled as $\omega_m$, is usually available from a speed encoder. Even though flux measurement can be available from Hall effect sensors in some applications, flux is usually estimated for cost and reliability reasons, and is not shown in Fig. 1. Current ($i_{abc}$) and voltage ($v_{abc}$) measurements are usually available, and are used in the flux estimation process. The mechanical load on the machine shaft could be a fan, propeller, vehicle gearbox, etc.

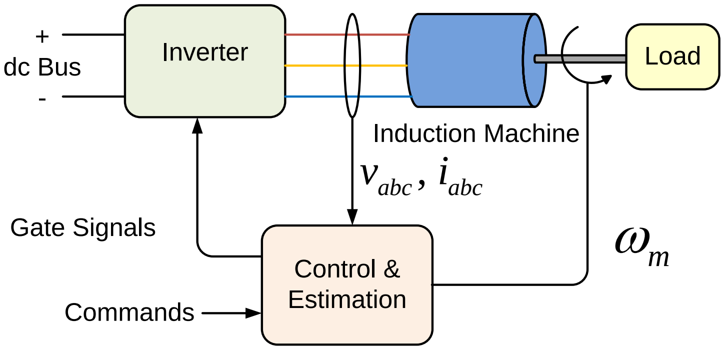

Fig. 1. Typical induction motor drive

Among the most important parts of the control and estimation process is the implementation platform. The decreasing price of powerful DSPs makes them suitable for motor control and estimation applications. Other options are microcontrollers, Field Programmable Gate Arrays (FPGAs) and

Complex Programmable Logic Devices (CPLDs). The choice of DSPs is more natural, as many have built-in pulse-width modulation (PWM) channels, analog-to-digital converters (ADCs), and even support speed encoder inputs. A natural companion to any control and estimation platform is an interface board that links this platform to the rest of the system. Such a board is essential when signals into and out of the platform are at power or voltage levels incompatible with the rest of the system. This board can also provide electrical isolation between the platform and high-power components, conditioning of sensor outputs, and amplification of the DSP outputs. Another essential subsystem is the three-phase inverter, which provides the machine with variable-frequency variable-amplitude three-phase voltages. In a laboratory or testing environment, the load is usually applied through a dynamometer which can be programmed for a specific torque-speed profile. The commands and monitoring can be available through a GUI, where the computer communicates with the DSP and the load simultaneously. An elaborate version of Fig. 1 is shown in Fig. 2 and shows more details.

