Implementation of a Speed Controlled Brushless DC Drive Using TMS320F240

Literature Number: BPRA064
Texas Instruments Europe
July 1997

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Contents

List of Figures

List of Tables

Implementation of a Speed Controlled Brushless DC Drive Using TMS320F240

ABSTRACT

The DSP Controller TMS320F240 from Texas Instruments is suitable for a wide range of motor drives. TMS320F240 provides a single chip solution by integrating on-chip not only a high computational power but, also, all the peripherals necessary for electric motor control. The main benefits are increased system reliability and cost reduction of the overall system. The present application note describes how a speed controlled brushless DC drive can be implemented using TMS320F240 and what kind of results can be achieved.

1. Introduction

Permanent magnet synchronous machines with trapezoidal Back-EMF and (120 electrical degrees wide) rectangular stator currents are widely used as they offer the following advantages: first, assuming the motor has pure trapezoidal Back EMF and that the stator phases commutation process is accurate, the mechanical torque developed by the motor is constant; secondly, the Brushless DC drives show a very high mechanical power density. This application report covers the TMS320F240 DSP Controller and some system considerations to get out high performances from a BLDC motor drive.

2. Brushless DC Drive Presentation

2.1. The Motor

The synchronous machine with permanent magnets used in this application note is a three phase Y wound motor. It has one magnetic pole pair on the rotor. The stator phase inductance is 40mH and the phase resistance is 190mΩ. The maximum permissible current at 5000rpm is 4.3A and the torque constant is equal to 17.2mNm/A. Its electromagnetic force waveform is trapezoidal and it is supplied with direct current. The DC bus voltage is 12V.

2.2. The Control Hardware

The control hardware can be either the TMS320F240 Evaluation Module or the MCK240 introduced by Texas Instruments. In this application, the second board can be plugged directly onto the power electronics board. The two boards contain a DSP controller TMS320F240 and its oscillator, a JTAG and an RS232 link and the necessary output connectors. See the figure below which depicts the EVM board.

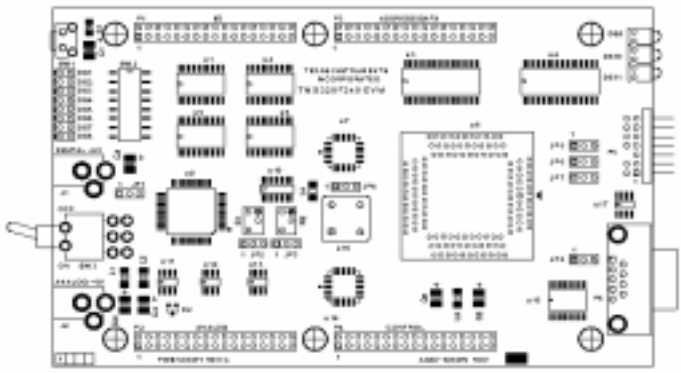

Figure 1: Top View of TMS320F240 EVM Board

2.3. The Power Electronics Hardware

The power board is designed to support a 12V DC voltage supply and a 300W power range. The converter topology support either sinusoidal currents (Three phases ON operation) or direct currents (Two phases ON operation). The latter control method is implemented in this application note. Figure 3 shows the converter which is used here. The power switches use the power MOSFET, type IRFP054. The selected pre-driver component is the IR2131 and the PWM output signals coming from TMS320F240 are directly connected to the pre-driver without any additional buffer. The pre-driver output signals go through a resistor and then directly to the power switches. The relative ground of the upper half bridge is implemented with bootstrap capacitors. This hardware configuration allows hard chopping as well as soft chopping operation. All the elements for protection of the power device securities are provided: Shutdown, Fault, Clearfault, Itrip, reverse battery diode, varistor peak current protection. The current sensing is ensured by a low cost shunt resistor, its voltage drop is directly interfaced with the TMS320F240 as shown in Figure 3. The break feature is accomplished with another MOSFET and a power resistor. Finally, the power board supports the voltage supply for position sensors such as Hall effect sensors and incremental encoders.

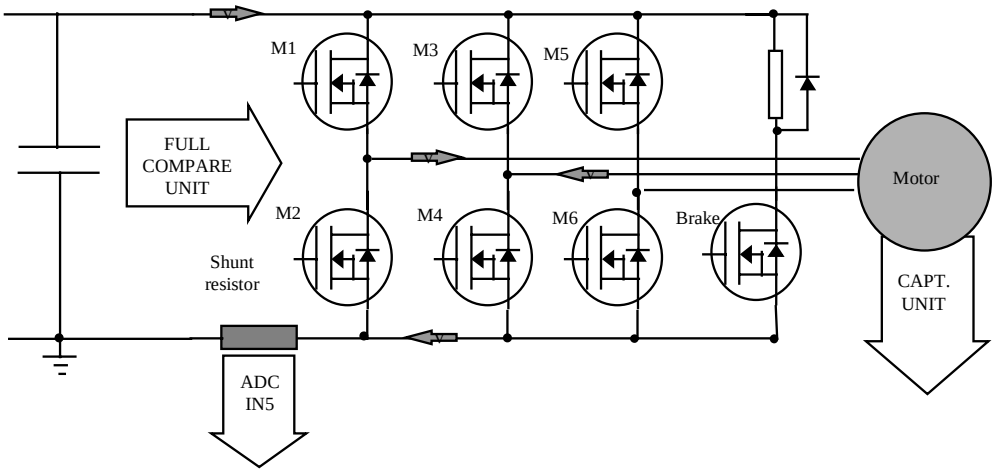

Figure 2: Power Electronics Topology

The bold arrows on the wires depict the Direct Current flowing into two motor phases during the pulsed signals Turn ON.

2.4. The Control Algorithm

According to Reference [3] (See list at the end of this report), torque production is almost directly proportional to the phase current. This statement gives rise to the following BLDC motor speed control scheme:

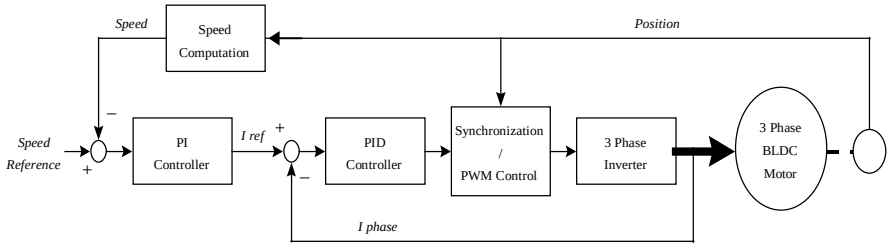

Figure 3: Speed & Current Control Loop Implemented in this Application

In this control scheme, torque production follows the principle that current should flow in only two of the three phases at a time. Only one current at a time needs to be controlled so that only one current sensor is necessary. The positioning of the current sensor allows the use of a low cost sensor as a shunt.

