
AVR32723: Sensor Field Oriented Control for Brushless DC motors with AT32UC3B0256

Features

- Standalone Space Vector Modulation library for AVR[®] 32 UC3 microcontroller.
- Park and Clarke mathematical transformation library for AVR32 UC3 microcontroller.
- Theory of Field Oriented Control.
- PC application for real time motor remote control and display of regulated variables.

1 Introduction

This application note delivers an implementation of Sensor Field Oriented Control algorithm for brushless DC motor on Atmel[®] AVR32 UC3B microcontrollers.

This software includes standalone libraries for all mathematical transformations required by the algorithm.

For more information about the AVR32 architecture, please refer to the appropriate documents available from <http://www.atmel.com/avr32>.

2 Requirements

The software provided with this application note requires several components:

- A computer running Microsoft[®] Windows[®] 2000/XP/Vista[®] or Linux[®]
- AVR32Studio and the GNU toolchain (GCC) or IAR Embedded Workbench[®] for AVR32 compiler.
- A JTAGICE mkII or AVROne! debugger



32-bit **AVR[®]**
Microcontrollers

Application Note

Rev. 32126A-AVR32-06/09



3 Theory of Operation

3.1 Overview

As continuity in the energy efficiency process, the increase of capability of microcontrollers allows to implement complex algorithm as Field Oriented Control methodology which allows to:

- Reduce power consumption (less power loss) of electrical motor driving.
- Provide smoother and more accurate motor control than sinusoidal command with hall feedback.
- Optimize motor choice (torque): find the best alternative between dimension and power.

But this algorithm requires more complex computation and requires more CPU bandwidth than standard algorithms.

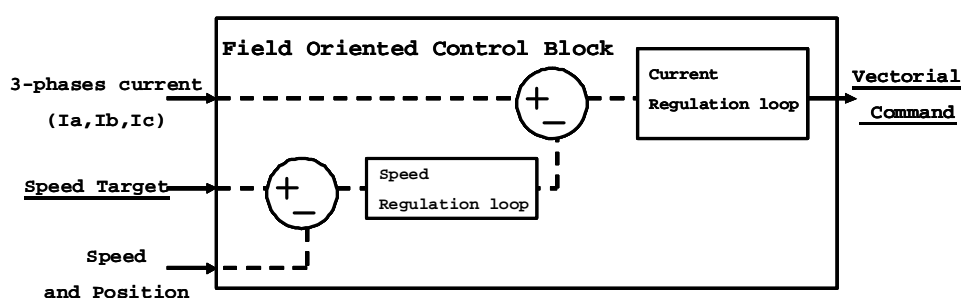
Moreover, this command strategy fits with some brushless DC motors with dynamical load changes due to reduction of response time. It allows regulating torque and speed at the same time.

3.2 Block Diagram

The following block diagram describes the principle of the algorithm. Indeed, it is based on:

- A vectorial command for 3-phases Brushless DC Motors.
- A periodic current sampling of the 3 shunts located on these 3-phases.
- 2 encapsulated regulation loops: one for the current (torque and power loss) and another for the speed.

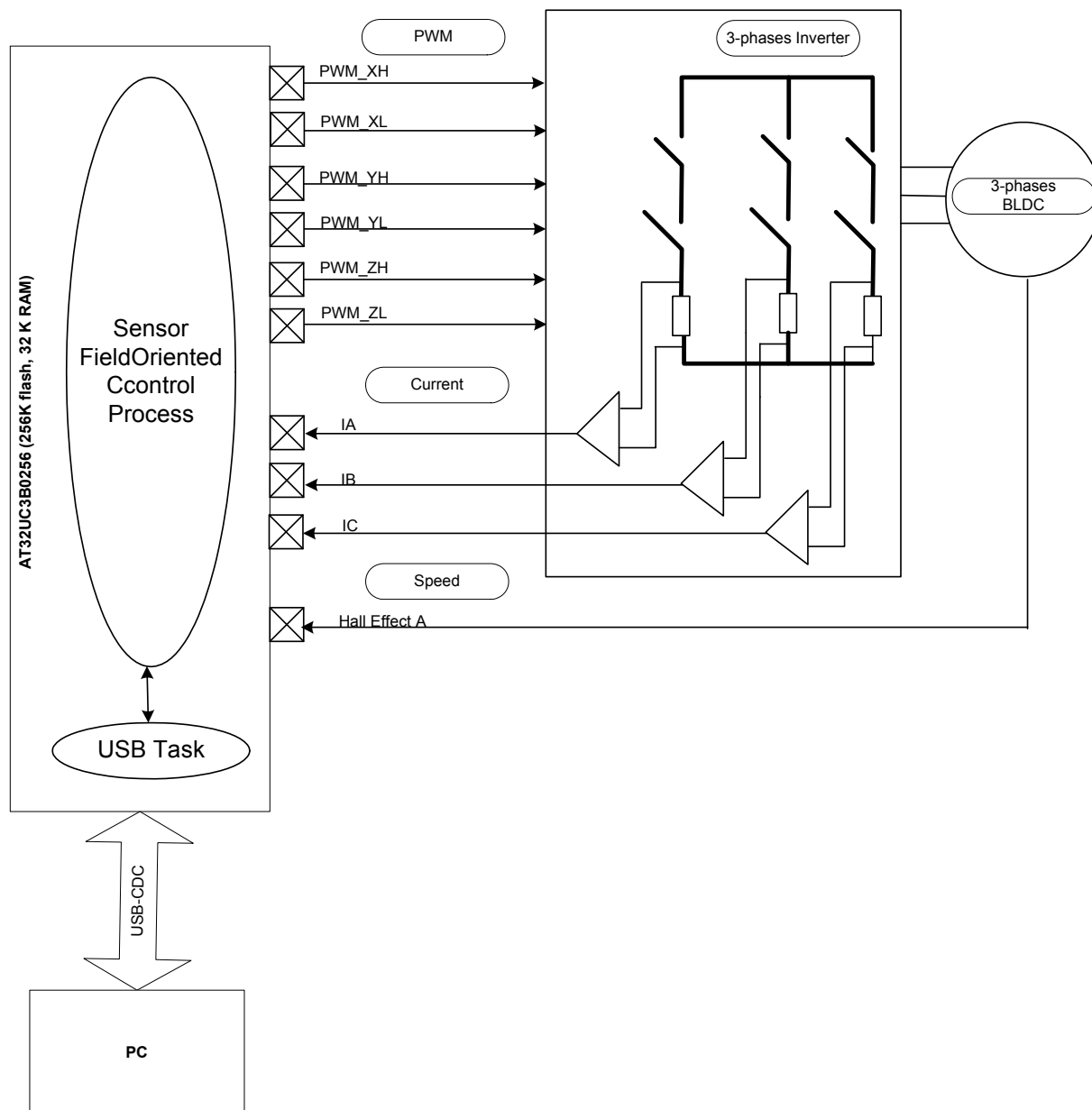
Figure 3-1. Block Diagram



3.3 System Overview

As shown Figure 3-2, the 3-phases current are measured and one Hall Effect feedback is used to compute field oriented control algorithm.

Figure 3-2. System Overview





The USB connection is used through a PC application for real time motor remote control and display of regulated variables. This data monitor displays feedback control remotely. This GUI provides:

- 3 graphs and 1 cursor display.
- 4 inputs control.

