Ground Work for Embedding a Field Oriented Motor Controller into a Single System on a Chip

Final Report, September, 2014

A PROJECT SUPPORTED BY THE NSF/IUCR CENTER FOR EMBEDDED SYSTEMS, AUG-2013 TO MAY-2014

PRINCIPAL INVESTIGATOR: C.J. HATZIADONIU

GRADUATE STUDENT: NIZAR QUAMAR

COLLABORATION FROM HS-UTAS: JEFF NELSON AND FRANK LUDICKY

Abstract

The scope of the project is to implement a motor drive on a system-on-chip (SOC) with the goal of reducing the number of the required discrete components. The motor drive considered in this project is for the permanent magnet synchronous motor and it is based on the field orientation control. The report provides an overview of the drive system, an overview of the implementation into SOC and a detail description of the design.

Contents

1. Executive Summary

The scope of the project is to design a motor drive system using the system-on-chip (SOC) approach with the goal of reducing the number of the required discrete components. An SOC device combines an FPGA and a multicore microcontroller. SOC devices, therefore, have the capability of processing a higher volume of data at higher speeds than a conventional microcontroller. In system design applications (e.g. motor drives), many of the subsystems can be designed into the SOC device thereby reducing the number of discrete components.

The motor drive considered in this project is for the permanent magnet synchronous motor (PMSM) and it is based on the field orientation control (FOC). The speed feedback loop uses a resolver sensor to measure the rotor position. Consequently, the system to be designed incorporates a resolver-to-digital converter (RDC). In addition, the system includes the following components which were incorporated into the SOC: the signal conditioning from the motor current sensors, the transformations of rotational frames, the main FOC loops, and the conversion of the control output to inverter gate signals.

In the previous project an RDC was developed using a microcontroller and a few other discrete components. The microcontroller implementation of the RDC reduced significantly the number of required components.

The approach followed in this project included three phases: in the first phase, the RDC was implemented into the fabric using the previous RDC design. In the second phase, the rotation frame transformations and the PWM module were designed into the fabric in order to take advantage of parallel computation. The main FOC loops, which do not require heavy computational effort were designed into the microcontroller. In the third phase, both the fabric and the microcontroller were tested to verify the design. For this purpose, specific test routines had to be developed.

This report provides an overview of the drive system, an overview of the implementation into SOC and a detail description of the design.

2. Project Description

Figure 1 shows the block diagram of the motor drive system considered in the project. The motor is supplied from a DC source through a PWM inverter. The inverter switching frequency is 10 kHz. A resolver sensor and current transducers are used to measure the motor speed and stator currents respectively. Two resolver voltages and, for symmetric operation, two phase currents are measured. ADC's are used (not shown on the figure) to convert the motor current and resolver voltage measurements to digital signals.

With reference to the same figure, the resolver signals are processed by the RDC subsystem. The RDC algorithm uses a type II loop to demodulate the resolver signals and track the motor rotor position and speed. The RDC subsystem was designed in the previous project.

The phase current signals pass through the ab-dq frame transformation providing the d and q currents of the motor. For the successful completion of the ab-dq and dq-abc transformations, the rotor position is needed. This is derived from the output of the RDC.

The next stage, shown in figure 1, is the FOC controller. The primary function of the FOC is to regulate the motor speed. A secondary function is to regulate the motor power factor (e.g. operating as over excited or under excited). The FOC contains two hierarchically structured outer loops. The speed control loop and the reactive current control loop. The two loops are decoupled through appropriate design so that the dynamic stability of the system is improved.

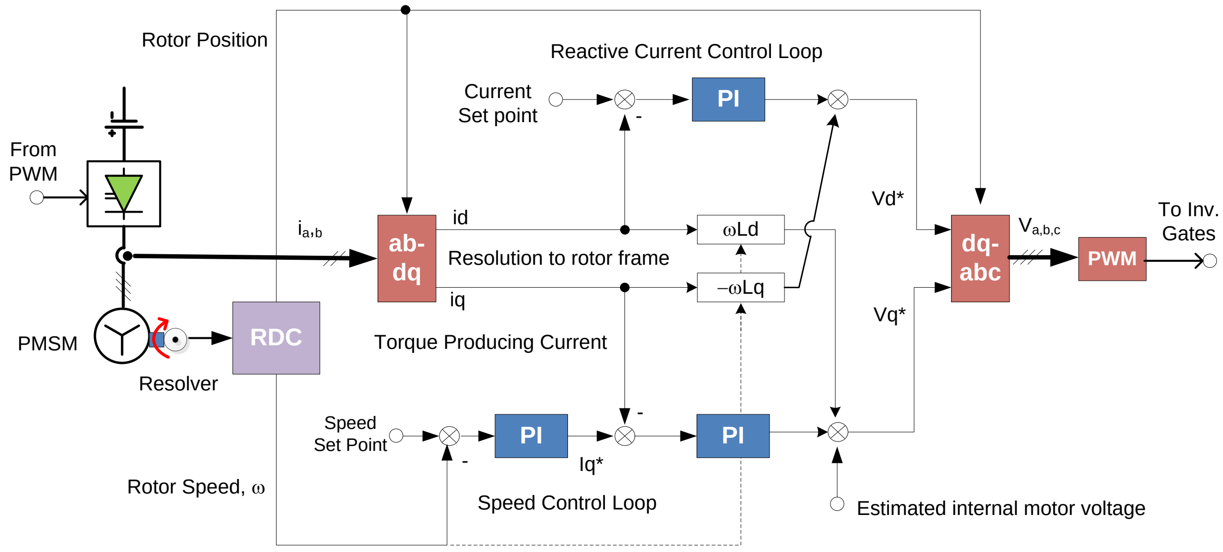

Figure 1: Representation of the Motor Drive.

The speed control loop operates by controlling the q current of the motor; this is the torque producing current. With reference to the figure, the motor speed feedback is derived from the RDC. The speed error is processed through a PI to produce the reference for the q current. An inner loop controls the q current. For this purpose the current error is processed through a PI controller to produce the reference of the q voltage of the stator.

The reactive current controller, shown in the same figure, controls the d current of the motor which is responsible for the power factor. This loop has only one level: the current error is processed through a PI controller to produce the reference of the d voltage of the stator.

The d and q reference voltages are, subsequently, passed through the dq-abc transformation module to derive the stator voltages in the abc frame, as in the figure. Finally, the abc reference voltages are converted to switching logic signals through the PWM module and sent to the inverter gates closing the loop. The PWM functions in the conventional manner utilizing a triangular carrier of fixed frequency (10 kHZ) and the abc reference voltages.

3. System on Chip Design

Figure 2 shows the design approach for the implementation of the motor drive system described in figure 1 into an SOC device. The ***Smartfusion2*** SOC was selected for this project.

With reference to Figure 2, the SOC includes the RDC and the entire FOC including the rotation frame transformations and the PWM signal derivation. The same figure also shows the design main architecture. The RDC, ab-dq, dq-abc and PWM subsystems are implemented in the fabric. The FOC loops are implemented in the microcontroller. In addition, the microcontroller plays a supervisory role for the entire system. The data paths are shown in the same figure. The RDC outputs are used for the ab-dq and dq-abc modules and are also uploaded to the microcontroller to use in the FOC inputs, as per the block diagram in Figure 1.

Data exchanged between the SOC and external devices include the data from the ADC's, the control signals for the resolver excitation, and the gate signals for the inverter. There are four ADC's: two for the quadratic voltages of the resolver and two for the phase a and b stator currents. The resolver excitation consists of a PWM signal with a frequency of 10 kHz.

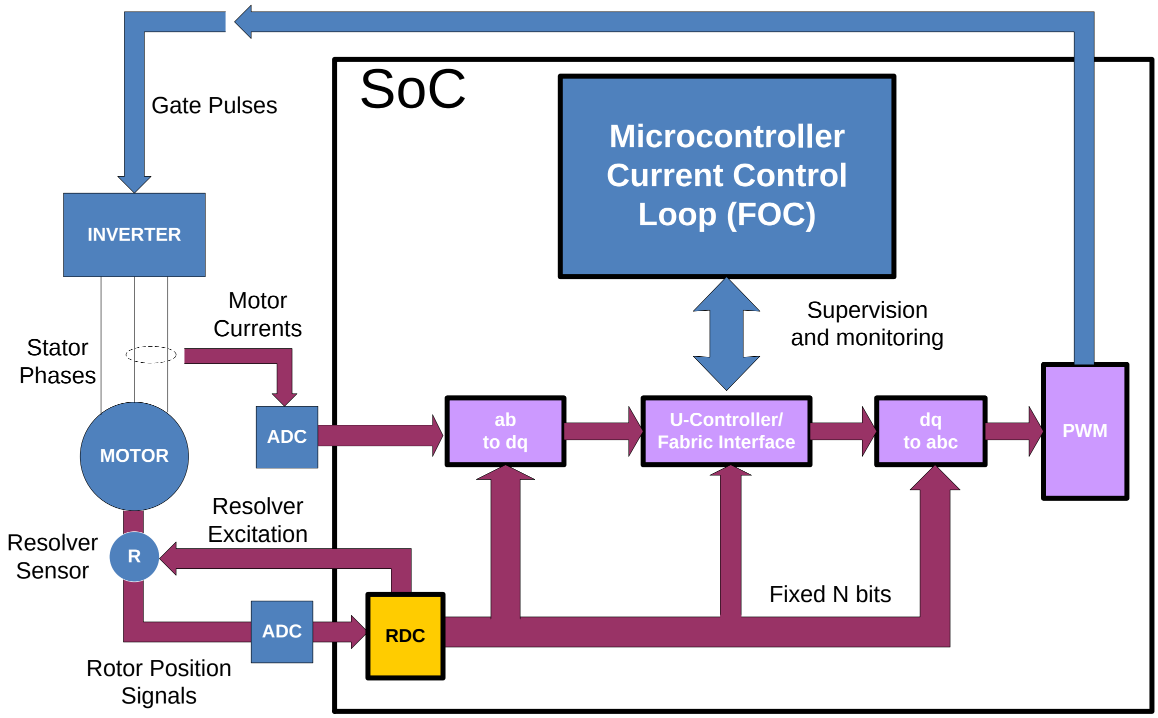

Figure 2: Representation of the SoC Design.

The design detail of the various parts is explained in the following sections of the report.

3.1. Peripheral Components

With reference to figure 2, the peripheral components include the resolver excitation, the ADC interface and the inverter PWM output.

Resolver excitation PWM. The resolver excitation component generates a 10 KHz sinusoidal wave based on the selective harmonic elimination technique (SHE) [1]. By utilizing this technique the 3rd through the 19th harmonic has been eliminated, which left only high order harmonics shown in Figure 3 that can be easily filtered. This component is composed of a counter clocked at 60 MHz with a number of comparator that switch the signal level based on a predefined switching angles shown in table 1.

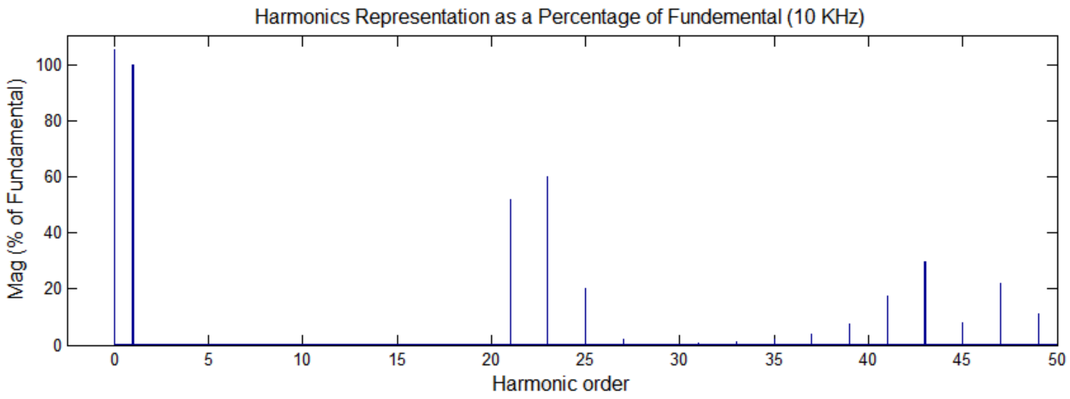

Figure 3: Harmonic spectrum of the designed PWM scheme.

Table 1: Switching angles of the PWM scheme.

Switching angle	Switching angle magnitude (radians)
$\propto_1$	0.148916144
$\propto_2$	0.261327547
$\propto_3$	0.44708158
$\propto_4$	0.525310346
$\propto_5$	0.746757968
$\propto_6$	0.795273217
$\propto_7$	1.051831591
$\propto_8$	1.077303206
$\propto_9$	1.379492596
$\propto_{10}$	1.391117437

ADC interface. The Smartfusion2 SOC does not feature an internal ADC; therefore, the resolver sine and cosine terms along with the motor I_a and I_b currents have to be sampled through an external ADC. The dual channel 12bits AD7912 has been selected due to it is outstanding performance and simple interface.