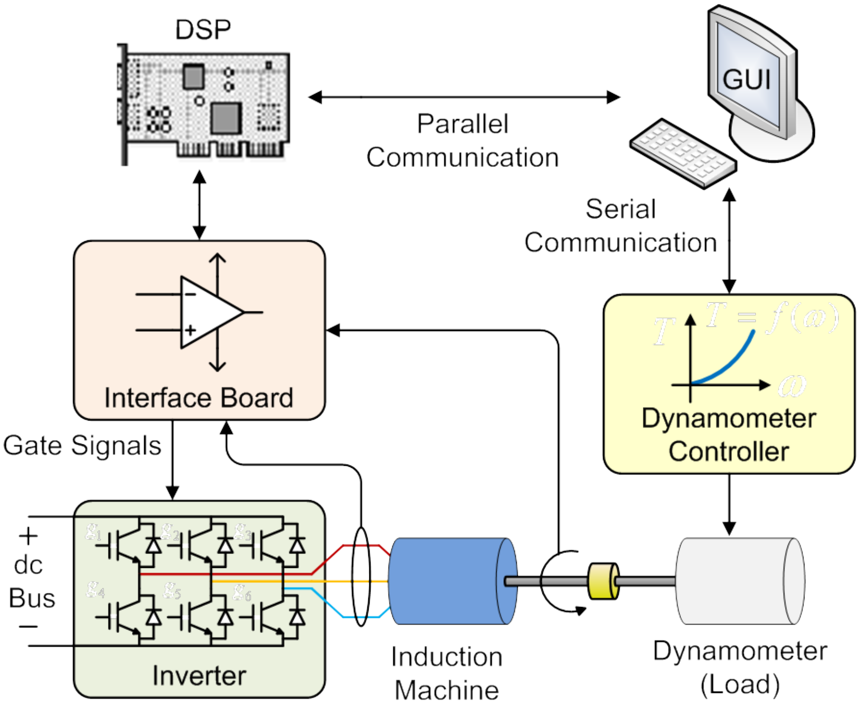

Fig. 2. Detailed laboratory setup

Fig. 2 shows that there are several important steps when building an induction machine drive for testing the control and estimation. These steps are summarized in Fig. 3, where the GUI is built in MATLAB/Simulink, and the DSP is programmed in Code Composer Studio (CCS). Simulink provides a user-friendly control and estimation interface where the designer can use signal-flow block diagrams similar to a simulation. The block diagrams built can then be automatically translated to C-code that can be compiled in CCS. It is essential that discrete-time blocks with fixed sampling rates and fixed-point math be used in the block diagram, although floating-point DSPs are currently available. The testing and calibration are first done with no load, then under different loads for further tuning and calibration.

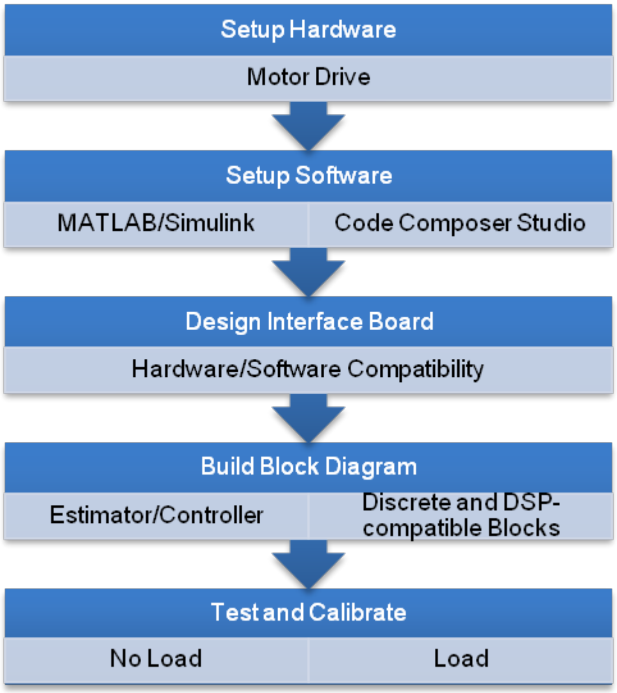

Fig. 3. Implementation procedure of the control and estimation

## Estimators

In the example presented here, the control is indirect FOC (IFOC), and the load is a fan. IFOC requires two inputs: the torque or speed command, and the d-axis rotor flux or d-axis stator current. IFOC is distinguished from direct FOC (DFOC) by estimating the rotor flux rather than measuring it.

Therefore, a flux estimation loop is usually implemented. A simple back-EMF estimator is implemented in the stator reference frame as shown in (1) and (2) where $L_m$, $L_r$, and $L_s$ are the magnetizing, rotor, and stator inductances, respectively; $R_s$ is the stator resistance, $v$, $i$, and $\lambda$ are the voltage, current, and flux linkage; $q$ and $d$ subscripts are quadrature and direct axes; $s$ and $r$ are stator and rotor terms; and $\sigma$ is the leakage factor.

$$\lambda_{qd}^{s} = \int \left( v_{qd}^{s} - R_s i_{qd}^{s} \right) dt \tag{1}$$

$$\lambda_{qd}^{r} = \frac{L_r}{L_m} \left( \lambda_{qd}^{s} - \sigma L_s i_{qd}^{s} \right) \tag{2}$$