The purpose is to be able to connect to the application without disturbing the behavior of the application. The application offers the following features:

- Implements vector control of a Brushless DC motor.
- Speed range from 400 rpm to 2000 rpm.¹
- With a 100 μ sec control loop, the CPU is used at 35% by the Sensor Field Oriented Control Process at 42MHz.
- Display in real time the current and speed measured.
- Control of speed in real time.

¹ See section 5.9 for the explanation of this limitation.

4 Motor Control Theory

4.1 General Model.

The mathematical model in a constant domain linked to the stator is linked to differential equation with constant coefficients defining the motor behavior. Applying the Lenz-Faraday model, we have:

$$[V_{abc}] = R[I_{abc}] + \frac{d}{dt}[\Phi_{abc}]$$

With:

$$[V_{abc}] = \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix}, \quad [I_{abc}] = \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \quad \text{and} \quad [\Phi_{abc}] = \begin{bmatrix} \Phi_a \\ \Phi_b \\ \Phi_c \end{bmatrix}$$

R is the statoric resistance, $[v_a \ v_b \ v_c]^t$ are the statoric voltages, $[i_a \ i_b \ i_c]^t$ are the statoric current and $[\Phi_a \ \Phi_b \ \Phi_c]^t$ are the global fields in the statoric solenoid.

$$[\Phi_{abc}] = [L][I_{abc}] + [\Phi_{sf}]$$

We can write :

$$[\Phi_{abc}] = \begin{bmatrix} L_s & M & M \\ M & L_s & M \\ M & M & L_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} \Phi_{sf} \cos(\theta) \\ \Phi_{sf} \cos(\theta - \frac{2\pi}{3}) \\ \Phi_{sf} \cos(\theta + \frac{2\pi}{3}) \end{bmatrix}$$

L_s (constant) is the inductance in a statoric gyres.

M (constant) is the mutual inductance between the 2 statoric gyres

θ is the electrical position of the rotor ($\theta = \theta_{elec}$)

Φ_{sf} is the max value (constant) of the flux create by the permanents magnet through the statoric gyres.

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = R \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} L_s & M & M \\ M & L_s & M \\ M & M & L_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \Phi_{sf} \cos(\theta) \\ \Phi_{sf} \cos(\theta - \frac{2\pi}{3}) \\ \Phi_{sf} \cos(\theta + \frac{2\pi}{3}) \end{bmatrix}$$

In order to simplify the mathematical resolution of this system, some mathematical transformations are used.

Using the Park P1(θ), the instantaneous power is kept and it allows to have a mathematical expression of the torque still acceptable for the real motor.



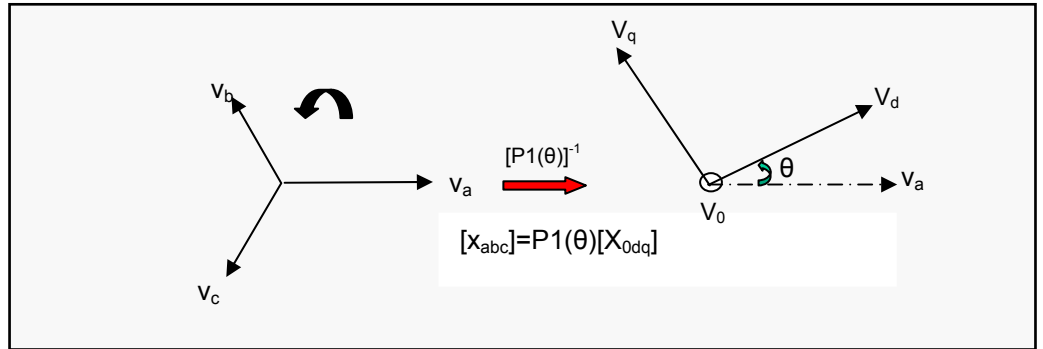
Using the Park inverse transformation, it is possible to return to the values space from regulated values and generate commands.

4.1.1 Park Transformation ($P1(\theta)$) and Clarke Transformation ($\text{inv } P-1(\theta)$)².

The Park transformation is defined from the matrix $P1(\theta)$, it defines the link between the real vectors $[V_{abc}]$, $[I_{abc}]$ et $[\Phi_{abc}]$, the new vectors $[V_{0dq}]$, $[I_{0dq}]$ et $[\Phi_{0dq}]$.

$$P1(\theta) = \sqrt{\frac{2}{3}} \begin{bmatrix} \frac{1}{\sqrt{2}} & \cos(\theta) & -\sin(\theta) \\ \frac{1}{\sqrt{2}} & \cos(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{2\pi}{3}) \\ \frac{1}{\sqrt{2}} & \cos(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{2\pi}{3}) \end{bmatrix}$$

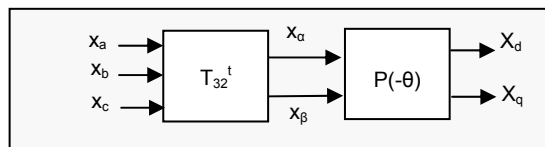
Figure 4-1. Park Transformation



With $V_0 = 0V$, $I_0 = 0A$ and $\Phi_0 = 0Wb$, the normalized Park transformation is the addition of the Concordia Transformation T_{32} and the rotating matrix $p(\theta)$:

$$T_{32} = \frac{1}{\sqrt{3}} \begin{bmatrix} +\sqrt{2} & 0 \\ -\frac{\sqrt{2}}{2} & +\sqrt{\frac{3}{2}} \\ -\frac{\sqrt{2}}{2} & -\sqrt{\frac{3}{2}} \end{bmatrix} \quad p(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Figure 4-2. 3D to 2D Transformation

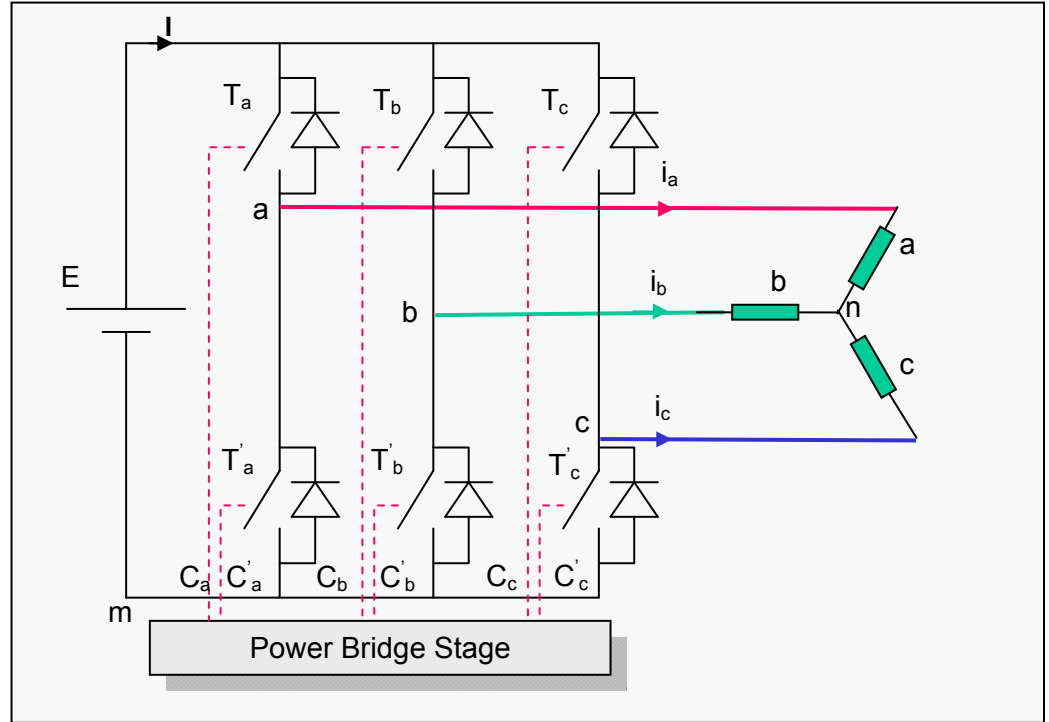


² The Park and Clarke transformations are implemented as a software library component in the software package associated with this application note. See also section 5.