3. Sensing and Regulation

3.1. Current Sensing

As shown in Figure 3, a shunt resistor is used to sense the current. The shunt is inserted between the sources of the power bridge lower side and the power board ground. Its value is set such that it activates the integrated over-current protection when the maximum current permitted by the power board has been reached. The voltage drop across the shunt resistor is converted by the dual ADC module, once it has been amplified to address the whole conversion range. The end of the conversion generates an interrupt flag (INT6 of the DSP Controller core) and the Interrupt Sub Routine is executed. The ADC reading for 30A will be 3FFh and for 0A will be 0h.

The phase current is sensed every 50μs in order to implement a 20kHz current loop. Each current measurement leads to a new PWM duty cycle loaded at the beginning of a PWM cycle. Note that, during Turn OFF, the shunt resistor does not have this current to sense, regardless of whether the inverter is driven in hard chopping or in soft chopping mode. The figure below depicts the shunt current in soft chopping mode and shows that in the Turn OFF operation the decreasing current flows through the M2 free wheeling diode and through the maintained closed M4 (so there is no current observable in the shunt in this chopping mode during Turn OFF). This implies that it is necessary to start a current conversion in the middle of the PWM duty cycle. The TMS320F240 event manager can handle this by starting a conversion on the Timer1 (dedicated in this application to PWM unit) period match.

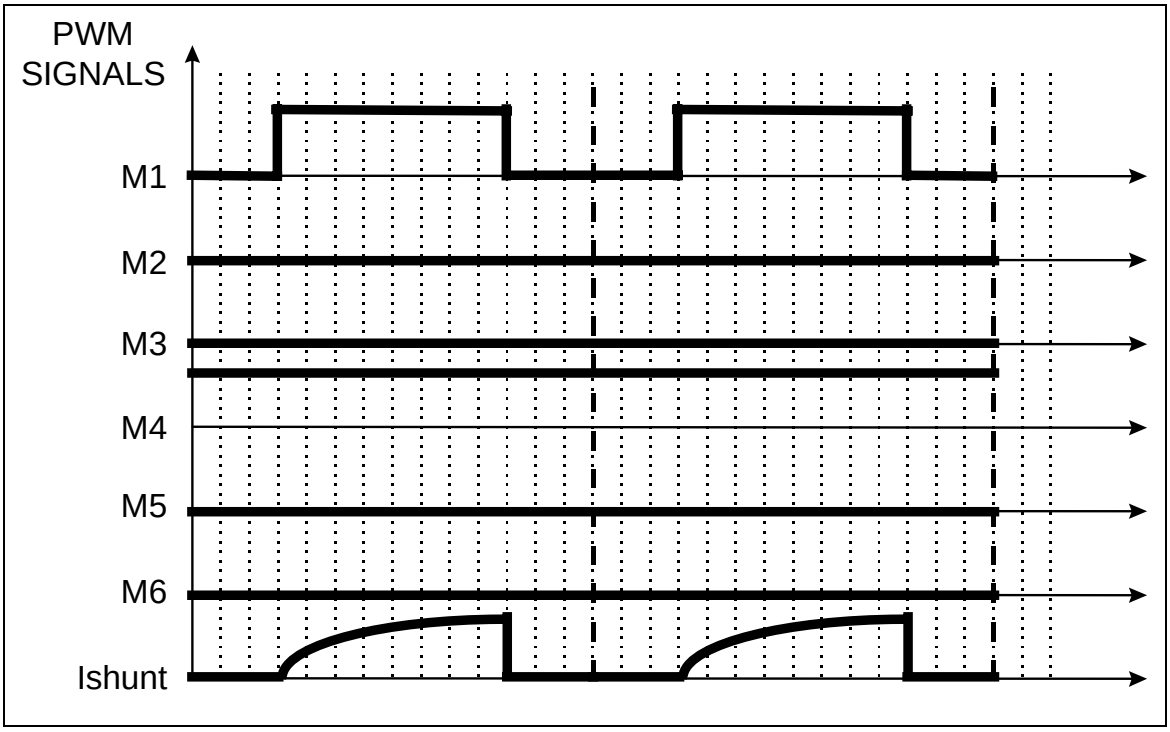

Figure 4: Shunt Resistor Voltage Drop according to PWM DutyCycles (Soft Chopping)

In the hard chopping mode during the Turn OFF neither M1 nor M4 drive current, so that the decreasing phase current flows from ground through the shunt resistor via M2 and M3 free wheeling diodes and back to ground via the capacitor. In this chopping mode it is possible to see the exponentially decreasing phase current across the shunt as a negative shunt voltage drop appears. Assuming that neither the power board nor the control board support negative voltages, this necessitates that the current be sensed in the middle of the Turn ON.

3.2. Position Sensing

The motor in this application is equipped with three Hall effect sensors. These sensors are fed by the power electronics board. The sensor outputs are directly wired to the TMS320F240 Capture Unit. This unit is configured by software to capture 4 inputs (only 3 necessary) and not like QEP. Timer2 is selected as the time base for the capture unit. As this is set in continuous up-count mode and as the slowest reachable speed is governed by the largest acceptable period, T2PER (Timer 2 Period) register is set to 0xFFFEh and the prescaler is set to 128. This setting allows the speed to go down to 24rpm (with CPUCLOCK=20MHz). The Hall effect sensors give three 180° overlapping signals, thus providing the six mandatory commutation points: The rising and falling edges of the sensor output are detected, the corresponding interrupt flag is generated and the Interrupt Sub Routine is served (Event Manager Interrupt group C). The ISR first

determines which edge has been detected, then computes the time elapsed since the last detected edge and commutates the supplied phases.