To sample the four signals two ADC's are arranged in the configuration shown in Figure 4. Even though AD7912 lacks a simultaneous sampling feature, this configuration allows the resolver terms pair and the motor currents pair to be simultaneously sampled by splitting each pair between the two ADCs and trigger both ADCs conversion at the same instance.

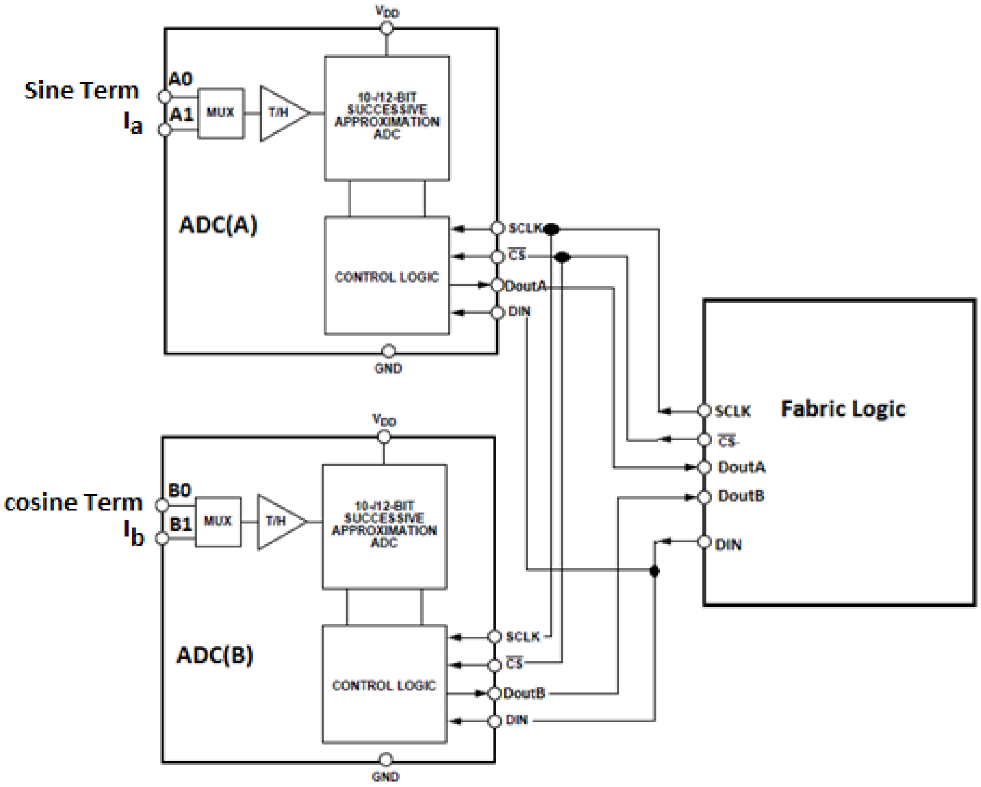

Figure 4: Physical interface between the fabric and ADC

The SOC interfaces the AD7912 and controls it to sample at 40 kHz through a state machine that writes two signals (DIN & $\overline{CS}$) and reads two other signals (DoutA & Dout B). The function and timing of these signals as well as a description of the ADC operation is elaborated in the datasheet [2]. The "ADC_FRAME" signal originated at the Resolver excitation component ensures all time synchronization between the ADC interface state machine and the resolver excitation PWM component. This is important, so the resolver terms are sampled at the excitation signal peaks. The states diagram is shown in Figure 5 and the states functions are as following:

- S0: Reset the counter and write $\overline{CS}$ ="1" & Din ="0".
- S1: write Din word "XX00XXXXXXXXXXXX" to sample (A0 & B0) in normal mode. Dummy read of (A0 & B0) is done at this state.
- S2: write Din word "XX11XXXXXXXXXXXX" to sample (A1 & B1) in normal mode. Read (A0 & B0).
- S3: write Din word "XX00XXXXXXXXXXXX" to sample (A0 & B0) in normal mode, which is ignored at S1. Read (A1 & B1).

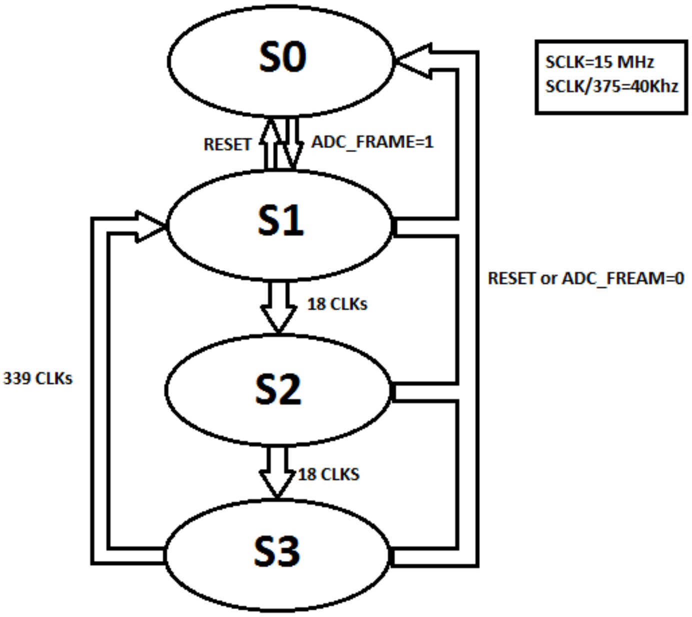

Figure 5: ADC interface state machine

Motor drive PWM. The motor inverter is driven by a sinusoidal PWM scheme with a 10 KHz triangular carrier. The positive zero crossing of the PWM carrier is synchronized with the motor current sampling instances. Functionally, a 2’s complement counter clocked at 60 MHz generates the triangular wave shown in Figure 6, which is compared to the reference signals (V_a, V_B, V_C) to trigger switching.

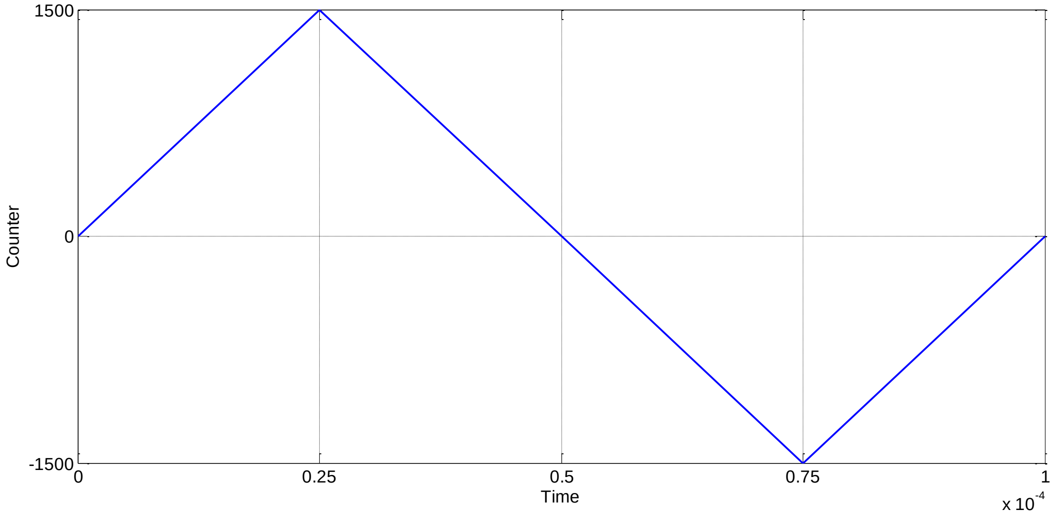

Figure 6: Triangular carrier generation

3.2. Fabric & U-controller Interface

The distribution of system functions between the Fabric & u-Controller demands a communication channel for data to be shared in a timely manner. The hardwired Fabric interface controller (FIC) in the u-controller subsystem is complemented with fabric synthesized cores, which provide flexibility on top of power and space efficiency.

From a variety of options the u-Controller AHBL master was selected, with an AHBL to APB conversion layer to allow fabric component to communicate using the simpler APB protocol as shown in Figure 7. Upon initialization the u-controller load the sine and cosine tables from the eNVM. During normal operation the u-controller read the resolver speed, $I_d$and I_q current from the DSP component, then process it to write back V_d and V_q.

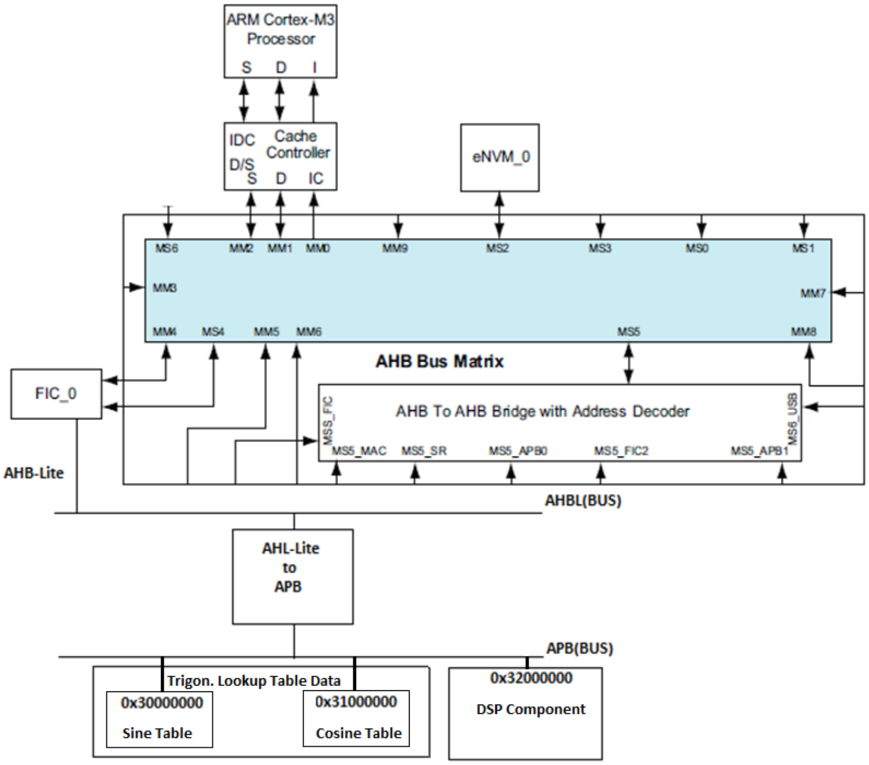

Figure 7: U-controller & Fabric Interface

3.3. Trigonometric Lookup Table Data

The trigonometric lookup table data component is a ram with a wrapper shown in Figure 8, which consist of a RAM accessible by the u-controller and DSP component, a state machine and a mux. On reset the state machine shown in Figure 9 arbitrates RAM access to the u-controller and executes an APB protocol to initialize data. Upon completion, the RAM access is handed over permanently to the DSP component for trigonometric functions evaluation.

