A Distributed Digital Control Architecture for Power Electronics Systems

Ivan Celanovic

Thesis submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirement for the degree of

Master of Science
in
Electrical Engineering

July 20, 2000
Blacksburg, Virginia

Keywords: power electronics, distributed digital control, voltage source inverter, real time control protocol, industrial automation protocols, PLD, DSP, fiber optic link

Distributed Digital Control Architecture for Power Electronics Systems

Ivan Celanovic

(ABSTRACT)

This thesis proposes a novel approach to power electronics system design that is based on the open-architecture distributed digital controller and modular power electronics building blocks (PEBBs). The proposed distributed digital controller partitions the controller in three levels of control authority. The power stage controller, designated as hardware manager, is responsible for low-level hardware oriented tasks; the high level controller, designated as applications manager, performs higher-level application-oriented tasks; and the system level controller handles system control and monitoring functions.

Communications between the hardware-oriented controller and the higher-level controller are implemented with the previously proposed 125 Mbits/sec daisy-chained fiber optic communication protocol. Real-time control and status data are communicated by means of communication protocol. The distributed controller on the power converter level makes the system open, flexible and simple to use. Furthermore, this work gives an overview and comparison of current state-of-the-art communication protocols for real-time control applications with emphasis on industrial automation and motion control. All of the studied protocols have been considered as local area networks (LAN) for system-level control in power converter systems. The most promising solution has been chosen for the system level communication protocol.

This thesis also provides the details of design and implementation of the distributed controller. The design of both the hardware and software components are explained. A 100 kVA three-phase voltage source inverter (VSI) prototype was built and tested using the distributed controller approach to demonstrate the feasibility of the proposed concept.

ACKNOWLEDGEMENTS

I sincerely believe that the very purpose of life is to seek happiness. Have I found it? That is a difficult question, and might lead to a discussion about the meaning of life, which certainly is not my intention at this point. Yet, the pursuit of happiness might be the answer. The chance that I have been given, to work on my thesis with so many great people, has enriched me and changed my perception of life forever. I feel greatly indebted to all of them.

Although the list of all the people whom I have met and with whom I have worked in Blacksburg is endless, I have tried to condense it as much as possible. Also, the nature of the paper as a medium and our inability to perform parallel reading forced me to list people in sequence. However, the position does not necessarily reflect the correlation between the stars.

First of all, I would to express my eternal gratitude to my brother, Nikola. I honestly believe that without him none of this would have happened. His continuous support, encouragement and parent-like care made me what I am today. Also his creative thinking and clear ideas contributed immensely to this work. Our infinite brainstorming sessions and continuous late hour discussions helped me change my perception of life and energy as well.

I cannot find the right words to express the admiration and sincere gratitude I feel towards my advisor Dr. Dushan Boroyevich. His broad knowledge, ahead of the time thinking and provocative teaching have been a source of inspiration throughout the course of this work. His continuous support and encouragement helped me survive many difficult moments.

Further, I would like to thank to Dr. Fred Lee for his teaching and guidance during my studies at Virginia Tech. His vision and profound way of thinking have always amazed me.

I would like to thank to Dr. Hugh VanLandingham and Dr. Pushkin Kachroo for serving on my committee and for teaching the courses that I have enjoyed so much.

Many thanks to Dr. Krishnan Ramu for his support, encouragement and his beautiful teaching I loved so much. Our nice and enriching talks helped to expand my horizons.

I can never exaggerate how grateful I am for getting the chance to be part of the big VPEC/CPES team. I am greatly indebted to all the students I have worked with and learned from, very often not only in the technical field: Cristophe Crebier, Raphael Caire, Xiukuan Jing, Kun Xing, Dengming Peng, Francisco Canales, Dr. Sriram Chandrasekaran, Choi, Jae-Young, Carlos Cuadros, Ivana Milosavljevic, David Wu, Peter Barbosa, Sam Ye, Luca Amoroso, Jinghong Guo, Anna Pasquini, Mauro Donati, Ivan Jadric and many others. Thank you guys.

I would also like to thank my dear friend Stanislav Licul for all the great times and nice talks we had.

I would like to thank Dr. Terry Ericson for his vision of the PEBB program. His advanced ideas and daring thinking have served as a big inspiration to me and many others.

I would like to thank Mr. Slobodan Gataric for his precious help during the course of this work.

Many thanks to Mr. Roger Cooley for his help. Many of his nice ideas were incorporated in this work.

I would like to thank Ms. Sue Downey for her help, encouragement and support on numerous occasions.

I would also like to acknowledge the great VPEC/CPES staff members who were always around to give me the hand and nice word whenever it was needed: Trish Rose, Teresa Shaw, Lesli Farmer, Mike King, Kelly Shannon and Joyce Moser.

My parents have been with me all these years. Their teaching, care, support and understanding have opened countless opportunities for me I had never dreamed of. Without them, none of this would be possible.

I cannot express the happiness and joy my beautiful girlfriend Marina has brought into my life. Her support, love and understanding has given me the will and strength to succeed and to continue my journey in the quest for happiness.

This work was sponsored by Office of Naval Research and in part This work made use of ERC Shared Facilities supported by the National Science Foundation under Award Number EEC-9731677.

To my family

and

to the memory of my grandfather Niko

Table of Contents

List of Illustrations

List of Tables

Chapter 1 Introduction

1.1 Thesis Motivation and Objectives

Over the past few decades, significant advancements in technology have advanced power electronics to the point where it has become the enabling technology for many engineering fields. Today, power electronics technology is used in a large number of applications, ranging from motor drive control, hybrid electric vehicles and power-distribution systems on one side all the way to the computer power supplies, telecommunications equipment and automotive applications on the other side.

Although power electronics has penetrated a variety of markets and has demonstrated the ability to provide efficient, reliable and compact solutions for energy conversion in many areas, there is still a large field of potential applications where power electronics solutions have not proliferated [1, 2, 3, 4, 6]. Indeed, there are many reasons behind this, but cost and reliability are probably the most significant ones [1, 4]. Today's power electronics is, for the most part, a business of custom designers [4]. There are very few universal and flexible, off-the-shelf solutions that can be easily applied to a particular problem, especially in the medium- and higher-power range. Rather, whole new designs are being developed or in some cases available designs are being customized to a fit particular application. In general, this approach, suffers from large development costs, significant time to market and low production volume [1, 4].

For some applications (e.g., the silver-box in personal computers), custom design is not a drawback, since it allows for overall converter optimization, yielding reduced costs and improved manufacturability. However, many applications (especially in the medium- and high-power range), can not justify single application development costs due to their relatively low volumes [4, 5, 6]. For these types of applications, engineering costs are still a predominant factor in the final product's costs.

Modularization and standardization of power converter architecture could overcome many of these shortcomings. Having multifunctional, flexible and universal building blocks that can be assembled and reconfigured easily to form a wide range of different applications would increase production volume. This in turn would reduce manufacturing costs due to the economy of the scale.

The term building block does not necessarily refer to a physical block. It can be understood to be an abstract object that can be either a power module or a digital signal processor (DSP) code module, or a whole control algorithm or something else. In that sense an object-oriented approach would allow for reduced time to market, which would cut the overall engineering costs.

Modularization of power converter architecture is not sufficient by itself and will not provide significant advancements in power electronics technology if standardization is not achieved in parallel. Finding common denominators for different power architectures and standardizing the building blocks--or at least standardizing interfaces and communications between objects (subsystems)--could bring a totally new twist to power electronics design and practice.

Therefore, this work seeks a way to standardize controls and communications in power electronics. We are not proposing how to achieve optimum power converter modularization and partition, but rather investigating standardized control and communication architectures that would complement different power electronics system partitions. This work investigates an open system design approach that will allow for different manufacturers and different modules to be used in continuously changing architectures.

Similar ideas have revolutionized the personal computer industry and influenced all segments of our lives [25]. Designing computer architecture in an open fashion that can be constantly modified, upgraded and used for many diverse applications opened new frontiers that were previously beyond comprehension. Different manufacturers were able to design their own cards, controllers, hard-drives and applications that had PC-compatible interfaces. The architectural openness brought a huge increase in the production volume. One was able to use the same elements for different applications by simple software management. Furthermore, this concept enabled healthy competition between manufacturers, further reducing the price and increasing availability. Achieving similar objectives in power electronics system design is the ultimate goal.

1.2 Digital Control Architectures for Power Converter Systems

In today's medium- and high-power converters, control architecture is (in the most cases) centralized. In centralized controller topology, all control and monitoring functions (regardless of the size, type of topology, power rating or spatial distribution of the power stage) are performed by a single, centralized controller. This approach exhibits several major drawbacks, such as. poor system modularity; limited system flexibility and reconfigurability; large number of point-to-point links and interconnections; and under utilization of transmission media.

As an example a two-level three-phase voltage source inverter (VSI) controlled by single centralized controller is shown in Fig. 1-1. The interface between power stage and controller has eighteen point-to-point mixed signal links. For the same type of application, if a three-level VSI (shown in Fig. 1-2) is employed, the interface between power stage and controller becomes significantly more complex, amounting to 30 point-to-point links. Furthermore, different power levels and different applications might require different levels of power stage modularization, and there is no industry-wide standard regulating interconnections and communications on the power converter level.

The complexity of the interface between the power stage and the controller, and its strong dependence on the type of converter that is used, hinders the potential for modularization and standardization of power converter architecture. Even if the power stage is modularized and standardized, the interface between the controller and power stage remains a big issue. Furthermore, paralleling converters that share a common controller (or part of the control) is becoming widely accepted practice. In this case, communication and control architecture becomes even more complex and a centralized converter approach proves to be inadequate.

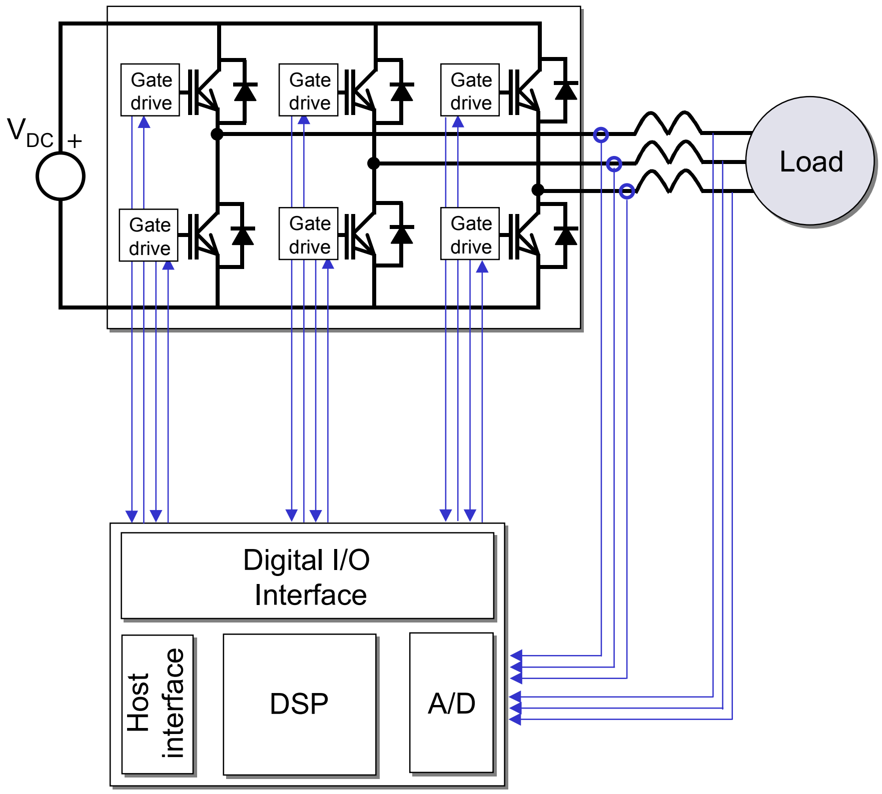


Fig. 1-1. Three-phase voltage source inverter controlled using centralized control architecture.

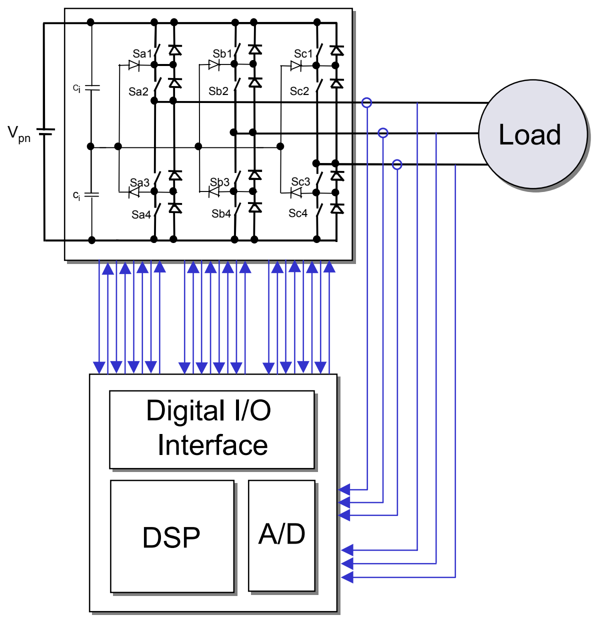


Fig. 1-2. Three-phase three-level voltage source inverter using centralized control architecture.

The problems of interface standardization and modularization of control architecture for power converters have been investigated to some extent, and some solutions have been proposed in literature.

In one case [14] authors proposed to split the control authority for large drives between the regulator block on one side and the bridge control block on the other side as shown in Fig. 1-3. The regulator block contains all the algorithms dedicated to the true control of the motor in terms of currents, voltages, speed, etc. Meanwhile, the bridge control block, contains all the algorithms that generate adequate gate signals for specific power converter topology. In other words, the regulator performs tasks not strictly related to the topology of the specific converter, while the bridge control must satisfy the requirements defined by the gating sequence and protection criteria of the power converter. The interface between the regulator and the bridge controller is implemented by means of a relatively slow parallel bus.

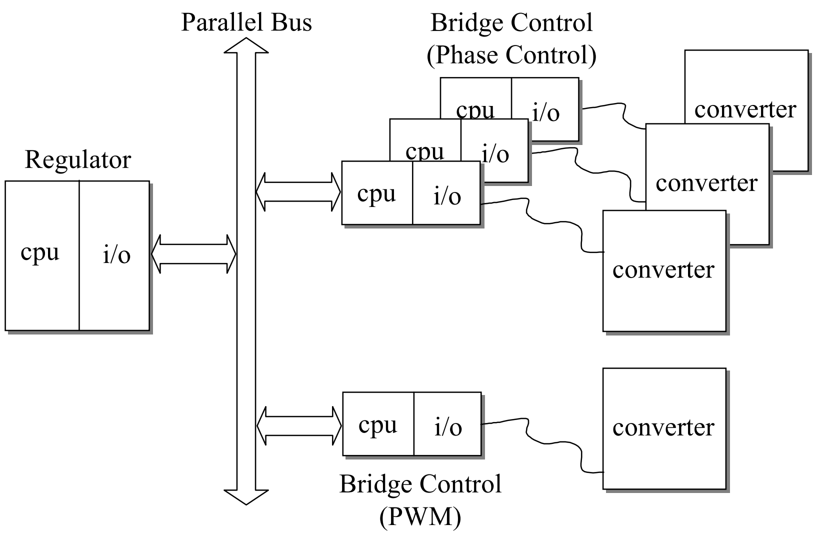


Fig. 1-3. Block diagram of control architecture for large drives [14].

A different approach to distributed controller, proposed in [13], where each phase leg is controlled by a separate module controller, is shown in Fig. 1-4. The module controller performs the inner-loop control, regulating either the current or the voltage. Communication between the outer-loop controller and the modules is achieved through a 2.5 Mbits/sec fiber-optic daisy-chained communication link, as illustrated in Fig. 1-4.

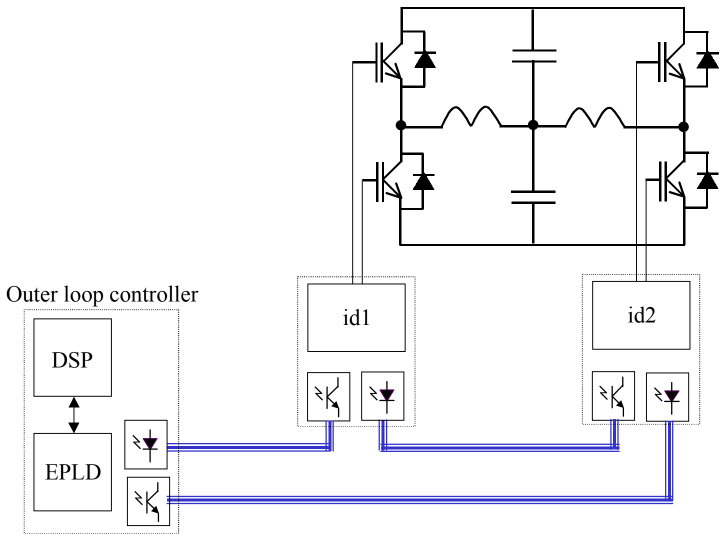


Fig. 1-4. Block diagram of distributed control architecture proposed in [13].

The issue of power electronics communications have also been investigated in [11]. The novel concept of optical daisy-chained high-speed protocol for communication between power electronics modules (phase-legs) and their controller have been proposed (Fig. 1-5.)

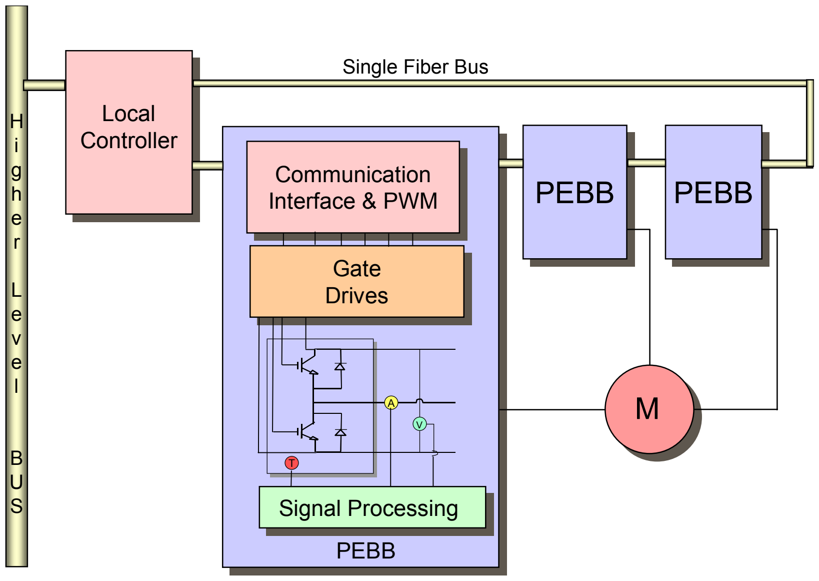


Fig. 1-5. Power converter system with daisy-chained phase–legs [11].

This thesis proposes a novel open system distributed digital controller for power converter systems based on previously proposed [11] digital communication protocol. The new control architecture is divided into two levels of control authority: the higher-level controller designated as applications manager and lower-level controller designated as hardware manager. The applications manager handles application-level tasks, such as control of three-phase VSI or control of the full-bridge DC-DC converter or some other application level control. The next level of control authority is the hardware manager. It is designed to accept the control commands from the applications manager and to port them to specific hardware implementation. The hardware manager becomes an integral part of the power stage and takes care of low-level hardware specific tasks, such as pulse width modulation (PWM), dead-time compensation, protection, and optimization of gate drive signals.

The backbone of the control system is the daisy-chained fiber optic communication network [11]. This communication protocol provides communication between the applications manager and hardware managers. In this work, the communication protocol has been extended to allow for much wider application. It is designed to support any type of module, regardless of topology and functionality, and to provide real-time data distribution. Furthermore it is designed to provide system flexibility, and reconfigurability in a plug and play manner.

Moreover, this thesis presents detailed description of the hardware manager and the applications manager. The hardware designs of the hardware managers and applications managers and the software design of the control units have been extensively explained. A prototype 100 kW three-phase inverter has been designed and tested using a phase-leg-based hardware managers and applications manager to demonstrate the feasibility of this approach.

Chapter 2 Analysis of Power Electronics System's Control Architecture

Today's power converter systems tend to exhibit the characteristics of distributed systems. Although it is quite difficult to measure the level of system distribution, the levels of hardware distribution, control distribution and data distribution can be adopted as general indications of systems distribution [21]. A similar approach can be used to analyze the distribution of power electronics systems.

Hardware distribution is, in most cases, the consequence of non-ideal behavior of power processing elements. Power switches are limited in maximum current voltage and temperature ratings. The maximum power that can be extracted through a certain area is limited by material properties. Therefore, the higher the power rating the bigger the size of the module. Also, the devices limited current and voltage ratings impose the need for paralleling and/or stacking in series multiple devices in order to meet the desired power requirements. This can significantly increase both the system's spatial (size) and control (number of controllable objects) distribution.

In addition, power switches have non-ideal switching characteristics. The limited switching frequency of the device sets a boundary on the size of passive components. This limits the level of integration and determines the converter size. On the other hand, non-ideal switching behavior of the switches introduces the need for snubber circuits, which introduce additional passive components.

Therefore, the number and spatial distribution of power modules vary significantly with power level, state-of-the-art in power integration, and type of application.

On the other side, control structure of power electronics systems can also be treated as a distributed system. In this case, the distribution can be qualified as temporal rather then spatial. Indeed, different segments of control architecture have control authority over different physical processes that are governed by different time constants.

Therefore, the following chapter will try to investigate some of the basic properties of hardware and control distribution that will point toward “optimum” and unified control architecture. This architecture would complement different types of system distribution in both control and hardware sense and would simplify the task of system design, while at the same time increasing the modularity and flexibility of the power electronics converters and systems.

2.1 Hardware Distribution

In recent years, power electronics have witnessed an exponential increase in the switching power of devices (mostly IGBT-s and IGCT-s). Although further advancements in this area are expected, the trend shows slight saturation [2,3]. The switching power for a certain device can be defined as:

$$P_S = U_M \cdot I_M ,$$

where U_M and I_M are defined as the maximum sustainable voltage and current across the device. Fig. 2-1. shows the historical development of the switching power P_s for different power semiconductor devices.

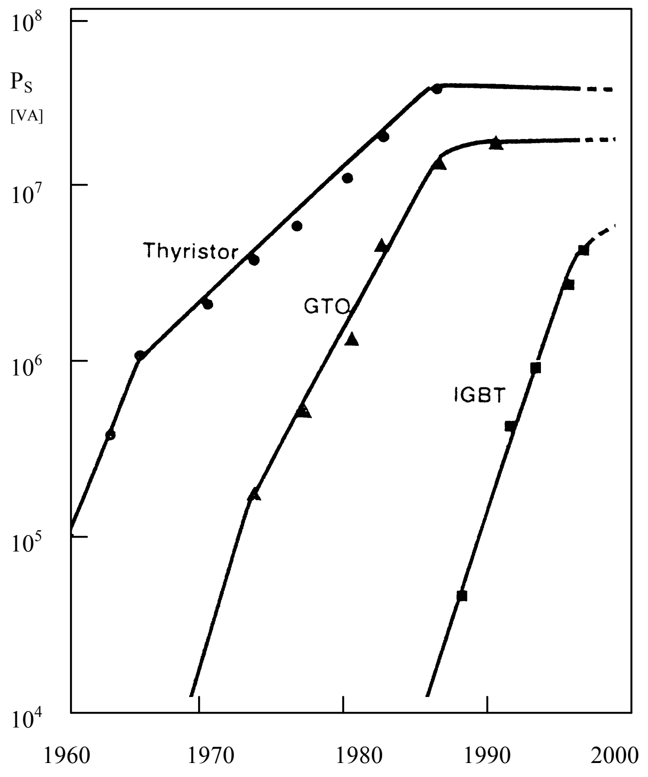


Fig. 2-1. Historical development of the switching power, Ps, for the power semiconductor devices [2,3].

The maximum switching power limitation for a device underscores the fact that in order to achieve higher ratings for certain applications, series and parallel connections of devices have to be employed. Paralleling and stacking devices in series increases hardware distribution and the number of interconnections and makes the control architecture more complex. All these factors contribute to increased size and cost of the power converters. Therefore, the industry has introduced different types of switching power modules connecting devices locally on the module level, thus increasing integration and reducing the number of interconnections. These modules consist of either series or parallel connections of devices, or both. In some cases, modules comprise the whole converter with a certain level of control. Most of the high-power IGBT modules are built using a parallel connection of multiple IGBTs within the module in order to increase the maximum current capability.

Yet this approach also has inherent limitation and cannot increase the power density nor increase the switching power capabilities of the converter beyond a certain level. Current state-of-the-art capabilities of switching power modules for different technologies are shown in Fig. 2-2. Although future advancements are expected, hardware distribution is the fact designers will have to live with.

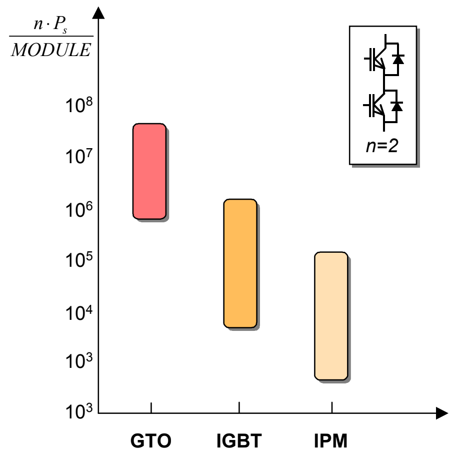


Fig. 2-2. Total switching power of different modules with various levels of integration (n equals the number of branches per module, for example if module is phase leg with two switches, n=2, as shown in the upper right corner of graph; IPM (intelligent power module))

Also, power dissipation becomes one of the limiting factors in the further integration of power semiconductor devices and modules. The physical properties of semiconductors, thermal limitations of packaging materials and fundamental limitation of two-dimensional packaging techniques all form a significant barrier against further integration and miniaturization of power electronics. Fig. 2-3 shows the maximum power dissipation per cm^2 for different types of switching devices and modules. Those numbers can be taken as a measure for the spatial distribution of the power converter.

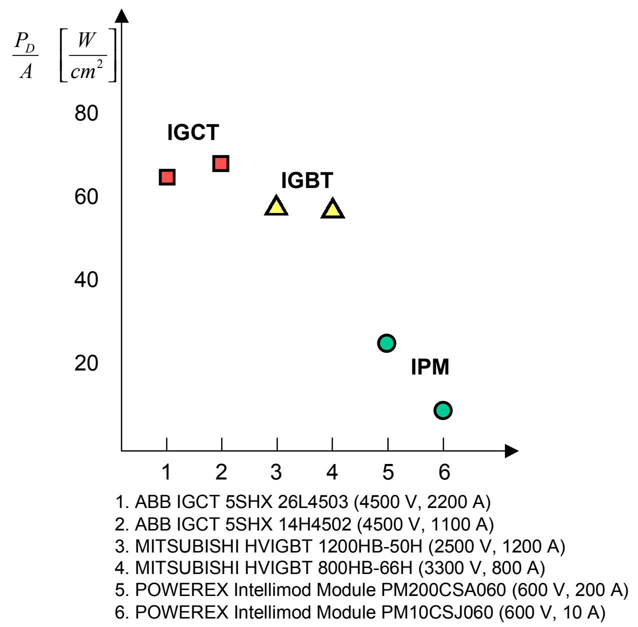


Fig. 2-3 Maximum power dissipation per square centimeter of module cross-section area.

Apart from the switching power limitation and maximum dissipation, the maximum switching frequency presents another significant limiting factor for overall size and cost reduction of power electronics systems. Fig. 2-4 shows switching power versus typical switching frequency for different devices. The empirical formula

$$P_S \approx f^{-1}$$

shows that for high-power applications, switching frequency tends to decrease, thus increasing the size of the passive components. This is also a significant indicator for the level of hardware distribution. This parameter points out that the size of the filter components remains significant for higher power applications even if larger integration of switching devices could be achieved.