3.3. Speed Sensing

The speed feedback is derived from the position sensor output signals. As mentioned in the previous paragraph, there are six commutation signals per mechanical revolution. In other words, between two commutation signals there are 60° mechanical degrees. As the speed can be written as:

$$\Delta\theta / \Delta T$$

where θ is the mechanical angle it is possible to get the speed from the computed elapsed time between two captures. Between two commutation signals the angle variation is constant as the Hall effect sensors are fixed relative to the motor, so speed sensing is reduced to a simple division. It is possible to overcome the floating point characteristic of the F240 DSP Controller with extended arithmetic algorithms to perform the division, but in this application a simple 16 bits division is sufficient. In addition, the error caused by neglecting the division remainder becomes smaller as the speed increases.

3.4. Current Regulation

Current regulation is achieved by Pulse Width Modulation (fixed frequency 20kHz) signals with varying duty cycles. PWM width is determined by comparing the measured phase current with the desired reference current as shown below:

$$I_{error} = I_{ref} - I_{measured}$$

$$duty\_cycle_{new} = duty\_cycle_{old} + I_{error} \times K$$

$$if\ \ duty\_cycle_{new} \geq Timer\_period$$

$$then$$

$$duty\_cycle_{new} = Timer\_period$$

$$if\ duty\_cycle_{new} \leq O$$

$$then$$

$$duty\_cycle_{new} = O$$

where, K is the proportional gain and depends on motor parameters and also on DC bus voltage and currents. The proportional gain K can be determined using the following procedure.

Let *'S'* be the number of steps allowed in one PWM cycle. Let Δi be the change in phase current for 100% change in PWM duty cycle. The proportional gain, K, can be defined as:

$$K = \frac{S}{\Delta i}$$

parameter Δi depends on motor and converter types.

3.5. Speed Regulation

The speed regulation loop is executed once every 62.5ms. This frequency can be modified according to the need of the application. In this control software the speed reference is directly written into memory space. It is possible to get the speed reference from an external potentiometer or from PC interface by using a ADC channel or the RS232 serial link. The new direct current reference is computed from the speed error and by means of a common digital PI algorithm:

$$Idcref_k = Idcref_{k-1} + Kp(E_k - E_{k-1}) + KiTE_{k-1}$$

where $Idcref$ is the speed regulator output, Kp the proportional gain, Ki the integral gain and T the speed sampling period. The sampling period is equal to 0.0625 that is to say 2^{-4}. From a numerical standpoint the multiplication KiT is done by right shifting the integral term four times before adding it to the others. See below for management of other speed loop sampling frequencies.

In the speed loop code every variable is stored into a stack. The stack pointer used is the auxiliary register AR2 and the stack entry address is 0x0300h.

Table 1: Speed Loop Variables Stack

STACK ADDRESS	VARIABLE
0x0300h	Capture k instant
0x0301h	Capture k-1 instant
0x0302h	Time elapsed between Capture k and k-1
0x0303h	Speed error k
0x0304h	Speed error k-1

4. Startup Operation

At zero speed the control software has first to determine which phase it should supply. So first of all the code reads the three Hall effect sensor output sequence and applies a voltage on the appropriate phase pair. This operation causes the motor to start rotating. The supplied phase pair is then changed at each Hall sensor output edge. The applied current does not change until the first speed loop entry. The user may want to immediately update the current reference, in this case just set *speed_count* to the speed loop frequency in the variable initialization.

5. PWM Strategy and Generation

The motor in this application is fed with Direct Current and two phases ON. In other words one phase feeds the motor with current, another phase is the current return path and the third phase is opened (See Figure 2). So at any time there are only two power switches conducting current. Let M1 and M4 be the switch pair considered. There are two ways to get the desired current into the right phase pair: either hard chopping or soft chopping operation. In the case of soft chopping the M4 power switch is maintained closed during all the 60° commutation while M1 switches according to the current loop output duty cycle. In the hard chopping case both M1 and M4 switch according to the same pulsed pattern. TMS320F240 supports the two chopping operation as proven below.

Soft chopping mode with CPUCLOCK=20MHz, PWM period=20kHz,
PWM duty cycle = comp, active switches M1 and M4.

```
LACC   COMP
LDP    #0E8h
SPLK   #0F3Dh,ACTR
SACL   CMPR1
SPLK   #0000, CMPR2
SPLK   #0000,CMPR3
```

Hard chopping mode with CPUCLOCK=20MHz, PWM period=20kHz,
PWM duty cycle = comp, active switches M1 and M4.

```
LACC   COMP
LDP    #0E8h
SPLK   #0F7Dh,ACTR
SACL   CMPR1
SACL   CMPR2
SPLK   #0000,CMPR3
```

Discussion of the advantages of one chopping mode versus another is beyond the scope of this application note. Hard chopping has been implemented here.

6. Experimental Results

The three following pictures show the current regulation results. The current reference was set to 1A, 2A and 4A respectively. The scope channel #4 is the phase A current measured by a current probe. The scope channel #3 is the phase A terminal voltage (this voltage has been measured between the phase A half bridge and the power board ground). This voltage is sensed directly on the upper side switch source. These figures show very small dV/dt and dI/dt.