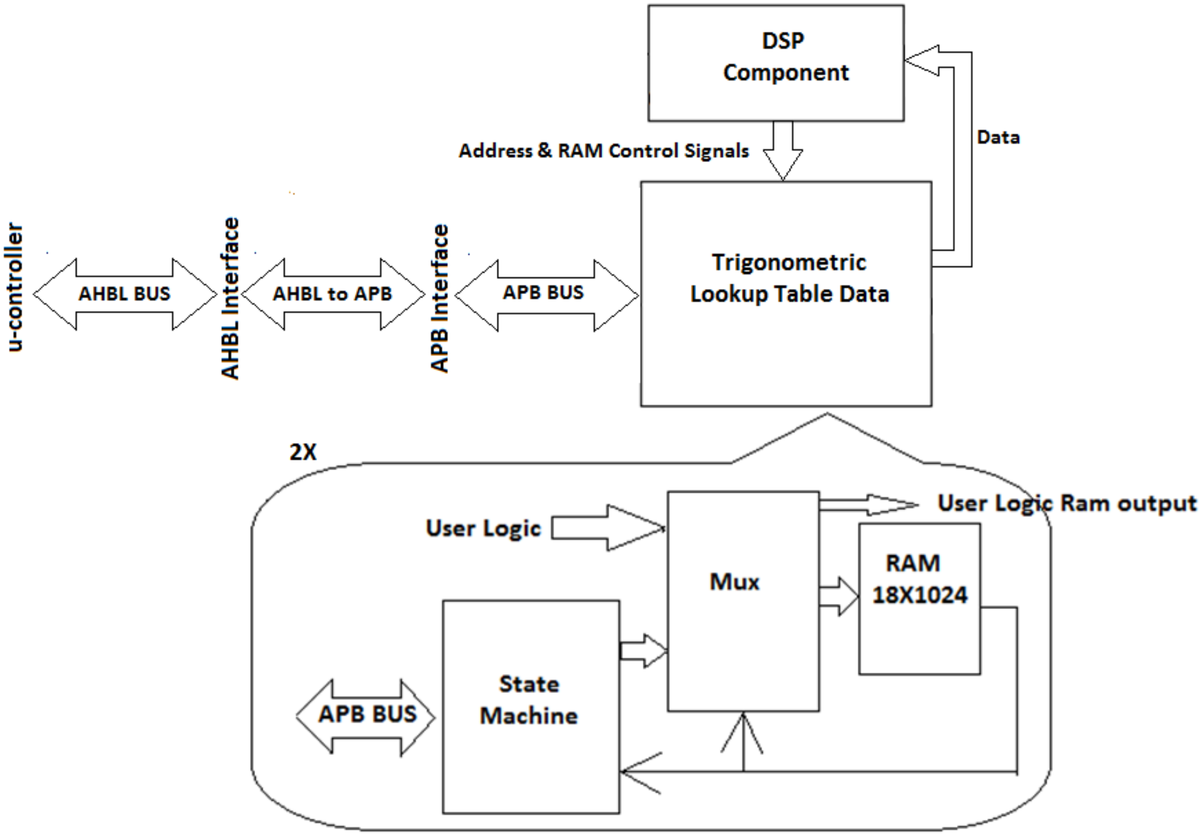

Figure 8: Trigonometric lookup table data component

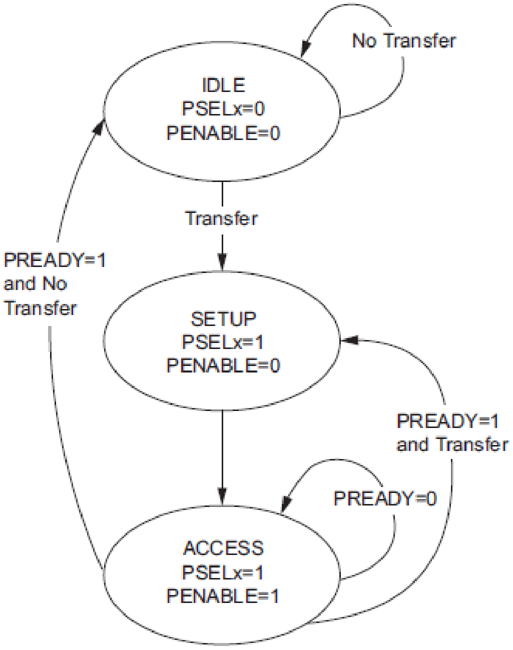

Figure 9: Trigonometric lookup table data component state machine

3.4. DSP Component

The DSP component is responsible of executing all mathematical operations of the fabric. It is built-up from the subcomponents shown in Figure 10. These subcomponents are subordinated by the DSP controller which controls the sequence of operations in the system. From the input/output prospective the system process the motor current and resolver terms from the ADC to evaluate the d and q axis motor currents. These currents are relayed to the u-controller to estimate and relay back V_{dq}. Finally, V_{dq} is transformed to V_{abc}, which is supplied to the triangular modulator.

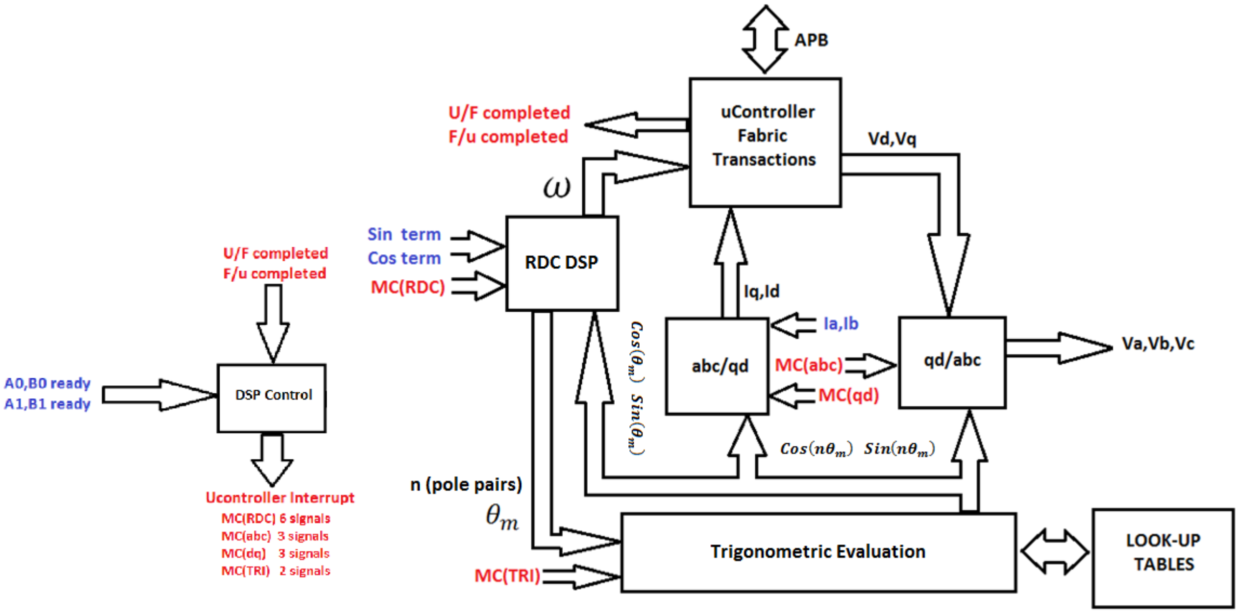

Figure 10: DSP component

RDC DSP. This subcomponent evaluates the resolver to digital converter closed loop shown in Figure 11. The ADC quantized resolver terms are processed in four stages to estimate the motor speed and position. These four stages are detailed in Appendix A and are summarized as follows:

- Stage 1: the ADC quantization of resolver terms is converted to voltages by evaluating (1)-(2).

$$ST = ADCSINE \times \frac{3}{4096} - 1.5\,, where\ ST\ is\ the\ voltage\ of\ the\ sine\ term \qquad (1)$$

$$CT = ADCCOSE \times \frac{3}{4096} - 1.5\,, where\ CT\ is\ the\ voltage\ of\ the\ sine\ term \qquad (2)$$

- Stage 2: the error X_1 represented in (3) is calculated.
  $$X_1 = MIXER\_AMP \times (\cos(P_1) \times ST - \sin(P_1) \times CT)$$
  $$, where\ P_1\ is\ the\ position\ estimation\ of\ the\ loop \qquad (3)$$
- Stage 3: the resolver speed V_1 in (4) is estimated, where n in X_n and V_n represents the order of the sample with 1 being the most recent sample.
  $$V_1 = 515.8 \times X_1 + 1951.19 \times X_2 + 116.985 \times X_3 - 1809.49 \times X_4 - 491.095 \times X_5 + V_2 \qquad (4)$$
- Stage 4: the resolver position P_1 in (5) is estimated.
  $$P_1 = P_2 + V_1 \times \frac{1}{40000} \qquad (5)$$

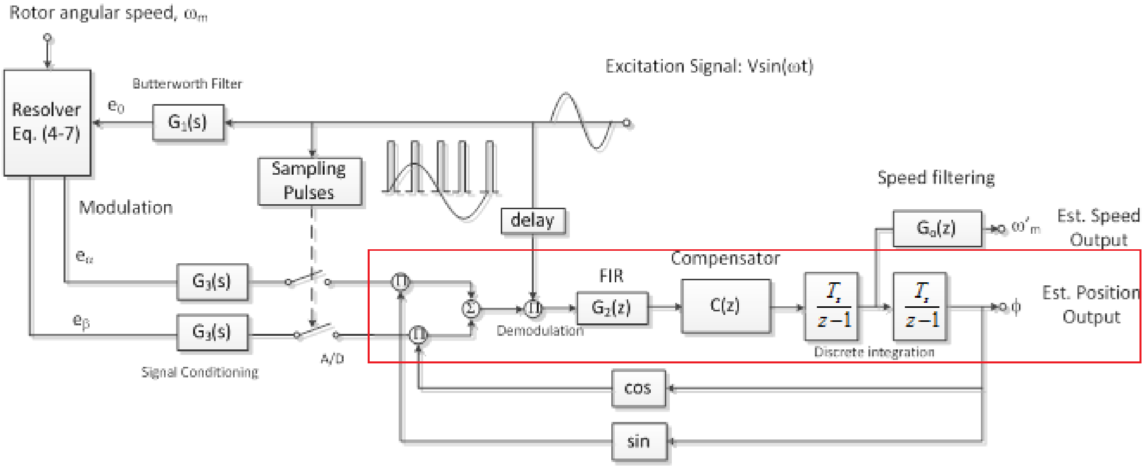

Figure 11: Resolver to digital converter

ABC-DQ & DQ-ABC transformation. The evaluation of the transformation is carried out in two stages with the trigonometric evaluation subcomponent responsible for evaluating the sine and cosine of the electric angle. The two stages of the ABC-DQ transformation are:

- Stage 1: I_α and I_β are calculated by (6).
  $$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{1}{\sqrt{3}} & \frac{2}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} I_a \\ I_b \end{bmatrix} \qquad (6)$$
- Stage 2: I_d and I_q are evaluated by (7)
  $$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} \qquad (7)$$

The stages of the DQ-ABC transformation are:

- Stage 1: V_α and V_β are calculated by (8).

$$\begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} V_d \\ V_q \end{bmatrix} \tag{8}$$

- Stage 2: V_d and I_q are evaluated by (9)

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} \tag{9}$$

Trigonometric evaluation. This subcomponent evaluates the sine and cosine of an angle simultaneously by implementing the state machine shown in Figure 12. The angle to be evaluated is registered and multiplied by $\frac{1024}{2\pi}$, where the resultant integer part is the lookup address, K, and the fraction part is the linear interpolation factor. The data in the lookup table addresses, K & K+1, are fetched and the evaluation is completed by using (10)-(11).

$$sin(Radian) = S(k) + \big(S(k+1) - S(k)\big) * Interpolation_{factor} \tag{10}$$

$$cos(Radian) = C(k) + \big(C(k+1) - C(k)\big) * Interpolation_{factor} \tag{11}$$

S0→S1	Register n*θ
S1→ S2	Register ADDRESS (K)
S2→ S3	Read and register sin(K) & cos(k) Register ADDRESS (K+1)
S3→S4	Read and register sin(K+1) &cos(k+1)
S4→S0	Register the interpolation result

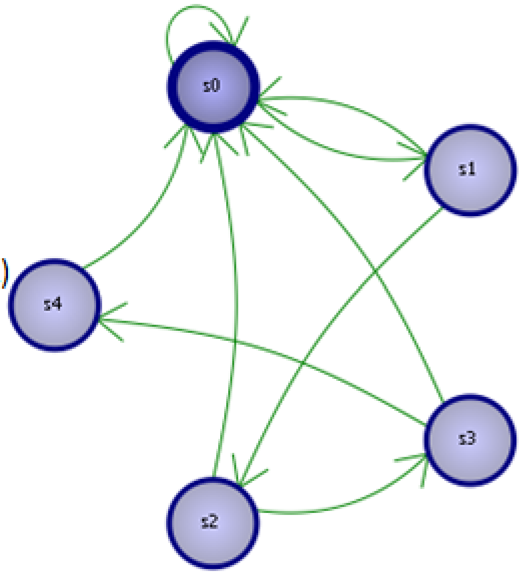

Figure 12: Trigonometric evaluation state machine

U-controller fabric transactions. This subcomponent shown in Figure13 transmits/receives the variables, which are stored in registers furnished in Table 2 to/from the u-controller. In addition to performing a zero wait APB communication protocol, the u-controller fabric transaction state machine decodes the APB bus to generate the proper signals to read and write the variable registers. Furthermore, the subcomponent originated signals “U_F_completed” and “F_U_completed” are used by the DSP control subcomponent to track the transaction status.

Table 2: Variables fabric addresses and destination.

