Real-Time Development Interface Embedded in a Compact Motion Controller

Josef, Reill, German Aerospace Center - DLR, Germany, Josef.Reill@dlr.de,
Cristina, Serrano Gonzalez, DLR, Germany, Cristina.SerranoGonzalez@dlr.de
Volker, Senft, DLR, Germany, Volker.Senft@dlr.de
Martin, Pfau, DLR, Germany, Martin.Pfau@dlr.de

Abstract

This paper presents a real-time development interface for motor control, embedded in a single board (MCB). The MCB's CPU is an ARM Cortex-A8 supporting EtherCAT implementation. This CPU computes the control algorithm that can be implemented with Matlab/Simulink toolchain. The host PC that runs Matlab environment is able to connect to the running code via EtherCAT bus by use of Ethernet over EtherCAT (EoE) directly. Via external mode the developer is able to monitor all Simulink signals of the deployed model on hardware board during runtime. Manipulation of constant block values is also feasible to e.g. investigate on different controller parameters online. The actual prototype board is fully functional and will now be miniaturized to fit into a modular drive unit. With this unit complex multi-axis systems can be installed, monitored and investigated by simply attaching EtherCAT communication bus and single power supply voltage. Thanks to Simulink development environment the controller code development is very efficient. Since the hardware is reduced to mass electronics components the cost is acceptable.

1. Introduction

It is always very time consuming to implement control algorithms to different hardware, the wish of an environment with good development support is very strong. Especially when developing new algorithms that deal with complex physics and mathematics a sophisticated toolchain is required. Otherwise a lot of developing time is to be invested on the implementation itself and cannot be used for elaborating the algorithm structure. Lots of industry motion controllers offer flexible and user defined controller structures that can be adapted to the application. Still when it comes to space vector modulation or special modulation procedures the controller simply do not offer the necessary interface to the power electronics hardware.

This is why a motion controller board was built that offers a Matlab/Simulink toolchain that is very familiar to all engineers in the field of control. The board consists of a computation unit that is an industry component, stacked to a power electronics board with the motor inverter. Fig. 1 shows a block diagram of the concept for the motion controller board and the communication interfaces. The concept is split into a real-time execution processor domain and a power electronics domain that interfaces the motor with sensors. The Interfaces on the left (UART, Ethernet and I2C) are optional and may be useful during prototyping status but not urgently required. The EtherCAT interface offers already all communication and external mode features. The prototype board is fully functional and will now be miniaturized to fit into a modular drive unit that will consist of motor, gearing, position sensor (if used), power electronics and controller.

The paper presents the concept and points out the benefits of having a toolchain like this available during control algorithm developing and use in application. Different from other rapid control development toolchains this proposed way is a very compact and cost efficient solution, so that the toolchain could be used in every single axis of complex systems and offer a very powerful way of monitoring in the sense of maintenance or online controller upgrade. Furthermore close investigation of the system is possible whenever needed. The host PC may be located somewhere else having access to the system via internet. The Simulink model has full access to hardware PWM and ADC registers and all monitoring is