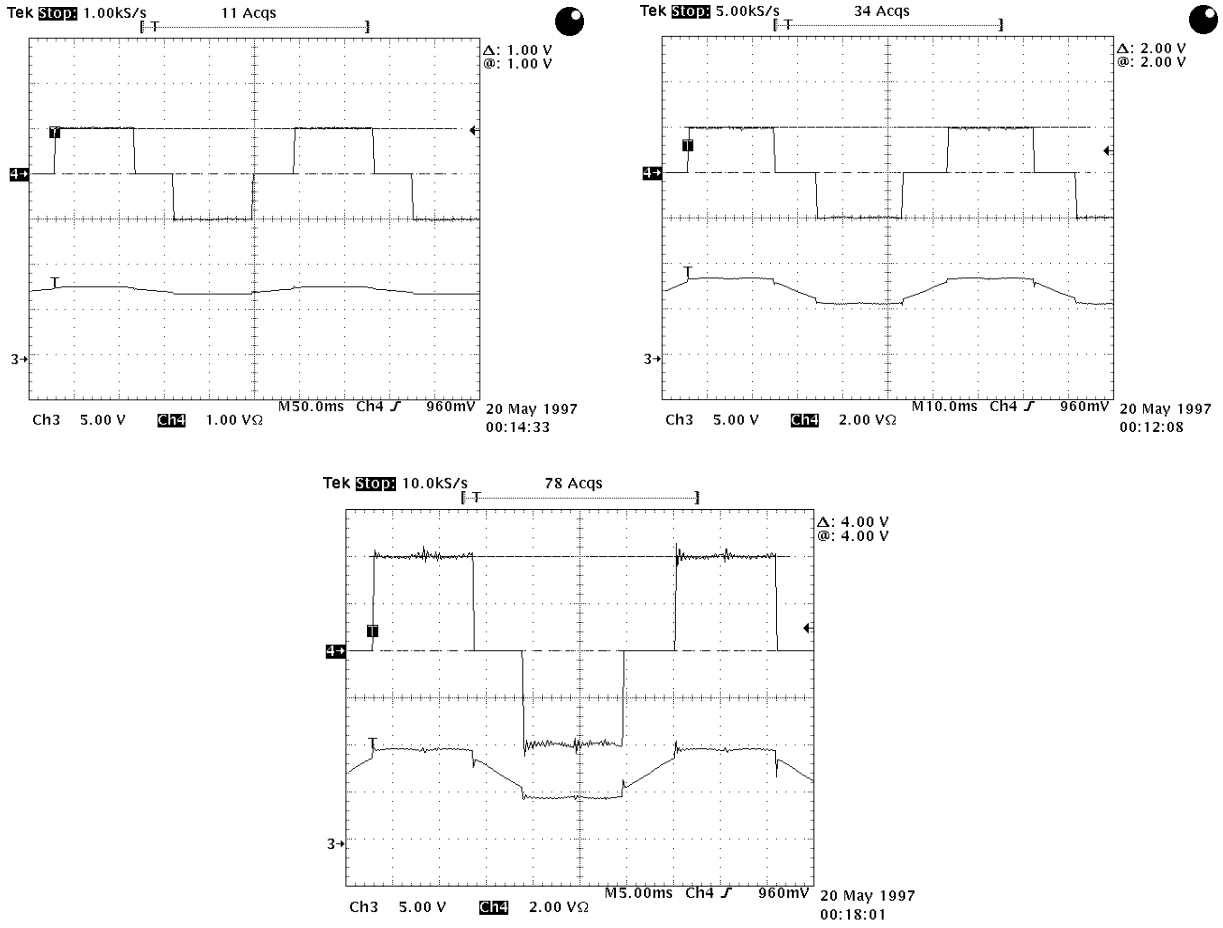

Figure 5: Current Regulation Results

The next two Figures are the Phase A current and half bridge voltage measured as the motor runs with closed loop speed control and a speed reference set to 700rpm. Two alternative braking torques are available to the regulation loop: 0.03 and 0.06Nm (Maximum Produced Torque =Torque constant * maximum continuous current = 17.2*4.37 =75.2mNm).

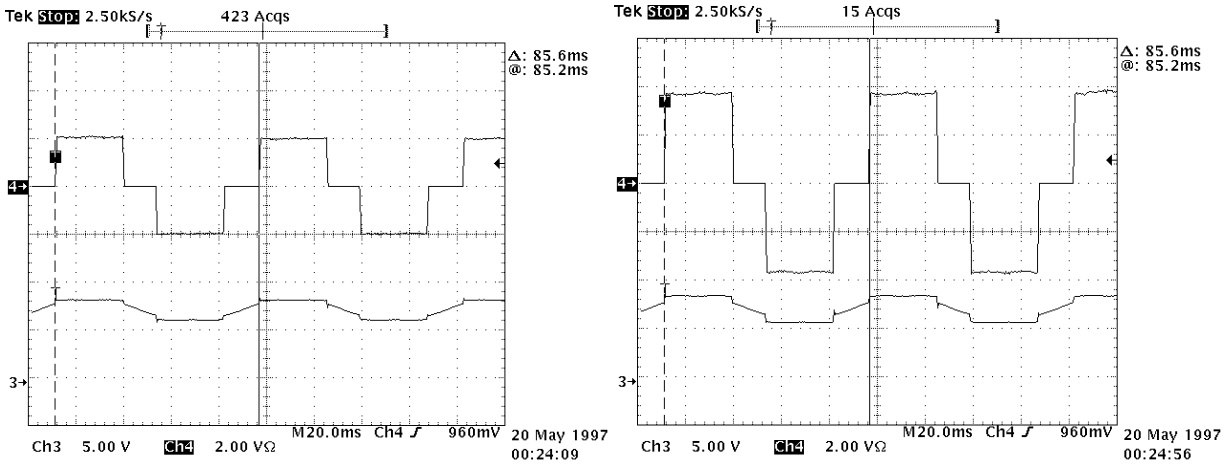

Figure 6: Speed Regulation Results (Spdref = 700rpm)

The next two pictures shows the motor running closed loop at the reference speed of 1500rpm with different braking torque (0.03, 0.06Nm respectively 40% and 80% of the maximum torque).

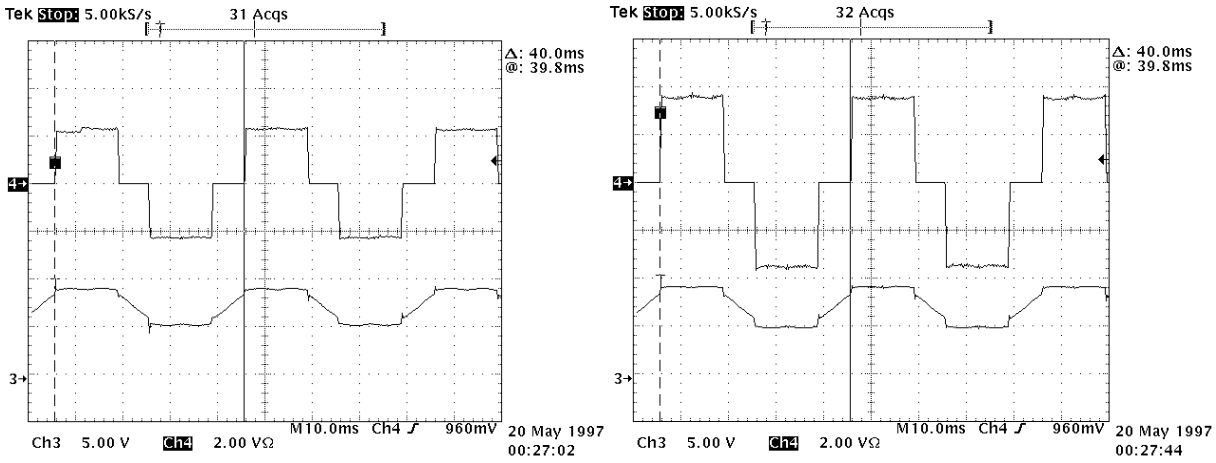

Figure 7: Speed Regulation Results (Spdref=1500rpm)

The last following figure shows the motor running closed loop at the reference speed of 2300rpm and its maximum deliverable torque.