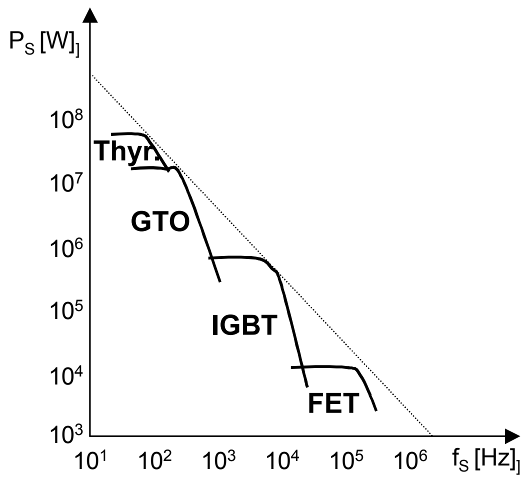


Fig. 2-4. Maximum switching power as a function of switching frequency for different devices [2].

This review of current state-of-the-art in power device integration reveals the well-known fact that for different power levels different levels of integration can be achieved. On a lower power level, a whole converter can be integrated in a module; on a medium power, level several devices can be integrated in a module; and for the highest power levels module consists of a single device.

To illustrate the difference in hardware and control distribution, two example converters that span the high-power range will be considered: a 100 kW three-phase rectifier/VSI [21] and a 19 MW 15-phase VSI for all-electric ship propulsion drive [39], shown in Fig. 2-5.

TABLE 2-1. Comparison of two example high-power converters

Converter type	Power stage volume	Number of modules	Switching frequency	Number of control units
100 kW three-phase rectifier/VSI	0.3m x 0.3m x 0.2m	3 x 2	20 kHz	1
19 MW 15-phase VSI	2m x 4m x 2.5m	15 x 16	2 kHz	3

illustrates basic features of both converters. While the three-phase rectifier/VSI is built with six power modules and the power stage can fit in a space approximately 0.015 m^3, the 19 MW inverter needs 240 power modules and occupies nearly 20 m^3. On the other hand, the switching frequencies of the converter exhibit an order of magnitude difference (20 kHz compared with 2 kHz). Also, the three-phase rectifier/VSI is controlled by a single DSP controller board, while the 19 MW inverter needs at least three independent controllers that can operate synchronously. The number of sensors and sensed variables differ significantly, too.

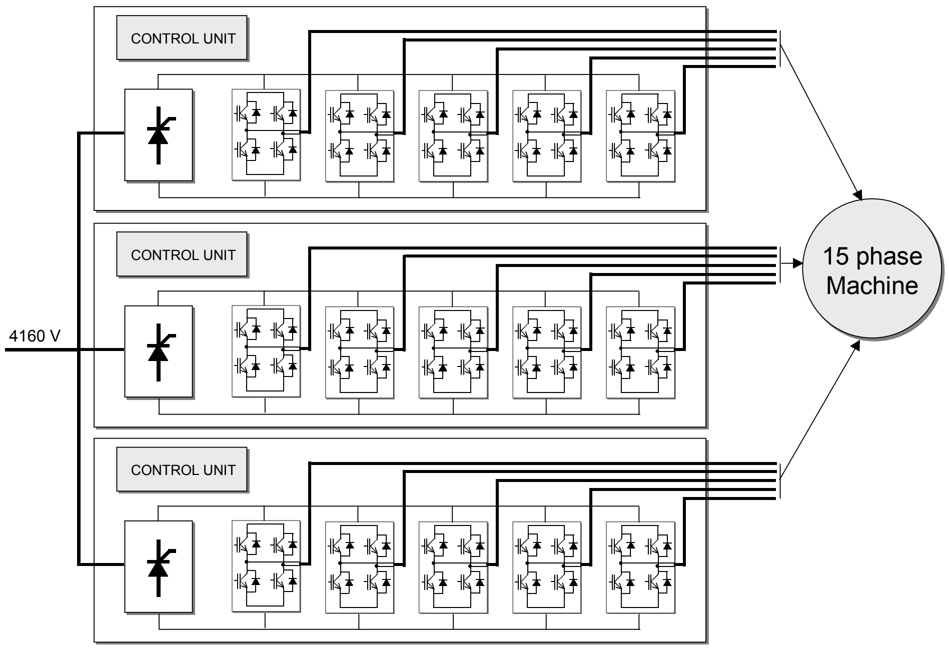


Fig. 2-5. Block diagram of a 19 MW 15-phase drive for electric ship propulsion [39].

It becomes obvious that different power levels tend to exhibit different levels of system distribution in terms of hardware, control and data distribution. Making power converter systems modular, flexible, and standardized, requires control architecture that is able accommodate all these differences in a uniform and consistent way, by means of a distributed control approach.

Splitting the controller between the power stage controller, higher-level controller and the system-level controller provides the system with many attractive features. Modules equipped with the intelligence become simple to use and control, while the control architecture becomes open and flexible.

2.2 Control Distribution in Power Electronics Systems

Control architecture of power electronics converters can be roughly divided into four segments as shown in Fig. 2-6. These segments are: gate drive, modulator with inner loop, load controller and system-level controller. Each segment of the controller governs a different time domain. The gate driver is controls the power switches with time constants that are less then 1us; modulator and inner loops are responsible for processes that are dominated by time constants in the order of tens of microseconds; load controller governs the domains with dominant time constants in order of tenth of millisecond; and the system controller is responsible for overall system monitoring and the highest-level control, usually governed by time constants in the order of tens of milliseconds.

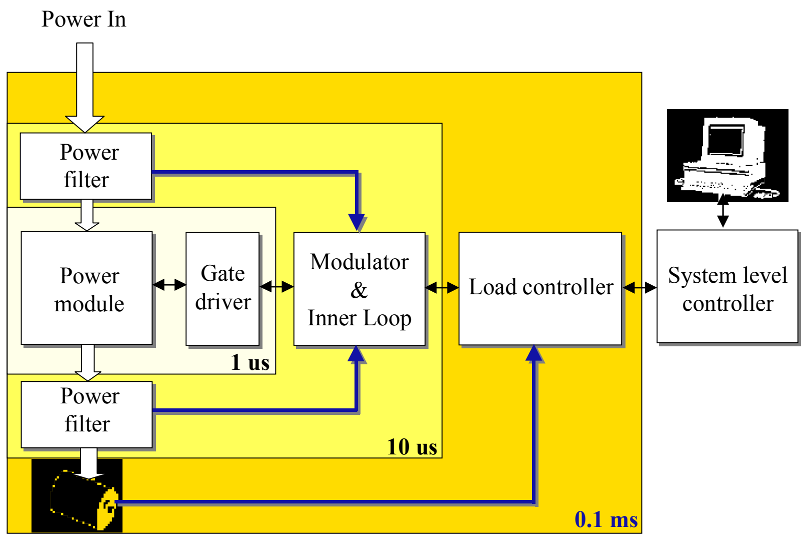


Fig. 2-6. Block diagram of power converter control architecture.

Although the power module in Fig. 2-6. is shown as a single block, depending on the power level and the level of system integration (as shown in and in Fig. 2-5), it can be composed of multiple modules distributed spatially, as explained in Section 2.1. Considering the examples given in, the “optimum” distributed control architecture—

which would minimize the number of communication paths between distributed controllers while providing enough flexibility and modularity—can be achieved by dividing the converter controller into three controllers, hardware manager, application manager and system controller. The principal block diagram of the proposed control authority partitioning is given in Fig. 2-7.

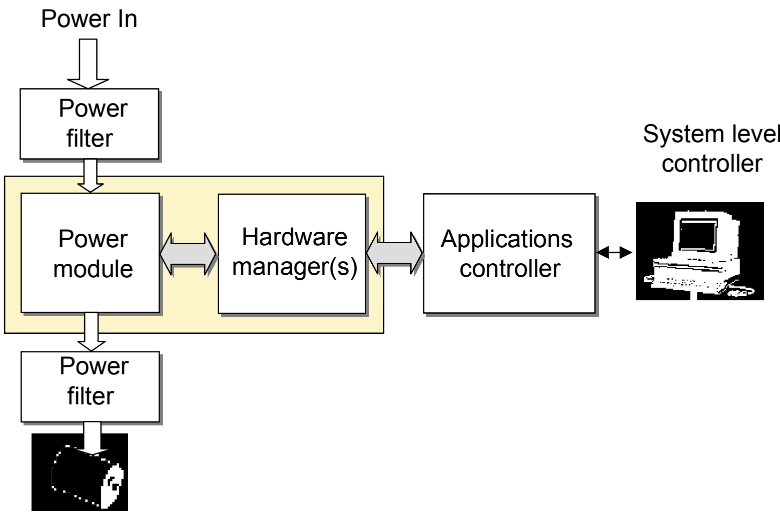


Fig. 2-7. Proposed distributed control architecture consisting of hardware manager(s) applications manager and a system level controller.

The hardware manager becomes an integral part of the power module. Regardless of the level of integration of the power module (single-switch module, phase-leg module or six-pack module), the hardware manager performs low-level hardware-oriented control tasks. Lumping the hardware manager with the modules makes the module intelligent and transparent for the user.

The applications manager assumes higher-level control tasks. These tasks are application-related (motor control algorithms, inverter control, etc.) and usually are independent of topology specifics. The system controller performs system level monitoring and control tasks.

Distribution of the control authority between the hardware and applications manager, determining the communication requirements for different spatial and temporal distributions, and determining which tasks certain partitions should assume in order to maximize system flexibility and modularity are the open questions. Design of the distributed control and tasks partitioning can largely improve system modularity, flexibility and user friendliness.

Therefore, the following section will look into optimal splitting of the control authority. Indeed, the question is where to locate the interface between the hardware managers and applications manager. Several prominent three-phase power converter topologies (three-phase VSI, three-phase boost rectifier, three-phase three level VSI) will be examined from the control architecture point of view and, a solution will be proposed.

2.2.1 Analysis of Three-Phase Boost Rectifier

The three-phase boost rectifier is increasingly used as the front-end converter in many medium- and high-power converter applications. Adjustable power factor, low harmonic content of input currents, bi-directional power flow capability and regulated DC output are fueling the drive for wider proliferation of those converters.

Modeling and control of three-phase PWM converters has been covered in details in [26]. This section concentrates on the analysis of control architecture for three-phase boost rectifiers, with special attention paid to the interface definitions between different segments of the control architecture. Also, certain common control objects that are used in other three-phase applications (such as space vector modulator, current controller etc.) are defined in terms of input-output properties and data formats. These definitions could be used to define data communication formats through different communication interfaces in distributed control architecture.

As shown in Fig. 2-10 the boost rectifier consists of six power switches with complementary freewheeling diodes, three input boost inductors, a DC-link filter capacitor, current and voltage sensors and a digital controller.

The digital controller consists of analog to digital (A/D) converters that sample converter currents and voltages, a voltage controller, a current controller and a space vector modulator with pulse generator that generates the right control sequence for power switches.

The controller is implemented in DQ rotating coordinates. Input currents i_a, i_b and i_c are being transformed using an abc to DQ coordinate transformation, given as:

$$T = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos\omega_r t & \cos(\omega_r t - \frac{2\pi}{3}) & \cos(\omega_r t + \frac{2\pi}{3}) \\ -\sin\omega_r t & -\sin(\omega_r t - \frac{2\pi}{3}) & -\sin(\omega_r t + \frac{2\pi}{3}) \end{bmatrix}, \tag{2-1}$$

where ω_r is defined as angular speed of the rotating coordinate frame. The transformed currents in a DQ coordinate frame are given by:

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = T \cdot \begin{bmatrix} i_{ab} & i_{bc} & i_{ca} \end{bmatrix} \tag{2-2}$$

and

$$\begin{bmatrix} i_{ab} \\ i_{bc} \\ i_{ca} \end{bmatrix} = \begin{bmatrix} i_a - i_b \\ i_b - i_c \\ i_c - i_a \end{bmatrix} \tag{2-3}$$

i_d and i_q currents are subtracted from reference values i_{dref} and i_{qref} that are obtained from the higher-level controller, which in this case is outer voltage regulator. The output of the current regulator (in general terms) can be expressed as:

$$\begin{bmatrix} \dot{d}_d \\ \dot{d}_q \end{bmatrix} = \begin{bmatrix} f_1(i_d, i_q, i_{dref}, i_{qref}, v_o, d_d, d_q) \\ f_2(i_d, i_q, i_{dref}, i_{qref}, v_o, d_d, d_q) \end{bmatrix} \tag{2-4}$$

Yet, most of the current controllers in industry are based upon linear proportional-integral (PI) regulators or some modification. In that case, the controller can be expressed as a linear transformation of input variables as follows:

$$\begin{bmatrix} \hat{d}_d \\ \hat{d}_q \end{bmatrix} = \begin{bmatrix} H_{dd}(z) & H_{dq}(z) \\ H_{qd}(z) & H_{qq}(z) \end{bmatrix} \cdot \begin{bmatrix} \hat{i}_d - \hat{i}_{dref} \\ \hat{i}_q - \hat{i}_{qref} \end{bmatrix}. \tag{2-5}$$

where ^ denotes a signal in z domain. Therefore, the output from the current controller is a desired voltage vector $\overrightarrow{V_{DQ}}$ given in DQ coordinates, where

$$\overrightarrow{V_{DQ}} = \begin{bmatrix} v_D \\ v_Q \end{bmatrix} \tag{2-6}$$

Finally $\overrightarrow{V_{DQ}}$ vector is transformed to αβ coordinates with inverse DQ transformation, using the following relationship:

$$\overrightarrow{V_{\alpha\beta}} = T^{-1} \cdot \overrightarrow{V_{DQ}} \ . \qquad (2\text{-}7).$$

Voltage vector $\overrightarrow{V_{\alpha\beta}}$, defined in (2-7) is the input variable for the space vector modulator, which approximates this vector (in average sense) over the switching period, using the appropriate combination of available switching state vectors. Accordingly, the output of the space vector modulator is a vector of switching state vectors and corresponding vector of duty cycles. The vector of duty cycles represents how long each switching vector should be applied at the output of the inverter. The vector of switching state vectors is given as:

$$\vec{V} = \begin{bmatrix} \overrightarrow{V_1} & \overrightarrow{V_2} & \overrightarrow{V_3} & \dots & \overrightarrow{V_n} \end{bmatrix}, \tag{2-8}$$

where

$$\overrightarrow{V_i} = \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix}, \qquad S_{abc} = \{p,n\}; \quad i=1,2 \dots n \tag{2-9}$$

depending on weather switch S_a is connected to the top (S_x=*p; x*={*a,b,c*}) or the bottom DC rail (S_x=*n; x*={*a,b,c*}). The row-vector of duty cycles is defined as

$$\vec{d} = [d_1 \quad d_2 \quad d_3 \quad \cdots \quad d_n], \tag{2-10}$$

where d_i is defined as the duty cycle of the *i*-th vector. The duty cycle is defined simply as the ratio between the time for how long the *i*-th vector is applied (T_i) and the switching period (T_s) of the converter:

$$d_i = \frac{T_i}{T_s}. \tag{2-11}$$

Equation (2-12) must always be true:

$$\sum_i d_i = 1. \tag{2-12}$$

For example, if

$$\vec{V} = \begin{bmatrix} p & p & p \\ p & p & n \\ p & n & n \end{bmatrix} \tag{2-13}$$

and row-vector of the corresponding duty cycles is given as

$$\vec{d} = [d_1 \quad d_2 \quad d_3] = [0.5 \quad 0.3 \quad 0.2], \tag{2-14}$$

this means that phases a, b, and c are connected to the top DC rail for 0.5 of switching period, then phases a and b are connected to the top and phase c to the bottom of DC rail for 0.3 of the switching period, and finally, phase a is connected to the top rail and phases b and c are connected to the bottom rail for 0.2 of the switching period.

The final step is converting the vector information into continuous-time switching functions that control the states of the power switches. These pulses must be generated in such a way as to take into account shoot-through protection, minimum turn-on time and other power stage limitations. The vector of switching functions can be defined as

$$\vec{S}=\begin{bmatrix}S_{ap} & S_{an} & S_{bp} & S_{bn} & S_{cp} & S_{cn}\end{bmatrix}, \tag{2-15}$$

where switching functions are defined as:

$$S_{xy}=\begin{cases}1, & \textit{if phase x is connected to top rail}\\ 0, & \textit{if phase x is connected to bottom rail}\end{cases} \quad x=\{a,b,c\}, \qquad y=\{p,n\} \tag{2-16}$$

Switching functions S_{xy} are continuous-time functions. A typical waveform of S_{xy} is shown in Fig. 2-8. While all other digital controller variables are discrete-time variables (exchanged once in a sampling time by means of a digital channel), switching functions are continuous-time functions requiring an analog communication channel.

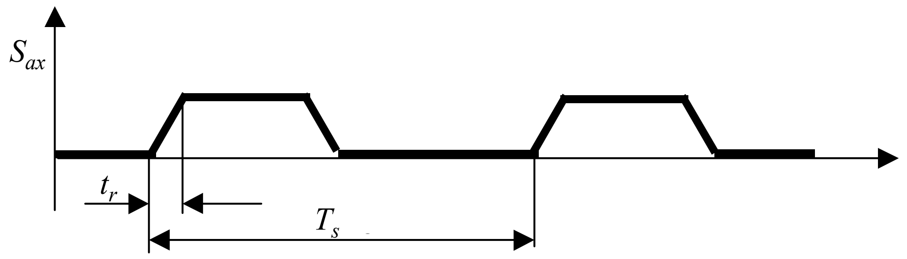


Fig. 2-8. Typical switch gate-drive waveform.

Therefore, the channel for communicating these switching functions is different in nature. The envelope of Fourier series spectrum of this type of gate drive signal is given in [23] and is shown in FIG. 2-9. From this graph a required communication channel bandwidth to transmit one gate drive signal without significant distortion can be estimated as:

$$BW = \frac{1}{\pi \cdot t_r}, \tag{2-17}$$

since after $\frac{1}{\pi \cdot t_r}$ the frequency spectrum envelope starts to decrease with 40 dB/dec and has no significant impact on the transmitted signal.

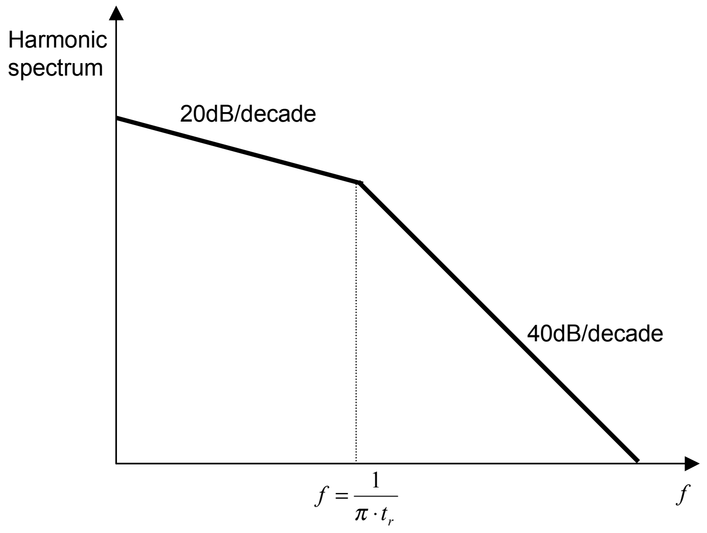


Fig. 2-9. Envelope of Fourier spectrum for a 50% duty cycle trapezoidal wave [23].

Since each switch needs one signal path for the control and one return path for sensing the switch fault state (desaturation protection), each switch requires at least $2 \cdot \frac{1}{\pi \cdot t_r}$ bandwidth where t_r is defined as the minimum required signal rise time.

To make this analysis conclusive and systematic we have proposed five possible interfaces along which the control algorithm can be distributed between the hardware manager and the applications manager. These five interfaces (designated **I_1**, **I_2**, **I_3** to **I_5**)

that are shown in Fig. 2-10 are the candidates for the physical interface between the hardware and applications managers.

Through each interface the data are communicated in both directions. However, the amount of the data the data format and the nature of the signal (analog, digital, etc.) that is being communicated through each interface and the update rate differ significantly, thus influencing the infrastructure of the communication hardware. Therefore, later sections will attempt to quantify data traffic and communication requirements of each.

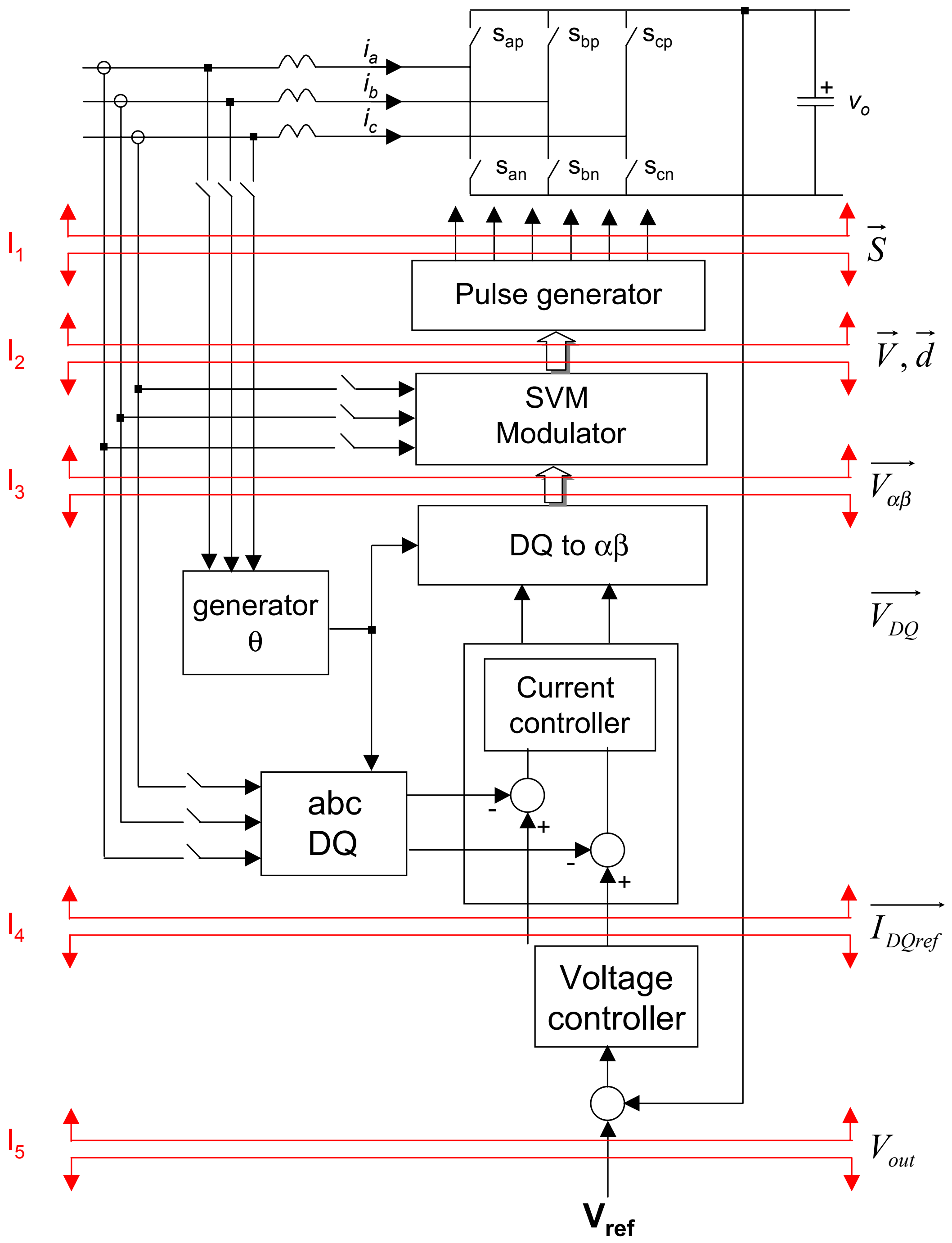


Fig. 2-10. Control structure for three-phase boost rectifier.

2.2.2 Analysis of the Three-Phase Two-Level Voltage Source Inverter

The three-phase VSI inverter is a very common topology used in high-performance motor drives, pumps, fans, air conditioning, UPS applications and many others. Although they vary in power, application and distributions, there are some pronounced commonalities in control architecture that are invariant and that can be found in the boost-rectifier circuit converter explained in Section 2.2.1.

The basic control architecture of a three-phase voltage source converter is given in Fig. 2-11. The power stage topology is basically the same as for the three-phase boost rectifier. Also, the control architecture is very similar to one explained in Section 2.2.1. The main objects in the control architecture are the pulse generator, the space vector modulator (SVM), the current and the voltage loop controller and coordinate system transformations.

The current loops are closed in the DQ coordinates. The output of the current controller is a voltage vector in DQ coordinates, the same as shown in equation (2-7). Vector $\overrightarrow{V_{DQ}}$ is transformed into αβ coordinates using inverse DQ transformation:

$$\overrightarrow{V_{\alpha\beta}} = T^{-1} \cdot \overrightarrow{V_{DQ}} \, . \tag{2-18}$$

Vector $\overrightarrow{V_{\alpha\beta}}$ is the input vector for the space vector modulator. The output of the space vector modulator is similar to the boost rectifier vector of the switching vectors, as defined in (2-8) and the vector of corresponding duty cycles, as in (2-10).

The final stage is the transformation of the vector of the switching vector states and the vector of corresponding duty cycles into a vector of switching functions, as defined in (2-15) and (2-16).

Although this application is significantly different from the three-phase boost rectifier, interfaces designated as I_1, I_2, I_3 to I_5 handle the same types of data format with slight variations. All this points towards the possibility to standardize communicated data formats and to modularize the control architecture in an object-oriented fashion.

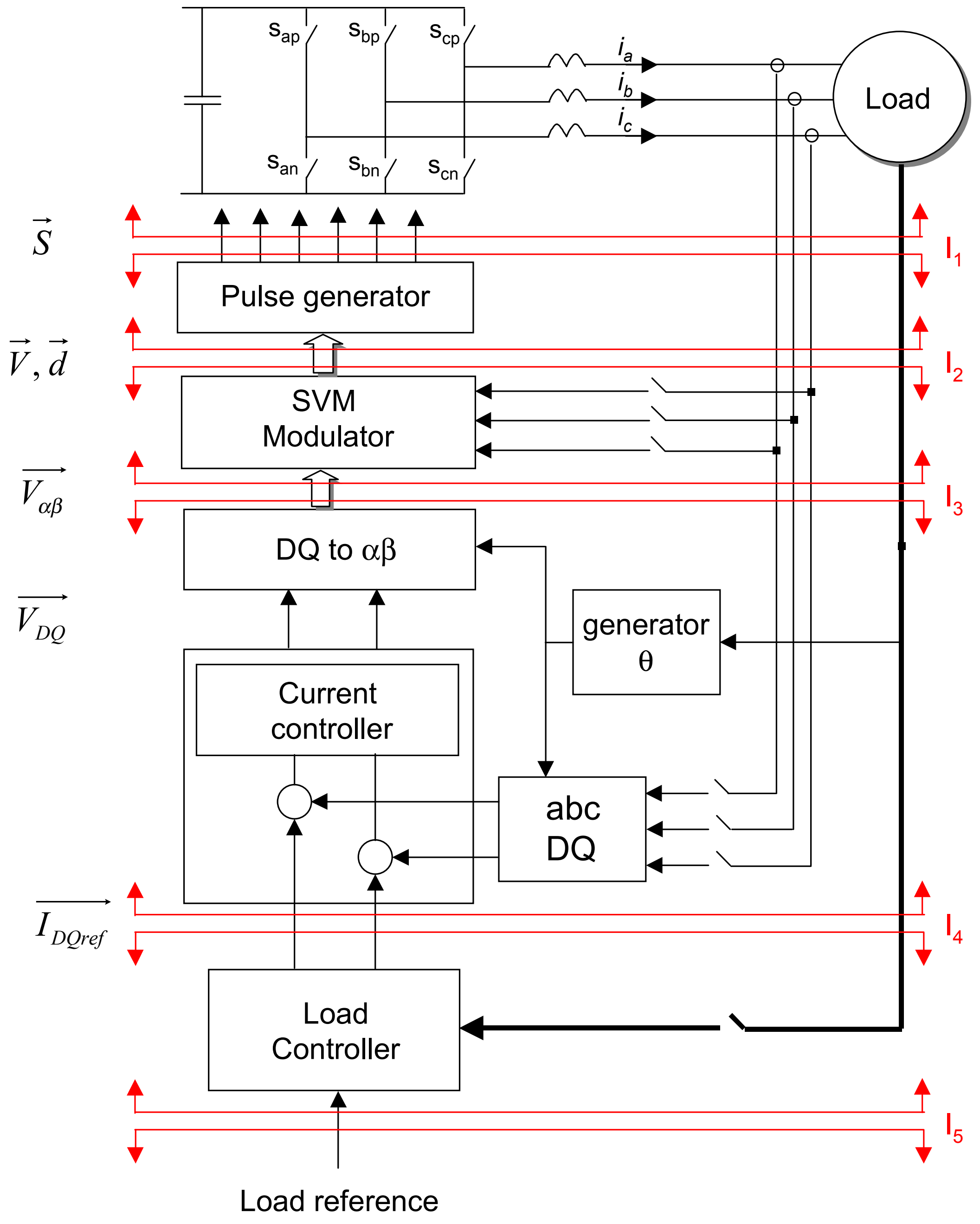


Fig. 2-11. Control architecture for three-phase VSI.