Flux estimation is an essential requirement for torque and speed estimation. When a flux estimate is available, a torque estimate ($T_e$) can be generated using (3), and the rotor angle ($\rho$) used in speed estimation can be found as shown in (4). The input power ($P_{in}$) can also be estimated from voltage and current measurements as shown in (5). It is important to note that back-EMF estimators can have problems at low speeds, and better estimators are available for high-performance drives.

$$T_e = \frac{3P}{4} \left( \lambda_d^s i_q^s - \lambda_q^s i_d^s \right) \tag{3}$$

$$\rho = \tan^{-1} \left( \frac{\lambda_q^r}{\lambda_d^r} \right) \tag{4}$$

$$P_{in} = v_a i_a + v_b i_b + v_c i_c \tag{5}$$

The implementation of (1) – (5) on a DSP is not straightforward even though the equations look simple. For example, any dc offset in the measurements of $v$ or $i$ could cause the integral to grow indefinitely, but on a DSP, this would saturate a register and give erroneous results. One of the suggested solutions is to use a low-pass filter (LPF) instead of an integrator. But an LPF will cause both magnitude and phase errors that need compensation. Therefore, appropriate scaling and filtering of all measured signals is essential, and even though the interface stage could help reduce noise and manage offsets, more digital filtering and scaling is required.

## Laboratory Setup

As seen in (1)–(4), a fundamental component is flux estimate. The work presented here employs an eZdsp F2812™ board as the control and estimation platform. This board is built around the TMS320F2812 DSP. This platform is compatible with Simulink®, and includes six dual pulse PWM channels (12 channels total), 16 ADCs, and a speed encoder input. The processor is a 32-bit DSP with fixed-point arithmetic; thus, discrete and fixed-point math blocks of Simulink can be used in the block diagrams. Once programmed, the DSP can run independent from Simulink, but the link is maintained through parallel communication for an interactive GUI. The GUI allows the user to place speed and flux commands, and monitor estimates in real time. For this platform, two primary software packages are available on the host computer where the development and control take place: MATLAB/Simulink, which support math and control development, and CCS, which supports detailed code development for the DSP. Compatibility is essential. For example, MATLAB 7.0.4 must be used with CCS 2.21, MATLAB 2006a must be used with CCS 3.1, etc.

Simulink provides a simple user interface where a designer can build the estimator using discrete-time blocks and special DSP-related blocks from the “C2000” library, such as the fixed-point math “C28x IQmath” library, which are compatible with the 28x family of DSPs. Simulink compiles a block diagram into C code and then calls CCS to generate the assembly code for the DSP. A project is generated in CCS to be loaded into the DSP. Note that programmers with experience in C or assembly languages can write their own optimized code for the DSP, but this is a time-consuming process. MATLAB is used to build the GUI for real-time communication with the DSP using real-time data exchange (RTDX) channels. These channels are set in the block diagram. Fig. 4 shows a high-level block diagram where the ADC is accessed through the “ADC” Simulink block, and RTDX channels are categorized as “Read” or “Write.” “Read” channels fetch data being processed on the DSP, and “Write” channels send commands to the DSP. PWM channels send gate signals to the switches in the three-phase inverter, and can be used as access points to monitor signals on an oscilloscope or logic analyzer.

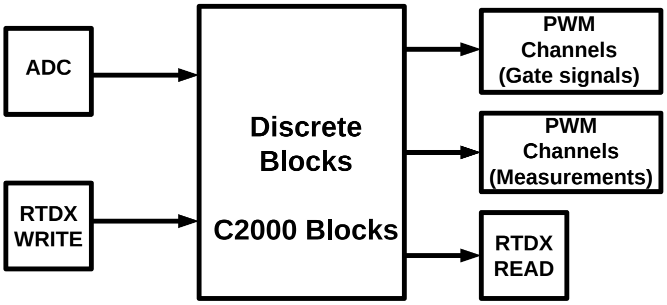

Fig. 4. Typical block diagram for control and estimation

On the hardware side, current and voltage sensors are usually built in the inverter. The interface board is used to amplify signals sent from the DSP to the inverter, and to filter and scale signals sent from the sensors to the DSP. For example, the eZdspF2812 requires all ADC inputs to be between 0 and 3 V. While simple voltage dividers with limited currents are straightforward, many current sensors have dc offsets and nonlinear input/output relations. Fig. 5 shows a sample current-sensor characteristic (voltage output versus actual current sensed), and the expected output of the interface board fed into the DSP. Op-amps are usually used on the interface board, but additional digital filtering and scaling is also required.