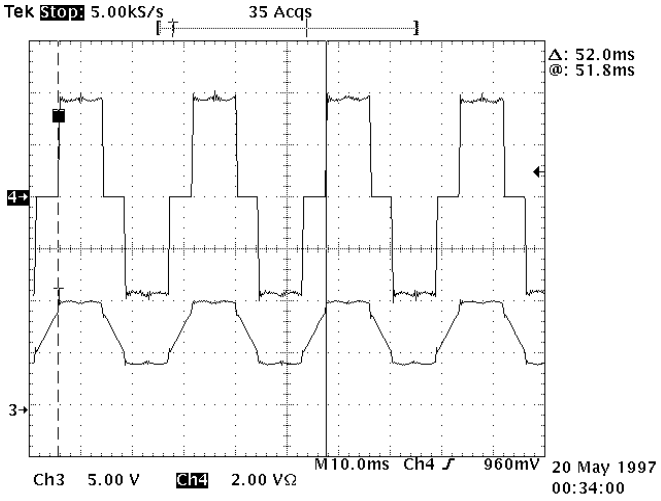

Figure 8: Speed Regulation Results (Spdref=2300rpm)

The speed response time in this application is below 2 seconds and the speed ripple is below 2%. The tested speed range is from 100rpm to 3000rpm.

7. How to Use the Code in another Application

The code given in the next chapter is suitable for one BLDC motor drive with constant braking torque and very high load inertia. As system parameters such as motor phase resistance and electrical and mechanical time constants change with the application, the software may be improved to get the best performances. The main part to change will probably be the regulator coefficients and the output limitation.

7.1. Current Regulator

As seen in chapter 3.4 the current regulator output is

$$duty\_cycle_{new} = duty\_cycle_{old} + I_{error} \times K$$

In the present software the gain K is set to 1. It may be very easily set to 2^x by right or left shifting the current error. If another factor is needed a new coefficient K should be set at the start of the software and the code should be modified as below

```
;Current error calculation
       CLRC    SXM
       LACC    ADCFIFO1,10
       LDP     #0
       SUB     Idc_ref,16
       SETC    SXM

;Current regulator
       SACH    Idc
       LT      Idc
       MPY     #K
       PAC
       ADD     COMP
       SACL    COMP
```

7.2. Speed Regulator

As the speed regulator implemented is a common PI algorithm, there are three parameters free to be changed in order to get good dynamic behaviour: the speed loop frequency T, the proportional gain Kp and the integral gain Ki. As a first step, only Kp and Ki should be adapted. Then it may be necessary to increase the speed loop frequency: change the maximum value of *speed_count* variable and pay attention to the KiT numerical dynamic.

The user may want a speed error limitation in order to avoid either numerical overflows or a too high input current. The following code performs this limitation and is inserted between the end of Speed and speed error calculation block and Speed regulator:

```
       LAR     AR2,#0303h
       BGEZ    LIMIT
       ABS
       SPLK    #-1,SPEED_COUNT
LIMIT
```

```
                SACL    *
                SUB     #01Ah           ;0X01Ah is the max speed error
                BLEZ    OK
                SPLK    #01Ah,*
        OK
                LT      SPEED_COUNT
                MPY     *
                PAC
```

A very similar code can be used to limit either the speed regulator output or the output rate of change.