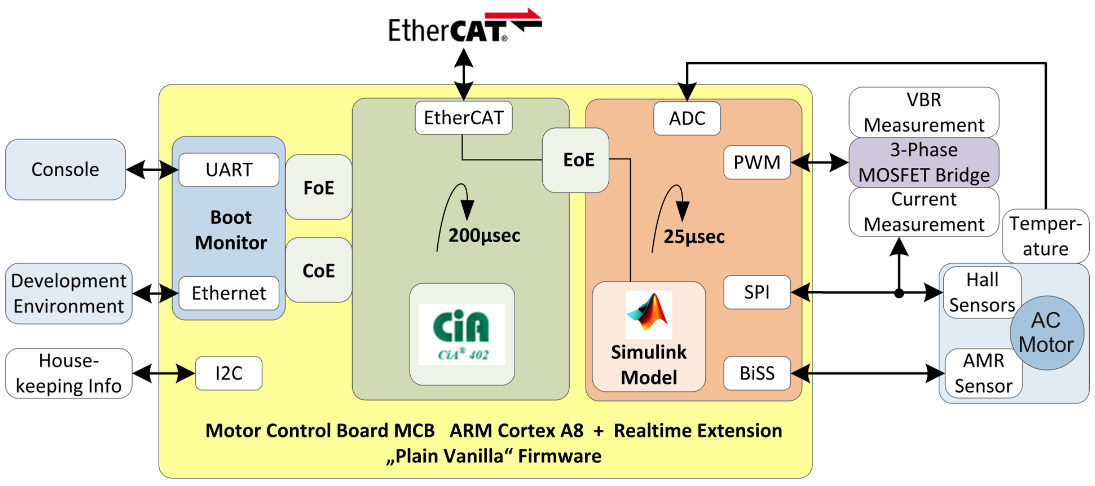


Fig. 1: *Block diagram of the Motor Control Board.*

possible with current control loop bandwidth.

By developing an appropriate software environment it is possible to draw Simulink block diagrams and compile them directly to the motion controller hardware. With the Matlab/Simulink external mode, all signals within the control loop algorithm can be investigated in real-time. This development platform is intended to be used in complex robotic systems (for example [1] and [2]) to investigate signals in detail and online while the system is in action. It is intended to analyze complex control theory issues. All signals of the Simulink model are available in a synchronous and direct way for close inspection. Also a backup of that signals is possible to perform offline analysis with potentially high demand for computation power. Matlab/Simulink models can be compiled and loaded to the board using EtherCAT fieldbus. The algorithm can be investigated without attaching additional cables simply by attaching Simulink scopes to the signal line. With just EtherCAT fieldbus and DC-power applied the board is ready to run and offers all debug options. This helps to use this motion controller in a modular way. The typical transfer from a rapid control environment hardware for developing to a compact hardware built for the final product becomes obsolete. The developing process is carried out on the final hardware without the drawback of reduced options of analysis.

The first two sections give a description of the software and hardware design of the board. The third section deals with the features that are helpful during controller development and shows an implementation of position sensorless control of permanent magnet synchronous motors as an example. The paper concludes with some aspects that are of special interest when synchronized multi-axis mode is needed.

2. Software Design of the Motor Control Board

The idea of the Motor Control Board (MCB) is to use a Simulink controller model running on a fast microprocessor, inside the innermost and fastest internal cycle. Fig. 2 shows a block diagram of the MCB from the developer's perspective. The MCB is based on an ARM Cortex A8 CPU including a real-time unit, to support EtherCAT as it is an industrial real-time Ethernet bus. The CPU is clocked by a frequency of 800 MHz and executes a regular command within 1.25 nsec. This requires an enabled cache for code and data. Because the CPU is an ordinary ARM microprocessor, the execution time of the firmware depends on whether the code is already loaded inside the cache memory or not. Parts of the firmware code run faster after a while, when executed completely from cache. Access to the peripheral hardware of the CPU could be much slower. It is a good idea to reduce register access to the peripheral hardware to a minimum to save runtime. It is also recommended to avoid integer division, because the ARM Cortex A8 architecture does not include a hardware integer divide unit [3]. The float-