4.2 Power Bridge Command.

The power bridge stage is here to power the BLDC motor with alternatives voltages from a continuous source. This bridge allows modulating frequency and amplitude.

Figure 4-3. Power Bridge Stage



The logical signals C_a , C'_a , C_b , C'_b , C_c and C'_c are command signals for the power bridge.

The output voltages are given from the ground ' m ' of source generator and from the virtual neutral point ' n ' of BLDC motor.

$$\begin{bmatrix} V_{am} \\ V_{bm} \\ V_{cm} \end{bmatrix} = E \begin{bmatrix} C_a \\ C_b \\ C_c \end{bmatrix} \quad (II-1) \quad \begin{aligned} V_{am} &= V_{an} + V_{nm} \\ V_{bm} &= V_{bn} + V_{nm} \\ V_{cm} &= V_{cn} + V_{nm} \end{aligned}$$

For an equilibrated load: $V_{an} + V_{bn} + V_{cn} = 0V$, so : $V_{nm} = \frac{1}{3}(V_{am} + V_{bm} + V_{cm})$.

In that way, we have:

$$\begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} V_{am} \\ V_{bm} \\ V_{cm} \end{bmatrix} = \frac{E}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} C_a \\ C_b \\ C_c \end{bmatrix}$$

Or :

$$T_{32}^t \cdot \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} = T_{32}^t \text{ and } \begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = T_{32}^t \begin{bmatrix} V_{an} \\ V_{bn} \\ V_{cn} \end{bmatrix}$$

We conclude so: $\begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = T_{32}^t \cdot E \begin{bmatrix} C_a \\ C_b \\ C_c \end{bmatrix}$, (T_{32}^t is the translation of T_{32})

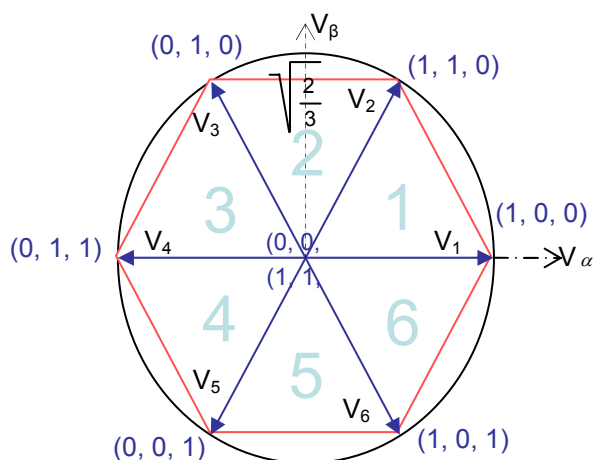
The following table gives the voltage values in the spaces (α , β) and (a, b, c).

Table 4-2. Voltage Values in the spaces (α , β) and (a, b, c)

Vector	C_a	C_b	C_c		V_α	V_β		V_{an}	V_{bn}	V_{cn}
V_0	0	0	0	$E \sqrt{\frac{2}{3}}$	0	0	$\frac{E}{3}$	0	0	0
V_1	1	0	0		1	0		2	-1	-1
V_2	1	1	0		$\frac{1}{2}$	$\frac{\sqrt{3}}{2}$		1	1	-2
V_3	0	1	0		$-\frac{1}{2}$	$\frac{\sqrt{3}}{2}$		-1	2	-1
V_4	0	1	1		-1	0		-2	1	1
V_5	0	0	1		$-\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$		-1	-1	2
V_6	1	0	1		$\frac{1}{2}$	$-\frac{\sqrt{3}}{2}$		1	-2	1
V_7	1	1	1		0	0		0	0	0

We show that all vectors components (V_α , V_β) are with modules $\sqrt{2/3} E$ and are located in the regular hexagon.

Figure 4-4. Space Vector values in the regular hexagon.



4.2.1 Space Vector PWM.³

The vectorial modulation uses directly the signal for the Concordia transformation. It supposes that a regulation loop is ever implemented for the generation of V_α and V_β components.

As stated below, at a instantaneous point, the power bridge is able to generate only 8 voltages (V_i , $i = 0, \dots, 7$) in the space of transformation Concordia (V_α , V_β) with two components equal to 0 and 6 with the module equal to $E\sqrt{2/3}$ and the angle to $(\pi/3)(i-1)$. Two successive vectors, called V_i and V_{i+1} defined a sector i (**Figure 4-4**) with t from the interval $[t_i, t_{i+1}]$.

$$V_i = \begin{pmatrix} V_\alpha \\ V_\beta \end{pmatrix}_i = E\sqrt{\frac{2}{3}} \begin{bmatrix} \cos\left\{\frac{\pi}{3}(i-1)\right\} \\ \sin\left\{\frac{\pi}{3}(i-1)\right\} \end{bmatrix} = E\sqrt{\frac{2}{3}} P \left[\left\{ \frac{\pi}{3}(i-1) \right\} \right] \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

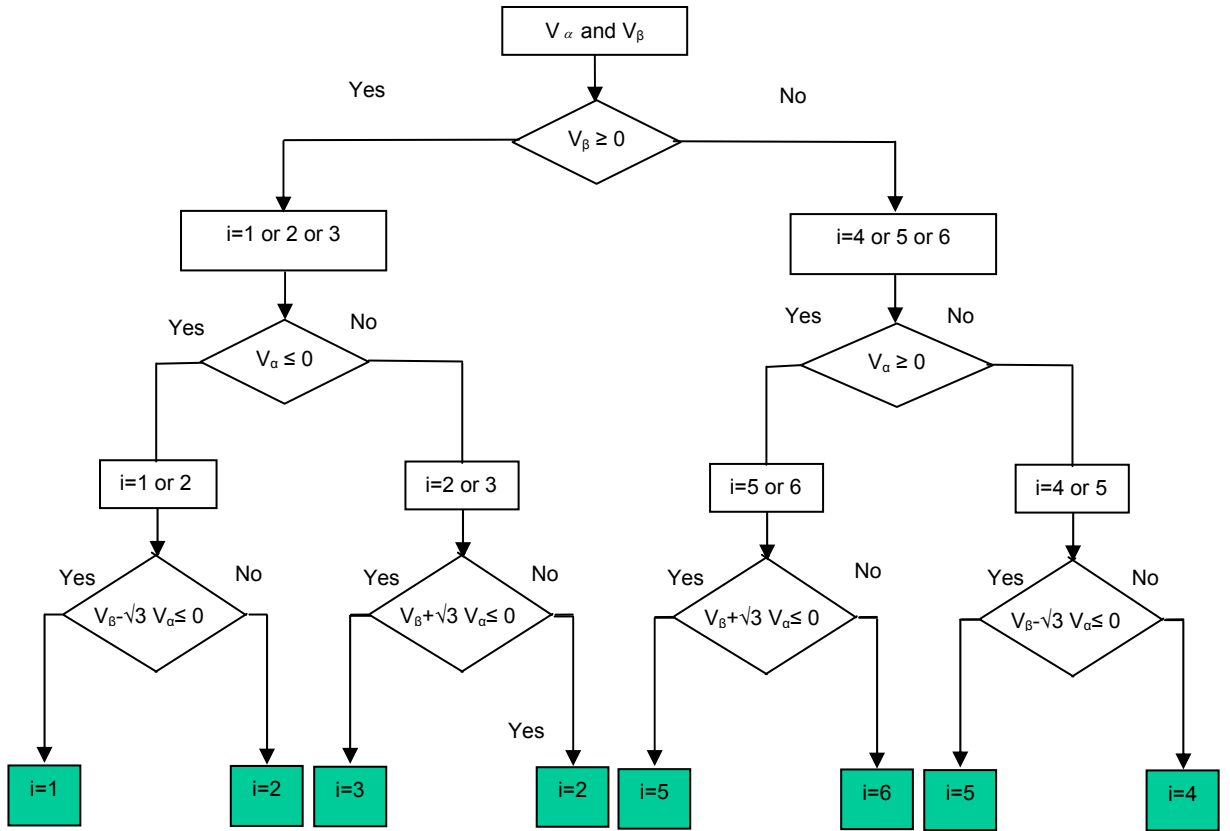
The power stage is only able to deliver voltages instantaneous voltages on a sampling period T , we can write:

$$(V_{\alpha\beta}) = \delta_i V_i + \delta_{i+1} V_{i+1}$$

Where δ_i and δ_{i+1} are the duration for relative values are the relative voltages when V_i and V_{i+1} ; $\delta_i V_i$ and $\delta_{i+1} V_{i+1}$ are the projection of vector $(V_{\alpha\beta})$.

³ The Space Vector Modulation is implemented as a software library component in the software package associated with this application note. See also section 5.

Figure 4-5. Sector Determination Algorithm.



In all cases, it is needed to determined sector number « i » where is located the vector V_i . To reduce time execution computing δ_i et δ_{i+1} , we have used this algorithm.

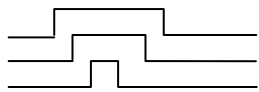
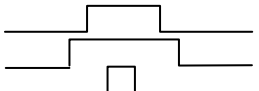
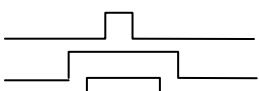
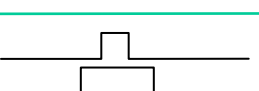


In that way, δ_i et δ_{i+1} are calculated directly from V_α and V_β :

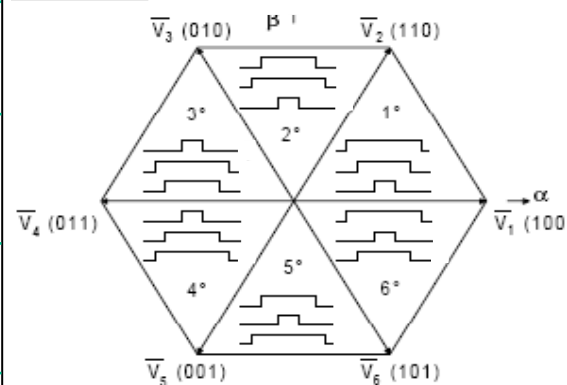
$$\begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = \delta_i \begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix}_i + \delta_{i+1} \begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix}_{i+1}$$

And so

$$\begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = E \sqrt{\frac{2}{3}} \left\{ \delta_i p\left(\frac{\pi}{3}(i-1)\right) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \delta_{i+1} p\left(\frac{\pi}{3}i\right) \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\}$$

Table 4-2. Sector expressions

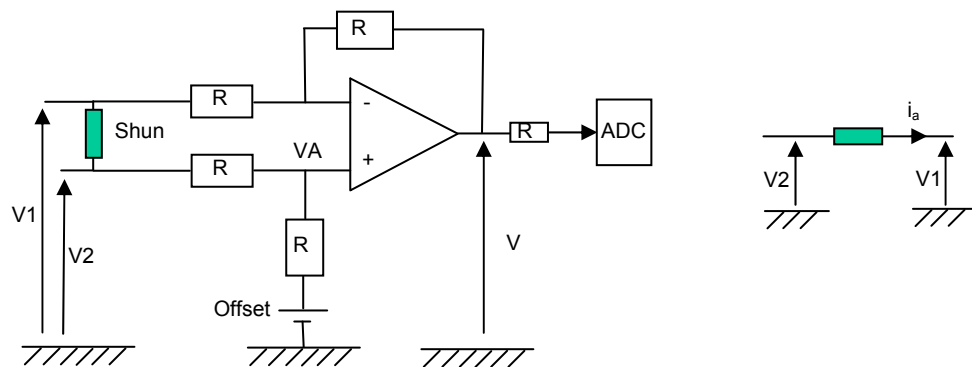
Sector 1	$T_1 = \tau_1 + \tau_2 + \tau_0$ $T_2 = \tau_2 + \tau_0$ $T_3 = \tau_0$	
2	$T_1 = \tau_2 + \tau_0$ $T_2 = \tau_3 + \tau_2 + \tau_0$ $T_3 = \tau_0$	
3	$T_1 = \tau_0$ $T_2 = \tau_3 + \tau_4 + \tau_0$ $T_3 = \tau_4 + \tau_0$	
4	$T_1 = \tau_0$ $T_2 = \tau_4 + \tau_0$ $T_3 = \tau_5 + \tau_4 + \tau_0$	
5	$T_1 = \tau_0 + \tau_6$ $T_2 = \tau_0$ $T_3 = \tau_0 + \tau_5 + \tau_6$	
6	$T_1 = \tau_0 + \tau_1 + \tau_6$ $T_2 = \tau_0$ $T_3 = \tau_0 + \tau_6$	



4.3 Current Measurement Stage

Three identical shunts are used to read the current I_a, I_b and I_c . These values should be amplified to be in the range of the ADC. Indeed, if we measure current of several milliamps (500mA), the maximum voltage is around 0.025V.

Figure 4-6. Amplifier stage

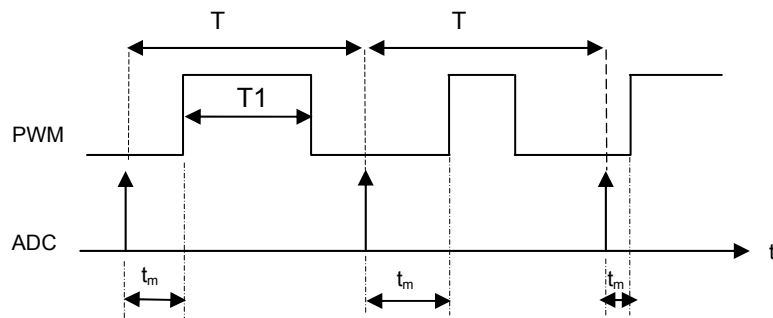


For example:

$$V_s = \text{Offset} + \frac{R_2}{R_1}(V_2 - V_1)$$

When measuring the three currents I_a, I_b and I_c , only two measures are required, indeed $I_a + I_b + I_c = 0$.

Figure 4-7. ADC and PWM synchronization



The ADC measures are done every period of the PWM. The first ADC measure should be done in the time range named t_m . This time is the minimum time where none of the 3-phases are driven.

For example:

In case of sector 1, T_1 is the longest time so it means that $T_1 > T - t_m$, in that case it is not possible to measure I_a . So I_b and I_c are measured and I_a is derived from the two others measures with the equation $I_a = -I_b - I_c$.