8. BLDC Motor Drive Code

8.1. Assembly Code

There are seven variables used in this software. The CAPT variable is used by the SEQUENCE sub routine to determine which 60° sector the rotor is in and thus activate the correct pulsed signal pair. The COMP variable is the duty cycle update, it varies from 0 to 500 in this application. Notice that the closer to zero this value becomes, the bigger the phase current. Idc is the variable used to store the line current sensed by means of the ADC module and of the voltage drop across the shunt resistor. Idc_ref is the speed regulator output. It can be interpreted as the torque reference value given by the speed regulation. SPEED_REF is the variable in which the reference speed value can be stored. This reference may come from the conversion of an external potentiometer voltage drop or from an user interface (the speed reference can be set via serial communication from a PC for example). SPEED_COUNT is a software counter (software clock frequency is equal to the PWM frequency 20kHz) which determines the speed regulation loop period. This variable can be set into the Current end of conversion Interrupt subroutine. Changing the counter period value also means changing the speed regulator integral gain. The final stack variable is the start address of the context switching necessary software stack. The memory space allocated to this stack is 6, so that it can be stored up to 6 registers.

```
;*******************************************************
; File Name      : sensor.asm
;
; Target System  : F240 EVM, JTAG, XDS510
;
;Originator      :Pierre Voultoury
;                 TI France
;                 May 97
;
; Description    : This is the BLDC motor drive code
;*******************************************************

                .include"c240app.h"
;-----------------------------------------
```

```
;Speed regulator coefficients setting
;----------------------------------------
Kp              .set    015
Ki              .set    004
;-------------------------------------------------------------
; Variable definitions
;-------------------------------------------------------------
                .bss    CAPT,1          ; Capture indication
                .bss    COMP,1          ;Duty cycle
                .bss    Idc,1           ;Line current
                .bss    Idc_ref,1       ;Current reference
                .bss    SPEED_REF,1     ;Speed reference
                .bss    SPEED_COUNT,1   ;Speed loop frequency
                .bss    stack,6         ;Context save stack
;=============================================================
; Reset & interrupt vectors
;=============================================================
                .sect   "vectors"
RSVECT          B       _c_int0         ; PM 0  Reset Vector
                .space  16*6
INT4            B       CAPINT
                .space  16*2
INT6            B       ADCINT
                .space  16*38



                .text
                .global _c_int0
_c_int0
                SETC    CNF
                CLRC    OVM             ; Reset overflow mode
                SETC    SXM             ; Reset sign extension mode
                CLRC    XF
                SETC    INTM            ; Set global interrupt mask

                MAR     *,AR2           ;Init speed variables stack
                LAR     AR2,#0300h
                SPLK    #0,*+
                SPLK    #0,*+
                SPLK    #0FFFFH,*+
                SPLK    #029H,*+
                SPLK    #029H,*+
                SPLK    #0,*
                LAR     AR2,#0300h
                LAR     AR1,#stack      ;Init context save stack pointer

          ;Disable watchdog (Vccp=5v)& watchdog counter reset p6-12
                LDP     #00E0h
                SPLK    #0006Fh, WD_CNTL
                SPLK    #05555h, WD_KEY
                SPLK    #0AAAAh, WD_KEY

          ; initialize WDT registers
```

```
                SPLK    #06Fh, WD_CNTL  ; clear WDFLAG, Disable WDT, set WDT
                                        ; for 1 second overflow.

        ; Set up CLKOUT to be SYSCLK p6-6
                SPLK    #40C0h,SYSCR
                LACC    SYSSR
                AND     #069FFh
                SACL    SYSSR

        ;set up PLL clockin=10Mhz,CPUCLOCK=20Mhz,SYSCLK=10Mhz
                SPLK    #0002h,CKCR0    ; PLL disabled
                SPLK    #00b1h,CKCR1
                SPLK    #0081h,CKCR0
        ;Set up zero Wait States for external memory
;               MAR     *,AR1
;               LACC    #0008h
;               SACL    *
;               OUT     *,WSGR

                ;I/O setting p11-11
                LDP     #00E1h
                SPLK    #0000fh, OCRA
                SPLK    #0070h, OCRB
                SPLK    #02800h, PBDATDIR    ;not shut down clearing fault
                LACC    PBDATDIR

        ; Clear EV control registers
                LDP     #0E8h
                SPLK    #0000h,T1CON    ;no timer enable
                SPLK    #0000h,T1PER
                SPLK    #0000h,T1CNT
                SPLK    #0000h,T1CMP
                SPLK    #0000h,T2CON
                SPLK    #0000h,T2PER
                SPLK    #0000h,T2CNT
                SPLK    #0000h,T2CMP
                SPLK    #0000h,T3CON
                SPLK    #0000h,T3PER
                SPLK    #0000h,T3CNT
                SPLK    #0000h,T3CMP
                SPLK    #0000h,COMCON
                SPLK    #0000h,DBTCON
                SPLK    #0000h,ACTR
                SPLK    #0000h,SACTR
                SPLK    #0000h,CMPR1
                SPLK    #0000h,CMPR2
                SPLK    #0000h,CMPR3
                SPLK    #0000h,SCMPR1
                SPLK    #0000h,SCMPR2
                SPLK    #0000h,SCMPR3
                SPLK    #0000h,CAPCON   ;no capture
                SPLK    #00ffh,CAPFIFO
                LACC    FIFO1
                LACC    FIFO2
```

```
       LACC    FIFO3

;GP Timer 2 Setting  start with GPT1 presc 128
       SPLK    #0fffeh,T2PER
       SPLK    #00000h,T2CNT
       SPLK    #17C0h,T2CON

;PWM Unit setting
       SPLK    #00500,T1PER
       SPLK    #0000h,T1CNT
       SPLK    #0FFFh,ACTR
       SPLK    #0508h,DBTCON
       SPLK    #00000,CMPR1
       SPLK    #00000,CMPR2
       SPLK    #00000,CMPR3
       SPLK    #0287h,COMCON
       SPLK    #8287h,COMCON
       SPLK    #2800h,T1CON
       SPLK    #2840h,T1CON
       SPLK    #0100h,GPTCON

;Capture Unit Setting
       SPLK    #0b0fch,CAPCON
       SPLK    #00ffh,CAPFIFO

;Core Mask Setting
       LDP     #0
       LACC    #028h
       SACL    IMR
       LACC    IFR
       SACL    IFR

;EV Mask Setting, Vector & Flag reset p11-46
       LDP     #0E8h
       LACC    IFRA
       SACL    IFRA
       LACC    IFRB
       SACL    IFRB
       LACC    IFRC
       SACL    IFRC
       SPLK    #0,IMRA
       SPLK    #0,IMRB
       SPLK    #7,IMRC
       LACC    IVRA
       LACC    IVRB
       LACC    IVRC

;ADC Unit setting p3-8
       LDP     #0E0h
       SPLK    #0403h,ADCTRL2
       SPLK    #1b0ah,ADCTRL1  ;ADCIN5

       CLRC    INTM
```

```
                LDP     #0                ;Variables init
                SPLK    #001FH,Idc_ref
                SPLK    #040H,SPEED_REF
                SPLK    #0500,COMP
                SPLK    #0000,SPEED_COUNT
                SPLK    #0000H,CAPT

*******************************
*STARTING PROCEDURE*
*******************************
          ;Disable CAPT Input p2-75
                LDP     #0E8h
                SPLK    #8000h,CAPCON
                SPLK    #00ffh,CAPFIFO

          ;Set I/O unit as I
                LDP     #00E1h
                SPLK    #0FF00h, OCRB
                SPLK    #0000h, PCDATDIR

          ;WHICH SEQUENCE?
                LACC    PCDATDIR
                AND     #0070H
                LDP     #0
                SACL    CAPT
                LACC    CAPT,12
                SACH    CAPT

          ;Disable I/O Unit
                LDP     #0E1h
                SPLK    #0FF70h, OCRB

          ;Capture Unit Setting
                LDP     #0E8H
                SPLK    #0b0fch,CAPCON
                SPLK    #00ffh,CAPFIFO
```

```
****************************************
*END OF STARTING PROCEDURE*
****************************************

FAULT_CLEAR                             ;Init the pre-driver
                LDP     #0E1h
                LACC    PBDATDIR
                AND     #010h
                BZ      FAULT_CLEAR
                SPLK    #02820h,PBDATDIR
                LDP     #0E8h

LOOP                                    ;Polling loop
                NOP
                NOP
                NOP
                B       LOOP

SEQUENCE
                LDP     #0
                LACC    CAPT
                SUB     #1
                SFL
                ADD     #CAPT_DETER
                BACC
CAPT_DETER
                B       FALLING3
                B       FALLING1
                B       RISING2
                B       FALLING2
                B       RISING1

RISING3
                LACC    COMP
                LDP     #0E8h
                SPLK    #0DF7h,ACTR
                SACL    CMPR3
                SACL    CMPR1
                SPLK    #0000,CMPR2
                B       END
FALLING3
                LACC    COMP
                LDP     #0E8h
                SPLK    #07FDh,ACTR
                SACL    CMPR1
                SACL    CMPR3
                SPLK    #0000,CMPR2
                B       END
RISING2
                LACC    COMP
                LDP     #0E8h
                SPLK    #07DFh,ACTR
                SACL    CMPR2
                SACL    CMPR3
```

```
                SPLK    #0000,CMPR1
                B       END
FALLING2
                LACC    COMP
                LDP     #0E8h
                SPLK    #0D7Fh,ACTR
                SACL    CMPR3
                SACL    CMPR2
                SPLK    #00000,CMPR1
                B       END
RISING1
                LACC    COMP
                LDP     #0E8h
                SPLK    #0F7Dh,ACTR
                SACL    CMPR1
                SACL    CMPR2
                SPLK    #0000,CMPR3
                B       END
FALLING1
                LACC    COMP
                LDP     #0E8h
                SPLK    #0FD7h,ACTR
                SACL    CMPR2
                SACL    CMPR1
                SPLK    #00000,CMPR3
END
                RET

SPEED_REG
                MAR     *,AR2
                LAR     AR2,#0302h
                LDP     #0
                SPLK    #1,SPEED_COUNT

          ;Speed and speed error calculation
                CLRC    SXM
                ZAC
                OR      #0FFFFH
                RPT     #15
                SUBC    *
                AND     #0FFFFH
                SUB     SPEED_REF
                NEG
                SETC    SXM

          ;Speed regulation
                LAR     AR2,#0303h
                SACL    *+
                SUB     *+
                SACL    *
                LT      *-
                MPY     #Kp
                LTP     *-
                MPY     #Ki
```

```
                LTD     *
                ADD     Idc_ref,4
                SFR
                SFR
                SFR
                SFR
                SACL    Idc_ref

        ;Reset Speed loop timer
                SPLK    #0,SPEED_COUNT

        ;Restore
                LAR     AR2,#0300H
                RET

ADCINT
        ;save status registers
                MAR     *,AR1
                MAR     *+              ; skip one position
                SST     #1, *+        ; save ST1
                SST     #0, *          ; save ST0

                LDP     #0
                LACC    SPEED_COUNT
                SUB     #1250
                BNZ     NO_SPEED_REG
                CALL    SPEED_REG
NO_SPEED_REG
                LACC    SPEED_COUNT
                ADD     #1
                SACL    SPEED_COUNT

                LDP     #0E0h
                LACC    SYSIVR

        ;current error calculation
                CLRC    SXM
                LACC    ADCFIFO1,10
                LDP     #0
                SUB     Idc_ref,16
                SETC    SXM

        ;current proportional regulator
                ADD     COMP,16
                SACH    COMP

        ;current regulator output limitation
                LACC    COMP
                BGZ     SUP_LIM
                SPLK    #0,COMP
                B       COMP_OK
SUP_LIM
                SUB     #0500
                BLZ     COMP_OK
```

```
                SPLK    #0500,COMP
COMP_OK

          ;output PWM
                CALL    SEQUENCE

          ;restore status registers
                MAR     *, AR1          ; make stack pointer active
                LST     #0, *-          ; load ST0
                LST     #1, *-          ; load ST1

                CLRC    INTM
                RET


CAPINT
                MAR     *,AR1
                MAR     *+              ; skip one position
                SST     #1, *+          ; save ST1
                SST     #0, *           ; save ST0

          ;Capture interrupt handling
                LDP     #0E8h
                LACC    IVRC            ;FLAG RESET

          ;speed calculation
                MAR     *,AR2
                LACL    T2CNT
                SACL    *+
                SUB     *+
                SACL    *-
                ADD     *
                SACL    *-

          ;Disable CAPT Input p2-75
                LDP     #0E8h
                SPLK    #8000h,CAPCON
                SPLK    #00ffh,CAPFIFO

          ;Set I/O unit as I
                LDP     #00E1h
                SPLK    #0FF00h, OCRB
                SPLK    #0000h, PCDATDIR

          ;WHICH SEQUENCE?
                LACC    PCDATDIR
                AND     #0070H
                LDP     #0
                SACL    CAPT
                LACC    CAPT,12
                SACH    CAPT

          ;Commutation
                CALL    SEQUENCE
```

```
;Disable I/O Unit
       LDP     #0E1h
       SPLK    #0FF70h, OCRB

;Capture Unit Setting
       LDP     #0E8H
       SPLK    #0b0fch,CAPCON
       SPLK    #00ffh,CAPFIFO

       MAR     *, AR1          ; make stack pointer active
       LST     #0, *-          ; load ST0
       LST     #1, *-          ; load ST1

       CLRC    INTM
       RET
```