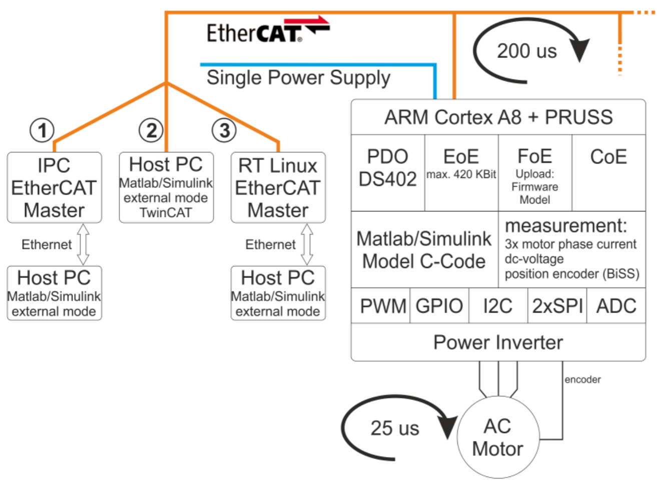

EtherCAT
Single Power Supply
200 us
1
2
3
IPC EtherCAT Master
Host PC Matlab/Simulink external mode TwinCAT
RT Linux EtherCAT Master
Ethernet
Ethernet
Host PC Matlab/Simulink external mode
Host PC Matlab/Simulink external mode
ARM Cortex A8 + PRUSS
PDO DS402
EoE max. 420 KBit
FoE Upload: Firmware Model
CoE
Matlab/Simulink Model C-Code
measurement: 3x motor phase current dc-voltage position encoder (BiSS)
PWM
GPIO
I2C
2xSPI
ADC
Power Inverter
encoder
25 us
AC Motor

Fig. 2: *Overview diagram of the MCB with all function blocks and interfaces. On the left, three different options to connect a host PC with the development environment installed are shown.*

ing point unit called NEON - supports a regular set of mathematical operations [4]. Square root is included as an own assembler command, but mathematical functions such as sin(), cos() and the popular atan2() must be calculated in software. With 512 Mbyte RAM more than enough main memory is available for the firmware and the Flash Memory can store the whole firmware as well as permanent parameters used after the next reboot. Software development is done using an Eclipse/GCC toolchain using C as programming language. The firmware does not include any operating system (OS). It reduces the software overhead dramatically and allows to develop a “plain vanilla” firmware code. It causes more effort during the development, but with the success of reduced jitter in the response of interrupts and communication. The EtherCAT communication is done mainly by the Beckhoff slave stack code [5].

In general the used ARM CPU is a fast microprocessor and allows an internal cycle time for the motor controller of 25 μsec. The included real-time unit allows a close access to the real-time Ethernet protocol (EtherCAT). There is no communication required using a slow peripheral hardware. The link is done by an internal shared memory, which can be used by both the microprocessor and the real-time unit at the same time. It allows a short cycle time for the external communication using EtherCAT of 200 μsec.

2.1. Internal Cycle Simulink Model Control Loop

The internal cycle (see orange part of block diagram Fig. 1) is computed within 25 μsec. In that turnaround time all actual sensor data needs to be read, the control algorithm is computed and the PWM register need to be updated for the next cycle. Every cycle starts by requesting the ADC values of motor current, the operating voltage of the dc-link and the motor position sensors. From the external AD converter the measurement results are transmitted through the SPI channel of the microprocessor (Fig. 3). The SPI is a serial bus using a master/slave architecture with an own clock line from the master. All connected members are linked as a chain and because of the SPI FIFO buffer transmission can be done with very limited control of the microprocessor. Because of speed limitation of the shift register the SPI clock speed is set to 25 MHz. The motor position is measured by an external chip using a magneto resistive AMR sensor. From an external chip the position information is transmitted to the microprocessor using a so called BiSS protocol [6]. This protocol is quite similar to the SPI channel; the microprocessor firmware must behave only slightly differently to support the BiSS protocol. As a chip requirement the BiSS clock speed is limited to 10 MHz.

After all requested measurement values are available, the calculation of the Simulink model can be started. The motor control model is designed using Simulink. Within the model only the control algorithm needs to be considered whereas IO operations are handled in the firmware. A Simulink S-function enables access to the measured sensor values of the hardware and an additional S-function is used to pass the computed PWM signals from Simulink to the firmware. For all this calculations the Simulink motor controller can use up to 10 μsec plus the time the internal cycle restarts (Fig. 4). This is the remaining time after the entire measurement requests are done. The next PWM values are set, when the new internal cycle starts. This ensures a precise and periodic behavior of the controller loop. The Simulink motor controller includes a parameter set for the so called Simulink external mode. During the calculations the parameters are stored inside a buffer and will be transmitted asynchronously to the

EtherCAT Master.