Variable	Address	Destination
$\boldsymbol{\omega}$	0x3200000XX	u-controller
$\boldsymbol{I_d}$	0x3200001XX	u-controller
$\boldsymbol{I_q}$	0x3200010XX	u-controller
$\boldsymbol{V_d}$	0x3200011XX	Fabric
$\boldsymbol{V_q}$	0x3200100XX	Fabric

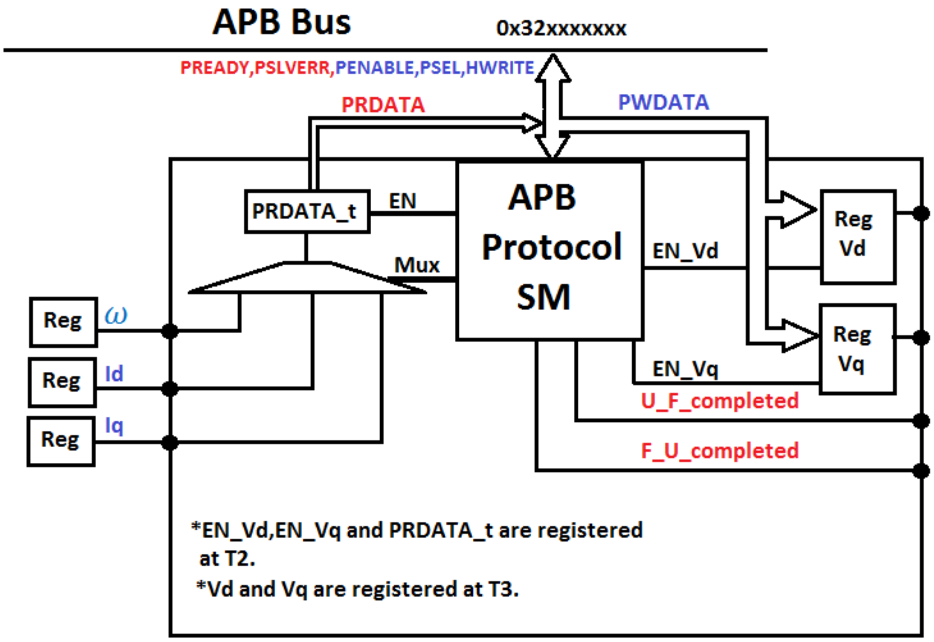

Figure 13: U-controller fabric transaction state machine

DSP control. The data dependency on the DSP component necessitates the sequencing of operations, which is control by a set of signals responsible of enabling a register or activating a sub-state machine. The DSP control state machine shown in Figure 14 is enabled upon sampling of the resolver's terms, where it generates the signal "ENTRI" to activate the trigonometric Evaluation state machine and "ENSHUFFLE","ENSTAGEI", "ENSTAGEII", "ENSTAGEIII", "ENSTAGEIV" shown in Figure 15 to compute the speed and position of the motor.

The state machine idles till the signal "I_ready" is received indicating the completion of the current sampling. Consequently, "STAGEI_A" and "STAGEII_A" converts I_{ab} to I_{dq}, where the u-controller fabric transaction transmits I_{dq} and generates the "F_U_completed" pulse, signaling the completion of the transaction. Upon receipt of V_{dq} from the u-controller the "U_F_completed" signal trigger the transformation of V_{dq} to V_{abc}, where the state machine toggles the "STAGEI_B" and "STAGEII_B" signals to complete the process.

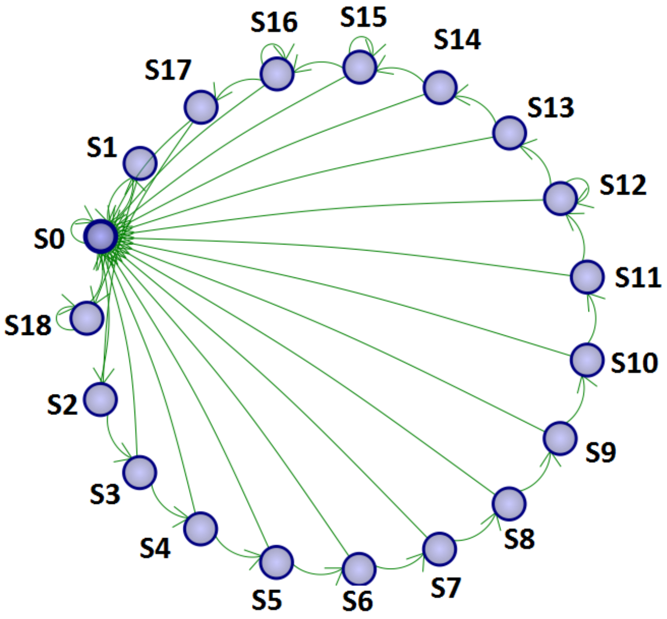

Figure 14: DSP control state machine

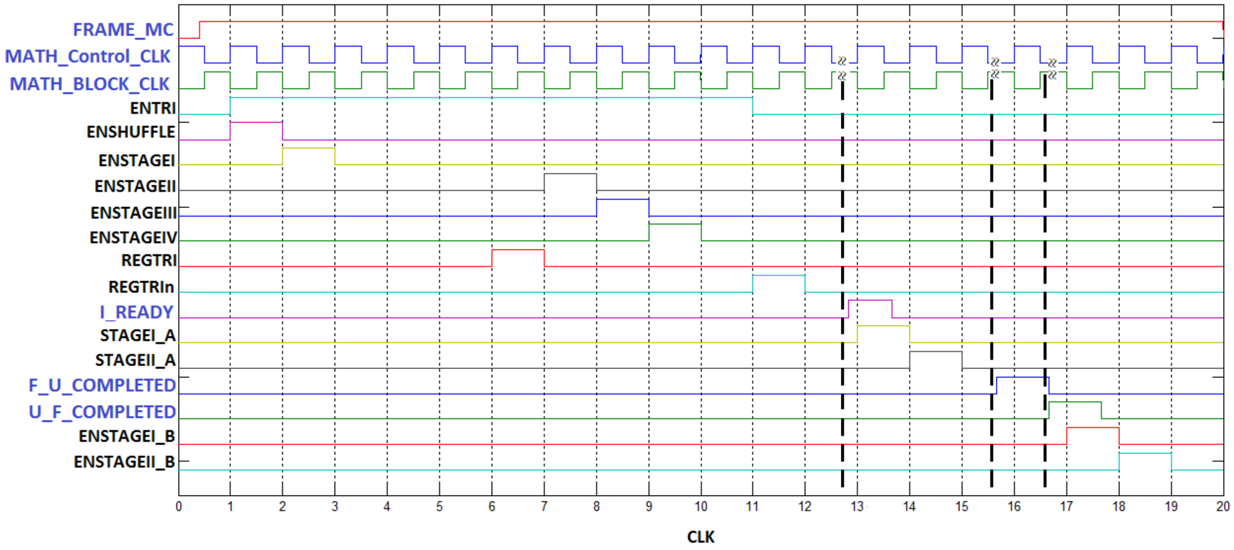

Figure 15: DSP control state machine generated signals

3.5. Fabric Reset Controller

The reset signals for all fabric units have been centralized in a single component to simplify the response to reset triggering events and allow the reset process to be sequenced. Upon device reset the component assert all reset signals and await “POWER_ON_RESET” and “RESET_N_M2F” to be de-asserted. Then, the reset controller starts with de-asserting the fabric & u-controller interface followed by resolver Excitation PWM & ADC interface reset signals. The component idles till the lookup tables are initialized and proceeds by de-asserting reset to the DSP and motor drive PWM. The component continues to monitor the PLL lock along with reset trigger signals and take the actions described in Figure 16.

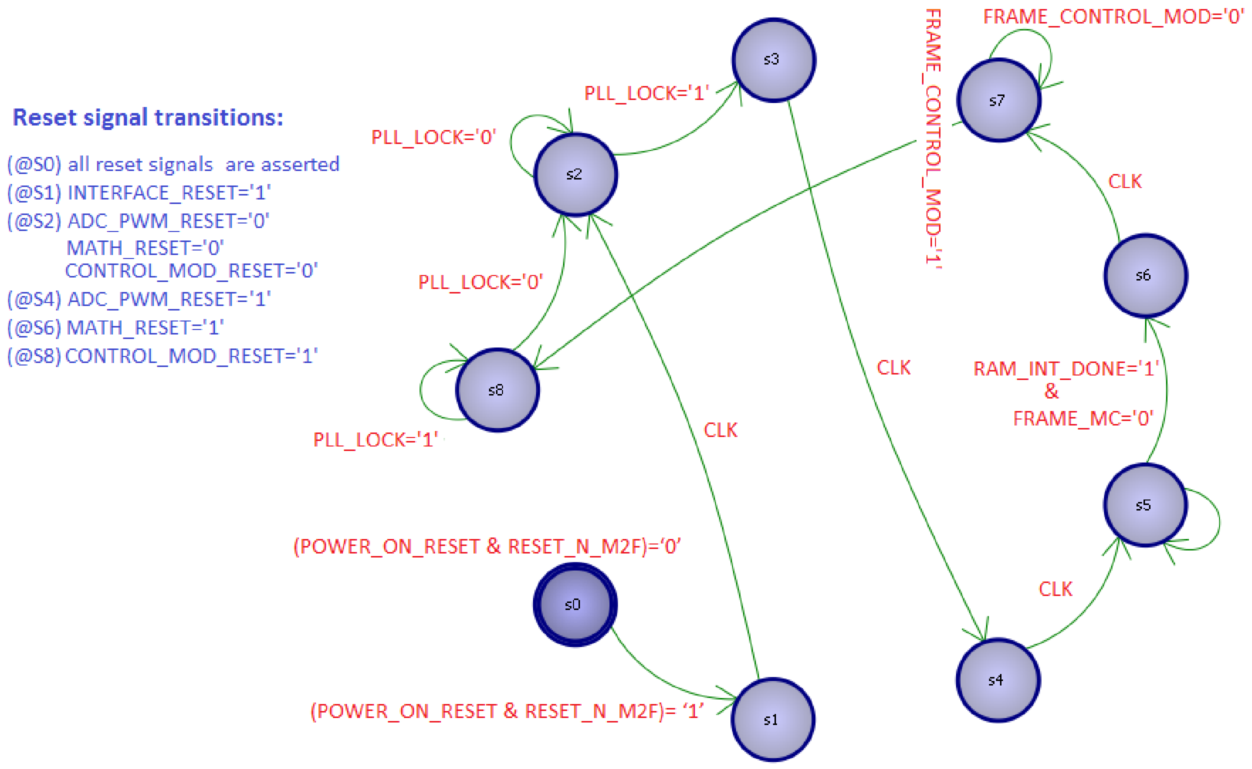

Figure 16: Fabric reset control state machine

3.6. Fabric synchronization

The fabric components shown in Figure 17 need to be in synchronization for the system to function correctly. The resolver excitation PWM which act as a reference generates two signals “FRAME_ADC” and FRAME_CONTROL_MOD” to synchronize the ADC interface and the motor drive PWM. In the event of synchronization lost these signals are capable of resynchronizing the referenced components.

The “FRAME_ADC” which is monitored by the ADC interface state machine is timed such that the resolver terms are sampled at the excitation signal peaks. Similarly, the “FRAME_CONTROL_MOD” aligns the carrier angles $\{0^0, 90^0, 180^0, and\ 270^0\}$ with the current samples. The “FRAME_MC” activates the DSP control state machine immediately after resolver terms sampling, while “I_ready” notifies the DSP control state machine of current sampling completion.

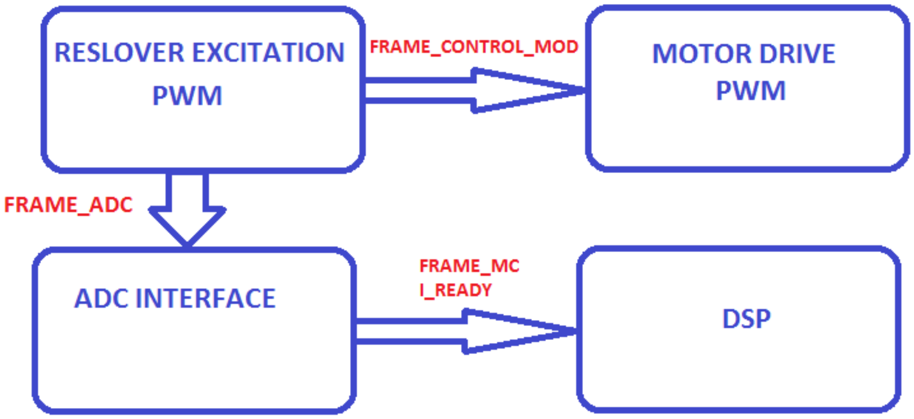

Figure 17: Synchronization signal diagram

3.7. U-controller Design

FOC motor model & controller structure. The dynamic equations of the d and q axis current are derived from the motor equivalent circuits shown in Figure 18 (a)-(b). The system model [2]-[3] is represented in (10)-(12), where λ_d, λ_q and T_e are shown in (13)-(15).