4.4 Current regulation and speed Regulation⁴

In order to reduce time consuming effect of regulator, we have used the Integrator Proportional (IP) regulator.

4.4.1 Current Regulation

In a continuous domain, the closed loop has the mathematical expression:

$$I_d = \frac{K_i}{L_c s^2 + (R + K_d)s + K_i} I_{dref}$$

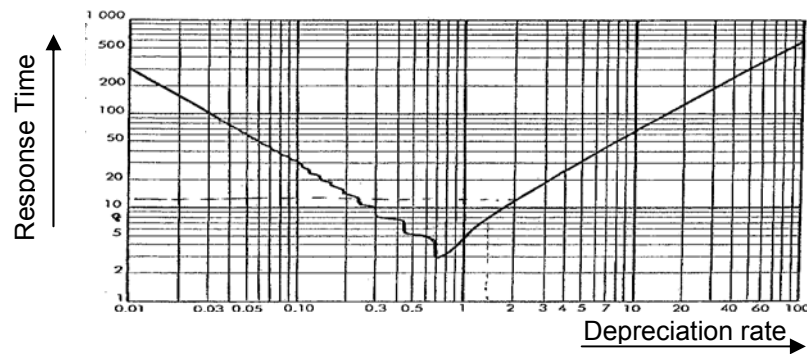
It is possible to easily identify the second order of the transfer function:

$$\frac{a}{s^2 + 2\varepsilon\omega_n s + \omega_n^2}$$

With $K_i = L_c \omega_n^2$ and $K_d = 2\varepsilon\omega_n L_c - R$

These correctors have been chosen to have a depreciation rate $\varepsilon = 2$ and a response time of 5% ($T_r \omega_n$).

Figure 4-8. Response time vs Depreciation rate.



For that

$$\omega_n = 2R/L_c \text{ with } K_i = 4R^2/L_c \text{ and } K_d = 7.R$$

4.4.2 Speed Regulation

The external closed loop for speed regulation has the following mathematical expression:

$$\frac{\Omega}{\Omega_{ref}} = \frac{K_i}{J s^2 + K_{dv} s + K_{iv}} = \frac{a}{s^2 + 2\varepsilon\omega_n s + \omega_n^2}$$

$$\text{With } \omega_n^2 = \frac{K_{iv}}{J} \text{ et } 2\varepsilon\omega_n = \frac{K_{dv}}{J}$$

⁴ The PI regulator is implemented as a software library component in the software package associated with this application note. See also section 5.



We choose $\omega_n = 0.1R/L_c$ and $\varepsilon = 2$, So $K_{iv} = 0.01 \frac{R^2}{L_c^2} J$ et $K_{dv} = 0.4 \frac{R}{L_c} J$

4.5 Execution Time and Regulation Loop

The following table gives the number of operations for each function

Table 4-2. Operations costs

Function	Add	Mul	Div	Shift
Concordia abc-to- $\alpha\beta$	7	0	0	17
Clark $\alpha\beta$ -to-dq	2	4	0	62
Clark Inv dq-to- $\alpha\beta$	2	4	0	62
Regulation	7	4	0	124
Decoupling ⁵	3	5	0	62
SVPWM	27	0	0	354

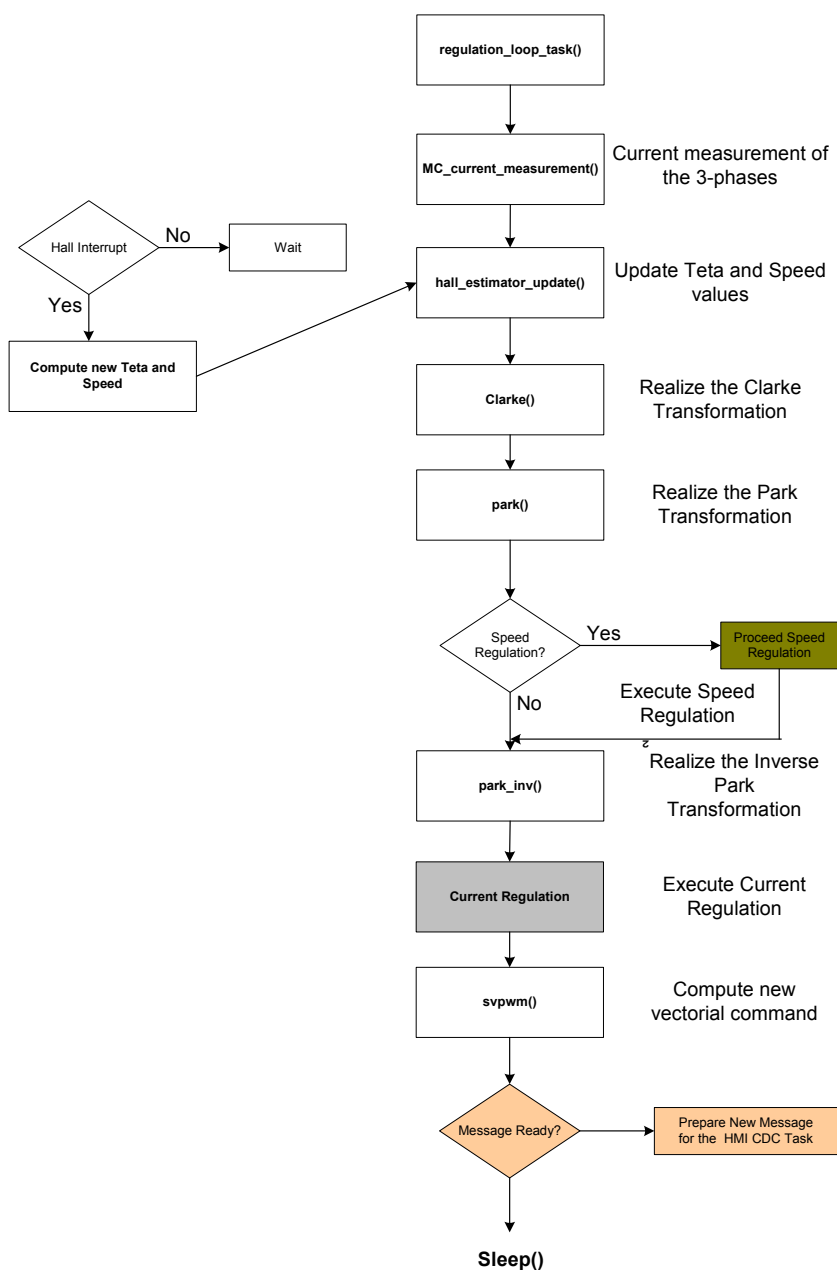
⁵ The decoupling stage is the stage to separate torque and power loss regulations after the current measurement stage. The idea is to be sure to only regulate torque (without disturbing the power loss loop).

4.6 General Algorithm

As shown in **Figure 4-9**, the `regulation_loop_task()` process is sequenced on tick reference (every 100 μ s). This task synchronizes current measurement and computes all mathematical transformations to execute the Field Oriented Control algorithm. Inside this task, there are two regulation loops interlinked:

- Current regulation loop.
- Speed regulation loop.