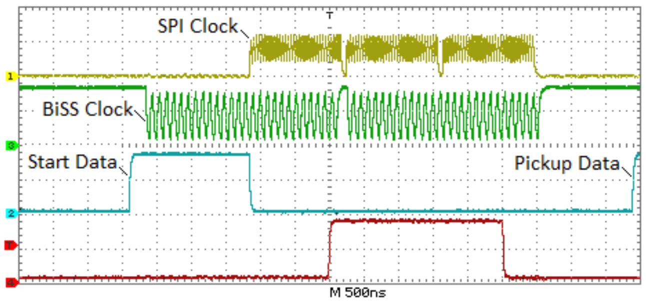


Fig. 3: *Data Transfer SPI and BiSS Clock at beginning of each cycle. When sensor data is successfully picked up the latest sensor data are provided for the Simulink model by means of S-function.*

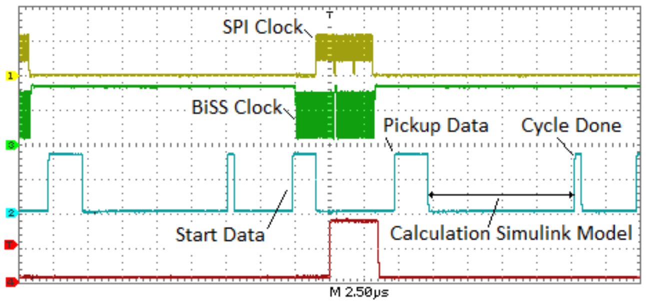


Fig. 4: *The cycle time of overall 25 μsec is split into sensor data reading for 10 μsec and additional 10 μsec time to calculate the Simulink model. Finally there is 5 μsec left to ensure real-time conditions and to update the PWM registers.*

2.2. External Cycle EtherCAT Communication

The external cycle (see green part of block diagram in Fig. 1) is computed within 200 μsec and handles mainly the EtherCAT process data communication (Fig. 5). The input and output process exchanges the relevant data between the motor controller and the EtherCAT Master PC. The structure and type of the process data are defined in the CiA402 CANopen device profile for drives and motion control description. For an easy usage of the motor control board, the firmware supports a set of mandatory parameters. Parameters definition is done in a separated XML file. It describes the MCB as an EtherCAT slave and allows usage of the MCB with different EtherCAT masters. EtherCAT

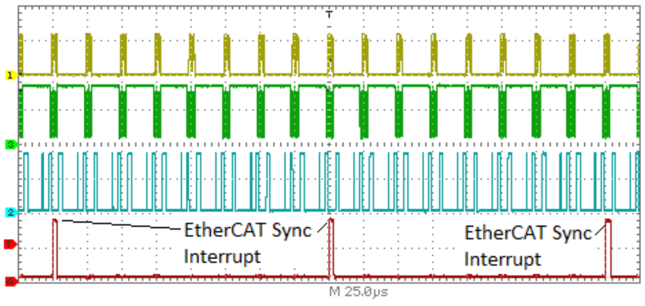


Fig. 5: *200 μsec EtherCAT PDO Cycle.*

supports also asynchronous communication using a mailbox for transmitted and a mailbox for received messages. The firmware can handle different types of services by using these mailboxes. One is called “Ethernet over EtherCAT” (EoE) and is being used to access the MCB by a regular Ethernet connection from the master. An application like Simulink can use this connection to exchange parameters from the external mode with the MCB. The connection appears transparent to the application no matter if transport layer is EtherCAT or Ethernet. The EtherCAT master inserts the mailbox communication between the process data communication. It happens in a way, that no disturbance or delay occurs for the process data and real-time conditions are ensured. Because of that, the mailbox communication gets just a fraction of the 100 Mbit/s EtherCAT bus. In our tests it was around 3% of the bandwidth or 420 Kbytes/s. Depending on the size of the Simulink control model, it usually allows a smooth observation of the model parameters at the master PC.

As long as the Simulink model can be adapted by parameter chances, the model remains unchanged in the firmware. Structural changes of the model require a Simulink generating step and an update of the MCB firmware. Which means the generated C code from Simulink must be recompiled with the other parts of the firmware. The whole firmware must be again loaded into the MCB and executed. The fastest way is to use the development connection to the board, which is available by a second Ethernet cable. Or using a further EtherCAT mailbox service called “File over EtherCAT” (FoE). The FoE service allows a whole firmware update when the MCB is connected just with the EtherCAT cable to the Master PC. An unique ID in the model and the same ID in the generated C code allows checking if the same model is used in Simulink application running on host PC and in the MCB firmware.