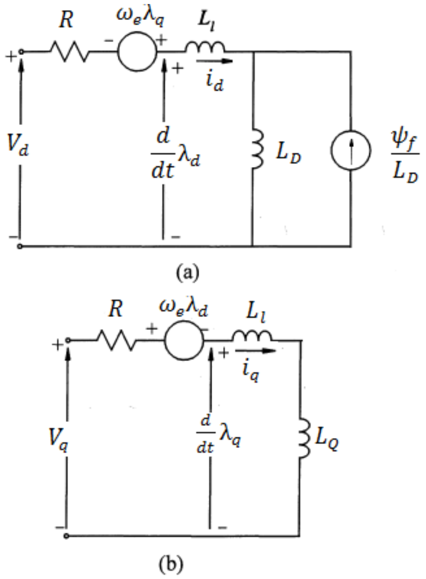

Figure 18: (a) d-axis equivalent circuit (b) q-axis equivalent circuit

$$v_d = Ri_d + \dot{\lambda}_d - \omega_e\lambda_q \tag{10}$$

$$v_q = Ri_q + \dot{\lambda}_q + \omega_e\lambda_d \tag{11}$$

$$T_e = T_L + B\omega_m + J\dot{\omega}_m \tag{12}$$

$$\lambda_d = L_d i_d + \psi_f \tag{13}$$

$$\lambda_q = L_q i_q \tag{14}$$

$$T_e = \frac{3}{2} P[\psi_f i_q + \left(L_d - L_q\right) i_d i_q\ ] \tag{15}$$

The system is coupled due to the speed terms $\omega_e \lambda_{d,q}$, which requires a decoupled controller for optimum performance. Alternatively, the problem can be simplified by designing the controller for the transfer function in (16)-(17) derived from (10)-(11) and compensate for the speed terms by estimating and subtracting $\mathcal{L}\{\omega_e \lambda_q\}/-\mathcal{L}\{\omega_e \lambda_d\}$ from the I_d/I_q current controller respectively. For $i_d = 0\ \ T_e$ is exclusively dependent on i_q, where the transfer function of ω_m is shown in (18) for a linear T_L.

$$\frac{I_d}{V_d + \mathcal{L}\left\{\omega_e \lambda_q\right\}} = \frac{1}{L_d s + R} \tag{16}$$

$$\frac{I_q}{V_d - \mathcal{L}\{\omega_e \lambda_d\}} = \frac{1}{L_q s + R} \tag{17}$$

$$\frac{\omega_m}{I_q} = \frac{\frac{3}{2} P \psi_f}{J \cdot s + (\beta + K_L)} \tag{18}$$

The complete controller structure shown in Figure 19 is based on an anti-windup PID controller. The three PID controllers $i_d, \omega\ and\ i_q$ produce $V_d + \mathcal{L}\left\{\omega_e \lambda_q\right\}$, i_q and $V_d - \mathcal{L}\{\omega_e \lambda_d\}$ respectively. Finally, the controllers output are compensated for the speed term as described previously and limited to the desired voltages.

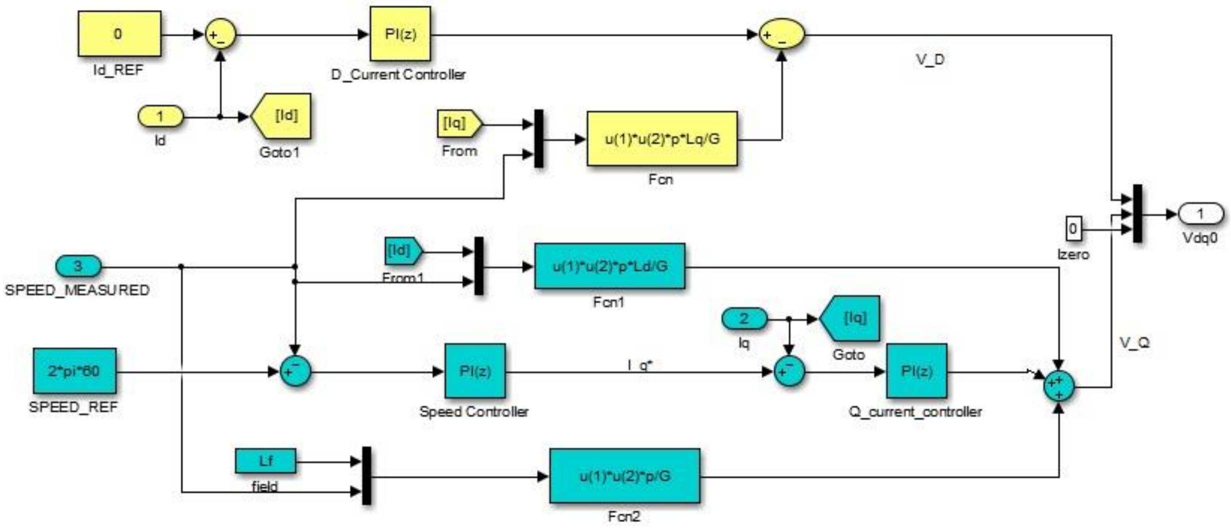

Figure 19: Controller structure

Adopted PU system. For a generic fixed point implementation with a wide pool of motor parameters the design process seems to be overwhelming. The problem becomes more complicated, when overflow and controller accuracy is taken into consideration. To deal with all these challenges a PU system furnished in Table 3 was adapted [5], where the controller structure has been modified as shown in Figure 20 (a). For computational efficiency and numerical precision V_b at the output is back propagated as in Figure20 (b).

Table 3: Adopted PU system.

Base	Description	Formula
$\boldsymbol{S_b}$	3-phase rates power (VA)	User defined
$\boldsymbol{V_b}$	Per phase peak –not RMS- voltage (V)	User defined
$\boldsymbol{\omega_{mb}}$	Rated mechanical speed in (Rad/s)	User defined
$\boldsymbol{\omega_b}$	Rated electrical speed (Rad/s)	$\boldsymbol{\omega_{mb}} * P(pole\ pairs)$
$\boldsymbol{I_b}$	Line-Line peak –not RMS- current (A)	$\boldsymbol{S_b/(1.5V_b)}$
$\boldsymbol{Z_b}$	Impedance (Ω)	$\boldsymbol{V_b/I_b}$
$\boldsymbol{L_b}$	Inductance Base (H)	$\boldsymbol{Z_b/\omega_b}$
$\boldsymbol{\lambda_{mb}}$	Field base (Wb-turn)	$\boldsymbol{L_b \cdot I_b}$

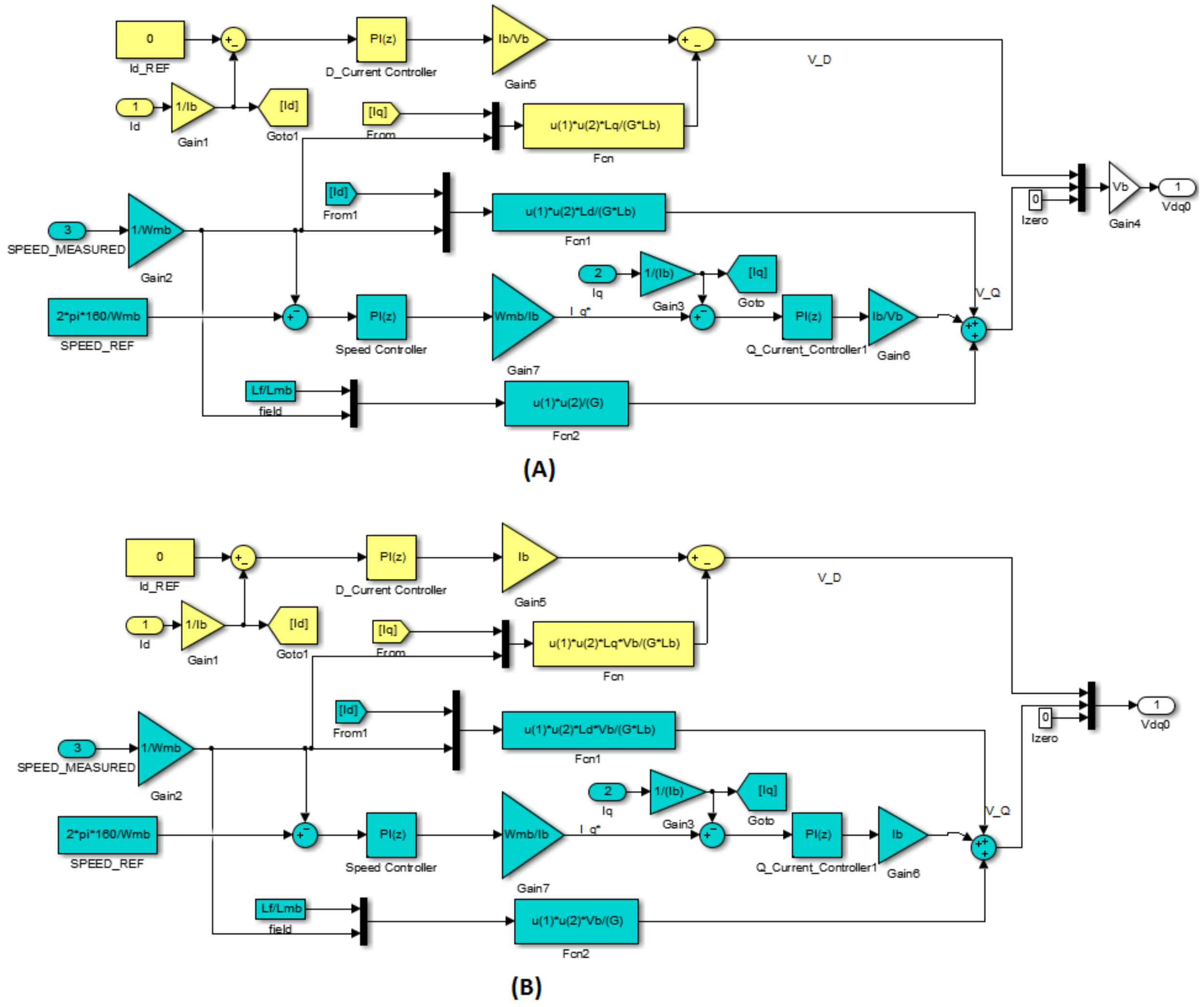

Figure 20: PU controller structure

FOC fixed point implementation. The fixed point math was architected to accommodate a wide range of motor rating, through the optimization of the PI coefficient and constant parameter ranges. In the process each stage of the controller was carefully contemplated to attain an overflow immunity and numerical accuracy.

The detailed design schematics in Appendix. B shows a low pass IIR filter in Figure B1-B4, which was added to improve the resolver speed estimation. The current conversion to PU along with the error evaluation are presented in Figure B5. An anti-windup speed, $I_d$and I_q current PI's characterized in (19)-(20), in addition to the speed terms estimation and $V_{d,q}$ evaluation are shown in (B6)-(B9).

$$Y_{K+1} = K_P(u_{K+1} - u_K) + K_i T_s u_k + Y_k \tag{19}$$

$$Y_{K+1} = \pm I_{sat} \qquad if \qquad (Y_{K+1} > |I_{sat}|) \tag{20}$$

4. Testing

4.1. Fabric Testing

A MATLAB model which duplicates the fabric and the u-controller math, besides interfacing the controlled motor is created to facilitate testing. When this model shown in Figure 21 is simulated the SOC inputs (I_{ab} and the resolver terms) along with the SOC outputs (I_{dq}, V_{abc}, resolver position and speed) are logged. The SOC inputs generated from simulation are used as a stimulus to the fabric test bench, while the test bench computed outputs are compared with their MATLAB counterpart. Since both models are equivalent the outputs of a successful test bench has to match the MATLAB simulated model. This process is implemented and the testing criteria where met.

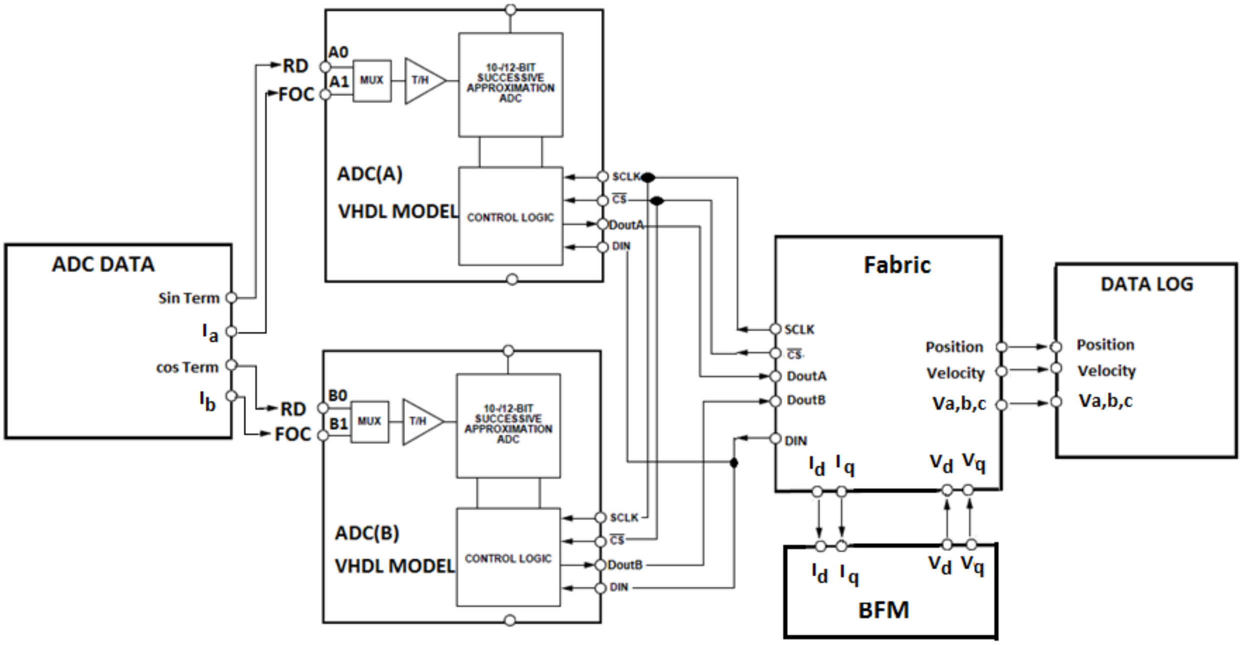

Figure 21: Setup of fabric test bench

4.2 Microcontroller Testing

As the test bench platform does not support the direct interface of the u-controller, V_{dq} were directly supplied by the bus functional model (BFM) in the previous procedure. To verify functionality of the u-controller the C code was uploaded to the physical device paired with I_{dq} MATLAB generated vector of inputs. The code were run to generate V_{dq}, which is also stored and compared with the MATLAB data.