2.2.3 Analysis of Three-Phase Three-Level Voltage Source Inverter

The three-level VSI topology is being widely accepted for medium- and high-power applications. Although the control architecture of this topology is more complex than those of two-level topologies, higher-level control loops do not differ significantly from the examples examined in previous sections.

The block diagram of the control architecture for the three-VSI is given in Fig. 2-12. The basic control strategy is very similar to that of the two-level VSI. The current loops are closed in the DQ reference frame. References for the current loop control are derived from the load controller. The load controller is typically dependent on the type of application for which the inverter is used. The output of the current controller is the voltage vector $\overrightarrow{V_{DQ}}$ in the DQ reference frame, as defined in (2-6). Using the inverse DQ transformation given in (2-7), voltage vector $\overrightarrow{V_{\alpha\beta}}$ is obtained.

The input to the space vector modulator is similarly to the previous example's vector $\overrightarrow{V_{\alpha\beta}}$. The output from the space vector modulator is again the pair of vectors:

$$\vec{V} = \left[\overrightarrow{V_1} \quad \overrightarrow{V_2} \quad \overrightarrow{V_3} \quad \ldots \quad \overrightarrow{V_n}\right] \text{ and} \tag{2-19}$$

$$\vec{d} = \left[d_1 \quad d_2 \quad d_3 \ldots \quad d_n\right], \tag{2-20}$$

where $\overrightarrow{V_i}$ is the vector of the switching state vectors, defined with

$$\overrightarrow{V_i} = \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix}, \qquad \text{where } S_{abc}=\{0,1,2\}, \tag{2-21}$$

and $\vec{d}$ is the vector of corresponding duty cycles. In case of three-level converter S_{abc} can take three values {0,1,2}, as compared with two-level converters where S_{abc} can only take values from the set $\{0,1\}$. And finally, the pulse generator converts a pair of vectors $\vec{V}, \vec{d}$ into switching functions for all 12 switches. The vector of the switching functions is defined as:

$$\vec{S} = [S_{a1} \quad S_{a2} \quad S_{a3} \quad S_{a4} \quad S_{b1} \quad S_{b2} \quad S_{b3} \quad S_{b4} \quad S_{c1} \quad S_{c2} \quad S_{c3} \quad S_{c4}] \text{ and} \qquad (2\text{-}22)$$

$$S_{xy} = \begin{cases} 1, & \textit{if switch xy is turned on} \\ 0, & \textit{if switch xy is turned off} \end{cases} \qquad x=\{a,b,c\}, \qquad y=\{1,2,3,4\} \qquad (2\text{-}23)$$

While the converter topology of the three-level three-phase VSI is significantly more complex then those explained in previous sections the control architecture and interface definitions remain consistent.

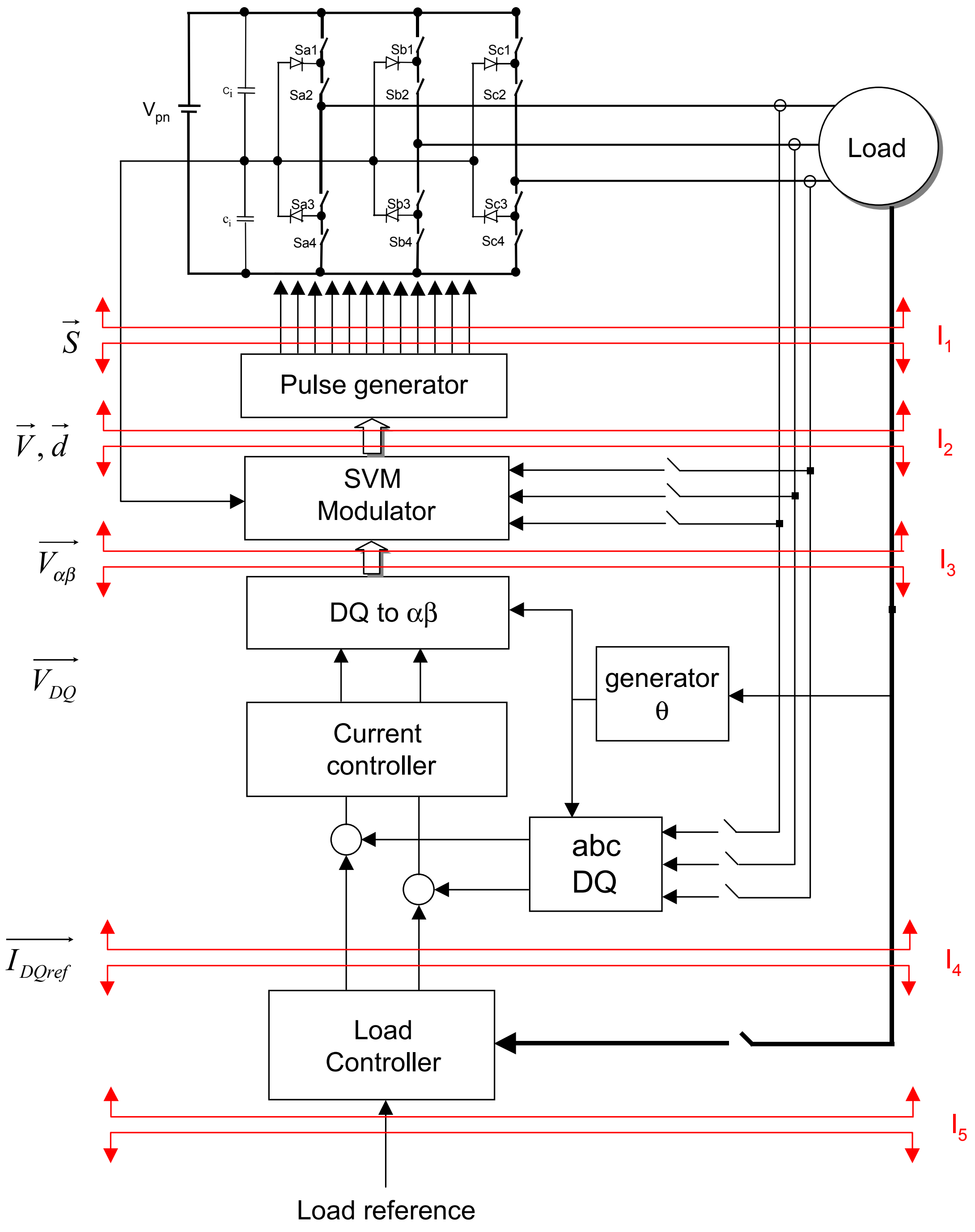


Fig. 2-12. Control architecture for three-phase three-level voltage source inverter (VSI).

2.3 Communication requirements for different interfaces

The previous analysis of control architecture for three-phase boost rectifier, three-phase VSI and three-phase three-level VSI has shown significant similarities between the three. Similar structure, large number of common control objects and similar data interfaces are pointing out that unified and object oriented approach to control distribution can be viable solution. The goal of this analysis is to find the "optimum" interface to split the control algorithm in two parts (hardware and applications manager) regardless of the topology and application.

I_1 interface communicates analog gate signals for each switch (from controller to power stage), analog fault signals for each switch (from power stage to controller) and communicates all the measured (sampled-data) currents and voltages (from power stage to controller).

Each switching function requires at least the bandwidth BW given with:

$$BW = \frac{1}{\pi \cdot t_r} \tag{2-24}$$

as explained in details in Section 2.2.1. Since each switch needs one input (control signal) and one output (fault indication) required bandwidth per switch is $2*BW$. Duty cycle (communicated through interface I_2), is represented with n_d bits yielding maximum resolution of PWM signal:

$$\rho = \frac{T_{sw}}{2^{n_d}} \tag{2-25}$$

Since duty cycle information is converted in switching functions in pulse generator, we can assume that maximum rise time will be:

$$t_r \leq \rho \tag{2-26}$$

Substituting (2-25) into (2-26) and then into (2-24), and multiplying by two (for input and output signals) gives the required bandwidth per switch as:

$$BW = 2 \cdot \frac{f_{sw} \cdot 2^{n_d}}{\pi \cdot t_r}. \tag{2-27}$$

For every sampled-data digital variable required the channel bandwidth *CB* can be expressed as:

$$CB = K_d \cdot f_{sw} \cdot n_X\ , \tag{2-28}$$

where K_d is defined as the ratio between the sampling period and transmission time, f_{sw} is defined as the sampling frequency (update rate), which in this case is assumed to be the same as switching frequency, and n_X is the number of bits used to describe the measured variable.

I$_2$ interface is a purely discrete-time digital interface. From the controller to the power stage vectors,

$\vec{V} = [\vec{V}_1 \quad \vec{V}_2 \quad \vec{V}_3]$ and

$\vec{d} = [d_1 \quad d_2 \quad d_3]$

are communicated as explained in (2-8) and (2-10). Equation (2-28) gives the necessary channel bandwidth per single variable. For the voltage vector $\vec{V}_{3x3}$ with nine variables in its matrix, and each variable represented by one bit--either zero or one as explained in (2-9)—the required bandwidth is calculated as:

$$CB_{V_{3x3}}=9\cdot K_d\cdot f_{sw}\cdot n_X. \tag{2-29}$$

A similar procedure gives the necessary bandwidth for the duty cycle and current and voltages' communications, as given in TABLE 2-2.

I_3, I_4 and I_5 interfaces are also purely sampled-data digital interfaces. All the variables are communicated in a digital form in both directions (to and from the power stage). Equation (2-28) gives the necessary channel bandwidth for all of them. TABLE 2-2 summarizes the types of data that are being communicated through each of the interfaces (I_1-I_5), and gives the expression for the channel bandwidth necessary to satisfy the requirements.

Therefore the channel bandwidth through each sampled data interface can be expressed as:

$$CB=N\cdot K_d\cdot f_{sw}\cdot n_X \tag{2-30}$$

where N is the number of variables communicated through the interface and K_d, f_{sw} and n_x are defined as in (2-28). In order to avoid notational clutter, TABLE 2-2 will only show the variables communicated, number of variables and number of bits used to represent each variable assuming that (2-30) is applicable unless otherwise specified.

TABLE 2-2. Interface communication requirements.

	Two-level VSI (variable, N, n_X)	**Two-Level Boost Rectifier** (variable, N, n_X)	**Three-Level VSI** (variable, N, n_X)
I_1 (in)	$\vec{S}_{(1\times6)}$, $(\frac{f_{sw}\cdot 2^{n_d}}{\pi}\cdot 12)$	$\vec{S}_{(1\times6)}$, $(\frac{f_{sw}\cdot 2^{n_d}}{\pi}\cdot 12)$	$\vec{S}_{(1\times12)}$, $(\frac{f_{sw}\cdot 2^{n_d}}{\pi}\cdot 24)$
I_1(out)	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(3\times1)}$, 3, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I
I_2 (in)	$\vec{V}_{(3\times3)}$,9 ,1 $\vec{d}_{(1\times3)}$, 3, n_d	$\vec{V}_{(3\times3)}$,9 ,1 $\vec{d}_{(1\times3)}$, 3, n_d	$\vec{V}_{(3\times3)}$,9 ,2 $\vec{d}_{(1\times3)}$, 3, n_d
I_2 (out)	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(3\times1)}$, 3, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I
I_3 (in)	$\overrightarrow{V_{\alpha\beta}}_{(2\times1)}$, 2, n_V	$\overrightarrow{V_{\alpha\beta}}_{(2\times1)}$, 2, n_V	$\overrightarrow{V_{\alpha\beta}}_{(2\times1)}$, 2, n_V
I_3 (out)	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(3\times1)}$, 3, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V $\overrightarrow{I_{measured}}_{(2\times1)}$, 2, n_I
I_4(in)	$\overrightarrow{I_{DQREF}}_{(2\times1)}$, 2, n_I	$\overrightarrow{I_{DQREF}}_{(2\times1)}$,2, n_I	$\overrightarrow{I_{DQREF}}_{(2\times1)}$, 2, n_I
I_4(out)	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V	$\overrightarrow{V_{measured}}_{(1\times1)}$, 1, n_V	$\overrightarrow{V_{measured}}_{(2\times1)}$, 2, n_V
I_5 (in)	$V_{outref,}$ 1, n_V, K_{d2}	$V_{outref,}$ 1, n_V, K_{d2}	$V_{outref,}$ 1, n_V, K_{d2}
I_5 (out)			

Key:

f_{sw}	is the switching frequency of the converter
n_V	is the number of bits to represent the voltage variable
n_I	is the number of bits to represent the voltage variable
n_d	is the number of bits to represent the switch duty cycle
K_d	is the ratio of sampling period of the converter and transmission time through the interface defined as in (2-28)

Fig. 2-13 shows the minimum necessary channel transmission capacity through different interfaces. The abscissa is discrete axis denoting interfaces (I_1-I_5) while the ordinate shows the necessary interface capacity for different types of converters. For this plot, the following parameters were used: f_{sw}=20 kHz; n_V = 12 bits; n_I = 12 bits; n_d = 10 bits, $K_d = 2$ and $K_{d2} = \frac{1}{5}$.

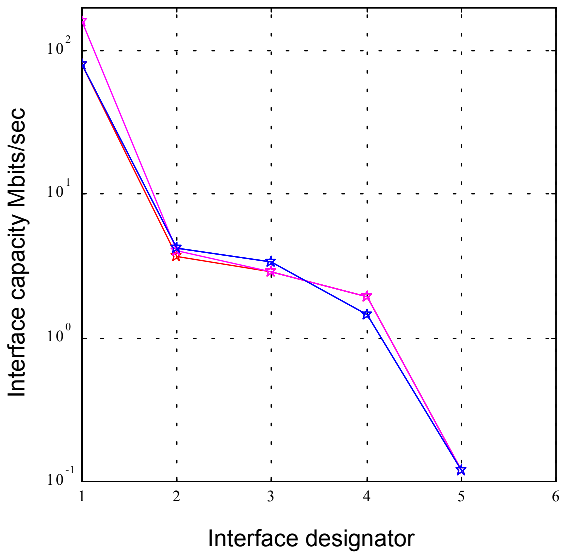


Fig. 2-13. Interface channel capacity requirements for different converters.

Fig. 2-13 reveals several facts important to the design of the distributed controller. If we split the controller along the interface I_1 the necessary communication channel capacity exceeds of 100 Mbits/sec, when 20 kHz switching frequency is chosen. Splitting the control authority along this interface would essentially render this controller the same as the centralized controller (which was not the initial goal), with the necessity for a large communication channel bandwidth. At the same time this approach would not yield increased flexibility, nor would it simplify the system integration tasks.

The further the interface is from the power stage, the less channel bandwidth is needed for communication. This was expected, considering that more loops can be closed locally on the hardware manager side, thus reducing the need to exchange excessive information through the other interfaces. The fact that the function of interface bandwidth becomes almost flat for I_2, I_3 and I_4 makes those three interfaces equally suitable candidates to be the physical interface between the hardware and the applications managers.

After the almost flat part of the curve, interface I_5 exhibits a significantly smaller channel bandwidth, indicating a good point for system-level communication interface.

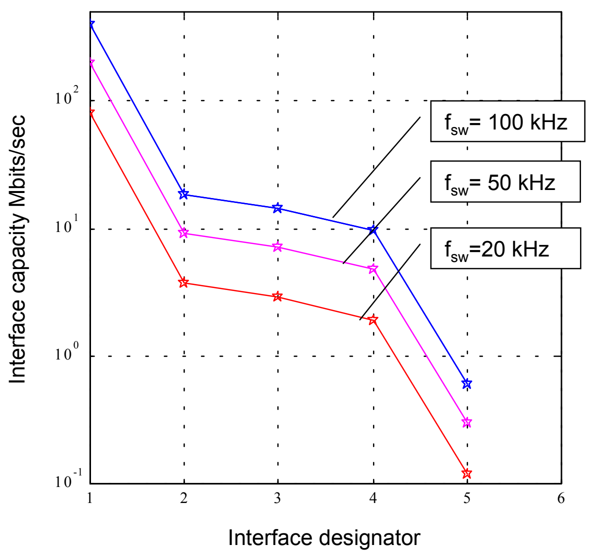


Fig. 2-14. Interface channel capacity requirements for the three-phase VSI for different converter switching frequencies.

As shown in TABLE 2-2 interface bandwidth is also a function of the converters switching frequency. Fig. 2-14 shows the interface bandwidth for three-phase VSI as a function of its switching frequency.

Another factor that must be considered when deciding where to split the control algorithm is the hardware distribution. So far the discussion has involved splitting controller along the vertical (time) axis, but has not touched yet on the distribution along the horizontal (hardware distribution) axis. The hardware related part that is to become an integral part of the power stage can also be distributed along the distributed hardware modules.

Consider the example of the three-phase VSI in which the power stage consists of three phase legs, as shown in Fig. 2-15. For this type of hardware distribution, the control algorithm can be distributed along only two interfaces, I_1 or $I_{2,}$ as shown in Fig. 2-15. This is because we made the assumption that the optimum distribution would involve two parts (hardware manager and applications manager), and in which case the hardware manager would become an integral part of the power stage module. Indeed, since the power stage consists of three separate modules only the pulse generator can be distributed on three parts while the space vector modulator, or DQ current controller can not be distributed among the phase legs. The space vector modulator treats the whole three-phase converter as a single switching network, and therefore, can not be decoupled on three phases. If sinusoidal PWM modulation is used instead, then the modulator or even per-phase current loops could be distributed among the modules.

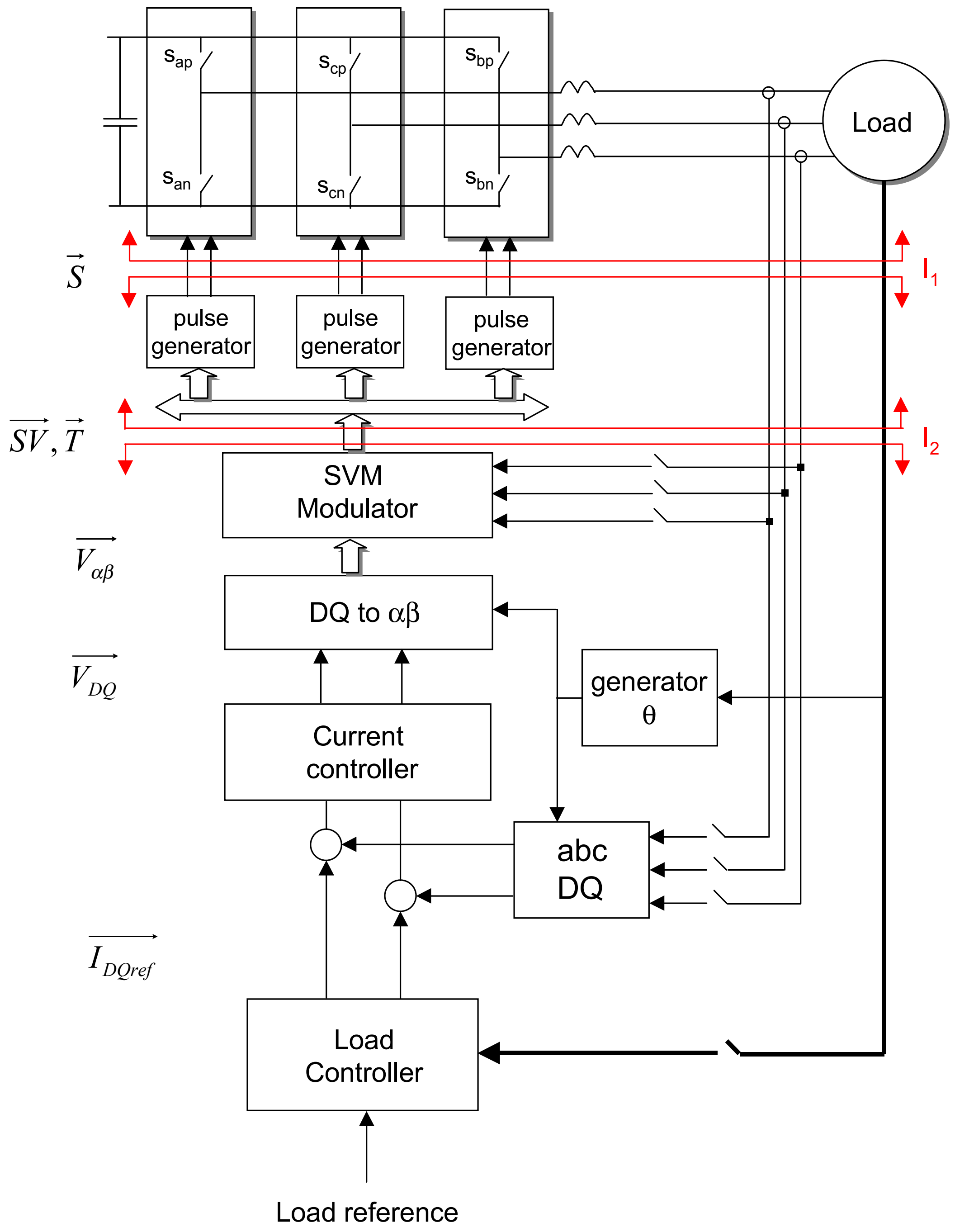


Fig. 2-15. Block diagram of the control structure for the three-phase VSI in which power stage is distributed among three modules.

On the other side, when the whole converter is lumped into one module, the distribution of the control between the hardware and applications manager can be accomplished along any of the interfaces, since that case does not require control algorithm distribution along the hardware only the temporal distribution. This is the case for which the total switching power of the module is less then $10*10^5$ [W] according to Fig. 2-2.

Another very interesting case for the analysis of distributed systems in power electronics is the paralleling of converters [28]. For paralleled converters, the hardware distribution is even more pronounced since it involves separate converters. Furthermore, each converter can be distributed along multiple modules depending on the power level and hardware realization. Additionally, the control algorithm is also distributed, as shown in Fig. 2-16, since every converter has its own local controller, while the outer loops and synchronization signals are shared between multiple converters.

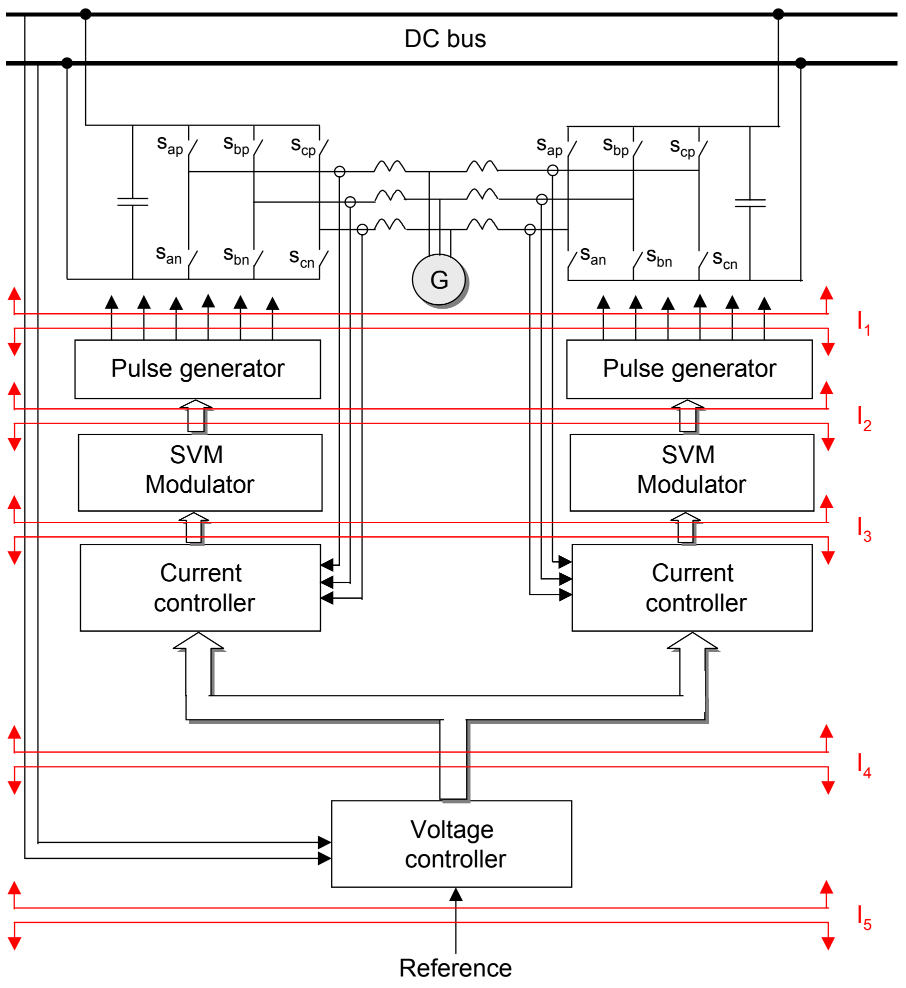


Fig. 2-16. Block diagram of the control architecture for two boost rectifiers that share a common load and source [28].

Paralleling of converters is a very nice example of strong hardware and control distribution. Unfortunately, this distribution is not supported by distributed control architecture. Indeed, in most cases, control of paralleled converters is performed by means of a centralized controller.

Therefore, to be able to apply the proposed distributed controller to a wide range of power converters and systems, we have adopted a communication network that can support communications through I_2, I_3 and I_4 interfaces, thus physically splitting the controller between the hardware and the applications managers. According to Fig. 2-14 125 Mbits/sec channel capacity is adopted, which is capable of providing enough bandwidth for I_2, I_3 and I_4 interfaces when the switching frequency is less then 100 kHz. How exactly the control algorithm is going to be distributed between the hardware and the applications manager depends on the particular application and implementation. In our case we have adopted to split the control authority along the I_2 interface. The reason for this is also because we adopted per-phase module as a unit of integration, which limited our options.

Chapter 3 Control Communication Protocols

Real-time distributed controls have been extensively used in factory automation and motion control systems for the last twenty years. Increased modularity, fault tolerance, expandability and significantly improved overall system flexibility have constantly fueled the drive for integration of factory elements into the computer integrated manufacturing environment [16]. The idea is to employ hierarchical control over the whole manufacturing process, and integrate all factory elements into a coherent control architecture, as depicted in Fig. 3-1.

To provide a consistent control and data communication environment for the distributed controller different manufacturers and organizations have proposed and implemented different communication protocols [16]. All the protocols have been specifically tailored to meet targeted system requirements.

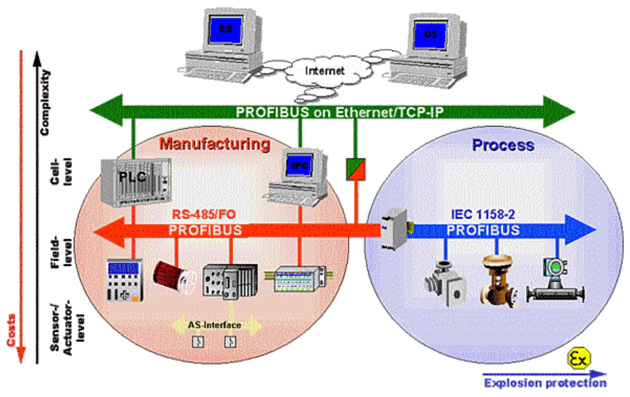


Fig. 3-1. Industrial communications for the example of the PROFIBUS standard [31].