For the EtherCAT master there are several topologies feasible. Fig. 2 shows the solution ① using a Beckhoff IPC to ensure precise EtherCAT master real-time cycle. A host PC with Ethernet communication to the IPC is running the Simulink environment for developing and analysis. The approach ② uses a host PC with TwinCAT software running. This way a single Windows PC is used for developing and analysis as well as for setting up an EtherCAT master. It turned out that the real-time cycle precision was reduced because the EtherCAT master cannot maintain a precise cycle on a normal Windows PC. Nevertheless it is sufficient for reliable communication. Solution ③ is similar to solution number one but uses a real-time Linux computer with an open source EtherCAT master featuring EoE and FoE.

3. Hardware Design of the Motor Control Board

The hardware design follows the concept shown in Fig. 2. As the focus of the development was put on the investigation of the software toolchain a quick prototype that is flexible for testing purpose was the goal. An approach of driving two motors with only one ARM CPU was also under investigation. Due to the background of robotics high dynamics and motors with low inductance were targeted. Thus a controller bandwidth of 40 kHz was more important to the application than dual axis operation. The investigation and interface testing capabilities is why the board envelope size is as big as 20 cm x 30 cm. A follow-up hardware is planned to fit to the volume of 50 mm x 50 mm x 25 mm and deliver electrical power of 750 W.

The following technical components are to be mentioned on the MCB. The ARM CPU which was integrated by means of phyCORE industry module. With every necessary interface option available this module reduced PCB design effort a lot. The power stage consists of a standard DRV8332 motor driver chip and a three-shunt inverter leg current sensing. A set of standard DC-DC converter modules provides 3V3, 5 V and 12 V from 24 V supply voltage.

The drive unit under test was taken from the light weight robot developments of the institute. This type of robotic joint served as base line for many different robotic systems and test benches (see [7] and [8]). The drive module consists of a RoboDrive ILM50 motor with a magneto resistive AMR absolute position sensor and harmonic drive gearing. The absolute position sensor makes use of two incremental tracks, with different amount of magnetic poles, that are computed by use of nonius principle. Additionally a force torque sensor completes the modular joint.

Because of the limited amount of SPI lanes on the ARM CPU it may be necessary to adapt the electronics depending on the needed sensor types.

4. Controller Development Example

Permanent magnet synchronous motors have taken over DC motors in industry and their importance is growing also in space-applications due to their advantages:

- High torque output per volume and weight
- Extreme durability and reliability
- No brushes: No brush sparking, no cold welding of brushes and motor collector, no brush wear debris that shortens motor windings and destroys the drive train
- Better heat dissipation because the copper windings are located in the stator only

However these type of motors, as opposed to DC motors, have to be electronically commutated and they don't move just by applying DC voltage. A commutation algorithm has to be developed. Furthermore an implemntation of the field oriented control algorithm is necessary. With the coordinate transformation of three motor phases to a stator fixed reference frame $\alpha\beta$ and a rotor fixed reference frame dq which is aligned to the north pole of the rotating permanent magnets, the motor control algorithm is split into d- and q-axis.

4.1. Space Vector Modulation

The Space Vector Modulation (SVM) is a well known technique to modulate PWM signals for 3-phase machines. This method treats the three sinusoidal phase voltages as a single rotating vector with certain amplitude and angle or frequency. It approximates the desired voltage vector by a combination of eight switching vectors (V_0, V_7), as seen in Fig. 6. V_0 and V_7 are not active vectors and are used to set the desired amplitude. The active vectors are V_1 to V_6. During the pulse period T_p a pattern of space vector combination results in a approximation of the desired voltage vector. Since