The system were tested for a reference speed of 10000(rpm) under no load, where the speed response of the motor is shown Figures 22 and 23. The 10 KHz oscillation in Figure.21 is attributed to the inverter switching.

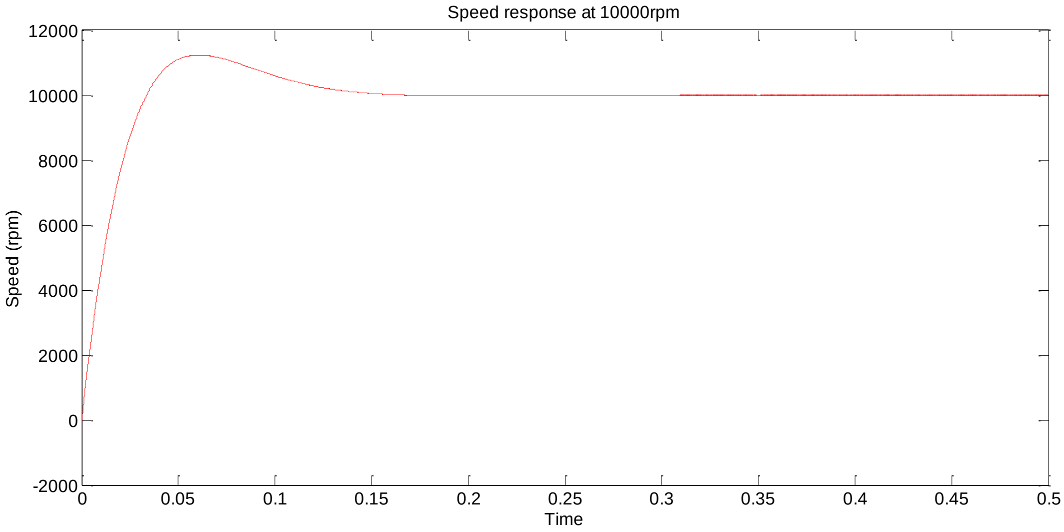

Figure 22: Motor actual speed response for a reference speed 10000rpm.

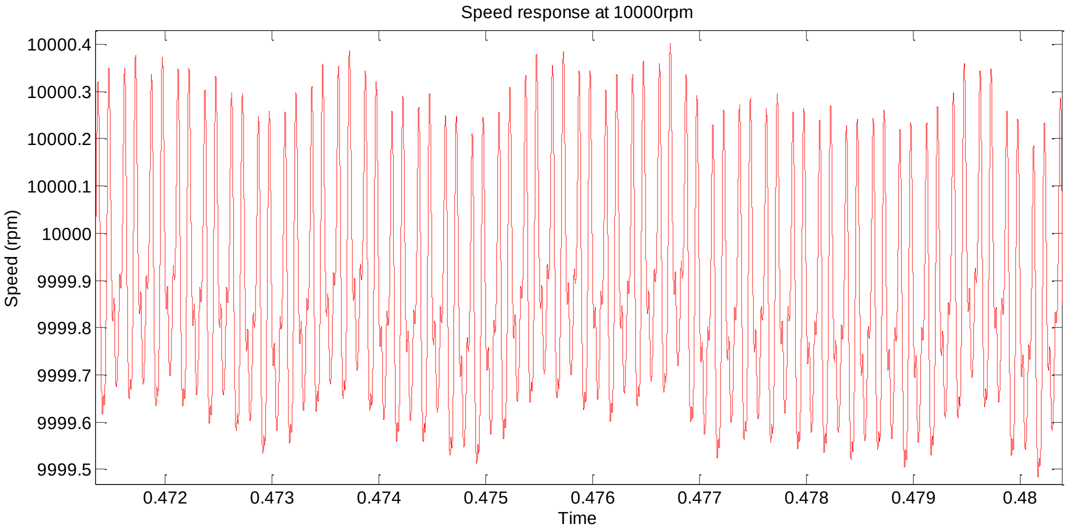

Figure 23: Zoom into the motor actual speed response.

5. Deliverables

In addition to this report, the project deliverables include a CD with the following folders:

- A folder containing the entire FPGA and microcontroller code developed using Libero SOC version 11.2. This code can be run directly using Libero software in order to upload it to the SOC. The code is written to be easily modifiable, if needed.
- A folder containing MATLAB files. Two are system simulations and one that can be used to tune the FOC PI controllers. The files are commented. Instructions of use are included.

6. Summary and Recommendations

A drive system for the PMSM incorporating a feedback resolver sensor was designed into a SOC device. The motor control is based on the FOC. Most of the time consuming mathematical computations such as the RDC math, the frame transformation math and the PWM conversion are done in the fabric. The microcontroller handles the FOC loops and supervises the system.

Testing of the system was done separately for the fabric and the microcontroller designs. Both tests verified the correct operation of the design.

The CPU load was estimated at 36% with the CACHE disabled. This can improve to 22% by running the interrupt routine in CACHE locked mode. The fabric resources utilized are summarized in the table below:

Table 4: Summary of Fabric Resources

Type	Used	Total	Percentage
COMB	2038	56340	3.62
SEQ	1686	56340	2.99
GLOBAL	9	16	56.25
RGB	9	1088	0.83
RAM64x18	0	72	0.00

RAM1K18	2	69	2.90
MACC	26	72	36.11
RCOSC_25_50MHZ	0	1	0.00
RCOSC_1MHZ	0	1	0.00
XTLOSC	1	1	100.00
CCC	1	6	16.67

Table 4 shows that the capacity of Smartfussion2 SOC is far greater than the needed resources for this project.

The major challenges in the project were exclusively implementation issues. These can be divided into the following groups.

- The design of the look-up tables so that they are accessible by various processes.
- The efficient pipelining and sequencing of operations.
- Time synchronization between the various processes and components and the regaining of synchronization in the event it was lost.
- The centralized reset and initialization of the components and processes. The design of the reset process was done to enable later addition of other stages such as the lock lost event handling.

A major contribution of this project to the art of designing in the fabric is the high level architecture in the math operation pipelining. Specifically the division of the math operation into stages that can execute in a single clock cycle. This is shown in the report in figures 10 and 14. Figure 10 shows the high level math architecture and figure 14 shows the DSP control state machine.

6. References

[1] H. S. Patel and R. G. Hoft, “Generalized Techniques of Harmonic Elimination and Voltage Control in Thyristor Inverter: Part I-Harmonic Elimination,” *IEEE Trans. Ind. Appl.*, vol. IA-9, pp. 310-317, May 1973.

[2] Analog Devices (2004). AD7912/AD7922. [Online].Available: http://www.analog.com/static/imported-files/data_sheets/AD7912_7922.pdf

[3] Pillay, P.; Krishnan, R., "Development of digital models for a vector controlled permanent magnet synchronous motor drive," *Industry Applications Society Annual Meeting, 1988., Conference Record of the 1988 IEEE* , vol., no., pp.476,482 vol.1, 2-7 Oct. 1988

[4] Pillay, P.; Krishnan, R., "Modeling, simulation, and analysis of permanent-magnet motor drives. I. The permanent-magnet synchronous motor drive," *Industry Applications, IEEE Transactions on* , vol.25, no.2, pp.265,273, Mar/Apr 1989
doi: 10.1109/28.25541.

[5] P. Kundur, “Synchronous Machine Theory and Modelling,” in *Power System Stability and Control*, McGraw-Hill, 1994, ch.3, sec.4, pp.75-88.

APPENDIX A: Resolver to Digital Converter DSP Structure

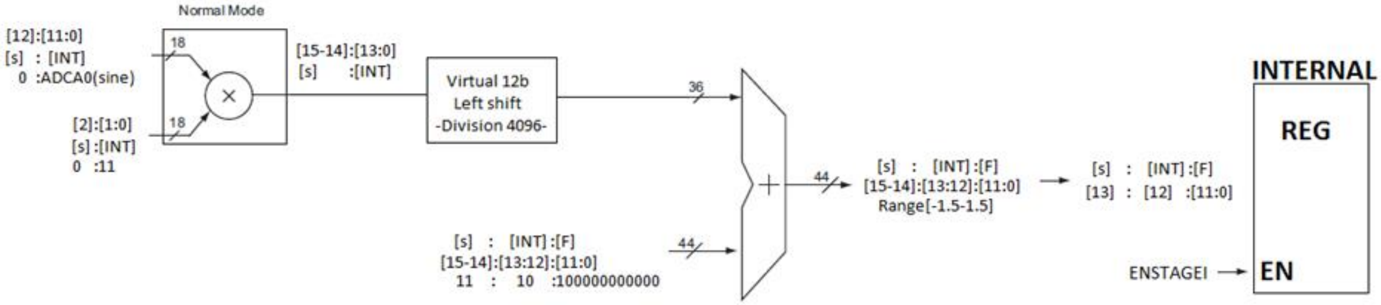

Figure A1: Stage I of RDC DSP.

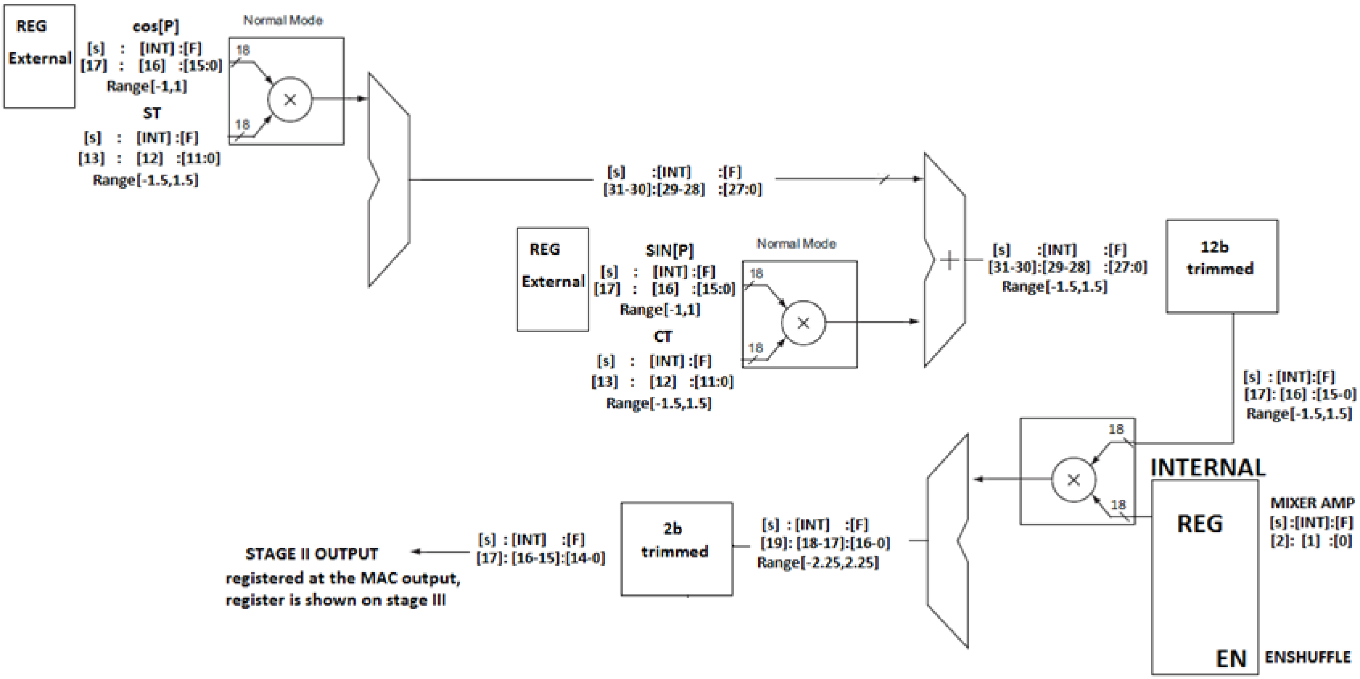

Figure A2: Stage II of RDC DSP .