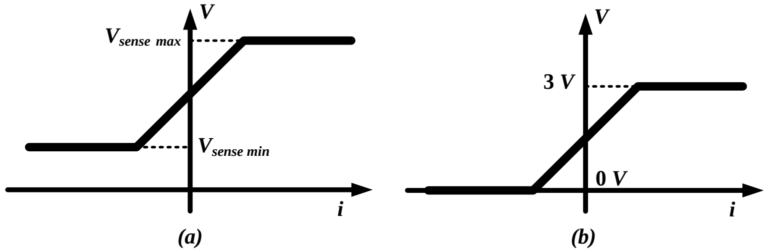

Fig. 5. (a) Sensor output (b) Scaled sensor output compatible with the DSP

After the sensors are scaled and conditioned for the ADC, the designer can read sensor and estimation information in Simulink. Since sensor outputs were manipulated as shown in Fig. 5, reverse scaling is required to find the real values. An example is shown in Fig. 6, where the ADC output is a 12-bit fixed-point integer with full scale of 4095 at an actual current of 20 A.

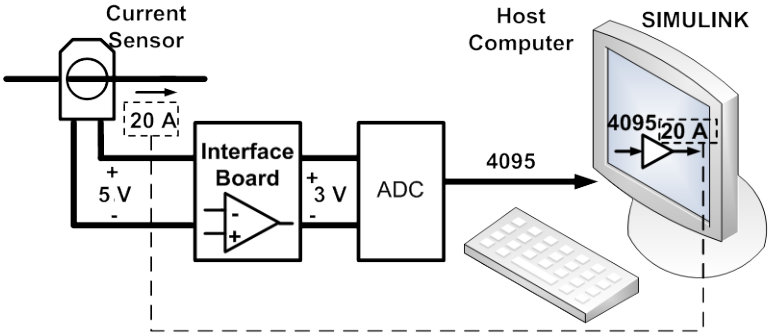

Fig. 6. Scaling and reverse scaling of sensor outputs

Building the control and estimation can start after the hardware is set up, ADC information is accessed and scaled, and PWM control channels are set. The focus here is on estimation, but building the controller follows a similar procedure. The target DSP is specified from the “C2000 Target Preferences” library in Simulink. It is essential to set a discrete step size for the simulator. In general, smaller step sizes are better, provided the computational burden can be managed.  In this system, the DSP clock frequency is 150MHz but the control and estimation usually run at slower sampling frequencies (34 KHz in this case). The ADC is accessed through the “ADC” block found in the “C28x DSP Chip Support” library.

To achieve the best decimal accuracy, fixed-point math is used on the DSP. The “Target for TI C2000” Simulink library has the “C28x IQmath” module in which basic accurate math operations can be found. The *Q* value of an IQ number is the bit before which the decimal point is placed. IQMATH blocks accept signed 16-bit or 32-bit fixed-point integers (sfix (16) or sfix(32)). Fig. 7 shows a sample sfix(32) number with *Q*=29. All data types must be converted to sfix numbers to be implemented on the DSP. The choice of *Q* depends on the number of the decimal digits. For example, the leakage term, $\sigma$, has a value close to 1, so *Q*=30 is used. Also, reference frame transformations involve the sine and cosine trigonometric functions, whose values are between -1 and 1, thus a higher number of significant decimal digits, e.g. *Q=30*, is used. On the other hand, the stator voltages have significantly higher values and can be represented with a smaller *Q*.