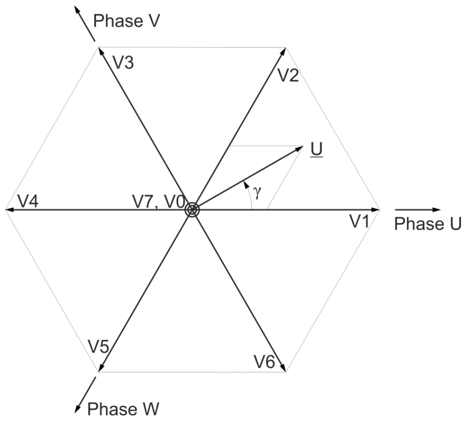


Fig. 6: *Space vector representation of motor phase voltages used for SVM.*

the field oriented control makes use of convenient Park-Clarke coordinate transformation the phase voltages to be realized are accessible as values in stator coordinate system $\alpha\beta$. In [9] coefficients a, b and c are computed (1).

$$a = |U_\alpha| + \frac{1}{\sqrt{3}}|U_\beta|\,;\, b = |U_\alpha| - \frac{1}{\sqrt{3}}|U_\beta|\,;\, c = \frac{2}{\sqrt{3}}|U_\beta| \tag{1}$$

A case differentiation considering the quadrant and sector of the desired voltage vector and simple addition and subtraction operations gives the time until switch to on (for quadrant 1 and sector 1). PWM counting is done in up-down fashion, allowing for a center alligned, symmetrical PWM shape as shown in Fig. 7. The SVM algorithm is part of the firmware and the Simulink model passes the PWM values for each half bridge to the firmware by use of a S-function block (see sSVM block in Fig. 9). If needed the algorithm can be adapted by writing appropriate C code.

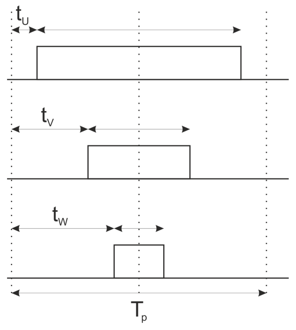


Fig. 7: *Diagram of center alligned, symmetrical PWM.*

4.2. Sensorless Algorithm Implementation

Direct feedthrough to the power electronics is for example needed when implementing sensorless control algorithms. Precise algorithms like SVM result in much smoother motor motion than the six-step algorithm, but need a position information with higher resolution to be doable. Typically two sensorless methods are to be implemented. The induced voltage is exploited by means of Kalman filter or sliding mode observers at higher speeds. Although an additional source of information is needed at low speed and standstill. Since the induced voltage is proportional to the motor speed there is no signal at standstill or only small signal amplitude in low speed region. The only source of information in that operating range is the magnetic saliency of the motor. In Fig. 8 a principal overview of that approach that was proposed in [10] by the author is illustrated. An advancement of that idea was for example shown in [11]. The idea is to add a high frequency signal to the d-voltage of the controller output right before the SVM block. The hf component needs to be extracted from the current to prevent the controller to react on that signal portion. The fundamental wave is used for field oriented control and the hf component is used to track the motor saliency and get the actual d-axis orientation of the coordinate system. The Simulink control algorithm shown in Fig. 9 can be setup according to block diagram Fig. 8. The resulting PWM wave with hf-signal modulation and the current reading from the motor are shown in Fig. 10. When tuning the angle estimator dynamics and investigating the filtering of motor phase currents the developed toolchain is absolutely essential.