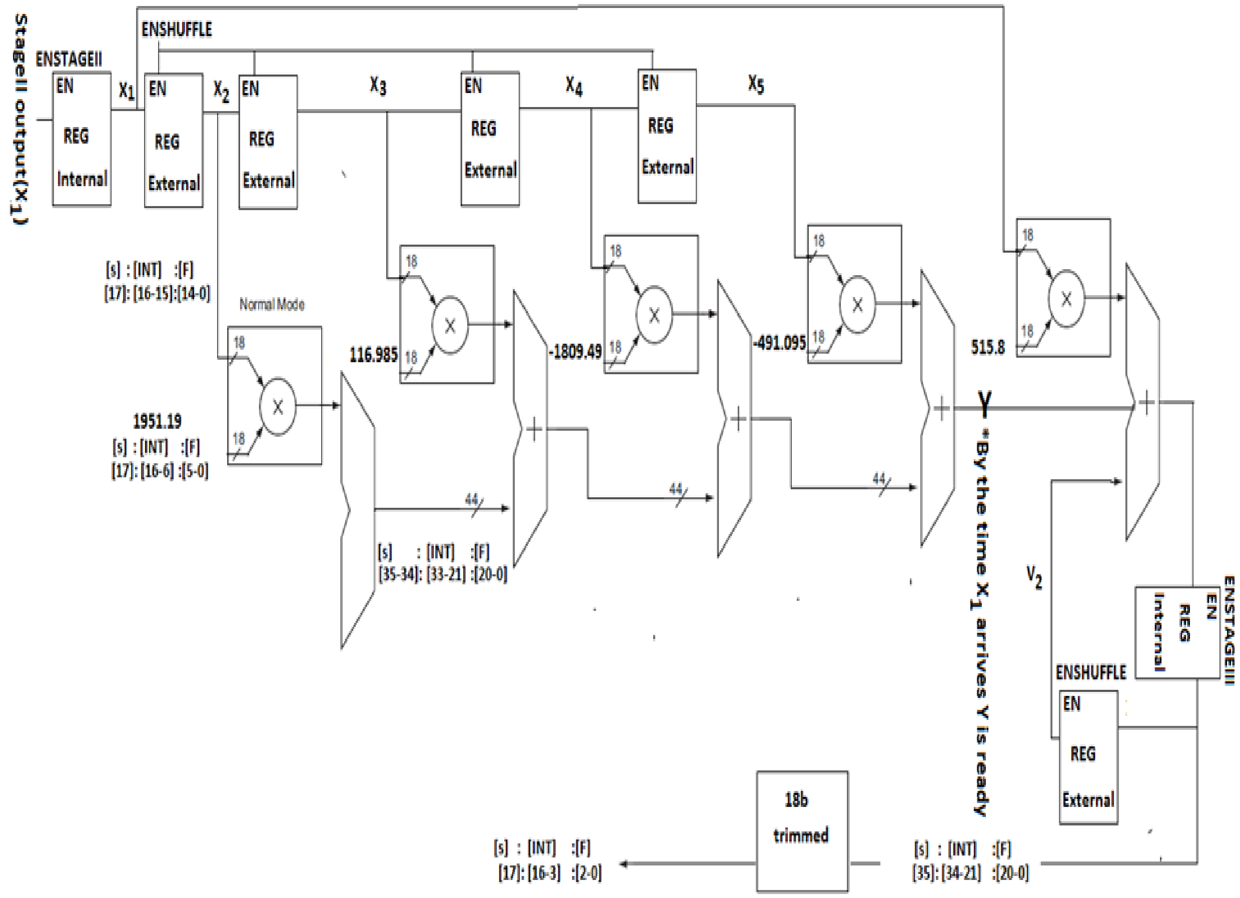

Figure A3: Stage III of RDC DSP.

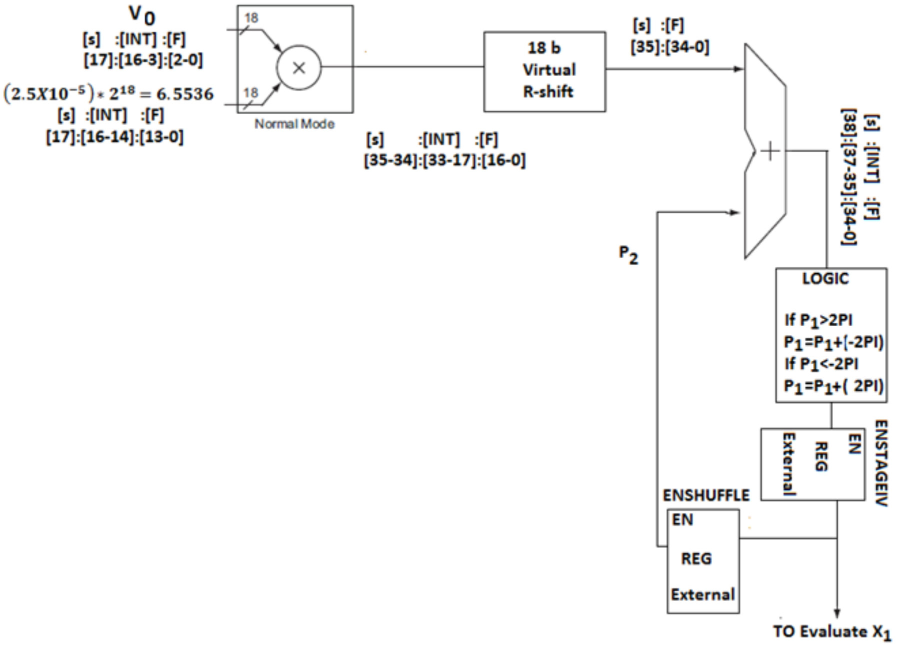

Figure A4: Stage IV of RDC DSP.

APPENDIX B: Fixed Point FOC Controller Structure

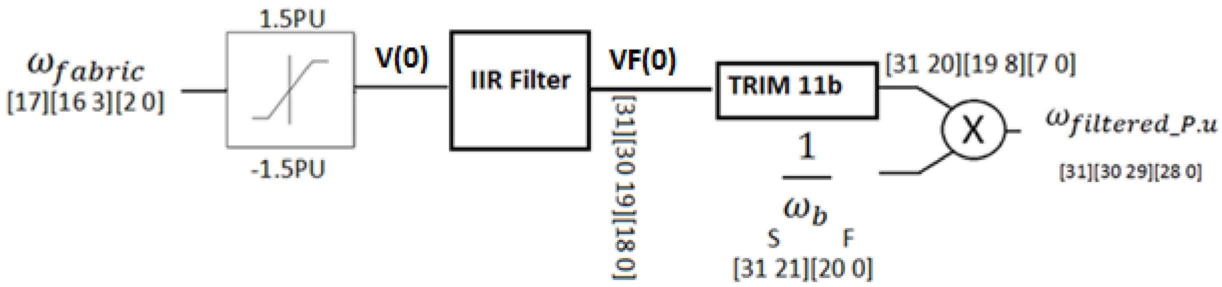

Figure B1: Speed filter top level structure.

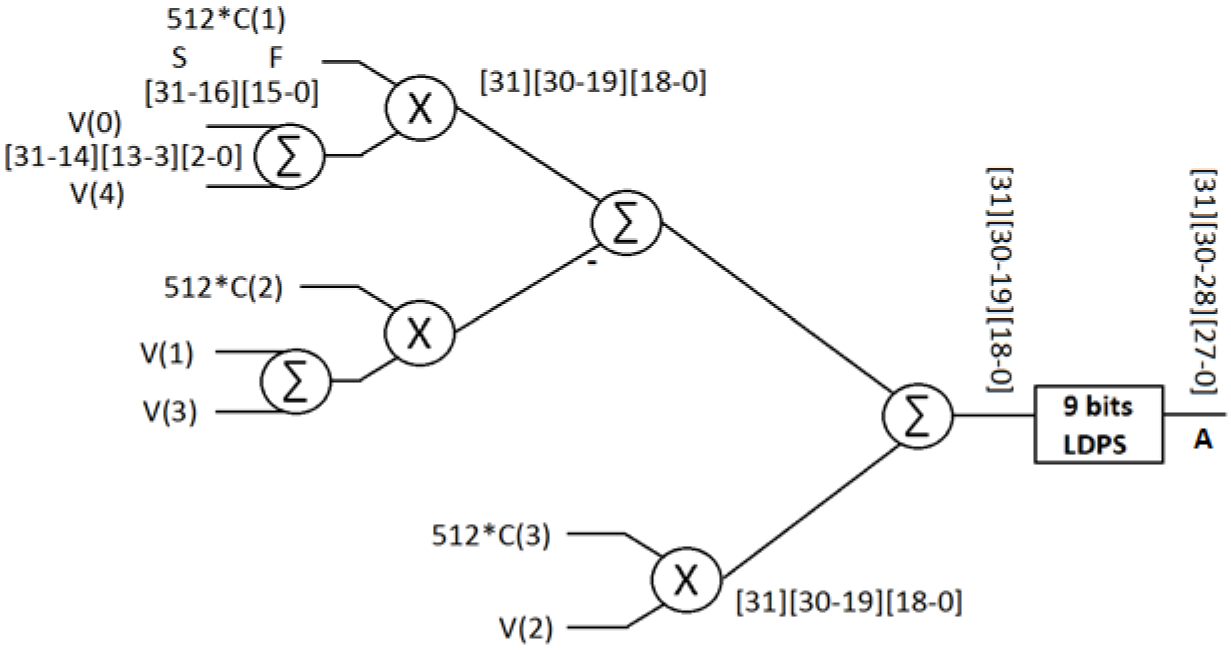

Figure B2: Speed filter part A.

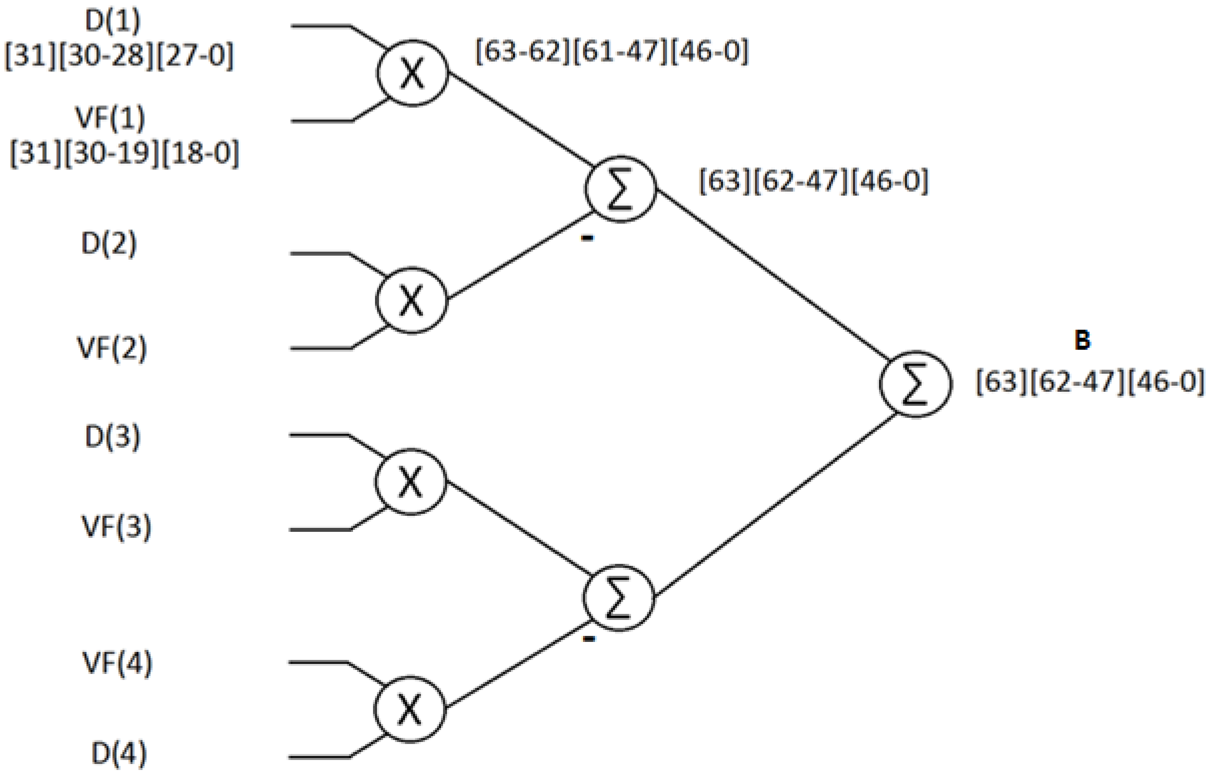

Figure B3: Speed filter feedback part B.

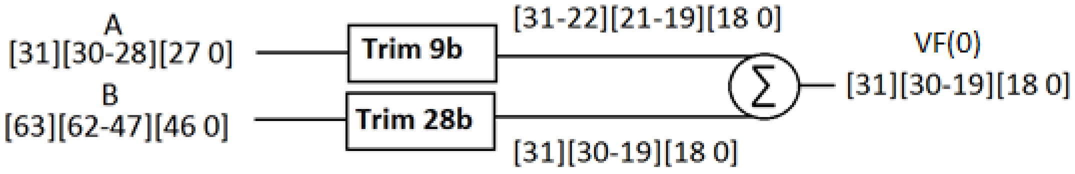

Figure B4: Speed filter output evaluation.

Table B1: Filter coefficient.