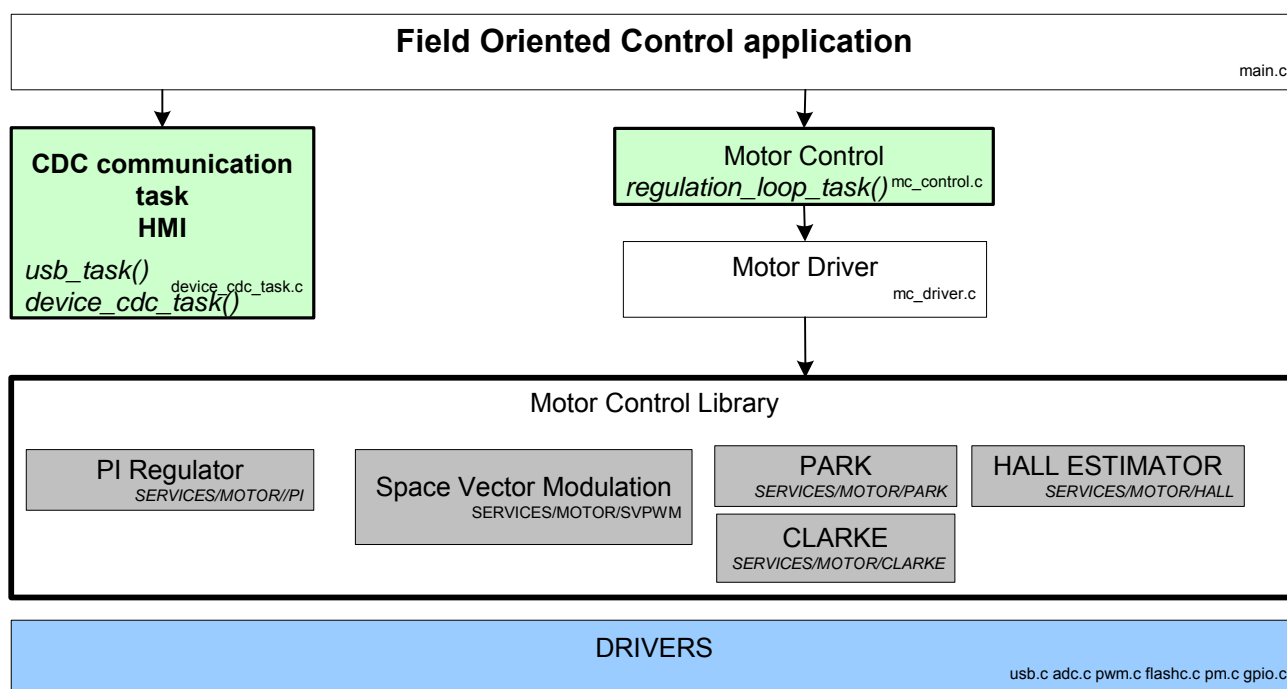
Figure 4-9. Regulation Loop Task Step Machine



5 Source Code Architecture

5.1 Software Architecture

Figure 4-10. Software Architecture



This application does not require any operating system to run. The main() function is in charge of calling the software «tasks» (using a scheduler) realizing FOC algorithm computation and USB communication with the Communication Device Class (CDC) for HMI management. There are 2 tasks:

- `regulation_loop_task()` The regulation task that computes all calculation for dedicated FOC algorithm.
- `usb_task()` / `device_cdc_task()` The USB Task: This task is in charge of the CDC communication management.

The main loop of the application is a simple free-running task scheduler:

```

while(TRUE)
{
#ifdef USB_DEBUG
    usb_task();
    device_cdc_task();
#endif
    mc_regulation_task();
}
  
```


5.2 Package

The EVK1101-SENSOR-FIELD-ORIENTED-CONTROL-X.Y.Z.zip contains projects for AT32UC3B0256 RevF or later:

- EVK1101-SENSOR-FIELD-ORIENTED-CONTROL-X.Y.Z

Default hardware configuration of the project is to run on the EVK1101 board, although any board can be used (refer to section 5.5.5)

5.3 Documentation

For full source code documentation, please refer to the auto-generated Doxygen source code documentation found in:

- AVR32723/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/readme.html

5.4 Projects/ Compiler

The IAR™ project is located here:

- `src/ APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/IAR/`

The GCC makefile is located here:

- `src/ APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/GCC/`

The Avr32Studio project is located in the root-dir of the package:

`./`

5.5 Implementations Details

5.5.1 Main()

The main() function of the program is located in the file:

- `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/main.c`

This function will:

- Initialize PWM outputs for Space Vector Generation.
- Initialize ADC inputs for 3-phases current measurement.
- Initialize USB connection for CDC HMI connection.

5.5.2 Motor Control Library

The motor control library is located here:

- `src/SERVICES/MOTOR_CONTROL/`

This folder delivers:

- The Park and Clarke transformations explained in section 4.1.1 are delivered in the folder `src/SERVICES/MOTOR_CONTROL/PARK_CLARKE/`
- The Space Vector Modulation explained in section 4.2.1 is delivered in the folder `src/SERVICES/MOTOR_CONTROL/SVPWM/`



5.5.3 HMI CDC Task

The HMI CDC Task is located here:

- `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/device_cdc_task.c`

This task receives and sends messages through the USB communication.

With:

- The message definition located in `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/ENUM/frame.h`

5.5.4 AT32UC3B Drivers

The example firmware uses the AVR32 UC3 driver library available in

- `src/UTILS/LIBS/DRIVERS/AT32UC3B/`

5.5.5 Board File Definition

The application is designed to run on the EVK1101. All projects are configured with the following

define: `BOARD=EVK1101`. The EVK1101 definition can be found in the `src/BOARDS/EVK1101` directory.

5.5.5.1 Board customization

For IAR project, open the project options (Project -> Options), choose the «C/C++ Compiler», then «Preprocessor». Modify the `BOARD=EVK1101` definition by `BOARD=USER_BOARD`. For GCC, just modify in the `config.mk` file (`src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/GCC`) the DEFS definition with `-D BOARD=USER_BOARD`. For Avr32Studio, open the project properties (Project -> Properties), go in the «C/C++ build», then «Settings», «tool settings» and «Symbols». Modify the `BOARD=EVK1101` definition by `BOARD=USER_BOARD`.

5.6 Project Configuration

The project configuration files can be found in the `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/CONF/` directory.

Configuration files are not linked to IAR, GCC or Avr32Studio projects. The user can alter any of them, then rebuild the entire project in order to reflect the new configuration.

- `/CONF`: configuration header files of demo modules:

CPU settings , Peripheral Clock settings and Motor settings

5.7 GUI Application

The GUI application installer is located here:

- `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/LABVIEW/FOC_Gui.msi`

The USB driver is located here:

- `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/LABVIEW/usb_cdc.inf`

5.8 CPU Cost and Memory Usage for Sensor Field Oriented Control algorithm.

All results are given using IAR Workbench 5.3 compiler revision 3.10A with speed optimization level and Hmatrix optimization.

Criteria	Result
CPU Occupation	35% with a tick value of 100 us at 42MHz.
Code Size	17Kb
Data	13Kb
Const	3.5Kb

5.9 Limitation of the Sensor Field Oriented Control algorithm.

All the application has been implemented with a fixed point library written in a 32-bit format (Q1.31)⁶.

The usage of this library allows accelerated computations but it generates some limitations in the variation range of each variable.

For example, when computing current regulation, every ADC samples are scaled by a variable named E. This variable matches a ratio of the bus voltage. In the current implementation, this variable has been fixed to half of the nominal voltage of the motor. In that case, the nominal speed is equal to 2000 rpm.

⁶ See the application note:

http://www.atmel.com/dyn/resources/prod_documents/doc32076.pdf for more details.



5.10 Compiling the application

The following steps show you how to build the embedded firmware according to your environment.