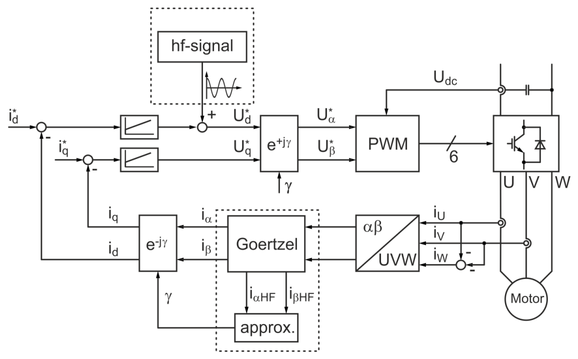


Fig. 8: *Block diagram of algorithm proposed in [10].*

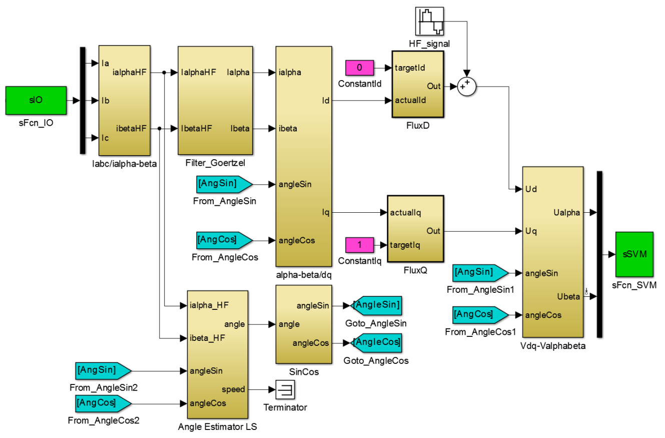


Fig. 9: *Simulink model*

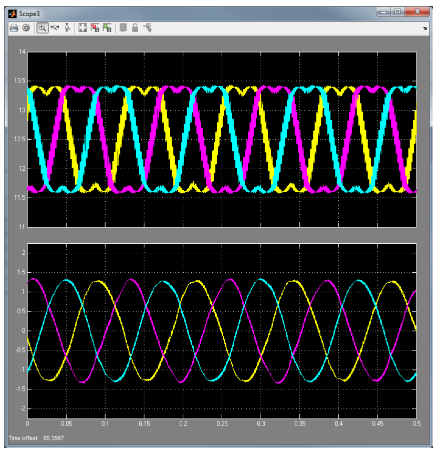
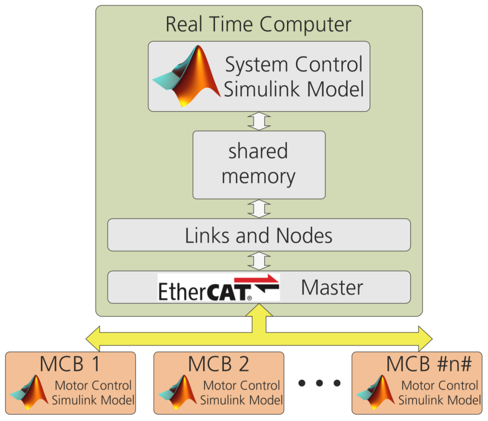


Fig. 10: *PWM wave with hf-signal modulation and filtered motor phase currents.*

5. Expansion to Multi-Axis Systems

Up to now the single-axis controller was considered and assumed to be investigated and tuned. Especially in the field of robotics but also in industry applications complex systems require the synchronized use of multiple degrees of freedom. In Fig. 2 it is indicated that it is possible to connect up several MCBs on one EtherCAT bus. To handle high degrees of complexity in real-time the approach to use Simulink for the high level controller programming is being pursued. The in house software development Links and Nodes (LN) enables the user to have access to system sensor data and controller values in real-time at EtherCAT bandwidth by use of Simulink model. DLR LN provides scalable process control and monitoring as well as two complementary interprocess communication paradigms: a real-time capable publish-subscriber and a performance optimized request-response mechanism. Supported operating systems include linux, qnx, vxworks and windows. All communication mechanisms can be logged and we aim to provide full control of all real-time features provided by the particular operating system. Individual hosts only need to be reachable via at least one ip link. LN will be released as GPLv3-licensed open source. In Fig. 11 an overview of the hierarchical software organization when extending to multiple axis systems is given. With growing amount of sensor data and degrees of



Real Time Computer
System Control Simulink Model
shared memory
Links and Nodes
EtherCAT Master
MCB 1 Motor Control Simulink Model
MCB 2 Motor Control Simulink Model
MCB #n# Motor Control Simulink Model

Fig. 11: *Combining several MCBs to a complex mechatronic system that is controlled via Simulink model.*

freedom to handle the need for hardware in the loop testing capabilities and rapid control development toolchains based on Simulink is an essential advantage. The MCB fits perfectly to that environment. All kind of control level use the same approach and toolchain. From the high level control loop down to the current control loop of a single motor the only difference is the scaling of the sample bandwidth.