The main drive behind this thesis is very similar to the previously explained idea but applied on a different level of control. While industrial communication protocols provide distributed control and communications on a field level (connecting drives, PLCs, controllers, sensors, actuators, etc.) this thesis proposes distributed control on a converter level. This would bring modular design, significant flexibility, hierarchical control and many other benefits on a converter control level, compared with industrial communications, in which hierarchical control starts above the converter and drive level. In factory automation, drives and converters are treated as a building block rather than as whole new distributed system.

This means that a distributed control has to operate on a much faster time scale, with a few orders of magnitude smaller time constants. Also, synchronization becomes much more of a pronounced issue due to the smaller time constants. Noise susceptibility and noise immunity of the distributed controller arises as another important issue. Since the communication link is getting physically close to the electro-magnetic interference EMI sources (power converters), noise immunity of a control and communication system starts to play a key role in the system reliability and robustness.

Therefore, this chapter will briefly review major industrial communication protocols and their most prominent features and compare them with the communication requirements for the distributed control of power converters that were established in chapter Chapter 2.

The two most widely used network topologies in industrial communications are point-to-point serial interface links and the multipoint connection offered by the local area network (LAN).

At the present time, many motor drives are connected to their steering units using point-to-point serial interface links (e.g. RS 232 or RS 422). This approach suffers from two major drawbacks. First of all, one control unit must drive several drives and handle communication with them simultaneously, which can seriously overload the main controller and limit the response time. This approach also limits the expandability of the control system. Yet, low cost makes this approach feasible for many applications.

LAN offer multipoint connections and are much more flexible and application-independent. This is the reason why most of today's control protocols are based on LANs. The basic requirement for LANs is that they have to be open and to support devices made by different manufacturers. This provides the system with high expandability and significant flexibility.

The LAN communication protocols for real-time control applications can be roughly divided into field busses and specific networks. The former is the general-purpose network devised for field-level control that can connect drives, PLCs, PCs, I/O modules, sensors, and actuators. They are designed to provide wider flexibility and to allow for interconnection of different units at the cost of speed and response time. Specific networks are less flexible and are designed to provide link between PLCs and controllers on one side, and sensors, drives and actuators on the other. Those specific networks are designed to provide faster response time and better synchronization at the cost of reduced flexibility.

Several widely used control communication protocols that are relevant to our study are briefly reviewed as follows. Some other relevant protocols are reviewed in [11], while a very comprehensive overview can be found in [16].

3.1 Field Busses

3.1.1 PROFIBUS

PROcess FIeld BUS is the German proposal for the open field bus standard for a wide range of applications in manufacturing and process automation [31,16]. PROFIBUS is designed for both high-speed time critical applications and complex communication tasks. The protocol architecture is oriented to the reduced ISO OSI (open system interconnection) reference model in which each layer handles a precisely defined tasks [31].

Layer 1 (the physical layer) defines the physical transmission characteristics. There are currently three transmission methods (physical profiles) available for PROFIBUS:

- RS 485 transmission for universal applications in manufacturing automation
- IEC 1158-2 transmission for use in process automation; and
- optical fibers for improved interference immunity and large network distances

In the course of further technical developments, it is intended to use commercial Ethernet components with 10 Mb/s and 100 Mb/s as the physical layer for PROFIBUS.

Layer 2 (the data-link layer) defines the bus access protocol. PROFIBUS is designed to support two types of devices, namely master-devices and slave-devices as shown in Fig. 3-2. Master devices determine the data communication on the bus. Master can send messages without an external request whenever it holds the bus access rights (tokens). On the other side slave devices are peripherals such as I/O devices, valves, drives and measuring transducers. They can only acknowledge received data or send messages when asked to do so by the master. The medium access control protocol includes the token passing procedure among masters. Each master (upon receiving token) takes control over the bus and starts communicating with slaves and other masters.

After the token hold time expires the token is passed to the following master node which then assumes the control over the bus. This allows for the following system configurations:

- pure master-slave system,
- pure master-master system (token passing), and
- combination of the two.

The actual token hold time of a master depends on the configured token rotation time.

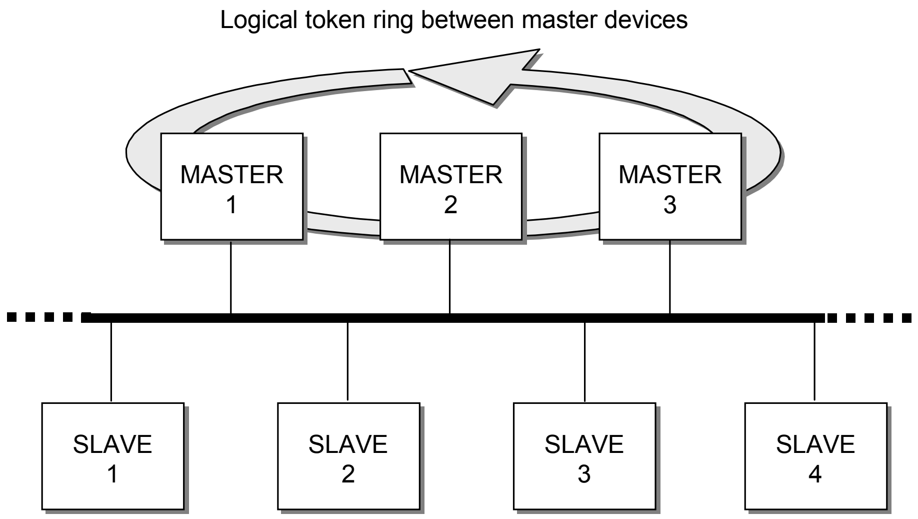


Fig. 3-2. PROFIBUS medium access control.

Layer 7 (the application layer) defines the application functions. It consists of two protocols, namely the DP protocol and FMS (field message service) protocol. DP is designed for efficient data exchange at the field level. This protocol is mostly used for cyclic data exchange between the automation system and distributed peripherals. FMS is the universal communication profile for demanding communication tasks. It allows object manipulation, with objects being variables, arrays, matrices, variable lists, program calls, subroutines and so on. It is divided into two sublayers, i.e.. FMS (fieldbus message specification) and LLI (lower layer interface). The architecture of the PROFIBUS is shown in Fig. 3-2

3.1.2 WorldFIP

Factory automation protocol is a French-Italian proposal for the field bus [30]. It is designed to provide links between level zero (sensors/actuators) and level one (PLCs, controllers, etc.) in automation systems [30].

WorldFIP is also designed according to the reduced ISO/OSI reference model, meaning that all protocol functions are implemented within the physical data-link and application layer. Although WorldFIP protocol supports both copper wire and optical fiber as a transmission meda, twisted pair copper cable is the one used most often.

The data link layer provides a medium access protocol. There is always one station, named the bus arbitrator (distributor; see Fig. 3-3), that handles the communication and bus access and a number of other stations which respond to the arbiter polling. When the system is configured, the bus arbitrator is given the list of variables to scan and the periodicities associated with each.

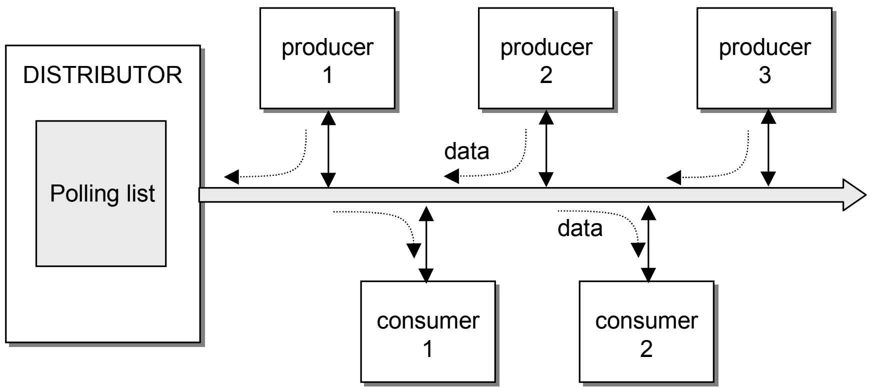


Fig. 3-3. Block diagram of WorldFIP medium access mechanism.

Each station on the bus has several produced and consumed buffers. When the arbitrator inserts the question command on the bus and states the buffer address the station that is producer of the buffer will respond by putting the buffer value on the bus. The stations that are consumers of that variable will respond with accepting the value from the bus and storing it. Data traffic consists of time-critical data (control variables) that are exchanged periodically and non-critical data such as messages. Cyclic variables always travel on time, regardless of other network traffic. Less time-dependent data (such as diagnostic reports), are sent via the message service. This is typically used for installing and setting applications, network supervision and diagnosis and integration with higher-level systems. Unlike token-passing systems, WorldFIP leaves no doubt about transmission time and regularity for cyclic data variables. WordFIP applications layer is divided into two distinct groups:

- MPS (manufacturing periodical/aperiodical services) and
- subMMS (subset of messaging services).

The MPS application layer provides the user with local read/write services, remote read/write services, variable transmission/reception indications and information on the spatial and temporal consistency of the data. All those services are used for real-time data distribution throughout the whole network. For example, the local read service provides the application layer with the variable from the consumed buffer that exists in the data link layer. The local read and write service uses the data-link layer services L_PUT.req and L_GET.req to place values in buffers or remove values from buffers. Those services generate no traffic on the bus since those produced and consumed buffers reside in the data-link layer. MMS services are used for non-critical traffic such as system configuration, monitoring, etc.

3.1.3 CAN

The controller area network (CAN) is a field bus proposed by Bosch for automotive applications [16]. It is a serial communications protocol, which efficiently supports distributed real-time control with a very high level of security.

CAN follows the ISO/OSI reduced model, with bus topology, client-server model and an original medium access method. In all protocol messages there are no origin/destination addresses but only an identifier. Every station first reads the identifier and then decides whether or not to read the rest. The priority field of the frame indicates the type of message transmitted and its priority. When a station wants to transmit the message it has to compare its relative priority to the network message priority. If it is less or equally important it has to wait until the bus is clear. This medium access control is a multimaster protocol with distributed priority arbitration.

The advantage of CAN protocol is that it was developed for automotive applications thus being optimized from the cost point of view. Also, market availability of the supporting components is very good, which makes the whole standard widely accepted. The fact that this protocol was designed for the inexpensive copper wire transmission medium makes it less suitable for applications in which EMI noise is a significant issue.

3.2 Specific Control Networks

3.2.1 MACRO

MACRO is a communication standard for distributed machine control. MACRO stands for a motion and control ring optical and was designed for multiple-axis precision motion control [32]. MACRO uses a ring topology to allow master controllers to communicate with both slave nodes and other master controllers.

A MACRO network is organized as ring architecture with multiple masters and slaves connected. Communication throughout the ring is started by a pre-designated ring-master.

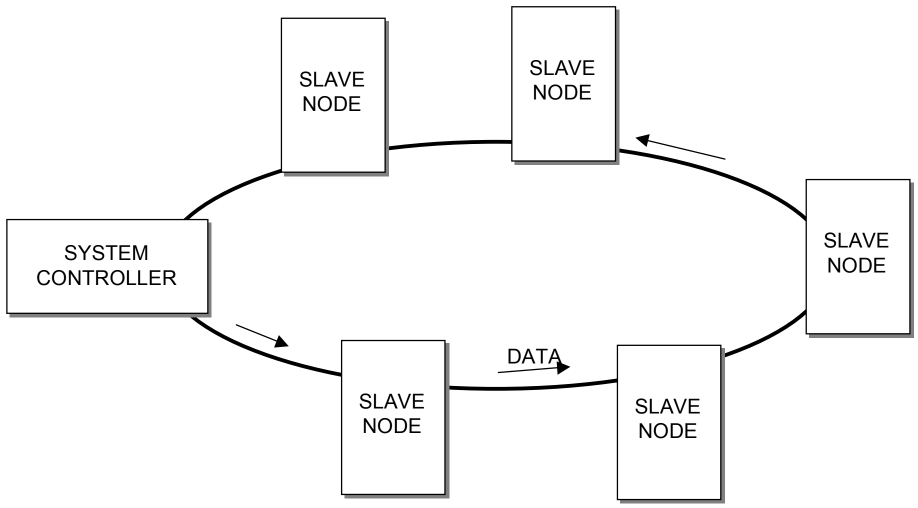


Fig. 3-4. The MACRO ring network consists of a group of master and slave stations.

Communication is always originated by a master that sends a data packet (with appropriate address) down the stream. The node that receives the data packets checks the address and if it is the same as the local address, it takes the data from the packet and loads the data packet with local data and sends the packet down the ring. If the local address is different the node just passes the data to the next one in the ring. A master's ring hardware shifts all active node data from the master's transmission registers across the ring to matching input registers of the appropriate slave node. The slave's ring hardware shifts the node data that is addressed to it into a set of receiving registers. Data is returned to the master at the same time the data is received from the ring by the slave's hardware transmission registers.

3.3 Power Electronics Network Solution

In Section 2.3 we have discussed communication requirements for different interfaces with respect to required channel bandwidth, and interfaces I_2, I_3 and I_4 were identified as optimum to distribute the controller between the hardware manager and the applications manager. Fig. 2-14 shows that for I_2 interface 100 Mbits/sec channel capacity can meet the requirements for the three-phase VSI switching at 50 kHz. None of the previously studied protocols (Sections 3.1 and 3.2) can meet those communication speed requirements. Furthermore, distribution along interface I_1 requires a network synchronization jitter less then 100 ns, which also can not be achieved with any of the commercially available control communication protocols. And finally, distributed controller approach for power converters requires noise immunity, which can only be achieved using a fiber optic link.

Therefore, in order to provide a link between the hardware and the applications manager in a distributed control network that will accommodate the need for channel bandwidth, network synchronization and noise immunity—a communication protocol called PESNet [11] —was adopted.

Fig. 3-5 shows the block diagram of a distributed controller for the three-phase VSI that is going to be used throughout the thesis as an illustrative example of the concept. In this example hardware distribution assumes that each phase-leg is an independent module. Each phase-leg has an associated hardware manager. Since the hardware manager and applications manager share the I_2 interface the data being communicated through the interface are given in TABLE 2-2: $\vec{V}_{(3\times3)}$, $\vec{d}_{(1\times3)}$, $\overrightarrow{V_{measured}}_{(2\times1)}$ and $\overrightarrow{I_{measured}}_{(2\times1)}$. Those four vectors are real-time control data. Communication protocol must provide the means for exchanging those data in a timely manner and with tight synchronization.

Besides, the real-time data that are exchanged on a switching cycle level (cyclic variables), in a distributed controller there is a need for communication of non-real time data. Non-realtime data can allow for initialization, monitoring and message exchange throughout the network. This can improve the openness of the whole system. Many of the studied control protocols have this feature, which makes them more flexible. Thus, we are proposing the non-real time communication addition to the PESNet protocol [11].

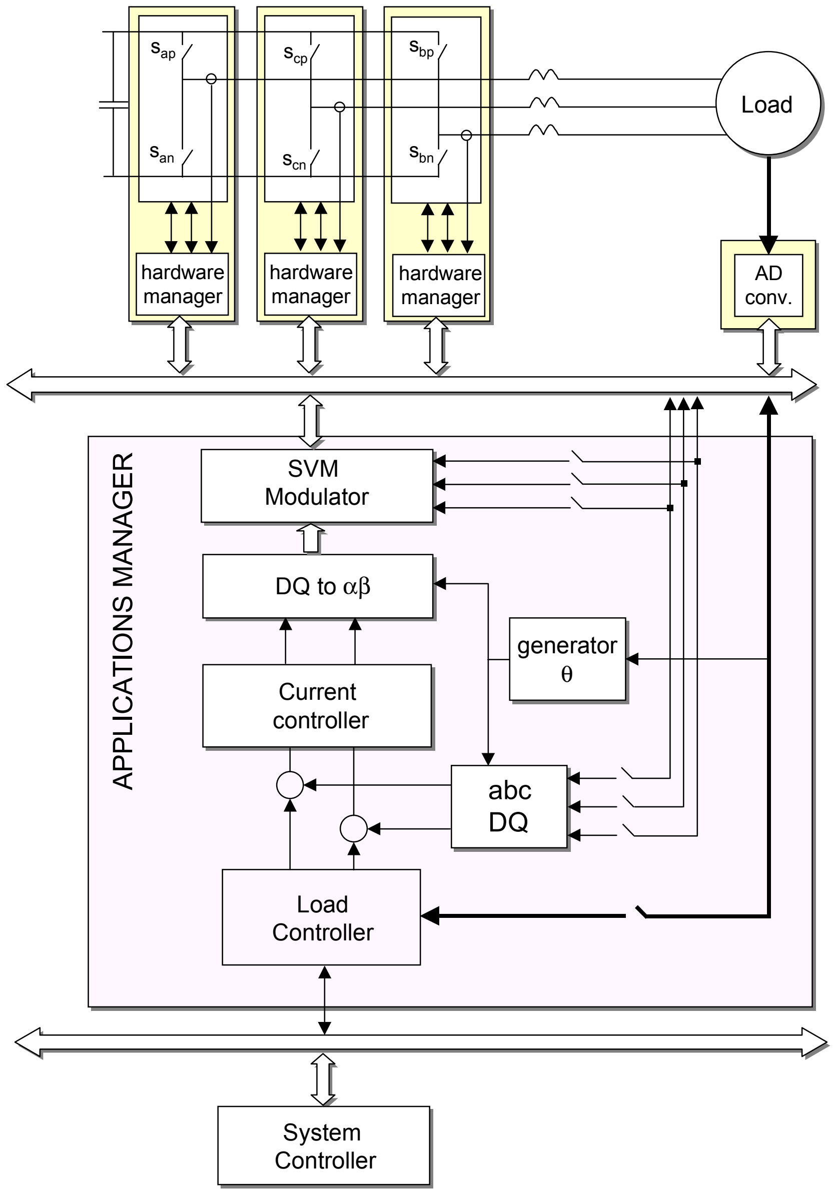


Fig. 3-5. Block diagram of proposed distributed controller for three-phase VSI.

3.3.1 Communication Protocol Functional Description

The master-slave protocol insures deterministic response of the network. If an error occurs during transmission, corrupt data is not used. Instead, the new data simply overwrite the previous data. This way the data flow is kept strictly predetermined. Fig. 3-6 shows two basic types of information communicated through the control network: time-critical data (exchanged in every switching cycle) and time non-critical data (transmitted only after all the critical time variables have been passed to all nodes). Time-critical information includes all the control variables such as: switching frequency information, duty cycle information and all the sensor information. The provision for non-critical data transfer is designed to support tasks such as initialization and software reconfiguration of the hardware managers. Non-critical data transfer is allowed only after all the time critical data is exchanged. Fig. 3-7 shows three types of time-critical data frames: the control data frame, the synchronization frame and the command frame.

The data frame consists of a command indicating the beginning of the data packet, the address of the node, the data field and an error check. The data field's configuration depends on the particular application and type of hardware manager.

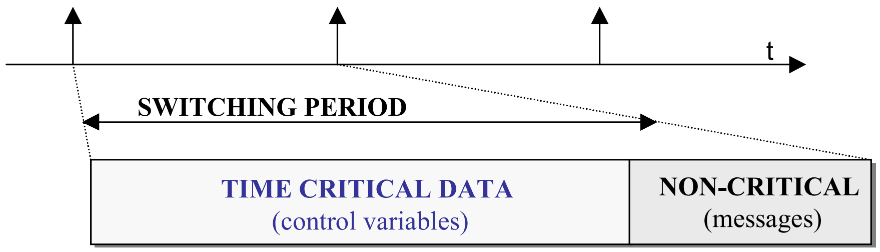


Fig. 3-6. Time allocation on the communications bus.

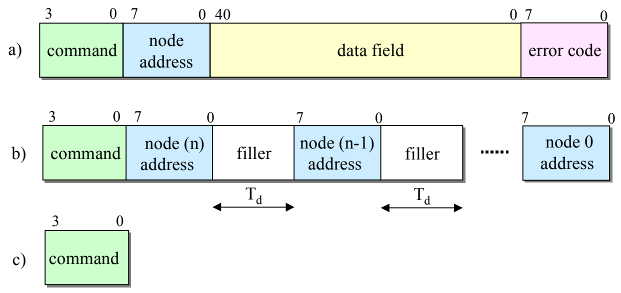


Fig. 3-7. Data formats in the distributed controller network:

a) data frame; b) synchronization frame; c) command frame.

In a ring network, each node introduces a delay in the data propagation path. This means that if a synchronization command is sent through the network, each node will receive the command with as many time delays T_d as there are nodes between that node and the master node. The time delay T_d in the hardware test-bed is typically around 460 ns. This means that the error in synchronization will generate time shifted PWM signals at the outputs, causing low-frequency harmonics. This problem is solved with the synchronization sequence.

The format of the synchronization frame is shown in Fig. 3-7(b). The frame starts with the synchronization command, and is followed by 8 bit long data blocks containing addresses of slave nodes and filler fields, for which it takes T_d to be transmitted and are used for propagation delay compensation. The first address to be transmitted is of the slave node that is last to receive the frame. The number of address data blocks sent equals the number of slave nodes on the ring, which need to be synchronized. The first field is a synchronization command that alerts the nodes to wait for their time to synchronize. Next are the address fields of the nodes being synchronized. After the synchronization command is passed, the node awaits its address field. When the address is received, the node generates the synchronization signal. Because all the addresses are in reverse order and time delayed for the node propagation delay, all the addresses will arrive at the destination nodes at almost the same time. Using this type of synchronization scheme in the proposed 125 Mb/s fiber optical link, the synchronization error is reduced to bellow 80 ns.

Chapter 4 Design and Implementation of Distributed Controller

The distributed digital controller proposed and implemented in this work consists of three main components, namely the application manger, hardware manager and communication protocol. The general block diagram of the distributed controller architecture is given in Fig. 4-1, while the specific example studied in this thesis is given in Fig. 3-5.

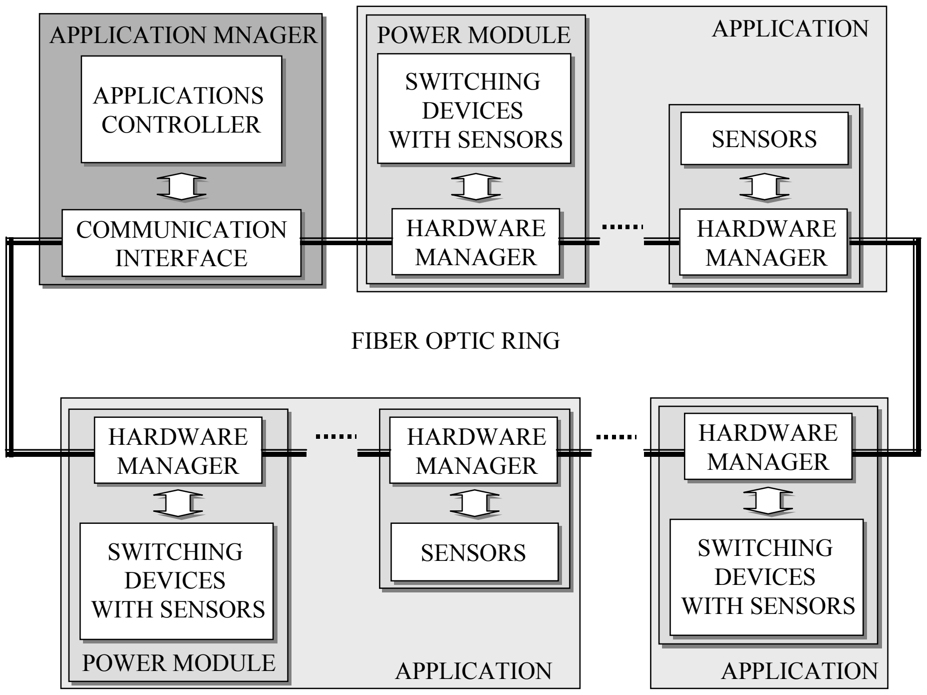


Fig. 4-1. Block diagram of proposed distributed control architecture.

4.1 Hardware Manager Design

The hardware manager provides two basic functions: communication interface and the module control and monitoring. In the following sections, two types of hardware managers will be discussed:

- phase-leg-based hardware manager design, and
- sensor-based hardware manager design.

4.1.1 Phase-Leg Based Hardware Manager Design

As previously explained, a single phase leg has been identified as the basic building block that provides good tradeoff between flexibility (defined as the number of realizable topologies with the basic building cell) and added control and communication complexity, especially in the power range from 100 kW up to 1 MW.

For two-level converters, using the integrated phase leg as basic building blocks, topologies such as the full-bridge converter, the three-phase VSI, or the three-phase boost rectifier can easily be assembled. For some multi-level topologies such as the flying capacitor, this cell can also be identified as a universal building block cell, while for others, such as neutral point clamped (NPC) converters, small changes are needed.

In addition to the main phase-leg, that is supposed to process the majority of the power, an auxiliary phase leg is added to reduce the switching stress on the main switches thus increasing the maximum switching frequency. Both main and auxiliary phase legs together with resonant tank form a zero current transition (ZCT) phase leg [21] shown in Fig. 4-2.

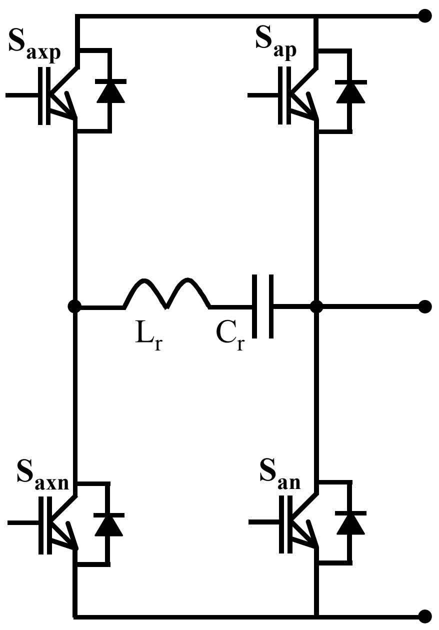


Fig. 4-2. Phase-leg topology with auxiliary soft switching leg and resonant tank.

Considering the topology of the phase leg shown in Fig. 4-2 and control and measurement issues, the phase-leg based hardware manager is designed to perform several major functions:

- PWM generation for main and auxiliary switches;
- isolated gate drives for both main and auxiliary switches;
- over-current protection and indication;
- current, voltage and temperature sensing with analog to digital (A/D) conversion; and
- communication of PWM, status and measurement information.

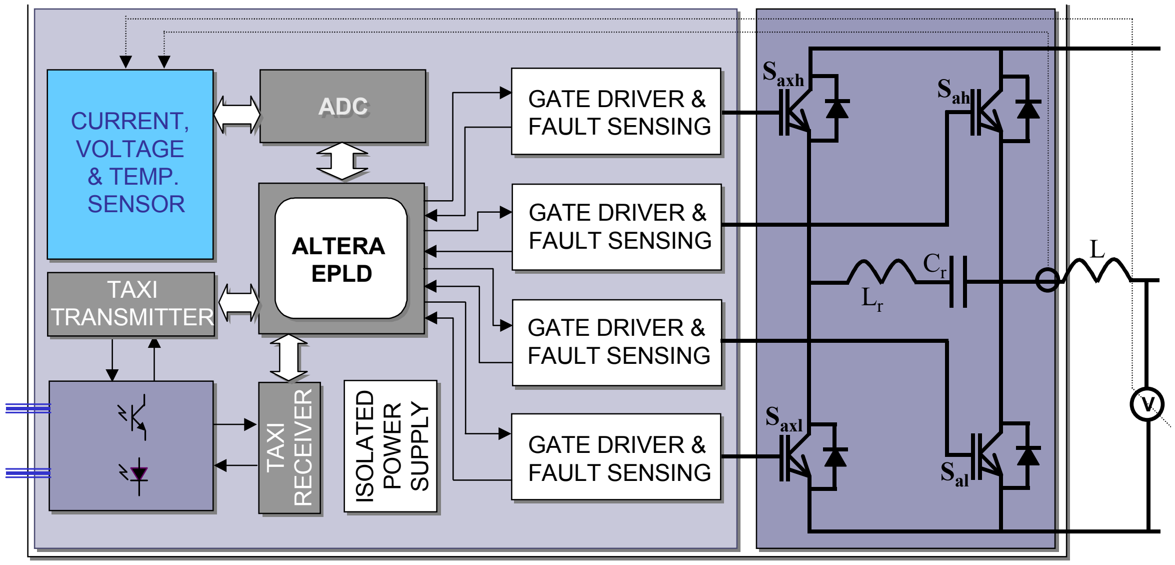

CURRENT, VOLTAGE & TEMP. SENSOR
ADC
ALTERA EPLD
TAXI TRANSMITTER
TAXI RECEIVER
ISOLATED POWER SUPPLY
GATE DRIVER & FAULT SENSING
GATE DRIVER & FAULT SENSING
GATE DRIVER & FAULT SENSING
GATE DRIVER & FAULT SENSING
S_{axh}
S_{ah}
S_{axl}
S_{al}
L_r
C_r
L
V

Fig. 4-3. Block diagram of phase-leg-based hardware manager.