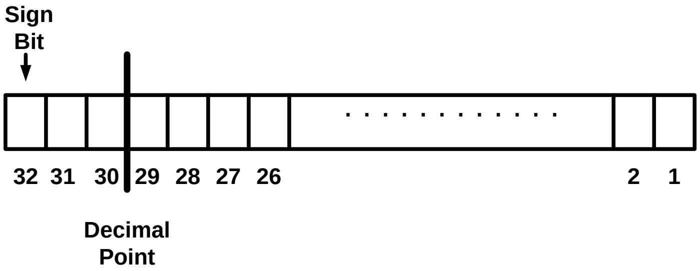

Fig. 7. 32-bit signed fixed-point number

## Estimator Implementation

The implementation of (1) is straightforward except for the integrator, where any minor dc offset could cause the integrator and integration registers to saturate. Several back EMF estimators replace the integrator with a LPF because of the integrator dc offset. A simple solution that ensures no dc offset is shown in Fig. 8. In the stator reference frame (hence the additional *s* subscript in Fig.8), *v*, *i*, and $\lambda$ are all sinusoids. The discrete integrator with a sampling rate $T_s$ and delay *z* can thus accumulate an offset due to any offsets in the sinusoids. A high-pass filter (HPF) is used to extract the sinusoid. Since the HPF is not ideal and minor dc offset could remain, another filtering stage is added where the remaining dc offset is extracted using a LPF and subtracted from the output of the HPF. This approach was verified to eliminate all dc offset and gave adequate results. The choice of 4 Hz for the HPF and LPF corner frequency is not strict, but the corner frequency should be low enough to avoid filtering the actual sinusoids when operating at low speeds. To further improve the estimation and achieve higher accuracy, the flux is estimated in mV·s. This gives more room for decimal bits in the 32-bit number.

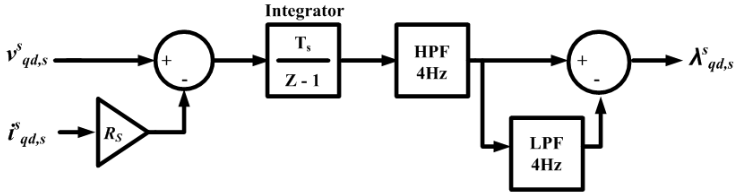

Fig. 8. Modified flux estimator

The rotor flux is estimated directly using (2). It is clear that this estimator is heavily dependent on motor parameters, but is shown to perform well. Estimating the torque is also a direct implementation of (3) where simple arithmetic blocks are used. A more challenging estimation is that of the speed, where $\rho$ is found using the “Arctangent IQN” block. Further integration and filtering of $\rho$ is thus required.

### Experiments and Results

The flux estimator described in (1) and (2) was implemented as shown in Fig. 8. The experimental setup is shown in Fig.9 and the eZdsp and interface boards are shown in Fig. 10. The flux computed by the estimator is monitored in real time using RTDX channels. The rms values of the voltage and currents were measured to estimate the real flux amplitude using continuous-time integration versus the DSP discrete-time integration. Comparing both methods of integration gives an estimate of the lumped estimation error including quantization error and error in parameter knowledge. Quantization error is demonstrated in Fig. 11, where the average quantized signal is different from the real continuous signal. The flux frequency is also essential, and this is verified using measurements of the electrical and mechanical frequencies. The frequency of the flux estimate matched, and the phase shifts were minimal since the speed was significantly greater than the HPF and LPF corner frequency of 4 Hz.