Coefficient	Numerical representation	Fixed point representation
$\boldsymbol{C_1}$	0.0005387986	18079
$\boldsymbol{C_2}$	0.001026802	34454
$\boldsymbol{C_3}$	0.0013654728	45818
$\boldsymbol{D_1}$	3.6460812183	978737474

D_2	5.0057462842	1343719786
D_3	3.065865926	822987118
D_4	0.7065903709	189673908

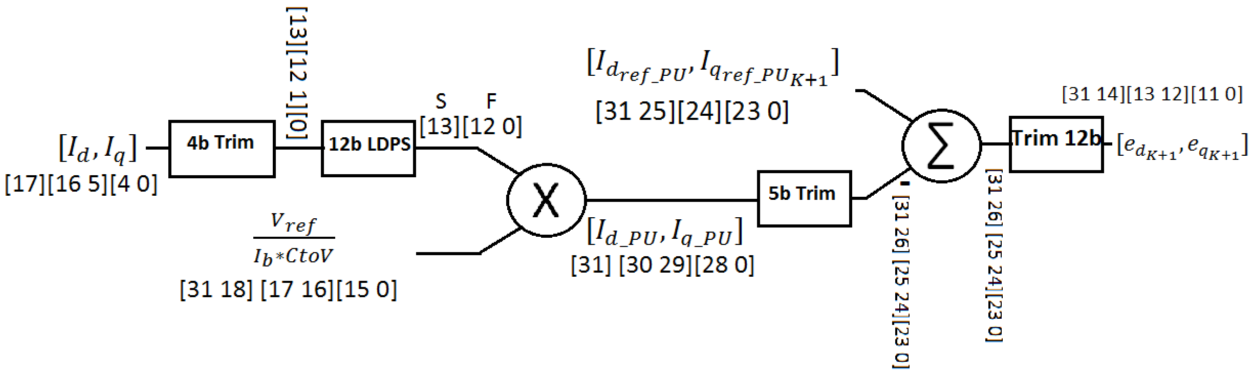

Figure B5: $\boldsymbol{I_{dq}}$ conversion to PU and error evaluation.

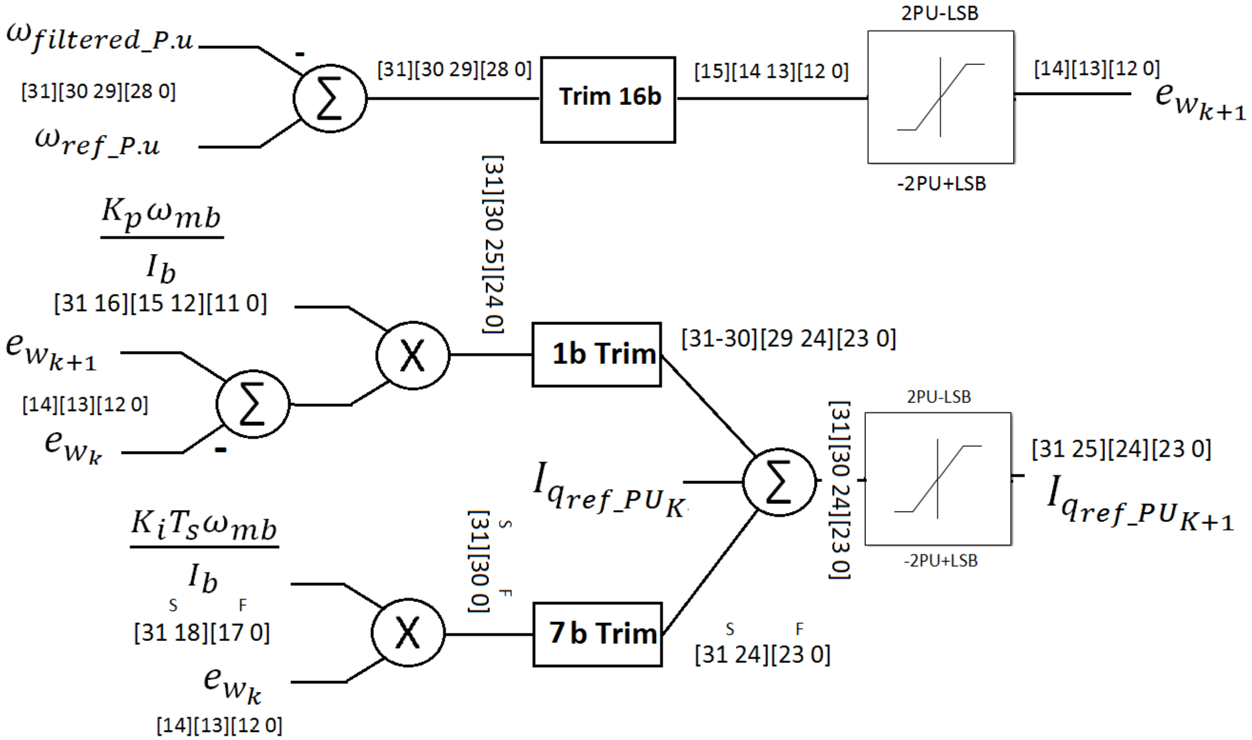

Figure B6: Speed PI.

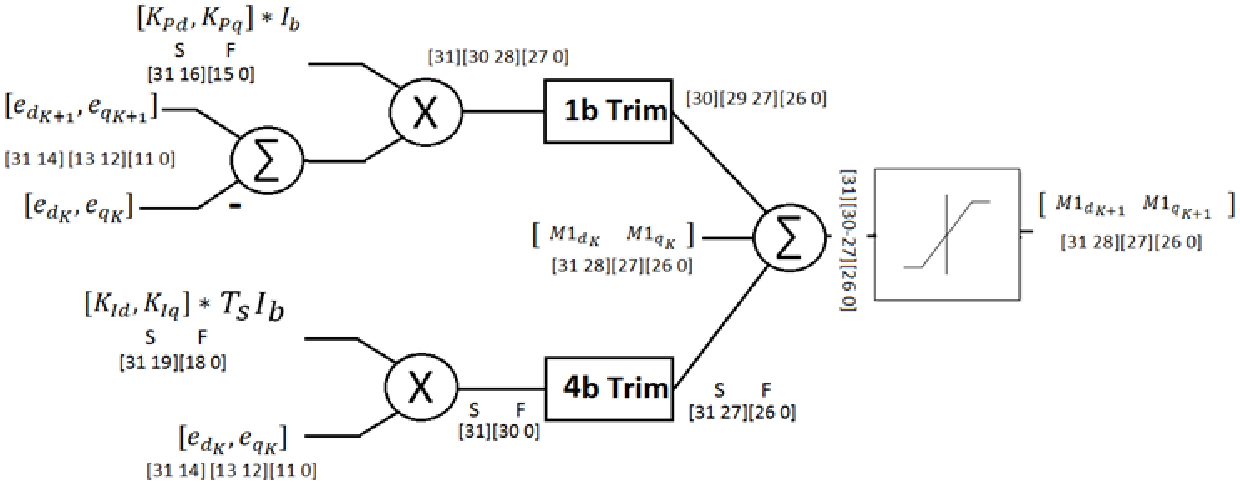

Figure B7: $\boldsymbol{I_{d,q}}$ PIs.

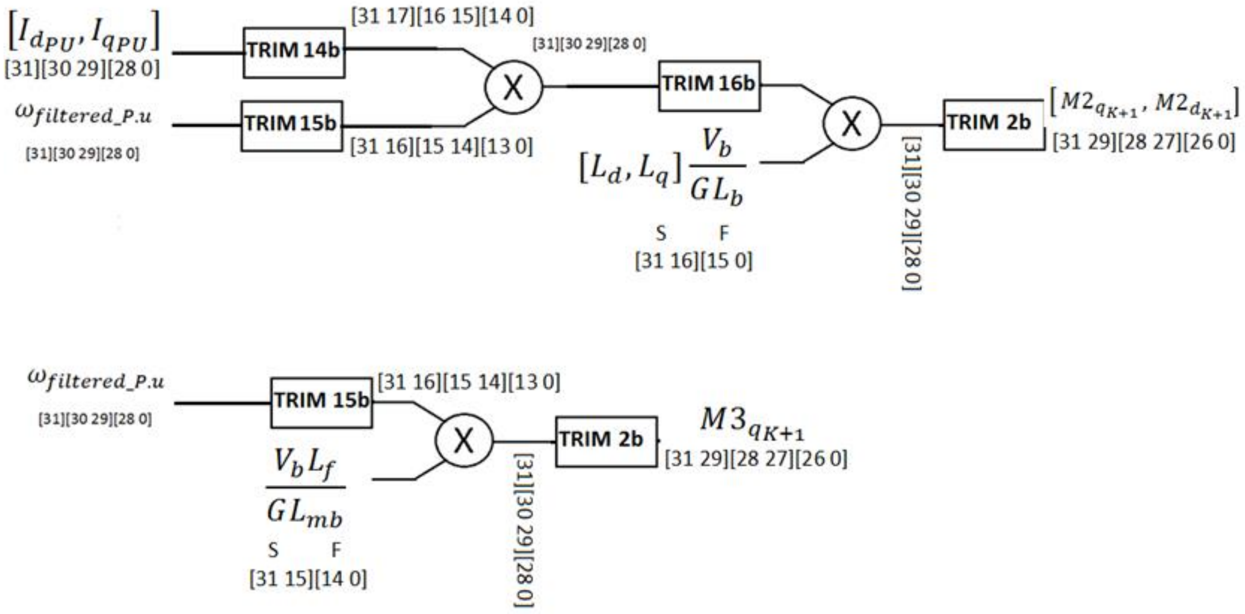

Figure B8: Estimation of speed terms.

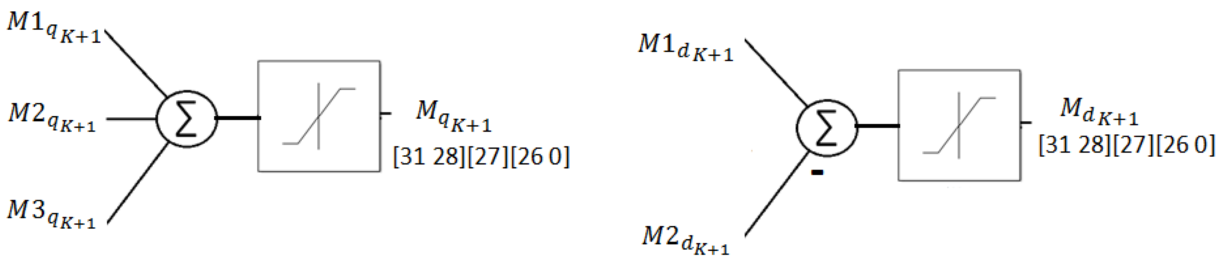

Figure B9: Controller outputs $\boldsymbol{M_{d,q}}$

Table B1. Variables and constant numerical range and fixed point representation

Variables/Constants	Overflow immunity range	Fixed point representation
$\boldsymbol{I_{q\_PU}, I_{d\_PU}}$	[-2PU,2PU-LSB]	Q2.29 [31][30 29][28-0]
$\boldsymbol{e_{I_{d\_PU}}, e_{I_{q\_PU}}}$	[-4PU,4PU-LSB]	Q2.12 [13 12][11-0]
$\boldsymbol{e_{\omega\_PU}}$	[-2PU,2PU-LSB]	Q1.13 [14][13][12-0]
$\boldsymbol{a_{0,1} = \frac{L_q V_b}{GL_b}, \frac{L_d V_b}{GL_b}}$	[0,1-LSB]	Q0.16 [31 16][15-0]
$\boldsymbol{a_2 = \frac{L_f V_b}{GL_{mb}}}$	[0,1-LSB]	Q0.15 [31 15][14-0]
$\boldsymbol{Conv_{ADC} = \frac{V_{ref}}{I_b(CtoV)}}$	[0,4-LSB]	Q2.16 [17 16][15-0]
$\boldsymbol{\omega_{b\_inv} = \frac{1}{\omega_b}}$	[0,1-LSB]	Q0.21 [31 21][20-0]
$\boldsymbol{K_{P_{\omega_S}} = \frac{K_{P_\omega} * \omega\_mb}{I_b}}$	[0,16-LSB]	Q4.12 [31 16][15 12][11-0]
$\boldsymbol{K_{I_{\omega_S}} = \frac{K_{P_\omega} * T_S * \omega\_mb}{I_b}}$	[0 , $\frac{1}{2}$--LSB]	Q0.18 [31 18][17-0]
$\boldsymbol{K_{P_{d,q_S}} = K_{P_d} * I_b}$	[0,1-LSB]	Q0.16 [31 16][15-0]
$\boldsymbol{K_{I_{d,q_S}} = K_{P_d} * T_S * I_b}$	[0, $\frac{1}{4}$ -LSB]	Q0.19 [31 19][18-0]

Notes:-

-Overflow immunity is guaranteed under the specified immunity range. For operation beyond this ranges overflow can occur theoretically.

-If a parameter does not fit into the specified range the user can rescale the necessary part of the design.

-CtoV is the motor sensing circuit current to voltage ratio $\frac{V}{I}$, where V is the voltage that appears at the ADC channel input and I is the motor current. For instance if 5Amp and -5Amp motor current induce a 3V and 0V at the ADC channel inputs the $toV = \frac{3}{10} = .3$.