4.1.1.1 Hardware Design

The phase-leg-based hardware manager (as shown in Fig. 4-3) consists of four isolated IGBT gate drives, an optical communication interface comprised of TAXI chips [17], an optical transceiver [18], a programmable logic device (PLD) [20], two A/D converters, current voltage and temperature sensors, and isolated power supplies for both sensors and the digital logic circuitry.

Integrating four different types of circuitry; (the 125 MHz pseudo ECL circuitry, the 12.5 MHz digital circuitry, the analog conditioning circuitry and the fast gate drivers) on a single board and providing EMI, functional and logical compatibility was extremely challenging. Furthermore, the hardware manager's close proximity to the 1200 V 400 A IGBT modules switching in excess of 20 kHz made the whole design even more difficult, especially in terms of EMI.

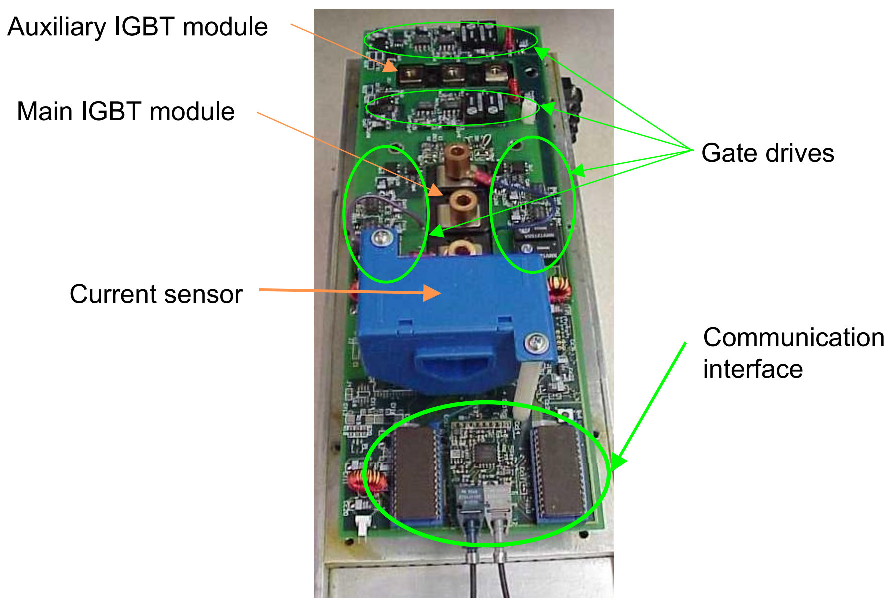


Fig. 4-4. Prototype 30 kW phase-leg based hardware manager.

Fig. 4-4. shows the actual hardware manager design, with spatial distribution of the main functional parts. The voltage sensor and digital logic power supplies are mounted on the bottom side of the board, while the ALTERA PLD and A/D converters are beneath the current sensor.

4.1.1.2 Gate Drive Design

Gate drive circuitry and IGBT modules are the dominant source of both radiated and conducted EMI noise. Therefore, in order to minimize noise propagation and to decouple this part of the circuit from other noise susceptible subsystems several design considerations have to be made.

To illustrate one of the EMI generation mechanism lets take a closer look at the switch commutation time instant. For example, in the phase leg configuration when the upper IGBT module is turning on, the whole gate drive makes a voltage swing of 800 V (DC-link voltage) in a few hundreds nanoseconds. Large dv/dt can create significant common-mode noise currents through the parasitic capacitive coupling paths, while di/dt creates noise through inductive coupling paths.

The three most prominent coupling paths in floating gate drive circuits are power supply isolation transformers, digital input, and output opto-couplers and stray capacitance coupling between floating power plains and other electronic components. Some of the common coupling paths are depicted in the Fig. 4-5 using lumped-parameter circuit representation.

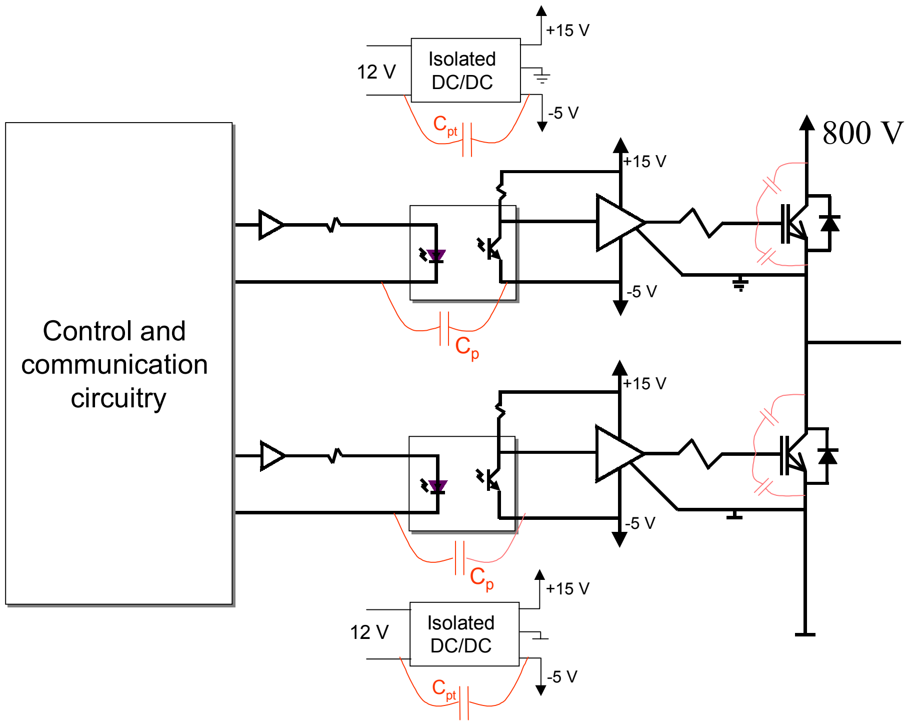


Fig. 4-5. Common coupling paths in floating gate drive circuits.

Opto-couplers that isolate gate-drives from digital control circuitry play an important role in common-mode noise suppression and noise source decoupling. If we consider that secondary side of the opto-coupler is exposed to dv/dt in excess of 5000 V/us, the common mode noise immunity of optocuplers becomes serious design consideration. Therefore, a special class of optocouplers—with built in shielding between primary and secondary side, that brings significant reduction in parasitic capacitance and that improves common mode voltage immunity—has been considered for this design [33]. To illustrate the difference, three opto-couplers were chosen (one with the shielding and two without) and compared. TABLE 4-1 illustrates the difference in common-mode noise immunity between different models. From TABLE 4-1 it becomes obvious that only the HCPL-4503 [33] can meet the requirements of common-mode noise immunity. That is why we it was chosen for the design.

TABLE 4-1. Comparison of three different HP opto-couplers common-mode noise immunity.

	Common-mode Noise Immunity (V/us)	Input-Output Capacitance
HCPL-0500	1000	0.6 pF
HCPL-4502	1000	0.6 pF
HCPL-4503	15000	0.5 pF

A Detailed design schematic with all the gate drive components is given in Appendix A, while Fig. 4-6 shows the conceptual design. The gate drive is designed around the Motorola MC33153 gate drive chip [35] with an additional output push-pull stage to increase the driver's current capability. The PWM logic signal is fed through the input opto-coupler. Fault indication that is generated by the driver chip when the collector-emitter voltage starts to rise above the threshold level is fed through the fault output opto-coupler.

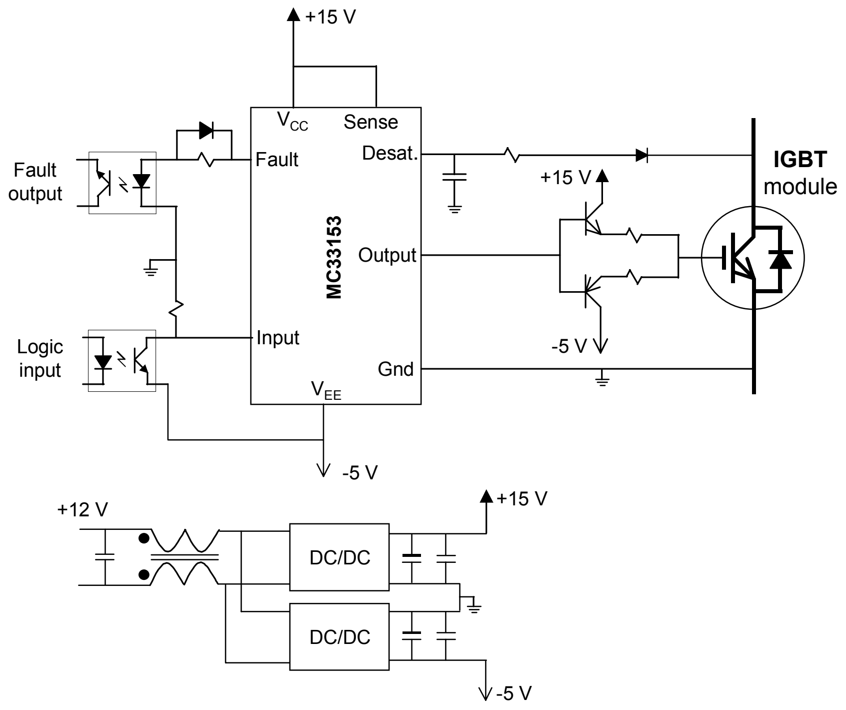


Fig. 4-6. Block diagram of gate drive circuit.

Two isolated power supplies provide +15/–5 volts for gate drive operation. To minimize common-mode current induced due to the parasitic capacitance of power supplies isolation transformer, a common-mode-choke (CMC) is placed in the path of the common mode current, as shown in FIG. 4-6. This CMC makes the noise path high-impedance, thus decoupling gate drive power supply from the power supplies for analog and high-speed logic circuitry [29].

Another coupling path for common-mode noise is the stray capacitance between floating gate drive power planes and the logic power planes. To minimize the stray inductance between the two, a large clearance was incorporated to separate them. Fig. 4-7 shows the printed circuit layout board where all floating power planes (for all four switches) were separated by at least 8 mm of clearance to minimize the coupling capacitance.

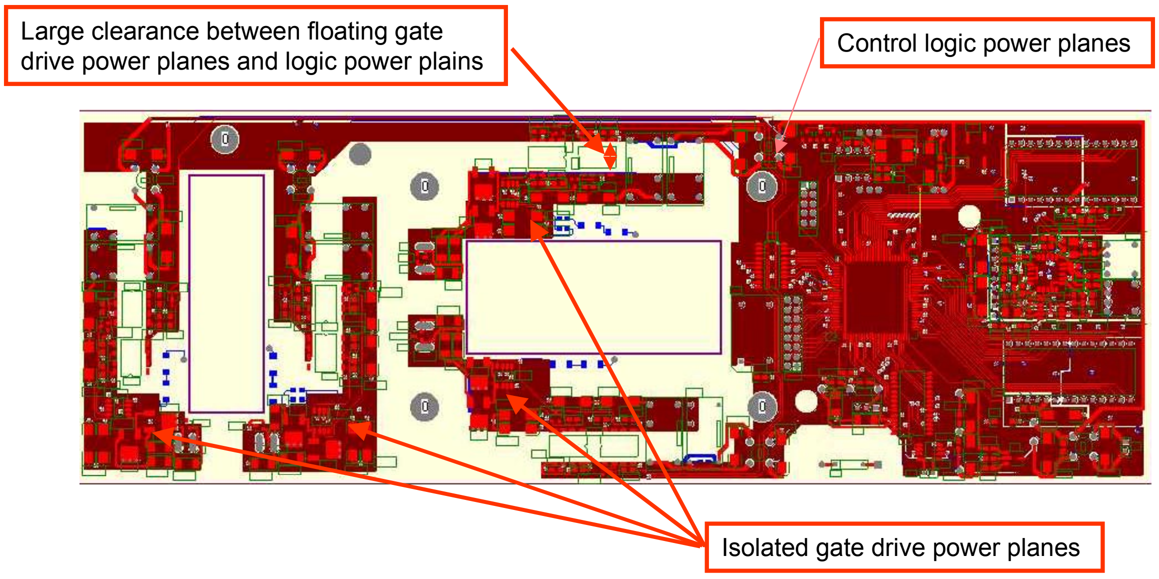

Large clearance between floating gate drive power planes and logic power plains
Control logic power planes
Isolated gate drive power planes

Fig. 4-7. Hardware manager's printed circuit board layout.

4.1.1.3 Communication Interface Hardware

The communication interface consists of a Hewlett Packard (HP) fiber-optic receiver and transmitter [17] and TAXIchip set (Transparent Asynchronous Xmitter-Receiver Interface) [18] comprised of a TAXIchip receiver and TAXIchip transmitter, as shown in Fig. 4-8. The fiber-optic data link receivers consist of a photo-sensitive diode and an amplifier. The photo-diode converts light pulses into currents of around a few hundred nano-amps. This signal current is then amplified and translated into a differential pseudo-ECL signal. The differential ECL signal is then fed into the TAXIchip receiver (Am 7969), that converts serial stream into parallel. On the other side, outgoing data are being fed to TAXIchip transmitter (Am 7968) which converts parallel data into a serial data stream. The serial data stream is fed into the optical transmitter in the form of a differential pseudo-ECL signal, which is amplified in the transmitter and then fed into the fast LED diode.

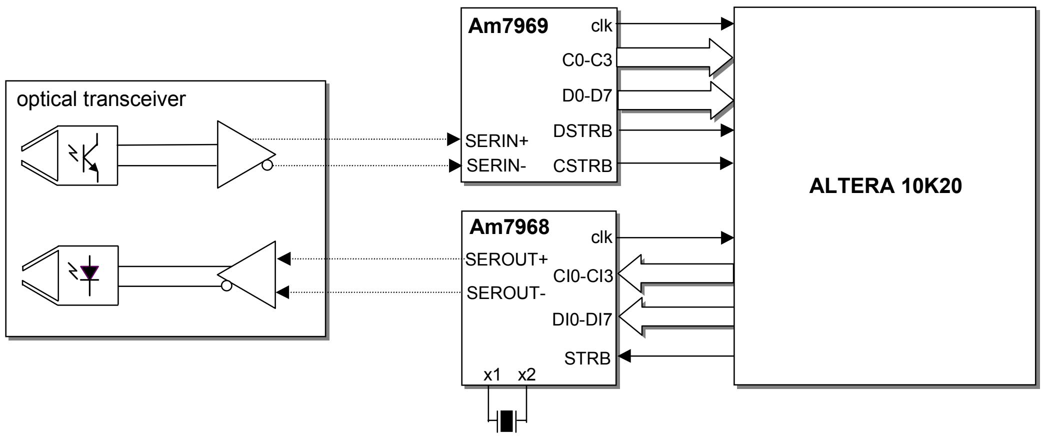


Fig. 4-8. Block diagram of communication interface architecture.

TAXI receivers and most other logical chips (such as the ALTERA PLD) switch hundreds of milliamps. If switching noise from the digital section of the board gets coupled into the optical data link, the signal from the light pulse data can be corrupted. To prevent the coupling of the optical data link output with other digital signals, the user must ensure that the small signal and digital switching currents do not flow in the same path. This is done by separating both the optical power and ground planes from power and ground planes used by other circuitry. The principle of the separation of power and ground planes is illustrated in the Fig. 4-9

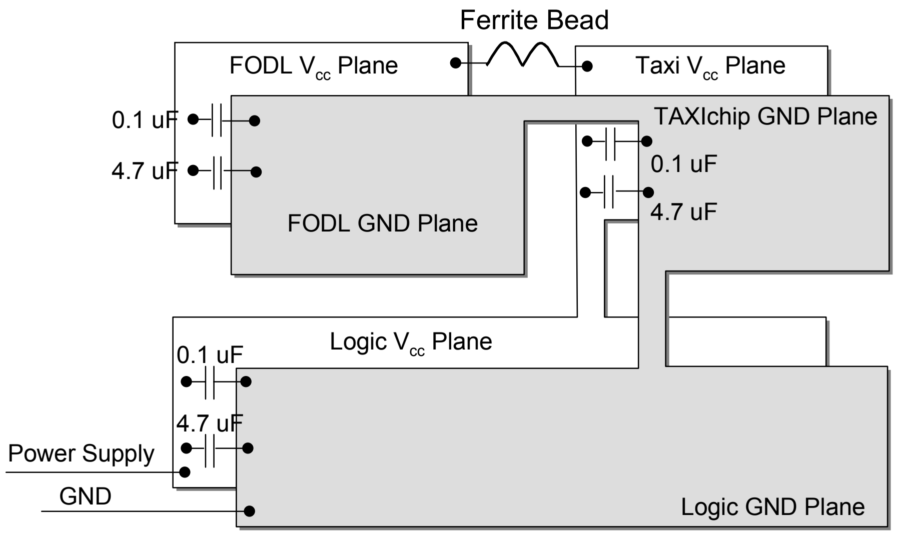


Fig. 4-9. Power planes designed for minimum cross-coupling effect.

For extremely sensitive circuitry such as example fiber optic data link (FODL) power plane should be fully separated from the rest of the circuit and then supplied through either the LC filter (single or two stage) or a ferrite bead. In this way high frequency noise will be filtered out and coupling will be avoided. Also, proper high frequency filter capacitors placed near the sources of the pulsating currents (digital circuitry) will localize the path of those pulsating currents, thus preventing them from coupling with other noise-susceptible circuits.

4.1.1.4 Measurement and Data Acquisition Subsystem

The measurement and data acquisition system on the hardware manager consists of a current sensor, a voltage sensor, a double A/D converter with four input channels and isolated power supply for sensors. Details of the design are given in the Appendix. For both current and voltage measurement LEM hall-effect sensors are used [36, 37].

Basic features of the dual 250 kSPS 12-bit A/D converter AD7862 [34] are as follows:

- two fast 12-bits ADC;
- four input channels;
- 4 us throughput time;
- single supply operation;
- input range ± 10 V and
- high-speed parallel bus interface.

This A/D converter has been chosen because of its high input range (thus better noise immunity), single supply voltage, fast A/D conversion (less then 4 us) and simultaneous sampling and A/D conversion of both channels. Simultaneous sampling allows for concurrent sampling of voltage and current without any phase shift between the two.

4.1.1.5 Digital Control

Control of the hardware manager is implemented in the ALTERA 10K programmable logic device (PLD) [20]. A large number of logic gates (around 20 000), great flexibility, simple and easy to use design software and simple system reconfiguration make this design approach very attractive. Cost-wise this solution may not be optimal (especially if large quantities are required), but unprecedented design flexibility and short design-to-device time are the major features of this class of the devices. Detailed description of ALTERA 10K family is given in [20].

4.2 Logic Design of Hardware Manager Controller

The hardware manager control is designed and implemented in the FLEX 10K20 ALTERA PLD. The controller main functions are to control and coordinate the serial communication link, to generate PWM signals for switches, to control and synchronize the data acquisition process (current and voltage measurements) and to provide local protection (over-current and over-voltage protection) for power switches.

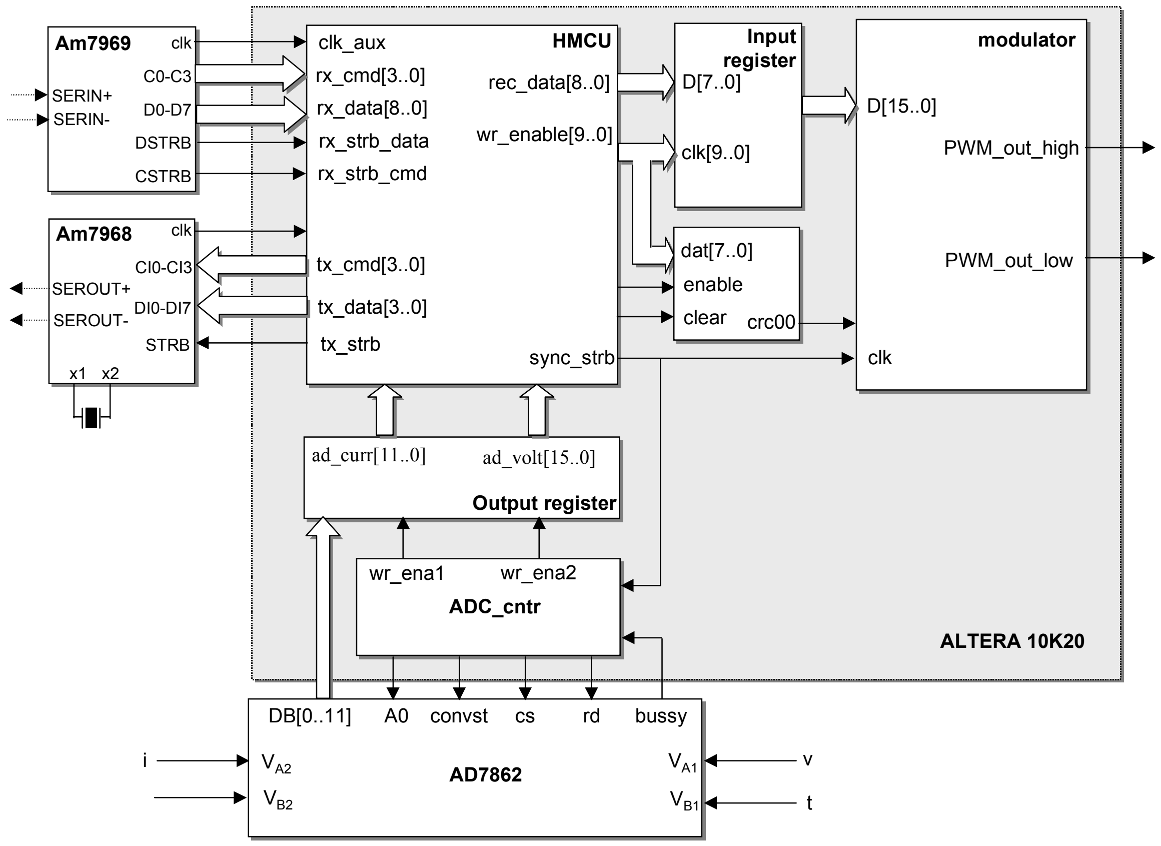


Fig. 4-10. Block diagram of hardware manager controller.

The basic block diagram of the module control architecture that is implemented in ALTERA PLD with main peripheral units is given in Fig. 4-10. The control data are received as a 125 Mbits/sec serial data stream through the optic fiber link. Serial data stream after being amplified and converted into differential pseudo ECL signal (SERIN+

and SERIN-) is fed into a serial to parallel converter (TAXIchip receiver Am7869) that converts serial stream into the 8 bit words (either data or commands) that are being passed to communication controller named “slave”. The hardware manager control unit (HMCU) is implemented as sub-design block within the main ALTERA controller design. HMCU is interpreting the incoming data and commands and stores received control data into data buffers-registers. While the data are being transferred from slave to the register CRC controller is simultaneously loaded with the data. CRC will perform error check and generate fault signal if the incoming data packet is corrupted.

The modulator block uses the register data to generate control signals for IGBTs. The modulator has another set of registers (double buffering), which are loaded from incoming register set only when the synchronization command (sync_strb) arrives from slave and if the data were not corrupted during transmission. This provides synchronization of all modulators in the network regardless of the clock differences and drifts. Every switching cycle the modulator is externally triggered by the network synchronization command.

The A/D controller (ADC_cntr) generates control signals for A/D converter. The A/D controller has to initiate data conversion process. In order to synchronize all the data measurements throughout the network, the A/D controller is also triggered by the network synchronization command. After the conversion is initiated appropriate signals are generated that read the measured value and store it in the buffer register.

Measured variables stored in the local data register are until the data packet with the address that corresponds to the local address arrives. Then the slave extracts the incoming data from the packet and inputs the measured data instead. The TAXI transmitter accepts the data in 8-bit parallel form from the slave and converts the parallel data into serial stream, then feeds the output differential pseudo ECL signal (SEROUT+ and SEROUT-) into the optical transmitter.

4.2.1 Hardware Manager Control Unit (HMCU)

The HMCU is the sub-design block within the main ALTERA design. It is designed to control the communication interface. As shown in Section 3.3, there are three types of network packets: data packets, synchronization packets and command packets. The task of the slave is to accept the incoming data, to interpret it, to process it, and to send it through the network.

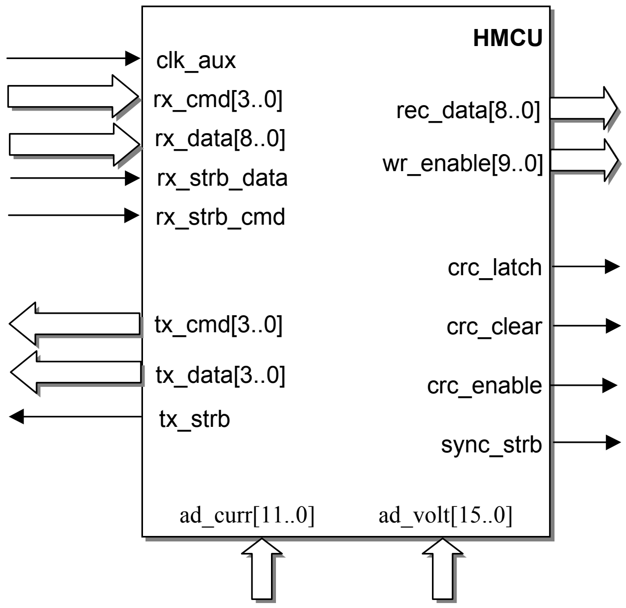


Fig. 4-11. Block diagram of the communication controller.

All the input and output pins (that are only visible in the ALTERA design file) and their functions are explained in TABLE 4-2.

TABLE 4-2. HMCU pin description.

Internal PLD pin name	Pin type	Description
clk_aux	input	Byte-rate clock reference to drive internal logic generated by TAXI Receiver
rx_cmd[3..0]	input	3-bit command from the receiver TAXIchip
rx_data[7..0]	input	8-bit parallel data from the receiver TAXIchip
rx_strb_data	input	Rising edge on causes input data (rx_data[7..0]) to be latched
rx_strb_cmd	input	Rising edge causes input command (rx_cmd[3..0]) to be latched
tx_cmd[3..0]	output	4-bit command that is to TAXIchip transmitter
tx_data[7..0]	output	8-bit data to TAXIchip transmitter
tx_strb	output	Data or command valid on the rising edge
rec_data[7..0]	output	8-bit received data
wr_enable[9..0]	output	10-bit latch signals for rec_data
crc_latch	output	Rising edge latches CRC result
crc_clear	output	Clears previous CRC results
crc_enable	output	Enables CRC calculation
sync_strb	output	Rising edge synchronizes PWM counter and A/D converter
ad_curr[15..0]	input	16-bit current measurement
ad_volt[15..0]	input	16-bit voltage measurement

The HMCU shown in Fig. 4-11. is implemented in AHDL (Altera hardware description language). It is implemented in the form of a state machine. A state machine has five distinct states. The transition between these five states happens when a particular command arrives, when the whole packet is received and retransmitted or when an error occurs. These five states (modes) of finite state machine are given as:

1. **idle** mode (waiting for the data packet),

2. **forward** mode: when the packet address is different from the local address, the data packet is passed to the following node;
3. **store** mode: when the address of the incoming packet is the same as local address, then data is stored locally and the packet is forwarded with the results of the current/voltage measurements and local status information,
4. **synchronization** mode: upon receiving synchronization packet, the state machine goes into synchronization mode; and
5. **initialization** mode: which initializes the whole system and dynamically assigns the node address (this mode has not been implemented).

The state machine block diagram is given in Fig. 4-12, with all the states and transitions that are implemented.