8.2. Linker File

```
/*----------------------------------------------------------*/
/*  LINKER COMMAND FILE - MEMORY SPECIFICATION for C240  */
/*  Last update 24 Sep 96                                   */
/*----------------------------------------------------------*/

MEMORY
{
 PAGE 0 :  VECS   : origin =    0h , length =   040h      /* PROGRAM */
           PROG   : origin =   40h , length =  0800h      /* Ext mem */

 PAGE 1 :  MMRS   : origin =    0h , length =   05Fh      /* MMRS    */
           B2     : origin = 0060h , length =   020h      /* DARAM   */
           B0     : origin = 0100h , length =  0200h      /* DARAM   */
           B1     : origin = 0300h , length =  0200h      /* DARAM   */
}
```

```
          /*--------------------------------------------------------*/
          /* SECTIONS ALLOCATION                                    */
          /*--------------------------------------------------------*/
SECTIONS
{
 .vectors : { } > VECS      PAGE 0   /* INTERRUPT VECTOR TABLE     */
 .text    : { } > PROG      PAGE 0   /* CODE                       */
 .mmrs    : { } > MMRS      PAGE 1   /* Memory Mapped Registers    */
 .blk0    : { } > B0        PAGE 1   /* Block B0 - page 4          */
 .bss     : { } > B2        PAGE 1   /* Block B2 - page 0          */
 .blk1    : { } > B1        PAGE 1   /* Block B1 - page ?          */
}
```