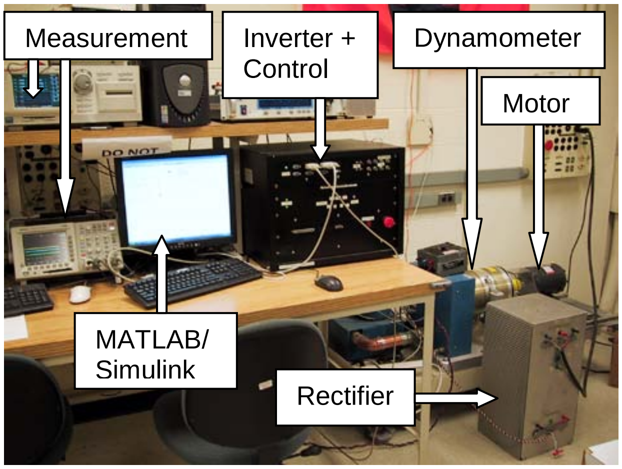

Fig. 9. Experimental setup

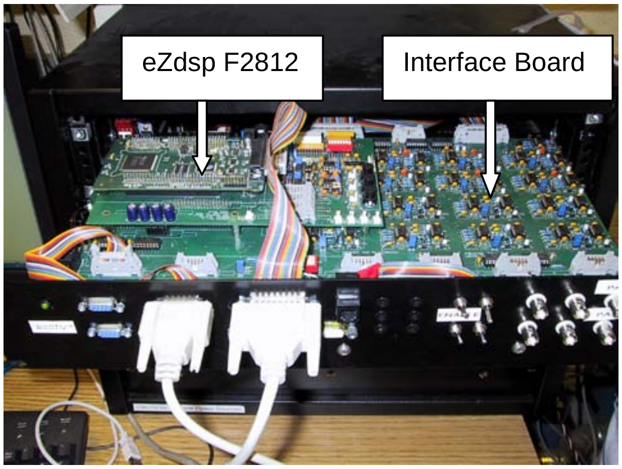

Fig. 10. DSP and interface boards

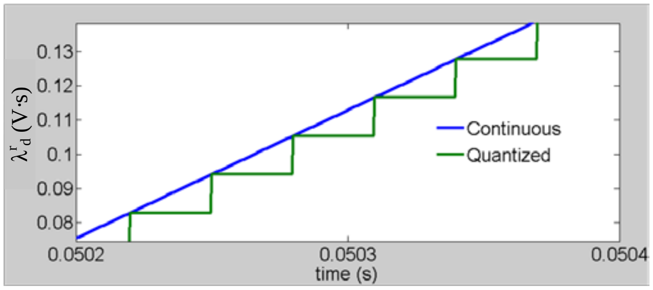

Fig. 11. Quantized vs continuous flux

The stator and rotor fluxes were estimated in real time under load and no load for both open loop (V/Hz) and closed-loop (IFOC). The filter gains were tuned iteratively to achieve the best performance. Table 1 summarizes the estimation results at different motor speeds and loading conditions for V/Hz, and Table 2 summarizes the results for IFOC. Fig. 12 shows a screenshot of the real-time estimate of $\lambda_d^s$ and $\lambda_d^r$. The errors shown in Tables 1 and 2 are acceptable for flux estimation, where the amplitudes are within 10% of the expected values. The errors are attributable to quantization and imperfect parameter knowledge.

Table 1. Flux amplitude from continuous-time calculation (Calc.) and experiments (Exp.) under V/Hz with and without load

| | 1226 rpm (42Hz) | | | 600 rpm (20.7Hz) | | | 1226 rpm (42Hz), 3N·m | | | 600 rpm (20.7Hz), 0.7N·m | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Calc. | Exp. | Error | Calc. | Exp. | Error | Calc. | Exp. | Error | Calc. | Exp. | Error |
| $\lambda_d^s$ (mV·s) | 415 | 400 | 3.6% | 325 | 350 | 7.7% | 355 | 315 | 11.3% | 301 | 290 | 3.7% |
| $\lambda_d^r$ (mV·s) | 410 | 390 | 4.9% | 320 | 310 | 3.2% | 345 | 305 | 11.6% | 295 | 265 | 10.2% |