5.10.1 If you are using AVR32Studio

- Launch avr32Studio
- Create a new AVR32 C project («File» -> «new» -> «AVR32 C Project»).
- Fill-in the dialogue box with project name, set target MCU to UC3B0256 and press finish.
- Choose Import archive file («File» -> «import»...), press the “next” button.
- Select the EVK1101-SENSOR-FIELD-ORIENTED-CONTROL-X.Y.Z.zip archive file with the browse button. Select «into folder», check «Overwrite existing resources without warning» and press the “finish” button.
- The project is now available in the given project name.
- Press the build button
- Load the Code: Please refer to the application note AVR32723: AVR32 Studio getting started

5.10.2 If you are using GCC with the AVR32 GNU Toolchain

- Open a shell, go to the `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/GCC/` directory and type:

`make rebuild program run`

5.10.3 If you are using IAR Embedded Workbench® for Atmel AVR32

- Open IAR and load the associated IAR project of this application (located in the directory `src/APPLICATIONS/EVK110x-MOTOR-CONTROL/BLDC-FOC/EXAMPLE/IAR`)
- Press the “Debug” button at the top right of the IAR interface.
- The project should compile. Then the generated binary file is downloaded to the microcontroller to finally switch to the debug mode.
- Click on the “Go” button in the “Debug” menu or press F5.

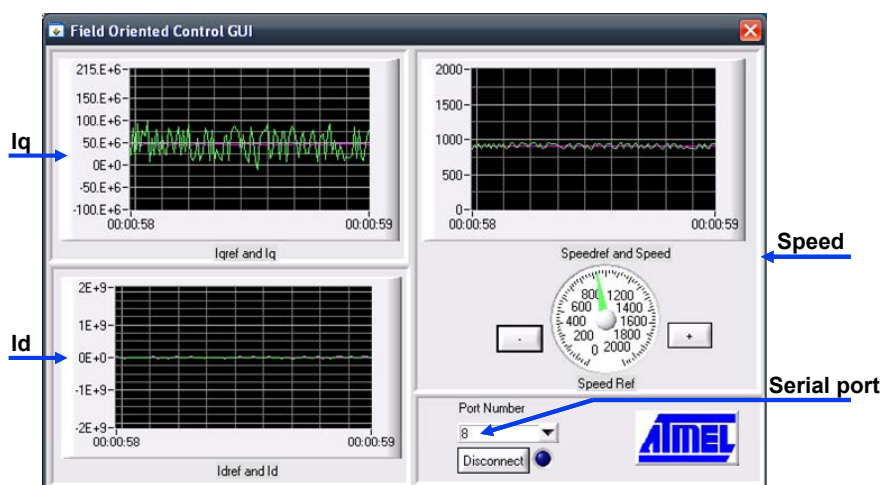
5.11 Start the PC application

- Plug the EVK1101 to the PC through a USB Connection.
- The USB enumeration should start; a new serial port appeared in Windows.
- Power-up the power bridge.

5.11.1 Field Oriented Control GUI

Once the GUI is launched, the user can select a serial port number and connect the application.

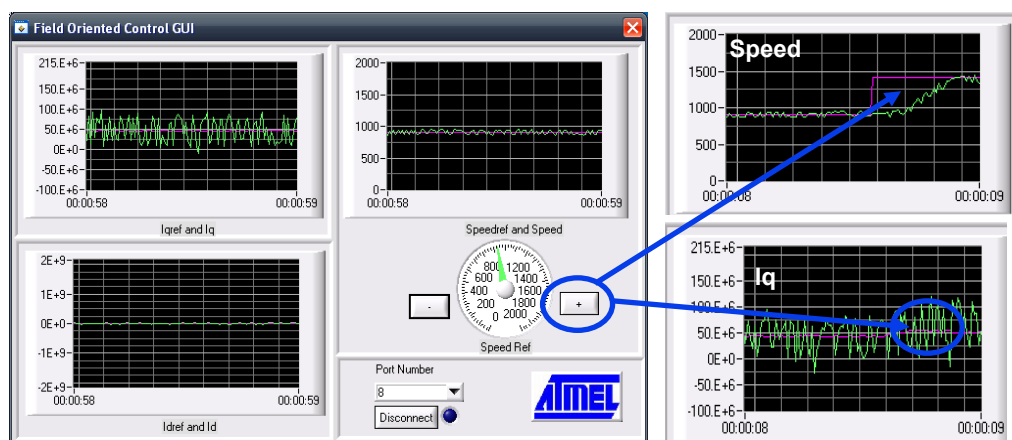
Figure 4-11. Field Oriented Control GUI



5.11.1.1 Increase Speed Value

When the speed reference increases with constant resistive torque value, the I_q increases smoothly to target the new speed value. The I_d remains at '0'.

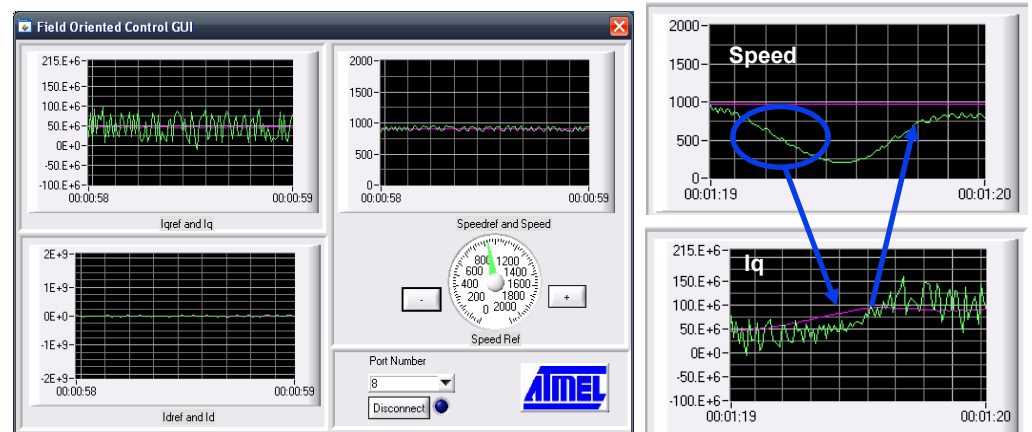
Figure 4-12. Increase Speed Value Step



5.11.1.2 Increase Resistive Torque Value

The resistive torque value increases so the measured speed value decreases. To compensate it, the FOC algorithm should increase the I_q reference and finally the speed is regulated. The I_d remains at '0'.