8.3. Header File

```
;***************************************************************************
; File Name:     C240APP.H
; Project:       C240 EVM Apps S/W suite
; Originator/s:  Auto/Ind DSP Apps group - Texas Instruments (Houston)
;
; Description:   C240 Header file with all Peripheral Register declarations
;
;***************************************************************************
;-----------------------------------------------------
; On Chip Peripheral Register Definitions (DATA SPACE)
;-----------------------------------------------------
;C2xx Core Registers
;~~~~~~~~~~~~~~~~~~~
     IMR           .set    0004h         ;Int Mask Register
     GREG          .set    0005h         ;Global memory allocation reg
     IFR           .set    0006h         ;Int Flag Register


;System Module Registers
;~~~~~~~~~~~~~~~~~~~~~~~
     SYSCR         .set    07018h        ;System Module Control Register
     SYSSR         .set    0701Ah        ;System Module Status Register
     DIN           .set    0701Ch        ;Device Identification Register
     SYSIVR        .set    0701Eh        ;System Interrupt Vector Register
     XINT1_CNTL    .set    07070h        ;Int1 (type A) Control reg
     NMI_CNTL      .set    07072h        ;Non maskable Int (type A)
     XINT2_CNTL    .set    07074h        ;Int2 (type C) Control reg
     XINT3_CNTL    .set    07076h        ;Int3 (type C) Control reg
     PDPINT_CNTL   .set    0742Ch        ;Power Drive Protection Int cntl reg

;Digital I/O
;~~~~~~~~~~~
     OCRA          .set    07090h        ;Output Control Reg A
     OCRB          .set    07092h        ;Output Control Reg B
     PADATDIR      .set    07098h        ;I/O port A Data & Direction reg.
     PBDATDIR      .set    0709Ah        ;I/O port B Data & Direction reg.
     PCDATDIR      .set    0709Ch        ;I/O port C Data & Direction reg.
     PDDATDIR      .set    0709Eh        ;I/O port D Data & Direction reg.
```

```
WSGR          .set     0FFFFh         ;Wait State Generator Register
```

```
;Watch-Dog(WD) / Real Time Int(RTI) / Phase Lock Loop(PLL) Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     RTI_CNTR     .set    07021h        ;RTI Counter reg
     WD_CNTR      .set    07023h        ;WD Counter reg
     WD_KEY       .set    07025h        ;WD Key reg
     RTI_CNTL     .set    07027h        ;RTI Control reg
     WD_CNTL      .set    07029h        ;WD Control reg
     CKCR0        .set    0702Bh        ;PLL control reg 0
     CKCR1        .set    0702Dh        ;PLL control reg 1

;Analog-to-Digital Converter(ADC) registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     ADCTRL1      .set    07032h
     ADCTRL2      .set    07034h
     ADCFIFO1     .set    07036h
     ADCFIFO2     .set    07038h


;Event Manager (EV) - EV Base Address = 0x7400
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
     EV_BASE      .set    7400h         ; Event Manager Base Address
     GPTCON       .set    00h + EV_BASE ; General Timer Controls
     T1CNT        .set    01h + EV_BASE ; T1 Counter Register
     T1CMP        .set    02h + EV_BASE ; T1 Compare Register
     T1PER        .set    03h + EV_BASE ; T1 Period Register
     T1CON        .set    04h + EV_BASE ; T1 Control Register
     T2CNT        .set    05h + EV_BASE ; T2 Counter Register
     T2CMP        .set    06h + EV_BASE ; T2 Compare Register
     T2PER        .set    07h + EV_BASE ; T2 Period Register
     T2CON        .set    08h + EV_BASE ; T2 Control Register
     T3CNT        .set    09h + EV_BASE ; T3 Counter Register
     T3CMP        .set    0ah + EV_BASE ; T3 Compare Register
     T3PER        .set    0bh + EV_BASE ; T3 Period Register
     T3CON        .set    0ch + EV_BASE ; T3 Control Register
     COMCON       .set    11h + EV_BASE ; Compare Unit Control
     ACTR         .set    13h + EV_BASE ; Compare Unit Output Action Control
     SACTR        .set    14h + EV_BASE ; S Comp Unit Output Action Control
     DBTCON       .set    15h + EV_BASE ; Dead Band Timer Control
     CMPR1        .set    17h + EV_BASE ; Compare Channel 1 Threshold
     CMPR2        .set    18h + EV_BASE ; Compare Channel 2 Threshold
     CMPR3        .set    19h + EV_BASE ; Compare Channel 3 Threshold
     SCMPR1       .set    1ah + EV_BASE ; S Comp Channel 1 Threshold
     SCMPR2       .set    1bh + EV_BASE ; S Comp Channel 2 Threshold
     SCMPR3       .set    1ch + EV_BASE ; S Comp Channel 3 Threshold
     CAPCON       .set    20h + EV_BASE ; Capture Unit Control
     CAPFIFO      .set    22h + EV_BASE ; FIFO1-4 Status Register
     FIFO1        .set    23h + EV_BASE ; Capture Channel 1 FIFO Top
     FIFO2        .set    24h + EV_BASE ; Capture Channel 2 FIFO Top
     FIFO3        .set    25h + EV_BASE ; Capture Channel 3 FIFO Top
     FIFO4        .set    26h + EV_BASE ; Capture Channel 4 FIFO Top
     IMRA         .set    2ch + EV_BASE ; Group A Interrupt Mask Register
     IMRB         .set    2dh + EV_BASE ; Group B Interrupt Mask Register
     IMRC         .set    2eh + EV_BASE ; Group C Interrupt Mask Register
     IFRA         .set    2fh + EV_BASE ; Group A Interrupt Flag Register
```

```
IFRB        .set    30h + EV_BASE ; Group B Interrupt Flag Register
IFRC        .set    31h + EV_BASE ; Group C Interrupt Flag Register
IVRA        .set    32h + EV_BASE ; Group A Int. Vector Offset
IVRB        .set    33h + EV_BASE ; Group B Int. Vector Offset
IVRC        .set    34h + EV_BASE ; Group C Int. Vector Offset
```

9. Summary

This application report deals with a BLDC motor drive based on the DSP Controller TMS320F240 manufactured by Texas Instruments. The electronic commutation is accomplished by three Hall effect sensors directly wired to the DSP Controller capture Unit. Accurate current regulation is maintained with help of the ADC unit and the speed regulation is performed by the DSP core. This software solution might be considered as a starting point to design a permanent magnet synchronous machine drive with trapezoidal Back-EMF and rectangular stator currents.

References

1. TMS320C24x DSP Controllers - Reference Set: Vol.1, Texas Instruments Inc., 1997.
2. TMS320C24x DSP Controllers - Reference Set: Vol.2 Texas Instruments Inc., 1997.
3. Digital Signal Processor Solutions for BLDC motor, Application Report Texas Instruments part #BPRA055.
4. Brushless Permanent Magnet Motor Design Duane C. Hanselman, McGraw Hill, Inc.