6. Conclusion

In this paper a toolchain for developing control algorithms for electrical drives is presented. The ease of use and comfort of investigating control algorithms in real-time that is known from large and expansive rapid control development systems is transfered to a compact solution. With the computation power of affordable ARM processors complete Simulink models can be downloaded to the hardware. Furthermore with EtherCAT a standard communication bus technology is used solely to interface the hardware to a host PC that is used for the development process. When the commissioning and parameterization is fully completed the hardware can remain unchanged in the mechatronic system. Whenever diagnosis functionality is needed the standard EtherCAT communication can be used to analyse system behavior. Also remote maintenance and updating is feasible with the presented toolchain. In the next development step the presented MCB is miniaturized to a compact size to fit to a modular drive unit.

7. References

[1] Christoph Borst, Thomas Wimböck, Florian Schmidt, Mathias Fuchs, Bernhard Brunner, Franziska Zacharias, et al. Rollin'Justin - mobile platform with variable base. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 1597–1598, Kobe, Japan, May 2009. IEEE.

[2] Martin Schuster, Sebastian Brunner, Kristin Bussmann, Stefan Büttner, Andreas Dömel, Matthias Hellerer, Hannah Lehner, Peter Lehner, Oliver Porges, Josef Reill, Sebastian Riedel, Mallikarjuna Vayugundla, Bernhard Vodermayer, Tim Bodenmüller, Christoph Brand, Werner Friedl, Iris Grixa, Heiko Hirschmüller, Michael Kaßecker, Zoltan-Csaba Marton, Christian Nißler, Felix Ruess, Michael Suppa, and Armin Wedler. Towards autonomous planetary exploration. In *Journal of Intelligent & Robotic Systems*, November 2017. `https://doi.org/10.1007/s10846-017-0680-9`.

[3] Chris Shore. Arm community blog which cores support hardware divide. `https://community.arm.com/processors/b/blog/posts/divide-and-conquer`.

[4] ARM. Cortex-a8 revision: r2p0 technical reference manual. `http://infocenter.arm.com`.

[5] Beckhoff. Ethercat slave stack code (ssc) et9300. `https://www.beckhoff.com/`.

[6] iC-Haus. BiSS interface protocol description c-mode. `http://biss-interface.com/download/biss-c-protocol-description-english/`.

[7] Gerhard Hirzinger, Norbert Sporer, Alin Albu-Schäffer, Matthias Hähnle, Rainer Krenn, Antonio Pascucci, and Manfred Schedl. DLR's torque-controlled light weight robot III - are we reaching the technological limits now? In *Proceedings 2002 IEEE International Conference on Robotics and Automation ICRA (Cat. No.02CH37292)*, volume 2, pages 1710–1716 vol.2, 2002.

[8] Johannes Englsberger, Alexander Werner, Christian Ott, Bernd Henze, Maximo A. Roa, Gianluca Garofalo, Robert Burger, Alexander Beyer, Oliver Eiberger, Korbinian Schmid, et al. Overview of the torque-controlled humanoid robot TORO. In *14th International Conference on Humanoid Robots (Humanoids) IEEE-RAS*, pages 916–923. IEEE, 2014.

[9] N. P. Quang and J.-A. Dittrich. *Vector Control of Three-Phase AC Machines - System Development in the Practice*. Springer-Verlag GmbH Berlin Heidelberg, 2nd edition, 2015.

[10] Josef Reill, Bernhard Piepenbreier, and Ingo Hahn. Utilisation of magnetic saliency for sensorless-control of permanent-magnet synchronous motors. In *International Symposium on Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM)*, Pisa, Italy, September 2010.

[11] Markus Seilmeier, Sebastian Ebersberger, and Bernhard Piepenbreier. HF test current injection-based self-sensing control of PMSM for low- and zero-speed range using two-degree-of-freedom current control. *IEEE Transactions on Industry Applications*, Vol. 51, No. 3, May/June 2015.