Figure 4-13. Increase Resistive Torque Value Step



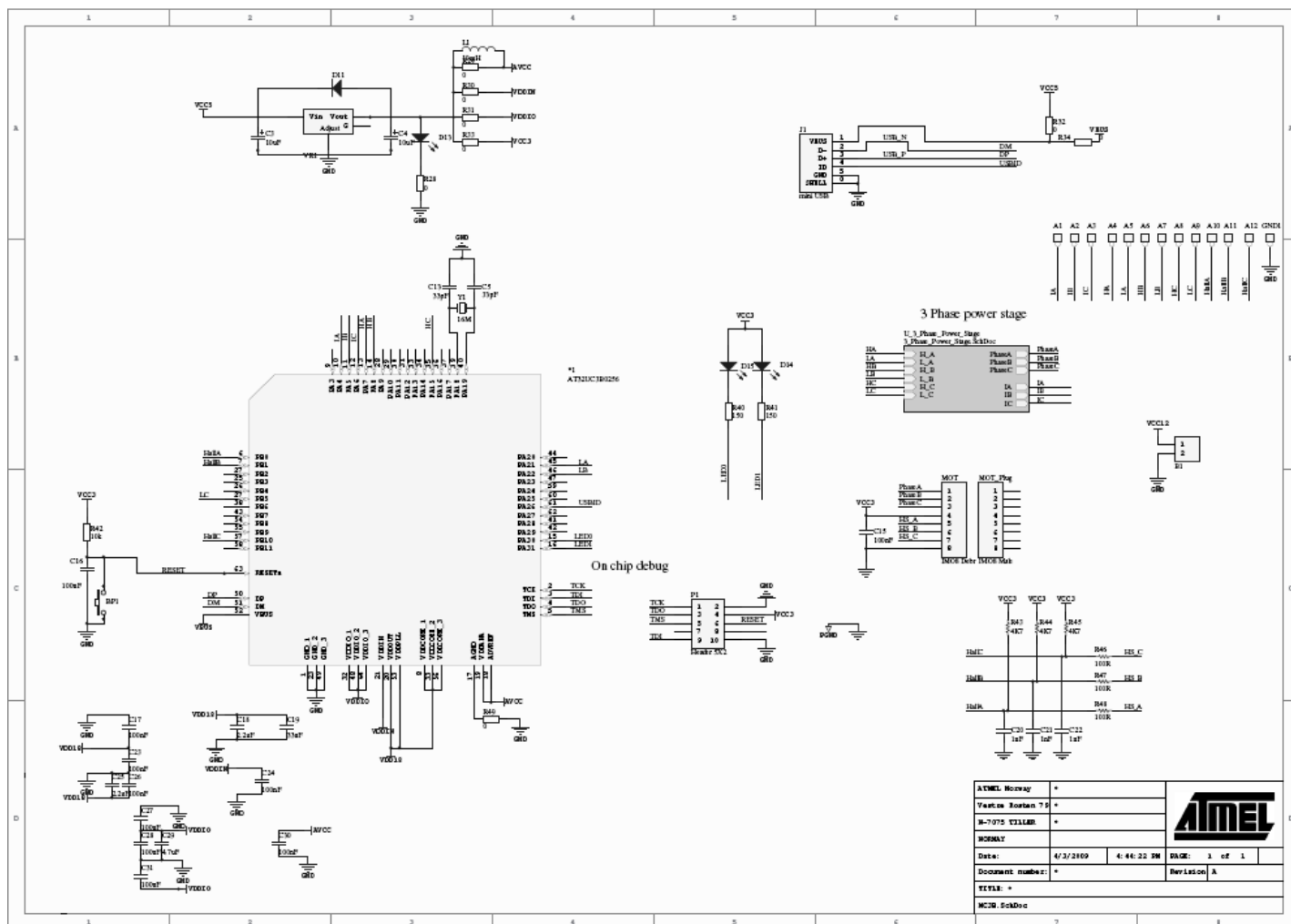
6 Reference

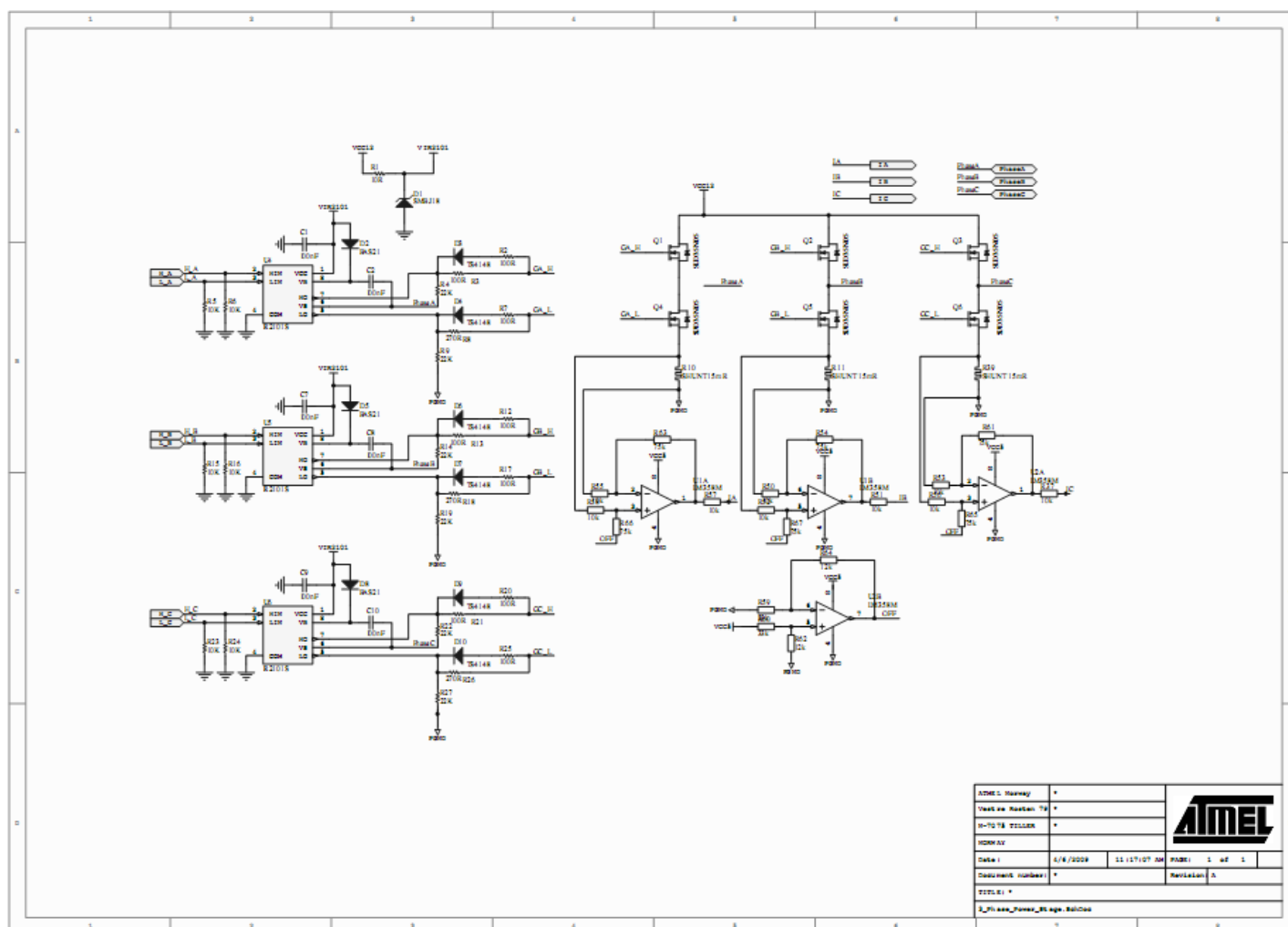
[1] Commande vectorielle sans capteur de position d'une machine synchrone à aimants permanents – Atmel Nantes & IREENA-

[2] NI LabWindows™/CVI <http://www.ni.com/lwcvl/>

7 Appendix

Figure 14. Hardware Schematic.





ATMEL Norway	*
Veritas Heston TS	*
Veritas Heston TS	*
Veritas Heston TS	*
Date:	1/8/2009 11:17:07 AM
Document Number:	Revision: 1
TITLE:	
S_Power_Frames_01.sch	





Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
Avr32@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Request
www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© 2009 Atmel Corporation. All rights reserved. Atmel®, Atmel logo and combinations thereof, AVR®, AVR Studio® and others, are the registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Windows® and others are registered trademarks or trademarks of Microsoft Corporation in the U.S. and/or other countries. Other terms and product names may be trademarks of others.