Table 2: Flux amplitude from continuous-time calculation (Calc.) and experiments (Exp.) under IFOC with load

| | 1226 rpm (42Hz), 3N·m | | | 600 rpm (20.7Hz), 0.7N·m | | |
|---|---|---|---|---|---|---|
| | Calc. | Exp. | Error | Calc. | Exp. | Error |
| $\lambda_d^s$ (mV·s) | 394 | 400 | 1.5% | 460 | 430 | 7% |
| $\lambda_d^r$ (mV·s) | 384 | 385 | 0.26% | 448 | 415 | 8% |

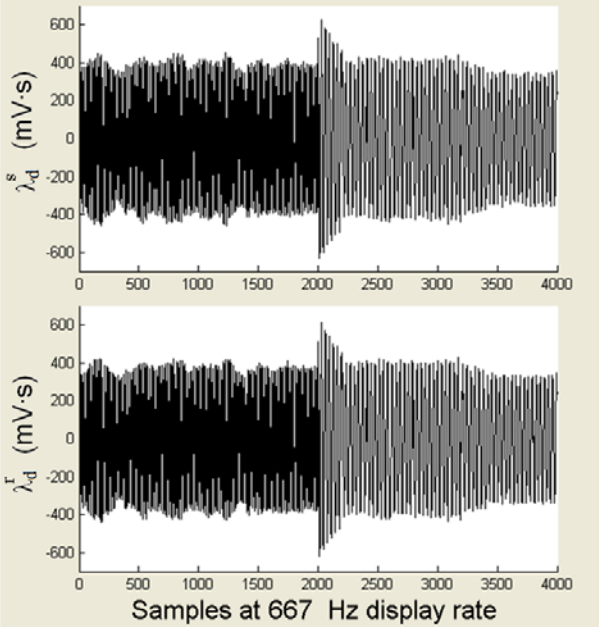

Fig. 12. Screenshot of estimation results: d-axis stator and rotor fluxes under IFOC and load

Even though in this specific setup the torque, speed, and input power are all measured, implementing (3) – (5) is straightforward once voltage and current measurements, in addition to flux estimates, are available. Torque is estimated as shown in (3) for a load step from 0.4 N·m to 2 N·m. The input power increased from 62 W to 430 W, and was estimated using (5). A screenshot of the real-time torque and input power estimates is shown in Fig. 13 for a constant speed of 1750 rpm. The torque was estimated in mN·m for higher resolution. It is clear that the estimators are all performing as expected. Speed estimation was tested by applying (4) to estimate $\rho$. The speed can then be inferred by estimated. The motor speed was also measured to verify the estimate ($\omega_{m,est}$) for a speed step from 1000 rpm to 1750 rpm. Fig. 14 shows the measured and estimated speeds. Both speeds match, and the estimator performs as expected. A negative aspect of the estimator is its convergence speed, where it has about 1.5 s of time lag.

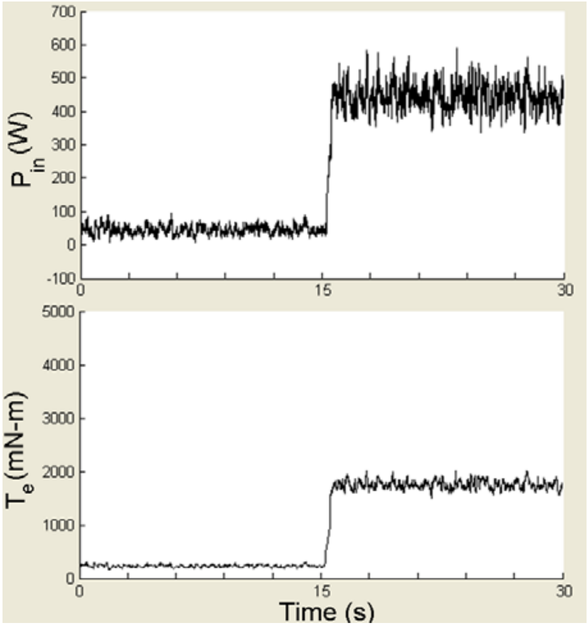

Fig. 13. Screenshot of estimation results: input power (top) and torque (bottom)