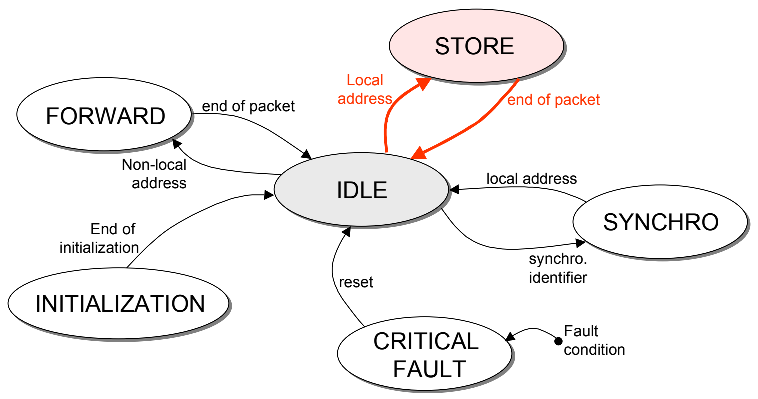


Fig. 4-12. Communication interface state machine transition diagram.

input	initial state	target state	output

CO[3..0] = “local address”;
CSTRB = 1
	Idle	Store	CI[3..0]=1
			STRB_TR=1

CO[3..0] = “not local addr.”;
CSTRB = 1
	Idle	Forward	CI[3..0]=CI[3..0]
			STRB_TR=1

CO[3..0] = 7;
CSTRB = 1
	Idle	Synchro	CO[3..0] = 7;
			CSTRB = 1

Counter = 10 (dec)	Store	Idle	
Counter = 10 (dec)	Forward	Idle	
DO[7..0] = “local address”	Synchro	Idle	Sync_strb = 1

To illustrate the operation of the state machine, several digital simulations of the slave sub-design are shown. CLK_AUX is the clock from the TAXIchip receiver that is synchronized (internal TAXIchip PLL) with the incoming data and commands (DO, CO) as shown in Fig. 4-10. Because of this, clk_aux is being used as state machine clock thus providing good timing and synchronization between incoming data and transitions between different states.

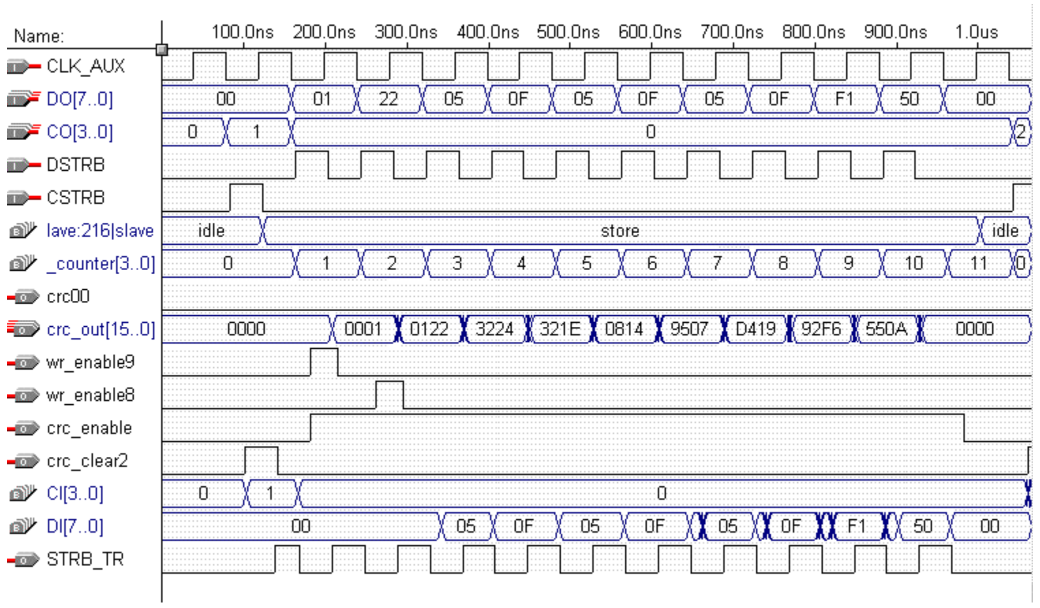


Fig. 4-13. Simulation diagram of communication interface controller.

This design adopts a lslightly simpler data packet then was previously proposed [11] format, such that the data identifier and node address are lumped into one command, thus making the design simpler and more robust. Variable slave is the name of the state machine. At the beginning of the simulation the state machine is in idle mode. The appearance of command “1” at command input CO[3..0] and the appropriate strobe signal triggers the state machine into store mode as shown in Fig. 4-13. Command “1” designates the first node and at the same time indicates that data packet follows. Since the state machine is triggered on the rising edge of clk_aux it changes the state according to this pattern. The change of state resets the local counter which begins to count the number of incoming data bytes. Upon reaching ten that is the expected number of data bytes in the packet, counter resets state machine back to the idle mode. State machine also generates write enable signals that are latching the incoming data into register. The wr_enable[9..0] lines signal when the data on the rec_data[7..0] are valid. Rising edge of the corresponding wr_enable signal is used to latch corresponding data byte. In this example, command “1” (CO[3..0]) is followed by then data bytes (DO[7..0]). Last two bytes in a row are CRC error check bytes. All data bytes are loaded into CRC checker sequentially as they are arriving. When the packet arrives, the data are being stored and simultaneously data packet is being forwarded with new (local) data incorporated in the packet. When the data or command that has to be sent is pushed to the output port (DI[7..0] or CI[3..0]) state-machine generates strb_tr signal. Rising edge of strb_tr signals that data or command is valid. TAXIchip receiver uses strb_tr to latch either data or command.

Another simulation example, shown in Fig. 4-14, explains the state machine transition from idle to synchronization state and the operation during the synchronization mode. When the TAXIchip receiver inserts a command “7” and CSTRB at the same time, the state machine changes the state from “idle” to “sync”. Idle state triggers the local counter. When the state machine receives the address that is the same as its local address, sync_strb signal is generated. Until the state machine receives the local address, all the other data are passed through the network. In this example, the sync_strb is inserted when one is received, since this is the local address of the node.

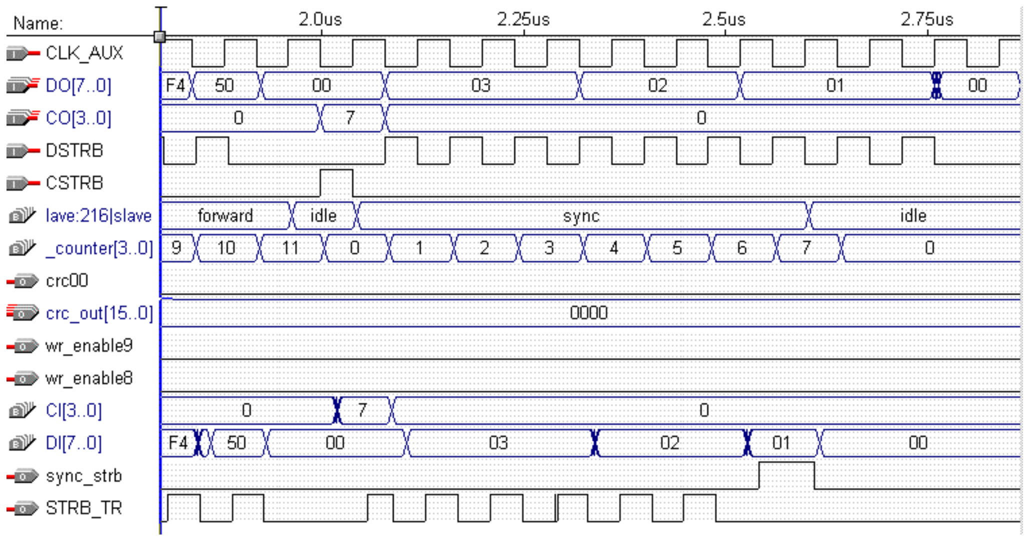


Fig. 4-14. Simulation diagram of communication interface in synchronization mode.

4.2.2 Modulator

A functional block diagram of the modulator sub-design is shown in Fig. 4-15. The two main parameters necessary for proper operation of the PWM generator are: duty cycle and the synchronization command. The duty cycle, when received and validated for proper transmission is being stored in the intermediate register as shown in Fig. 4-10. The duty cycle data from intermediate register is transferred (D[15..0]) into the modulator active register on rising edge of sync_strb line. The rising edge of sync_strb latches the data into the modulator active register. A digital comparator compares the content of the counter with duty cycle information and creates a control pulse for the switch. The dead time generator provides shoot-through protection by introducing dead-time blanking period between top and bottom switches turn-on signals.

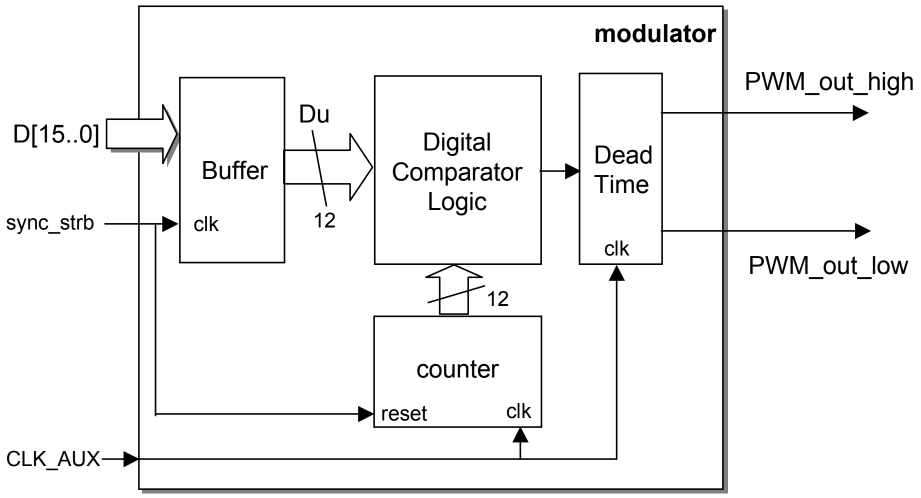


Fig. 4-15. Block diagram of PWM controller.

Fig. 4-16. shows the simulation diagram of the modulator sub-design block. Bus D[15..0] is the input from the intermediate register. Data on the bus are latched into the active buffer on the rising edge of the sync_strb signal. Du[15..0] is the output of the active buffer. PWM_average is the output of the digital comparator that reflects the actual duty cycle. PWM_out_high and PWM_out_low are the top and bottom signals after the dead-time block. In this simulation, the ratio of dead-time to switching period is set very high for the sake of reducing the simulation time and for a more clear presentation of simulation results.

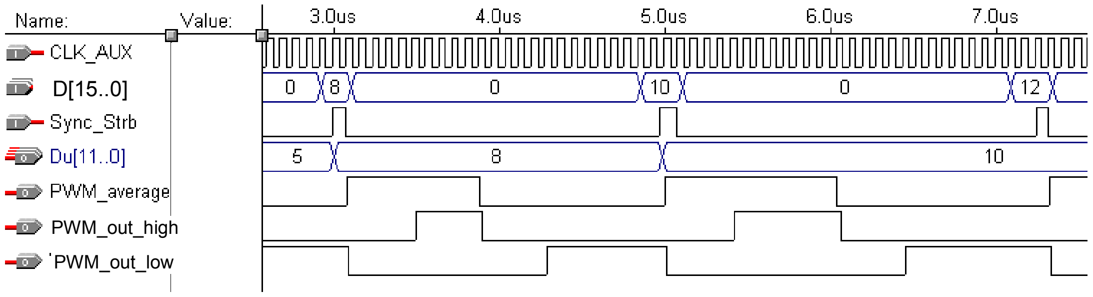


Fig. 4-16. Simulation diagram of modulator.

4.2.3 Data Acquisition Subsystem

Most of today's control systems in power electronics are based on full-state feedback, which requires per-switching cycle current and voltage measurement. If the measurement is done locally on the module several benefits can be achieved as follows:

- Measurement of current, voltage and temperature locally on the module can be utilized for fast over current/voltage/temperature protection.
- The A/D converter can be placed close to the sensor thus reducing noise problems and at the same time reducing unnecessary wiring and connections.
- Sensors are scaled (sized) to the power stage ratings and not controller; they are designed and built together with power stage
- Hardware manager local control can be implemented (e.g. minimum-loss SVM, variable-timing soft switching, dead-time compensation). Similarly, voltage feed-forward can be done locally.

The analog-to-digital controller is a sub design block implemented within the main controller design as a part of the ALTERA design. It is designed using AHDL language. Its task is to provide control signals to the A/D converter and to store the converted data in the buffers. The block diagram of the ADC_cntr is given in Fig. 4-17.

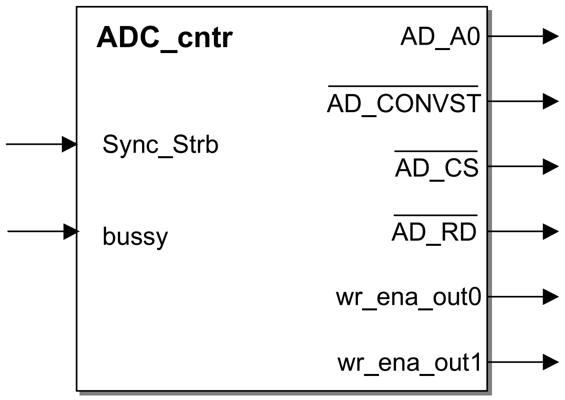

ADC_cntr
Sync_Strb
bussy
AD_A0
$\overline{\text{AD\_CONVST}}$
$\overline{\text{AD\_CS}}$
$\overline{\text{AD\_RD}}$
wr_ena_out0
wr_ena_out1

Fig. 4-17. Block diagram of ADC controller realized in ALTERA.

A high level on the input pin Sync_Strb initiates the whole conversion process, as shown in simulation in Fig. 4-18. When Sync_Strb becomes high internal counter starts counting. After the counter reaches the pre programmed value ADC_cntr initiates the start of AD conversion. AD conversion is initiated when AD_CONVST goes from high to low. The timing between Sync_Strb and AD_CONVST determines the relative position of sampling time from the beginning of switching period. This interval can be changed by reprogramming the inner counter thus changing the sampling instant accordingly. After 3.6 us from the beginning of the conversion period (the time it takes for AD converter to finish conversion) ADC_cntr inserts read signal (AD_RD). Falling edge of AD_RD signal causes the data to appear on the A/D converter data bus. Output signal wr_ena_out0 latches the data on converter data bus into the buffer. Next falling edge of the AD_RD signal causes the result of the second A/D converter to appear on the data bus. The rising edge of the wr_ena_out1 latches the data into the second register. After both results of the both A/D conversions are latched the ADC_cntr goes into the idle state.

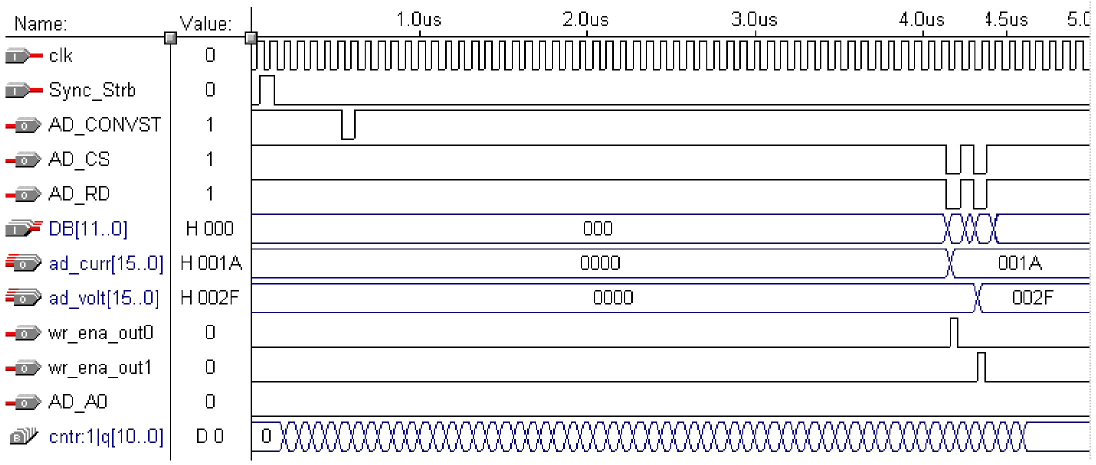


Fig. 4-18. Simulation diagram for ADC controller operation.

4.2.4 Cyclic Redundancy Code (CRC) Generator

Error detection codes enable data transmission through a noisy environment (channel). There are many techniques for obtaining error detection and error correction. Some of them are very simple like parity check or checksum. Others, such as convolutional codes, provide not only error detection but can also extract the true message.

CRCs have been accepted in many applications mainly due to the following:

- excellent protection against common errors such as burst errors, in which consecutive bits in data stream are corrupted during transmission, and
- the original data is the first part of the transmission, which makes systems that use CRCs easy to understand and implement.

The basic idea behind the CRC algorithms is to treat the binary message as the polynomial and to divide it by another polynomial and then making the remainder of this division as error check code. For example, the message 10111 in binary system can be represented as a polynomial:

$$1 \cdot x^4 + 0 \cdot x^3 + 1 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0 \qquad (4\text{-}1)$$

All the coefficients are represented in modulo-two base, which means that they can either be 1 or 0. The generator polynomial is the fixed polynomial with which we divide the message polynomial to obtain the CRC error check. We have to have in mind two things while performing the division: this is polynomial arithmetic (no carries) and it is performed in MOD 2 base (1+1=0; 1+0=1, 0+1=1, 1-0=1; 0-1=1). Few examples of polynomial representation of binary numbers are given in TABLE 4-3.

TABLE 4-3. Polynomial representation of binary numbers.

	Binary representation	Polynomial representation
Original message	1101011011	$x^9 + x^8 + x^6 + x^4 + x^3 + x^1 + x^0$
Generator polynomial:	10011	$x^4 + x^1 + x^0$
Message after appending w zeros	11010110110000	$x^{13} + x^{12} + x^{10} + x^8 + x^7 + x^5 + x^4$

The width of polynomial 10011 is w=4, where w is defined as the maximum length of the remainder of the division. If the length of the divisor polynomial is for example five (10011), maximum length of the remainder will be w=4, since it will produce the remainder that has one power less. To obtain the error code for the original message w zeros has to be appended to the message before the CRC is calculated. After the remainder is calculated it is put in the place of the appended zeros. The new message with appended error check divided by the generator polynomial gives zero remainder. This is the basic for error detection. If some of the bits are corrupted during the transmission the message will not give zero remainder when divided with generator polynomial.

Therefore, generating the error checksum can be mathematically represented as:

$$x^{13} + x^{12} + x^{10} + x^8 + x^7 + x^5 + x^4 : (x^4 + x^1 + x^0) \qquad (4\text{-}2)$$

or in binary representation as:

(11010110110000):(10011) (4-3)

The fully worked-out example of finding the remainder (CRC error check code) for previously given polynomials is following:

```
11010110110000 : (10011)
10011
 10011
 10011
  00001
  00000
   00010
   00000
    00101
    00000
     01011
     00000
      10110
      10011
       01010
       00000
        10100
        10011
         01110
         00000
          1110 = remainder = CRC error code
```

Therefore the message with appended CRC code for given message and the generator polynomial is 11010110111110.

Implementation of the polynomial division in digital form can be done in two ways. First way is very straightforward and resembles the procedure of hand calculation but is very slow. First we load the data buffer with highest four bits of message. Then we shift left the message for one bit. If the bit that went out of buffer is one then we XOR the buffer with generator polynomial. The same as what we did in previous example. Then we shift left buffer for one bit, examine the last bit to go out and repeat the procedure till we reach the end of the message. Detailed description of this algorithm can be found in [27]. This algorithm is slow since for every bit of the message we need one clock cycle.

In our case incoming data are bytes (8 bits) therefore the ALTERA implementation is modified to calculate CRC code in parallel. Every byte is loaded in parallel and CRC code is calculated instantaneously. Details of this parallel implementation can be found in [27]. CRC code generator/checker block implemented in AHDL is given in Fig. 4-19.

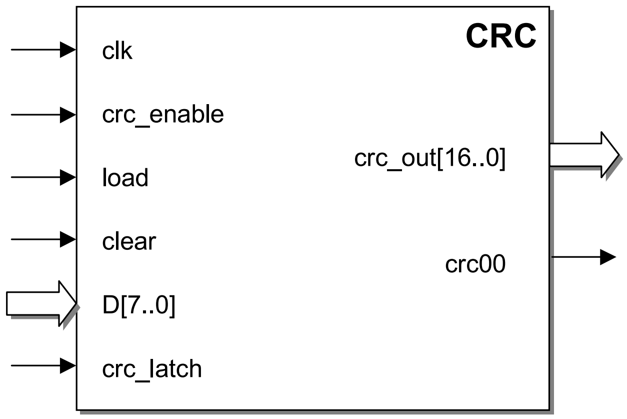


Fig. 4-19. Block diagram of CRC generator.

The message that is to be coded or decoded by CRC generator is loaded in parallel form. Incoming message is being fed in pieces of eight-bits to the D[7..0] while the result appears on the output parallel bus (CRC_out[16..0]) after the last byte is latched into CRC generator. CRC generator internal logic is clocked with external clock signal. Detailed description of each pin is given in the TABLE 4-4.

TABLE 4-4. CRC generator pin description.

Internal PLD pin name	Type	Description
clk	input	Byte-rate clock reference for internal logic
crc_enable	input	High-level on this pin enables CRC calculation
clear	input	High-level clears all registers in CRC generators Before new message all the registers should be cleared.
D[7..0]	input	8-bit input into CRC generator/checker
crc_latch	input	Rising edge on this pin causes result of CRC check to be latched on **crc00** output
crc_out[15..0]	output	Result of CRC calculation which is valid on the first falling edge of input clock after the last message byte was loaded.
crc00	output	CRC error check indicator

The simulation diagram shown in Fig. 4-20 illustrates the process of CRC code generation for the incoming message (01 34 00 00 hex) while the generator polynomial is given as $\mathbf{x^{16} + x^{12} + x^{5} + x^{0}}$. Every byte is loaded on the rising edge of the clock signal. The result is ready on the next falling edge of the clock after the last byte of the message has been loaded. In this case 16 bit wide CRC code is 45 E6 (hex).

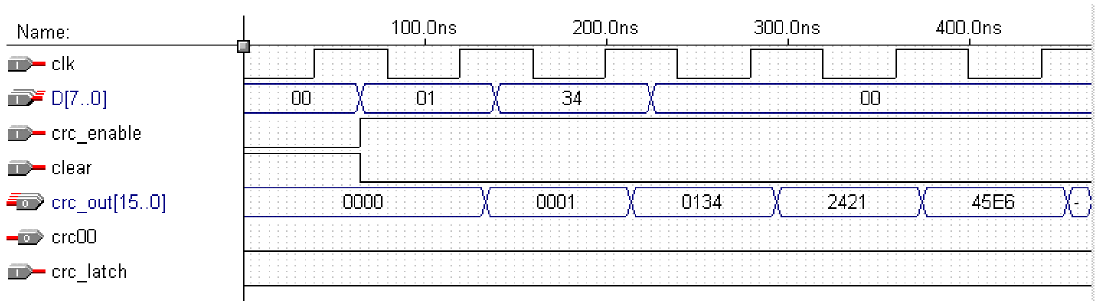


Fig. 4-20. Simulation timing waveform of a byte-wide CRC generator.

4.3 Sensor-based Hardware Manager Design

Although the hardware manager has built-in current, voltage and temperature sensors for some applications, additional system sensors such as position, velocity or electromagnetic field measurements may be needed to provide feedback to the application controller.

To overcome this limitation a new architectural block, smart sensor, is introduced. The smart sensor can be based on any type of sensor or transducer. Smart sensor has a lumped intelligence and communication interface together with sensor(s) and AD converters. Therefore this unit is fully network transparent. Built-in intelligence is providing full control over sampling and conversion process while communication interface is allowing for simple data access and management.

The block diagram of sensor based hardware manager is shown in Fig. 4-21. It consists of 125 Mbits/sec optical transceiver, a TAXI chip receiver-transmitter pair, an ALTERA 10K PLD, two A/D converters and a current and voltage sensor (although it can be any type of transducer).

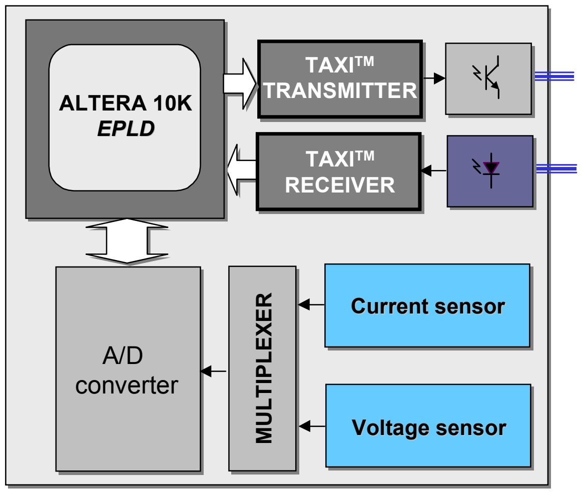


Fig. 4-21. Block diagram and hardware prototype of the distributed current sensor.

Hardware-vise the sensor based hardware manager is similar to the phase-leg based hardware manager except that no isolated gate drives are needed for the former. Implementation of the communication interface and the state machine is very similar to the previously explained phase-leg based hardware manager, except that in this case the state machine has only to control data acquisition process and AD conversion. Therefore, whole control can be embedded into the smaller PLD chip or it can be manufactured as separate less expensive ASIC chip with reduced capabilities.

Block diagram of sensor based controller is given in Fig. 4-22. Since the principal sub-designs are similar detailed explanation of the block diagram will not be given assuming the similarity between this implementation and phase-leg based hardware manager.

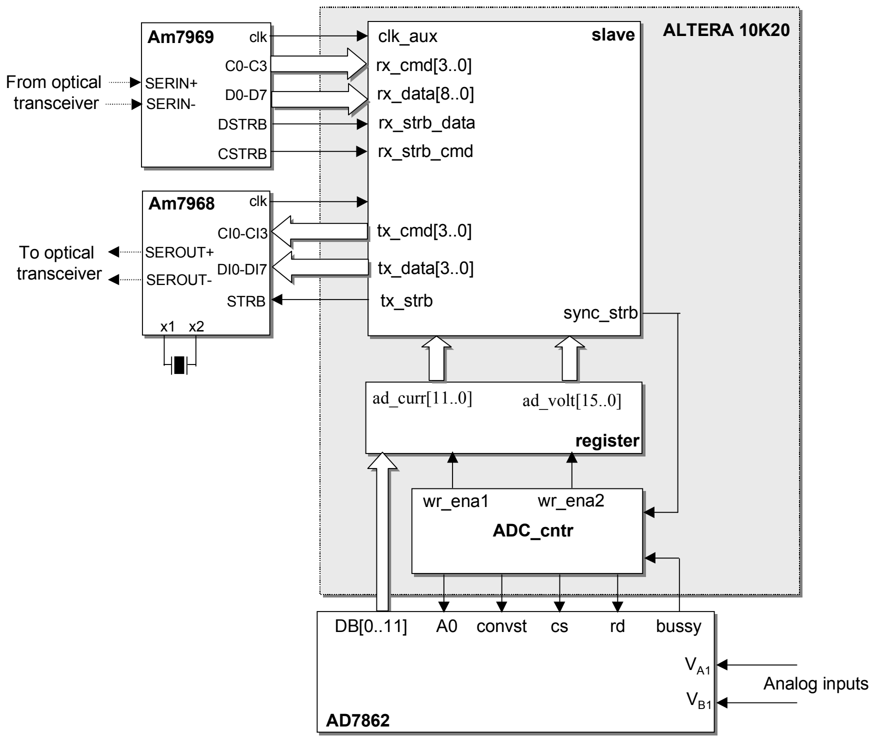


Fig. 4-22. Block diagram of control block for sensor-based hardware manager.

The block diagram in Fig. 4-22 shows the basic control architecture of the sensor-based hardware manager. The local controller accepts the data and commands from the TAXIchips and either stores the incoming data (if the address corresponds to the local address) and replaces them with the measured data (rearranging the packet), or directly forwards the packet to the following node (if the address of the packet is different from the local address). The local controller is also extracting the synchronization time from the synchronization sequence in order to achieve synchronize sampling and switching states throughout the whole network.

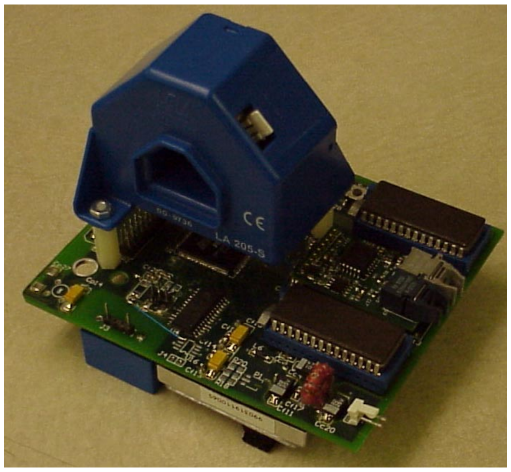


Fig. 4-23. Picture of current sensor based hardware manager.

Fig. 4-23 shows the prototype implementation of sensor based hardware manager. Although in this case Hall-effect current sensor has been used almost any type of sensor can be incorporated in this design.

4.4 Application manager

As described before in Section 3.3, the application manager tasks are:

- high-level control (current, voltage load control);
- distributed control network arbitration and synchronization;
- system initialization; and
- monitoring and protection.

4.4.1 Application Manager Design

The prototype applications manager is built around Analog Devices 21062 SHARC DSP EZ-LAB Development System [28, 40], with piggy-back board with two ALTERA 8K PLD chips and fiber-optic communication interface board, as shown in Fig. 4-24.

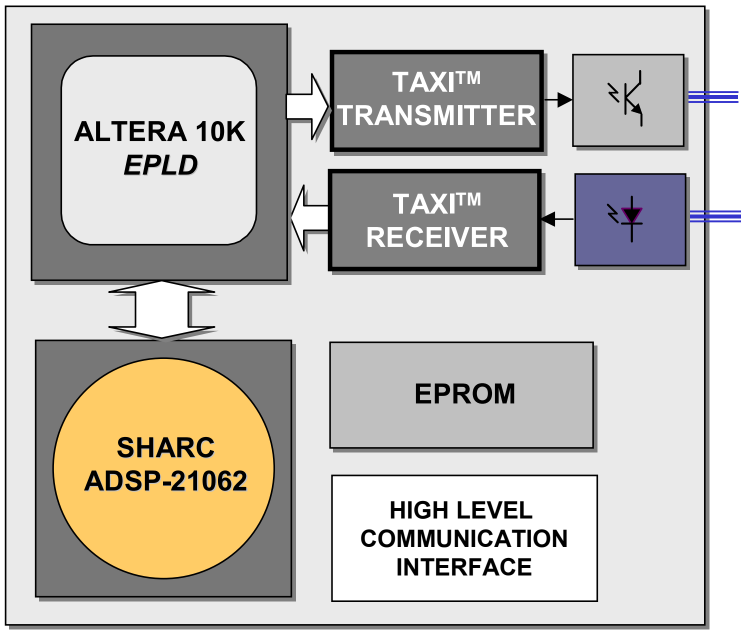


Fig. 4-24. Block diagram and application manager hardware prototype.

The ADSP-21062 EZ-LAB development system from Analog Devices is based on the 33 MHz floating point ADSP-21062 DSP processor. It allows for in-circuit emulation using a JTAG port and facilitates communication with additional modules through the SHARCPAC expansion connectors. Digital interface board, consisting of two 8K ALTERA PLD chips, two EPROM-s and connection ports, is connected two the DSP development system through SHARCPAC [41]. The third element of the application manager is the 125 Mbits/sec communication interface board. Block diagram is given in Fig. 4-25.

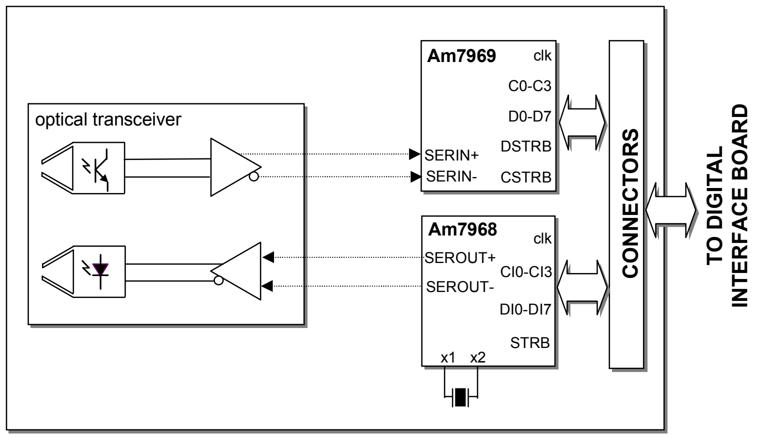


Fig. 4-25. Block diagram of communication interface board.

The communication controller is implemented in the ALTERA 8K PLD, which serves as the interface between the DSP and the communication interface. Block diagram of implemented communication interface is given in the Fig. 4-26. Main design blocks are receiver, transmitter, input and output registers address decoder and timer.

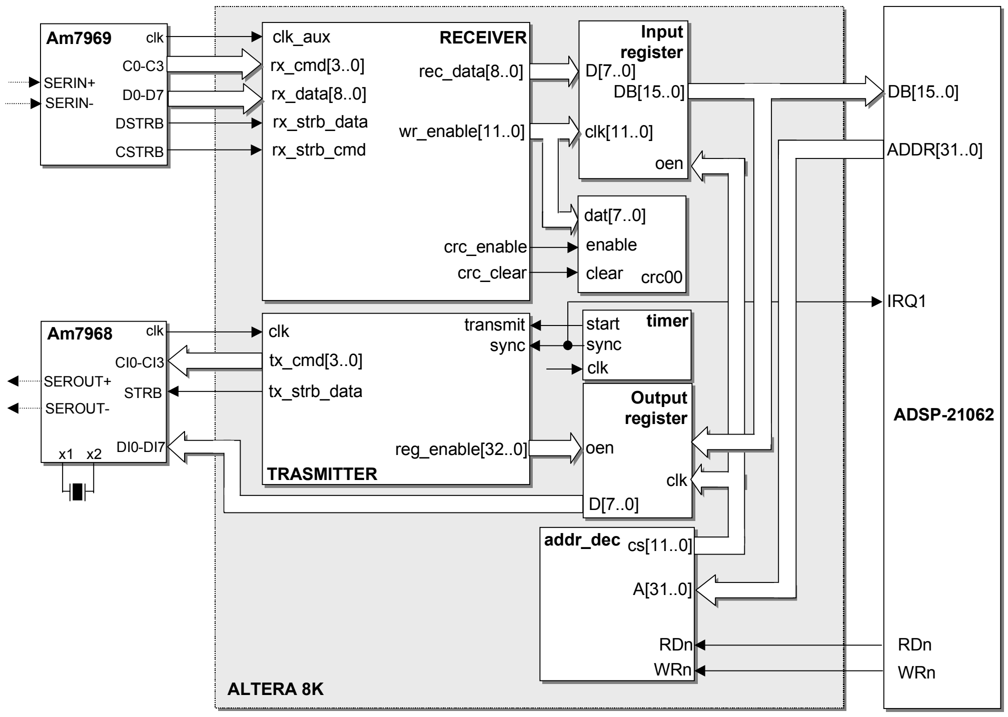


Fig. 4-26. Block diagram of communication controller for the applications manager.

Receiver block is implemented in AHDL as a state machine that handles incoming data packets from the TAXI chip receiver and stores them into the input buffers. DSP processor can address all the input registers through the address bus. Input and output pins are described in the TABLE 4-5.

Transmitter block is activated when the transmit pin goes from low to high. Then transmitter sends all the data to the hardware managers. When the sync pin goes high that initiates sending of the synchronization sequence. The transmit and sync pins are fed from the timer which insures synchronous operation of the applications manager. Detailed pin descriptions are given in the TABLE 4-6.

TABLE 4-5. RECEIVER PIN DESCRIPTION.

Internal PLD pin name	Pin type	Description
clk_aux	input	Byte-rate clock reference to drive internal logic generated by TAXI Receiver
rx_cmd[3..0]	input	3-bit command from the receiver TAXIchip.
rx_data[7..0]	input	8-bit parallel data from the receiver TAXIchip
rx_strb_data	input	Rising edge on causes input data (rx_data[7..0]) to be latched.
rec_data[7..0]	output	8-bit received data
wr_enable[11..0]	output	12-bit latch signals for rec_data
crc_latch	output	Rising edge latches CRC result
crc_clear	output	Clears previous CRC results
crc_enable	output	Enables CRC calculation

The timer block initiates the data sending while the sync signal that initiates network synchronization is also fed to the DSP as interrupt request (IRQ1 pin). Interrupt request in DSP invokes interrupt service routine, which first reads the data from the input buffer and then performs the calculation and writes calculated control variables into output buffers. Those data will wait till the next transmit signal to be passed to the hardware managers.

TABLE 4-6. Transmitter pin description.

Internal PLD pin name	Pin type	Description
clk	input	Byte-rate clock reference to drive internal logic generated by TAXI Receiver
tx_cmd[3..0]	output	4-bit command that is to TAXIchip transmitter.
tx_strb	output	Data or command valid on the rising edge
reg_enable[32..0]	output	33-bit output enable signals for output registers
transmit	input	Initiates data transmission
sync	input	Initiates network synchronization sequence

Timing diagram of the applications manager data flow is given in Fig. 4-27. Sync signal is dictating the switching frequency, sampling instant and network synchronization. Immediately after the sync signal communication interface sends synchronization sequence and DSP start executing interrupt routine. Upon completion DSP writes the data into the input buffer. In our case AD conversion plus calculation delay is always less then 2 switching periods. $T_{send}+T_{prop}+T_{sync} < 4$ us (for three-phase closed loop VSI), where T_{send} is the time necessary for data to be send through the network, T_{prop} is the propagation delay of the network and T_{sync} is the time necessary for transmission of synchronization sequence. Therefore, according to the Fig. 4-27 for $f_{sw} =$ 20 kHz; more then 47 us is available for DSP calculation.

The DSP code that performs the control of hardware managers is structured as a interrupt service routine. Every time sync signals goes high, IRQ1 line will signal external hardware interrupt to DSP, which will invoke interrupt routine. The structure of the interrupt routine is given in Fig. 4-28. Therefore the code design is not affected with the communication interface. Whole interface is designed to be fully transparent to the DSP and thus simplify the design process.

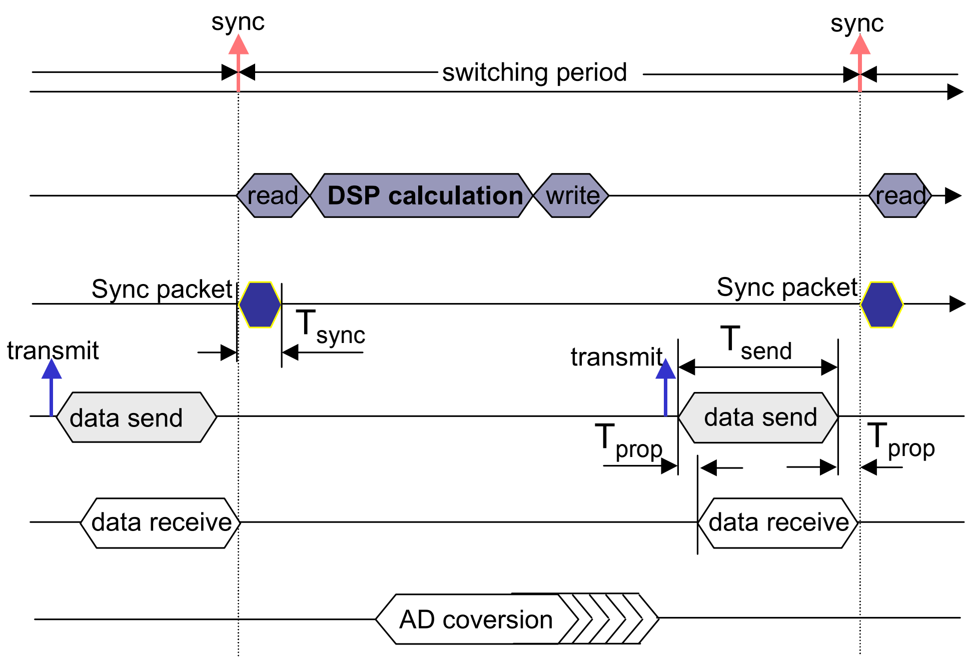


Fig. 4-27. Applications manager data communication timing diagram.

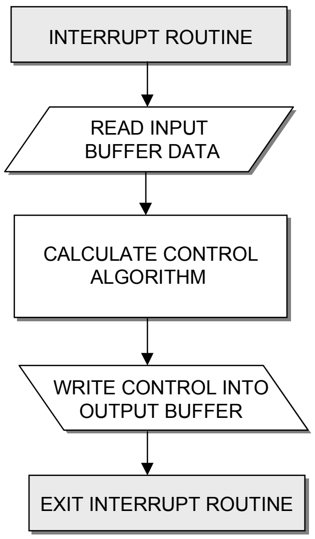


Fig. 4-28. Interrupt service routine (IRQ1) code structure.

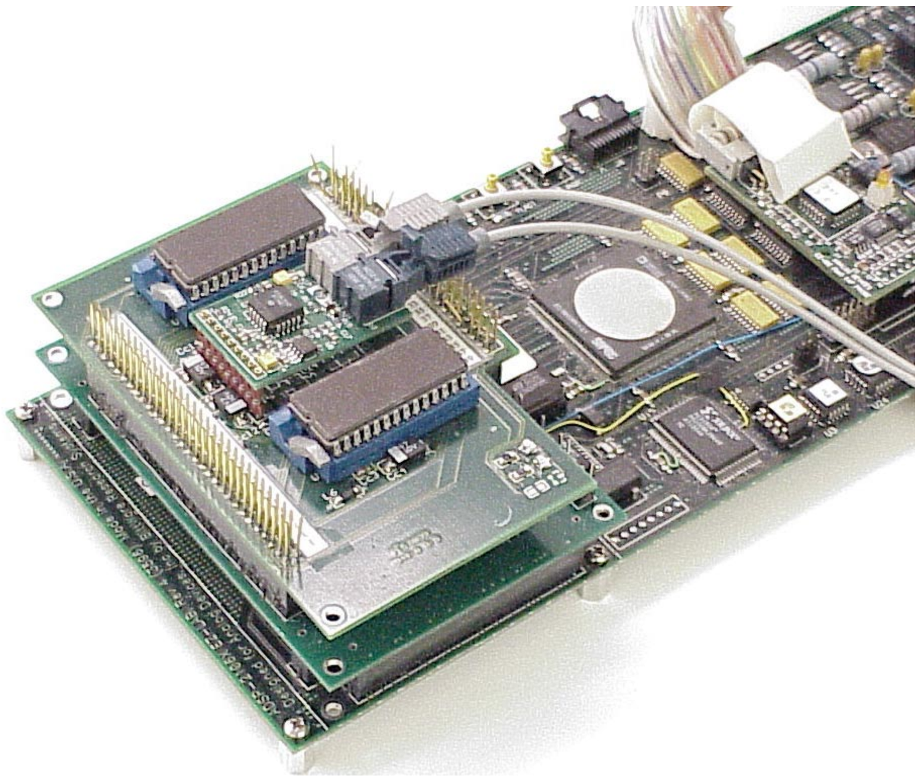


Fig. 4-29. Prototype applications manager.

The power of the distributed controller and application manager design is in the universal and flexible communication interface. Regardless of the power stage, number of hardware managers or power level architecture of application manger is the same. This means that the same application manager can be used in whole range of different applications thus increasing the production volume of application managers and eventually reducing the costs. Instead of constant struggle with different hardware interfaces for different converters and applications proposed approach is promoting standard serial interface. All the flexibility for interfacing different applications and hardware managers is now achieved through the software reconfiguration rather than hardware. Picture of built prototype applications manager is shown in Fig. 4-29.

Chapter 5 Experimental Verification

As previously explained the idea of distributed control has been around for quite some time. Yet distributed and hierarchical control has neither been accepted in power electronics systems nor extensively explored.

Bringing the distributed controller to the converter level introduces many technically challenging issues up front. EMI immunity of the control system becomes very important factor. Another unexplored issue that begins to play a major role in control architecture is synchronization of the distributed control. Measurement delay and sampling frequency jitter with distributed data acquisition becomes a significant issue if fast, safe, and accurate control of a converter is to be achieved.

Therefore a distributed system on a power converter level was built in order to test and verify some of the most critical design aspects of the new approach. The details of the design and operation are given in the following section.

5.1 Three-Phase Voltage Source Inverter Design

To verify the idea of a distributed controller, a two-level three-phase VSI was designed. The inverter is coupled with a simple R-L load. Block diagram of the system is given in Fig. 5-1. The prototype of the converter consists of three phase-leg based hardware managers connected in three-phase voltage source topology. The applications manager performs control of the three-phase VSI. The phase leg modules are controlled via daisy chained fiber optic network. The application manager is also connected with the local PC through JTAG port which allows for simple software debugging and fast code downloading. Also, this link could be used for higher level (system) control and monitoring.

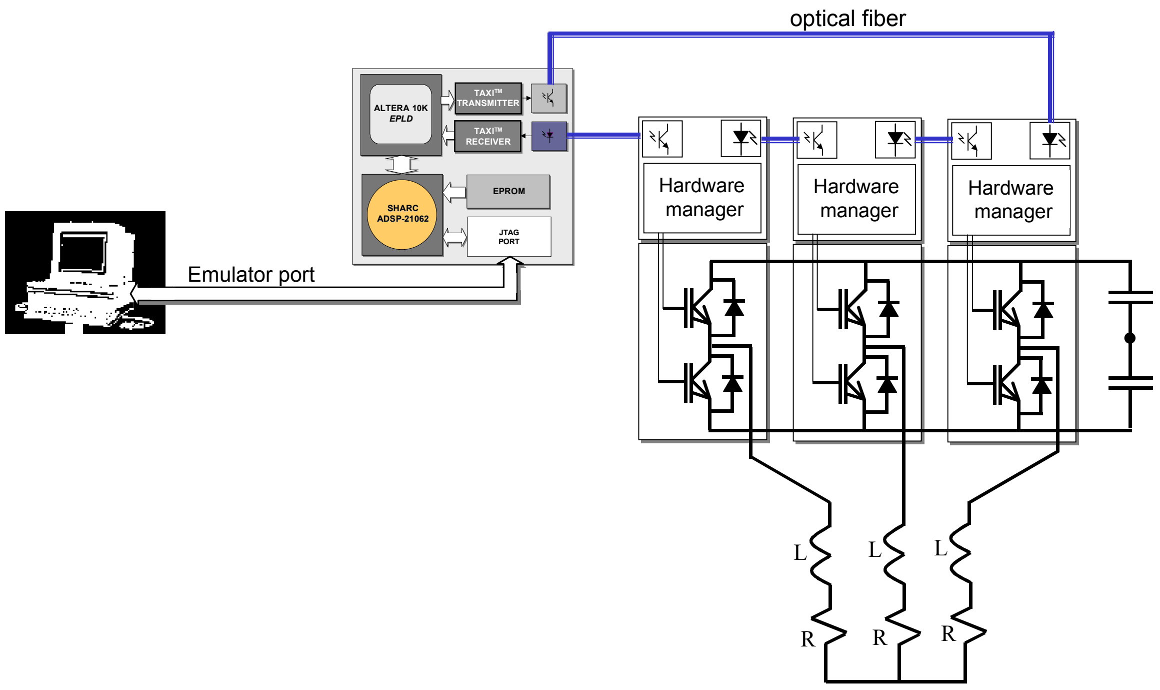


Fig. 5-1. Block diagram of three-phase VSI built using distributed control approach.

The general specifications for this three-phase VSI are given as follows:

• DC input voltage:	800 V
• Output power:	100 kW
• Switching frequency:	20 kHz

The IGBT devices that were used for main and auxiliary switches are as follows:

• POWEREX	1200 V,	300 A (main modules),
• POWEREX	1200 V	150 A (auxiliary modules)

An actual picture of the prototype is given in Fig. 5-2. Its modular structure is a very prominent feature of this design.

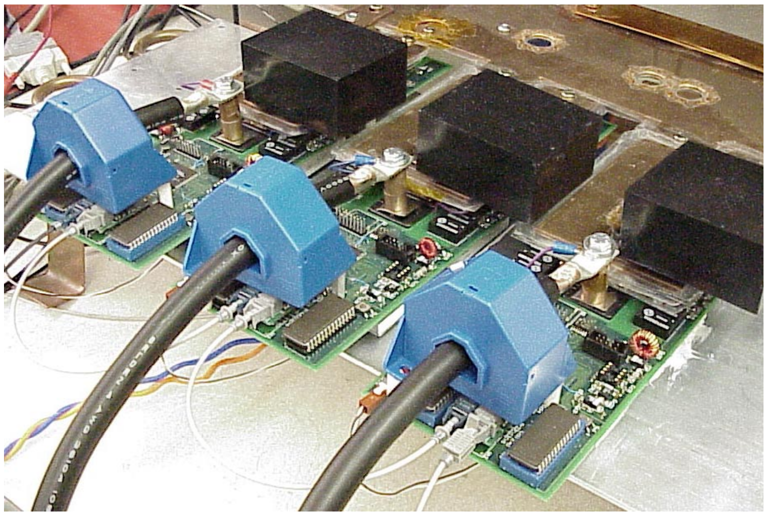


Fig. 5-2. Prototype three-phase VSI built using smart modules and distributed controller approach.

5.1.1 Synchronization

Each node in the daisy-chained network introduces a propagation delay into the signal path. For the previously explained example of the distributed controller that consists of an applications manager and the three phase-leg based hardware managers the total propagation delay was measured. Fig. 5-3 shows that total delay through the three hardware managers is around 1.7 us, measured from the time instant when the packet is sent from applications manager until its return to the applications manager. Therefore, the propagation delay introduced by each node is around 560 ns. In order to compensate for this intrinsic delay, method proposed in [11] was used. The effectiveness of this compensation scheme is shown in Fig. 5-4. Signals SYNC_1, SYNC_2 and SYNC_3 are synchronization signals generated locally on the hardware managers that are used as a start signals for PWM generation and time synchronization of the modules. It can be seen that maximum synchronization jitter is always less then 80 ns (network transmission byte rate). Typical value of the synchronization jitter is around 40 ns which is one half of the communication byte rate. CMDR and DATAR are command and data received at the second node. That is the synchronization packet as explained in 3.3.1.

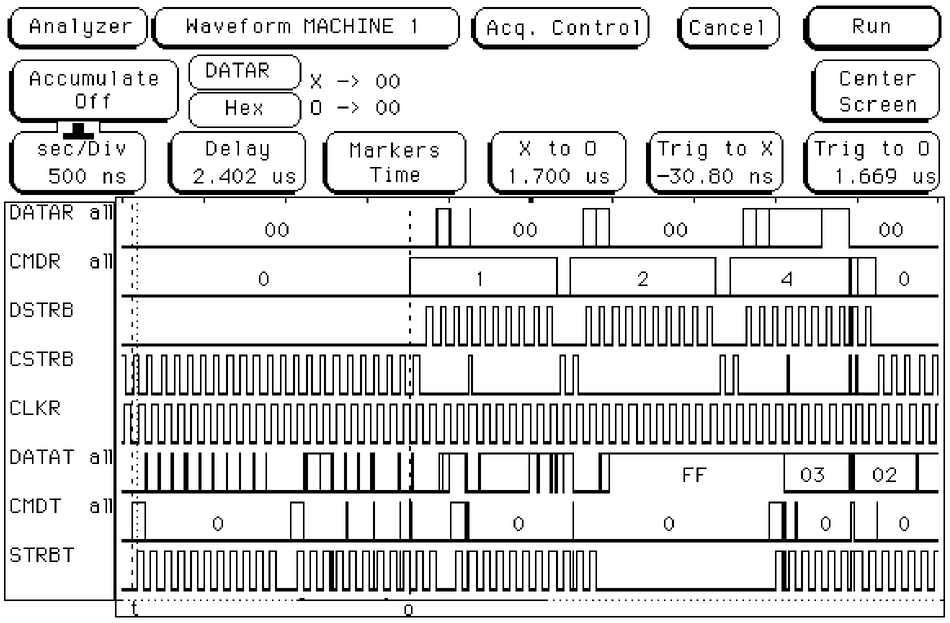


Fig. 5-3. Propagation delay measurement through the network.

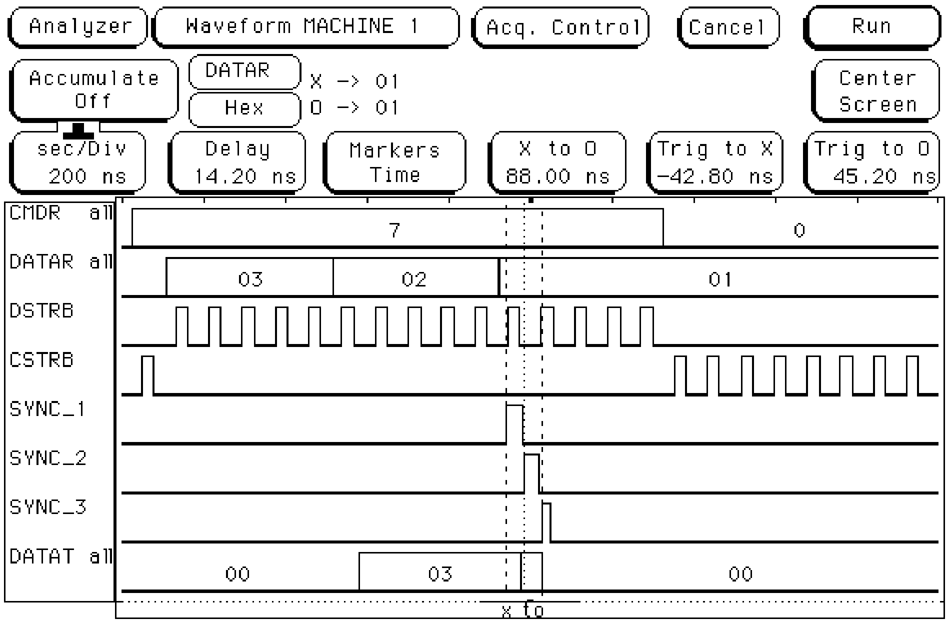
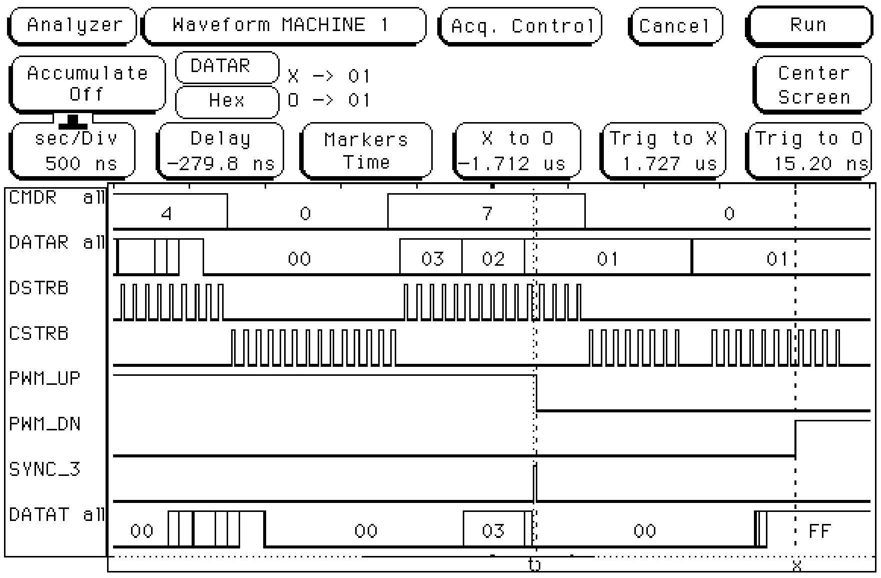


Fig. 5-4. Measurement of synchronization delay at each node with compensation.