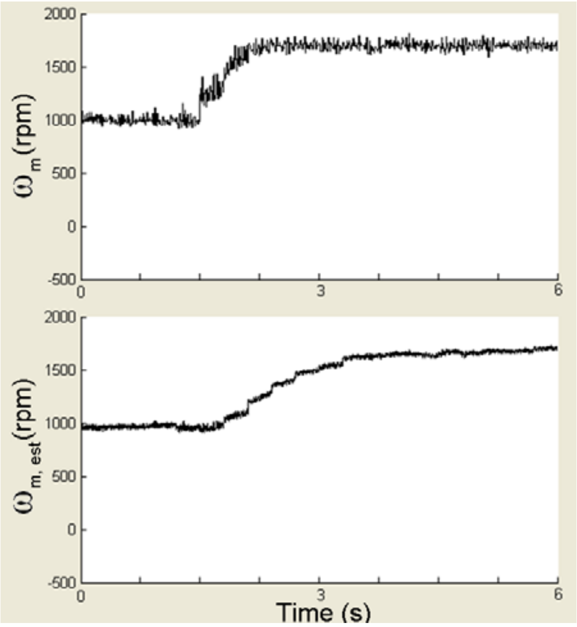

Fig. 14. Screenshot of measured (top) and estimated speed (bottom)

**Conclusions and Final Remarks**

This article presented a comprehensive procedure for implementing control and estimation algorithms for induction machines. The procedure includes important considerations for both software and hardware setups, and can be extended to other software, digital platforms, and motor drives. Tests showed that even though the estimation used is parameter dependent and lacks error minimization, the results are acceptable. More complex or advanced control and estimation can be implemented using a similar approach. As a final remark, this work on estimation started from the need to find a flux estimate to minimize losses in induction machines. Other applications of flux, torque, and speed estimation include speed-sensorless control and dynamic torque control among several others.

## Acknowledgements

This work is supported in part by the Granger Center for Electric Machinery and Electromechanics at the University of Illinois, and by the Office of Naval Research under grant ONR N00014-08-1-0397. The authors would like to acknowledge the valuable help of Mr. Kevin Colravy and Mr. Sanghun Choi.

## Further Readings

- A.M. Bazzi and P.T. Krein, "Comprehensive Flux Estimator Implementation Procedures for Advanced Control of Inverter-Fed Induction Machines," *in Proc. Electrical Manufacturing and Coil Winding Conference*, 2009, pp. 1-9.
- P. C. Krause, O. Wasynczuk, and S. D. Sudhoff, *Analysis of Electric Machinery and Drive Systems*, 2nd ed. New York: Wiley - IEEE Press, 2002.
- G. C. Verghese and S. R. Sanders, "Observers for Flux Estimation in Induction Machines," *IEEE Trans. Industrial Electronics*, vol. 35, pp. 85 - 94, Feb. 1988.
- "eZdsp(TM) F2812 Technical Reference 506265-0001 Rev. F," Spectrum Digital, Stafford, TX 2003.
- B. Nikolic and B. I. Jeftenic, "Fixed-point DSP algorithm for flux estimation in DTC IM drive," in *Proc. International Conference on Computer as a Tool*, 2005, pp. 1582 - 1585.
- D. A. Andrade, A. W. F. V. Silveira, P. B. Severino, and T. S. Tavares, "Dsp based torque estimation in three-phase cage induction motors," in *Proc. IEEE International Electric Machines and Drives Conference*, 2007, pp. 1726 - 1731.
- Consoli, G. Scarcella, and A. Testa, "Speed - and current - sensorless field - oriented induction motor drive operating at low stator frequencies," *IEEE Trans. Industry Applications*, vol. 40, pp. 186 - 193, Jan. 2004.