5.1.2 Modulation

The synchronization packet is broadcast once in a switching cycle to re-synchronize all the PWM generators. Upon receiving a synchronization command every node generates its local SYNC signal, as shown in Fig. 5-4. This signal is used as a start signal for the PWM generator.



```
Accumulate Off
DATAR  X -> 01
Hex    O -> 01
sec/Div 500 ns
Delay -279.8 ns
X to O -1.712 us
Trig to X 1.727 us
Trig to O 15.20 ns
```

Fig. 5-5. PWM and synchronization signals measured on a phase leg.

In order to avoid a shoot-through between the top and the bottom switches a dead time is added. When the synchronization signal (SYNC_3) is inserted (as shown in Fig. 5-5), the gate drive signal that was high changes the state to low (PWM_UP) immediately while the rising edge of complementary switch gate drive signal (PWM_DN) is delayed for a dead time. Dead time is implemented as variable counter. In the given example dead time was set to 1.7 us as the measurement in Fig. 5-5 shows.

5.1.3 Closed Loop Waveforms

As a final test, a three-phase VSI was tested with closed current loops in a DQ coordinate frames. Detailed control design procedures and implementation of VSI control in DQ coordinates can be found in [26]. For the space vector modulation, a sixty degrees clamped (highest-current not switching) algorithm was used. Since the pulse generators are implemented on a hardware manager the pulse alignment (left-aligned, central-aligned or right aligned) depends on the hardware manager implementation. In this design we used left-aligned scheme which tends to exhibit low-frequency distortions on a sixty-degree sector crossings as shown in Fig. 5-6. Center aligned modulation has been shown that can alleviate those problems.

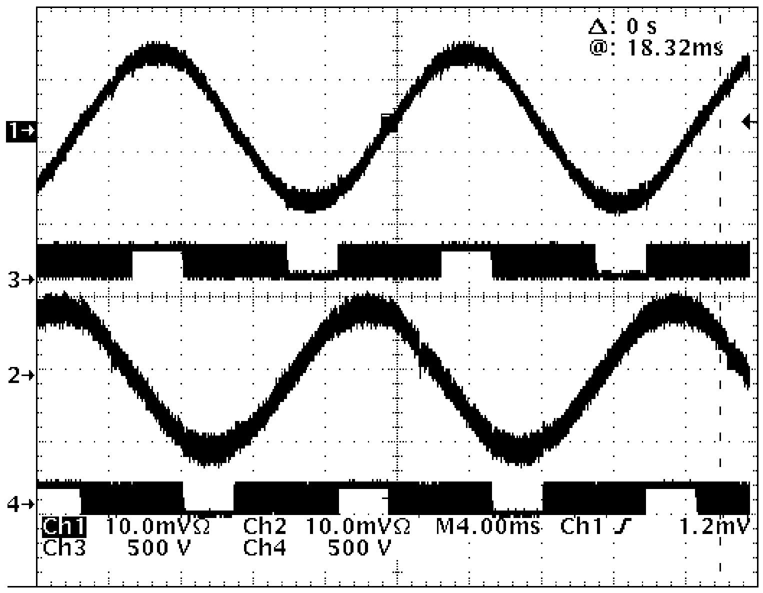


Fig. 5-6. Output phase currents (50 A/div) and output phase voltages (500 V/div) PWM waveforms.

The three-phase VSI was tested under different DC bus voltages and under different loading conditions. Phase currents and PWM phase-leg output voltages are shown in Fig. 5-6. Low harmonic distortion and stable converter operation over the wide range of operating conditions show the robustness of this design.

Chapter 6 Future Work on Distributed Controllers and Possible Improvements

Work in this thesis was mostly aimed at the design and implementation of a distributed controller on the converter level. However, a converter is rarely operated by itself; most often a power converter is part of a large-scale control and automation system.

For example, on a DC distribution system for a ship a large number of DC/DC, AC/DC, DC/AC and AC/AC converters have to operate in parallel. In order to avoid undesirable system interactions and to maintain system stability, close coordination between converters is needed. Fault conditions and exception handling on a system level requires a system level control too. This can be achieved by employing a system level communication network that will control the system on the top level and connect all of the lower level objects (converters, drives, sensors, actuators etc.) in a coherent control structure. A typical block diagram of such hierarchical control architecture is given in Fig. 6-1. The distributed controller on a converter level is the one described previously in this work providing converter control (fast time scale), while all the converters and actuators are linked in a system-level distributed controller.

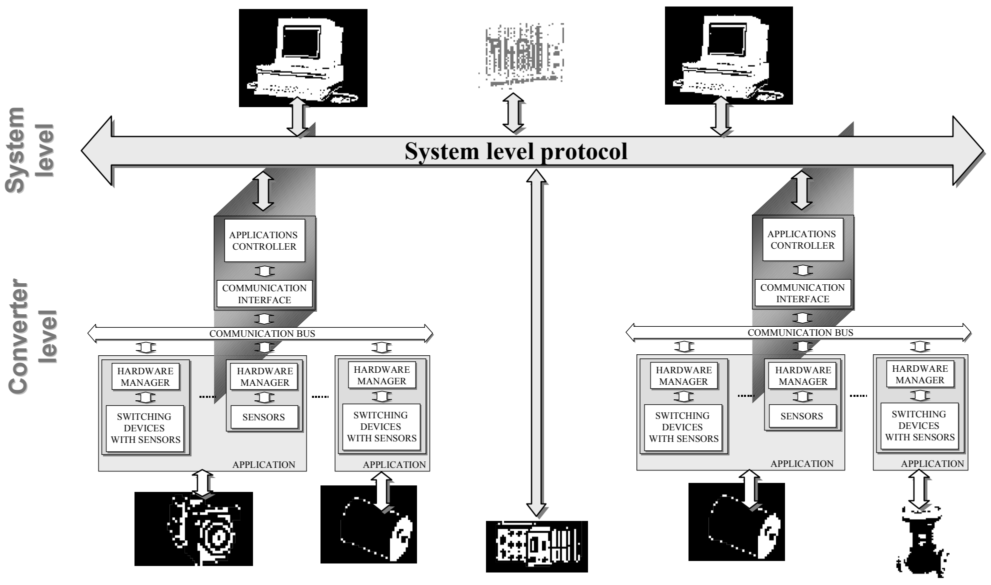


Fig. 6-1. Block diagram of overall system level control (e.g. DC distribution system on a ship).

6.1 Choosing the Right High-Level Protocol

As shown in Chapter 2, system-level control authority begins after the load controller (or what we have designated as I_5 interface). According to the estimate given in Fig. 2-14. for the I_5 interface capacity, a necessary communication channel capacity can be estimated if more then one converter is connected into the system level communication network including the necessary communication overhead. Fig. 6-2. shows the estimated bus cycle time for different communication speeds under different network loading conditions. This graph refers to real time communication requirements for the purpose of real time control.

Another very important aspect of a system level communication network is non-real-time data traffic. Non real time data traffic include downloading and setting up processes, initializing converters, monitoring and data logging, terminating applications and many others.

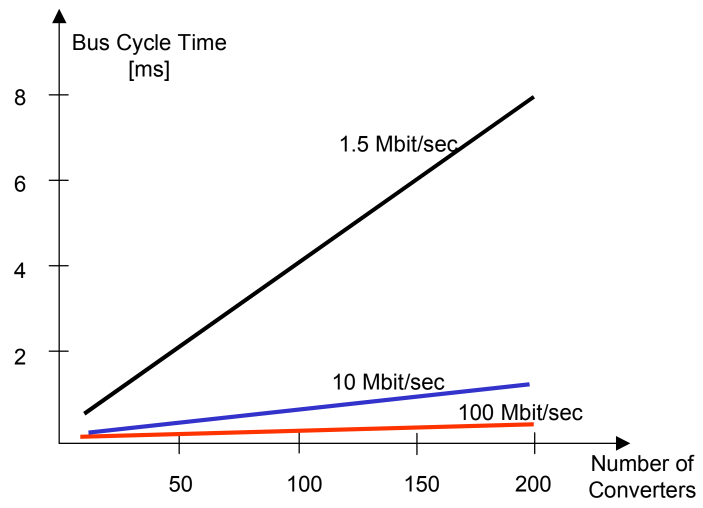


Fig. 6-2. System level communication network bus cycle time for different baud rates and different numbers of converters on a network.

It is very important for high-level protocol to be able to handle non-real time tasks. Non-real time tasks include communication of longer messages that have complex data structure but are not restricted with time constraints. This includes communicating initialization data, code downloading and applications setup. This is usually achieved with some of the higher-level communication protocols which support non-real time tasks. Some of the industry recognized high level protocols are: MMS (Manufacturing Message Specification [30]) or FMS (Fieldbus Message Specification [31]) services which are a subset of MMS (manufacturing message specification, ISO 9506). These higher level protocols provide the tools for handling larger data structures.

Another important aspect of communication protocol for distributed control applications is synchronization. A rule of thumb, it can be accepted that the synchronization jitter should be less then one tenth of a fastest time constant that is governing controlled process. For converter level network where fastest time constants were in order of microseconds synchronization jitter was not to exceed 100 ns. For the system-level controller time constants are in order of hundreds of microseconds therefore a synchronization jitter that is less than 10 us can be accepted.

And finally, the cost of control network should be minimized. The only way the costs can be reduced is by increasing the production volume. Since the power electronics is a small field that can hardly drive the prices down, one possible solution would be to leverage some of the already available solutions widely accepted in other fields. Therefore using the ethernet technology as the communication vehicle (low-level protocol) for system level control network would minimize the infrastructure costs. At the same time this would allow for seamless integration of system level control in wider internet information structure.

Considering all of the above mentioned requirements for the system level control network in power electronics and considering features of available industrial automation protocols (chapter Chapter 3), PROFIBUS emerges as one of the most suitable solutions. PROFIBUS has all of the required features summarized as:

• Synchronization jitter:	less then 1us;
• Transmission speed:	up to 12 Mbits/sec;
• Medium access:	master slave type with logical token ring;
• Communication allocation:	supports both real time and non real-time communications;
• Transmission medium:	RS 485, IEC 1158-2, fiber optic.

Apart from widespread application of PROFIBUS for industrial automation and its acceptance as a standard the latest innovations that are concerning development of transparent coupling between PROFIBUS and Ethernet are putting it into a leading position in the field. Features such as mapping of engineering services from PROFIBUS to TCP/IP, direct routing from TCP/IP to PROFIBUS and connecting complex field devices directly through Ethernet can provide a new dimension to system level control, monitoring and information flow integration. The emerging possibilities for power electronics systems integrated in a global communication network are countless. If the momentum the information technology (IT) has gained over the past few years can be leveraged in the field of power electronics significant gains and improvements could be induced.

6.2 Power Electronics Communication Protocol Improvements

Communication protocol for power electronics systems proposed in [11] and explained in Section 3.3 was designed as a master-slave daisy chained type of network. It is designed as single master (applications manager) multiple slaves (hardware managers) system. Furthermore in order to meet stringent real-time requirements we have adopted a fixed communication packet size. Also the network addressing is fixed and each node has a hard-wired address.

In order to provide a system with more flexibility we are proposing several major communication protocol improvements that can be summarized as:

- Variable packet length,
- Multiple master system with token,
- Dynamic address assignment,
- Improved synchronization scheme.

6.2.1 Variable Packet Length Protocol

Instead of the data packet format shown in Fig. 6-3. where the packet length was fixed and predetermined we are proposing a variable packet length that very much corresponds to the FDDI data packet format.

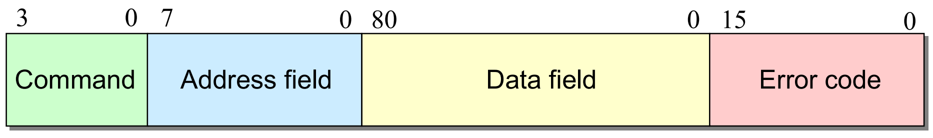


Fig. 6-3. Data packet format with fixed packet size.

Previously proposed and implemented data packet format with fixed size is shown in Fig. 6-3. Command indicates the data type and flags the beginning of the data packet while address field identifies the node address (note here it is assumed single master system). Data field is fixed length and is followed by sixteen bits error check code.

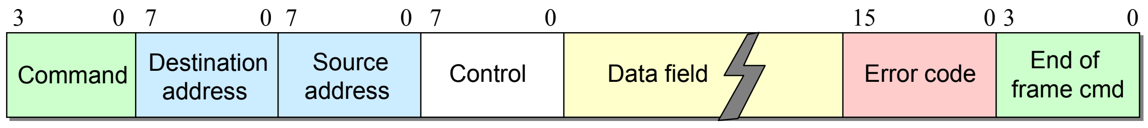


Fig. 6-4. Data packet format with variable packet size.

The data packet that we are proposing as the logical upgrade of the previously explained one is shown in Fig. 6-4. Beginning and the end of the frame are designated with special commands. After the beginning designator is the destination address that is followed by the source address that allows multiple master system. Control byte is followed by the data field which can be of a variable length. Error code contains the error check information and is most often realized as CRC error check.
The approach with variable packet length allows more flexibility and provides tools for more complex communication tasks.

6.2.2 Medium Access Protocol-Timed Token Algorithms

When multiple masters are connected in a ring type of network a medium access mechanism has to be developed. This mechanism has to provide algorithm that controls when each node is allowed to transmit. These algorithms have been studied extensively [32] and are known as Timed-Token Algorithms. Each master node is able to transmit data upon receiving a token. When master node has finished transmission it forwards the token to next master station which then holds it for certain time and passes it forward. This mechanism allows for setting the boundaries on token rotation time thus guaranteeing the timeliness of communicated data.

6.2.3 Improved Synchronization Scheme

Each node in the network introduces a delay into the data propagation path as shown in Fig. 5-3. If we send a synchronization telegram through the network, each node is going to receive it delayed for the time T_d as shown in Fig. 6-5. If the number of the nodes in the network is large synchronization delay between distant nodes might become significant which could lead to performance degradation of a converter and eventually to fatal failure. This is due to the fact that PWM generators if not synchronized will implement a wrong switching state for some amount of time.

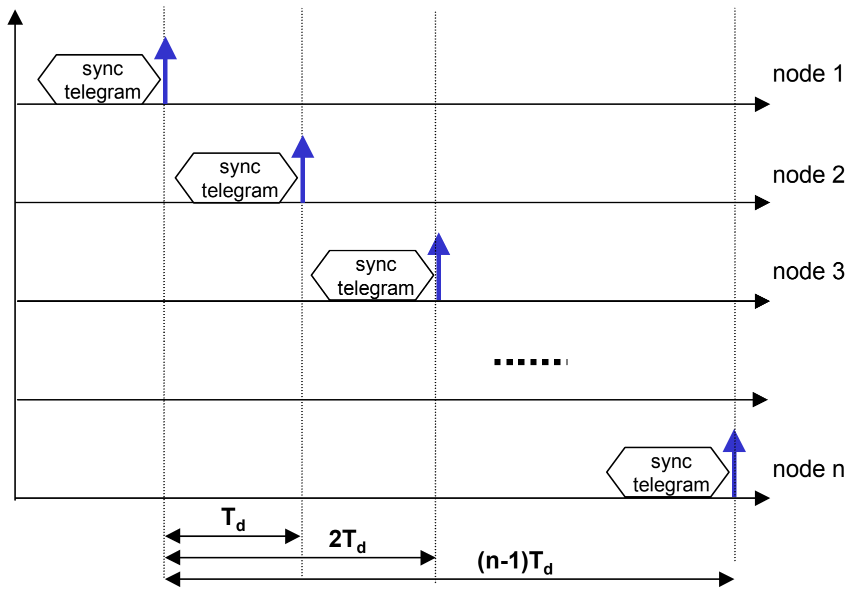


Fig. 6-5. Cumulative effect of synchronization delay in ring type of networks.

One way to alleviate this problem would be to add a digital delay circuits in each node. If each node can delay generation of the synchronization impulse, with reference to the synchronization telegram arrival, for

$$T_{di} = (n-i) \cdot T_d$$

where n is the total number of nodes and i is the relative address of the node, T_d is the node propagation delay than all the network nodes could be synchronized. This simple delay algorithm is shown in Fig. 6-6. The logical question at this point is how do the network node know its relative position in the network and how much delay should it implement. Since the network has to be configured prior to its operation and all the nodes will be assigned a dynamic address that information can also be passed to each node to program a delay. Indeed, master node is responsible for system configuration and it can pass the information to each node how much delay should be introduced upon arrival of the synchronization telegram.

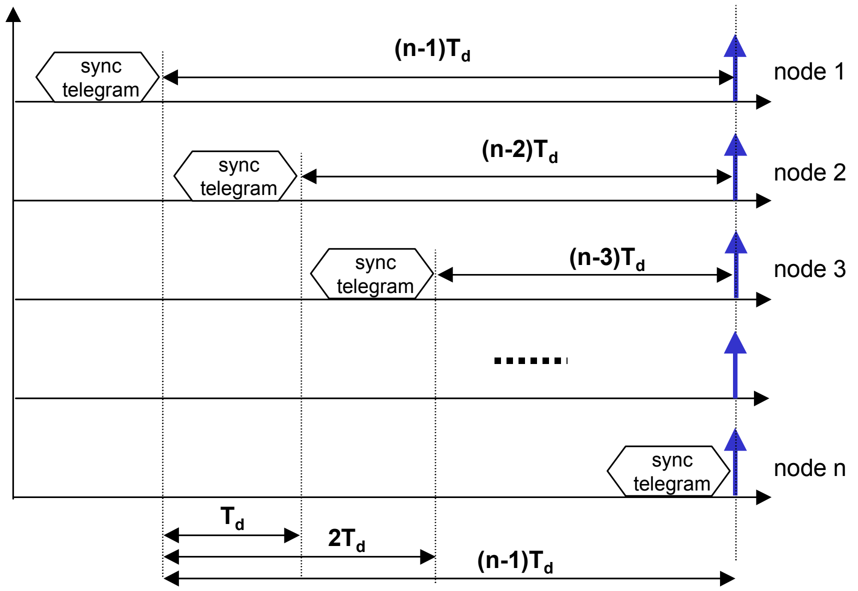


Fig. 6-6. Timing diagram for network synchronization with delay compensation algorithm.

The delay circuit can be implemented as simple counter that is clocked with the bus clock. This would reduce synchronization jitter between nodes to typically one half of the bus clock. The advantage of this method over previously proposed one [11] is that in this method a synchronization telegram length is typically 4 bits compared to the [11] where synchronization telegram is linearly dependent on the number of network nodes as shown in Fig. 6-7. Implementation of this method can increase the available bus time for communication of useful data rather then wasting it on the synchronization telegram.

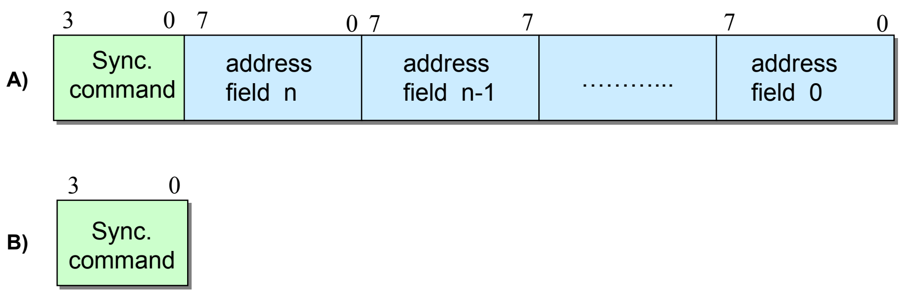


Fig. 6-7. Comparison of synchronization telegram lengths (old method [11] and new method)

Another important feature of this scheme that each node has a self-running clock that is just network re-synchronized. So even if the network synchronization telegram is not broadcasted every switching cycle local clock will be able to run PWM modulator and all switching related functions. When next synchronization telegram arrives it will re adjust the local clock without any interruption.

6.3 Moving Towards Plug and Play Power Electronics Systems

Modularization of the power converter structure together with distributed control approach is a significant leap towards future simple to use, flexible and modular power electronics systems. But, standardization and modularization of hardware is not sufficient unless software modularization and software portability is achieved in parallel. In order to achieve Plug and Play system design, standardization and modularization on a software level (embedded control) is necessary.

In order to provide a modular design on a software level operating system for power electronics systems is needed. An additional software layer that is going to provide transparent environment for control application regardless of the hardware platform would simplify integration of control algorithms in real control hardware. Operating system kernel would be able to provide all the services to the control algorithm in a uniform way regardless of the hardware platform thus fully decoupling control algorithm domain from the hardware specifics. All the hardware specifics would reside in a form of drivers that operating system would invoke every time some hardware operation are executed.

Another benefit that modularization and plug and play approach will eventually bring is integration of control design process. Instead of separately simulating the system then using different tools to program the control algorithm then using another tool for compiling and emulating all those operations could be automated within single design environment thus reducing design cycle enormously. Design and simulation tasks that are done in MATLAB/SIMULINK graphical tool could be directly compiled and linked with operating system kernel and necessary hardware drivers and immediately downloaded into the digital controller platform. This design process can bypass multi-step design and debugging process. If at the same time we use the distributed controller and power electronics building blocks then the design process could be accelerated many times.

Chapter 7 Conclusion

This thesis presented a novel approach to power electronics system design based on the open architecture distributed digital controller and modular power electronics building blocks. The most important features of the new concept are as follows:

- open and flexible communication protocol for distributed control of power electronics systems;
- flexible and easy-to-use power converter modules;
- hardware-independent, applications manager capable of controlling several different types of converters in parallel; and
- simple system integration and re-configuration.

The designed prototype of distributed controller for the three-phase VSI has shown the feasibility of the approach. Prototype system shows that successful design and satisfactory performance can easily be achieved. Furthermore, extreme flexibility of the system lends itself to a reusable and object oriented design.

Also, the concept of the distributed controller can easily be extended to almost any type of medium- and high-power converter regardless of the converters distribution, control complexity and type of application.

Besides, we believe that the distributed digital controller together with open-system communication protocol and modular power electronics building blocks provides solid ground for application of object-oriented software design in power electronics. This approach provides the tools for easier integration of higher-level, graphically oriented design and simulation tools with real control of power electronics hardware.

These trends are driving power electronics towards system-oriented design and thinking. New tools and systems will allow for a shorter development cycle, better system performance and improved reliability, thus providing power electronics solutions for a large number of power electronics-free applications and systems. An increased number of applications can provide greater volumes which will eventually drive the costs further down and provide the additional increase in production. This positive economic feedback is what power electronics has been missing and what might eventually revolutionize the field.

References

[1] J. D. Van Wyk and F.C. Lee, “Power Electronics Technology at the Dawn of the New Millennium-Status and Future,” *IEEE PESC Conference Proceedings,* 1999, vol. 1, pp. 3-12,

[2] A. A. Jaecklin, “Future Devices and Modules for Power Electronic Applications,” *European Conference on Power Electronics and Applications, EPE,* September 1999.

[3] A. A. Jaecklin, ‘Integration of Power Components – State of the Art and Trends,” *European Conference on Power Electronics and Applications, EPE*, September 1997.

[4] L. Lorenz, “System Integration – a New Milestone for Future Power Electronic Systems,” *European Conference on Power Electronics and Applications, EPE*, September 1997.

[5] V. Vlatkovic and D. Borojevic, “Digital-Signal-Processor-Based Control of Three-Phase Space Vector Modulated Converters,” *IEEE Transactions on Industrial Electronics,* June 1994, Vol. 41, No. 3, pp. 326-332.

[6] T. Ericsen and A. Tucker, “Power Electronics Building Blocks and Potential Power Modulator Applications,” *IEEE Conference Record of the Twenty-Third International Power Modulator Symposium*, New York, NY, 1998, pp. 12-15.

[7] T. Ericsen, A. Tucker, D. Hamilton, G. Campisi, C. Whitcomb, J. Borraccini, W. Jacobsen, “Standardized Power Switch System Modules (Power Electronics Building Blocks),” *Proceedings of the PCIM ’97 Power Electronics* Conference, September 9-13, 1997, pp. PEO3.5-1 to PEO3.5-21.

[8] T. Ericsen, "Open Plug and Play Power Architectures Simplifies System Integration," *PCIM Magazine*, pp. 70-71, February 2000.

[9] T. Ericsen, "Control Architectures and Distributed Intelligence for Plug and Play Power Blocks," *PCIM Magazine*, pp. 38-41, March 2000.

[10] I. Celanovic, I. Milosavljevic, D. Boroyevich, G. Jinhong and R. Cooley, "A New Distributed Controller for the Next Generation Power Electronics Building Blocks," *IEEE APEC Proceedings,* February 2000, pp. 889-894.

[11] I. Milosavljevic, "Power Electronics System Communications," Thesis, Virginia Tech, 1999,

[12] I. Milosavljevic, Z. Ye, D. Borojevic and C. Holton, "Analysis of Converter Operation with a Phase-Leg in Daisy–Chained or Ring Type Structure," *IEEE PESC Conference Proceedings*, June 1999.

[13] J.A. Du Toit, A.D. Le Roux and J. H. R. Enslin, "An Integrated Controller Module for Distributed Control of Power Electronics," *IEEE APEC'98 Proceedings*, February 1998, pp. 874-880.

[14] P.L.G. Malapelle, G. Torri, R. Moruzzi, A. Oliva, "A New, Modular, Programmable, High Speed Digital Control for Large Drives," *IEEE IECON 20th International Conference on Industrial Electronics, Control and Instrumentation,* 1994; vol.1, pp. 210-214.

[15] I. Milosavljevic, D. Borojevic and I. Celanovic, "Modularized Communication and Control Structure for Power Converters," *8th European Conference on Power Electronics and Applications, EPE,* September 1999.

[16] E. Bassi, F. Benzi, L. Lusetti, G.S. Buja, "Communication Protocols for Electrical Drives," *IEEE IECON Proceedings, 1995*, Vol. 2, pp.706-711.

[17] "TAXIchip™ Integrated Circuits," *Data Sheet and Technical Manual, Publication#07370*, AMD, April 1994.

[18] "Plastic Optical Fiber and HCS® Fiber Cable and Connectors for Versatile Link," *Technical Data 5963-3711E,* Hewlett Packard, November 1994.

[19] Jeffrey J. P. Tsai, Yaodong Bi, Steve J. H. Yang and Ross A. W. Smith, "*Distributed Real-Time Systems*," New York: John Wiley & Sons, Inc.

[20] "FLEX 10K-Embedded Programmable Logic Family Data Sheet," San Jose, California: Altera Corporation, October 1998, ver. 3.13.

[21] J. Wu, H. Dai, K. Xing, F. C. Lee, D. Boroyevich, "Implementation of a ZCT Soft Switching Technique in a 100 kW PEBB Based Three-Phase PFC Rectifier," *IEEE PESC Conference Proceedings*, June 1999, pp.647-658.

[22] J. T. Carlo, R. D. Love, M. S. Siegel, K. T. Wilson, "*Understanding Token Ring*," Artech House, 1998.

[23] R. Jain, "*FDDI Handbook, High-Speed Networking using Fiber and Other Media,*" Addison-Wesley Publishing Company, 1994.

[24] D. MacKinnon, W. McCrum, D. Shepard "*An Introduction to Open System Interconnection,*" W. H. Freeman and Company, 1990.

[25] F. Seitz and N. G. Einspruch, "*Electronic Genie: The Tangled History of Silicon,*" Urbana and Chicago: University of Illinois Press, 1998.

[26] S. Hiti, "Modeling and Control of Three-Phase PWM Converters," Ph.D. Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, 1995.

[27] Williams, Ross, "*A painless Guide to CRC Error Detection Algorithms,*" Hazelwood Park, Australia: Rocksoft PTY Ltd. 1996. (http://www.rocksoft.com), ver. 3.

[28] Kun Xing, "Modeling, Analysis, and Design of Distributed Power Electronics System Based on Building Block Concept," Ph.D. Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, 1999.

[29] Henry W. Ott, "*Noise Reduction Techniques in Electronic Systems,*" New York: John Wiley & Sons, Inc. 1988.

[30] "WorldFIP Basics," http://www.worldfip.org.

[31] "Profibus Technical Description," http://www.profibus.com, September 1999.

[32] "MACRO: Motion And Control Ring Optical," http://www.macro.org, May 1998.

[32] Larry Peterson and Bruce Davie, "*Computer Networks: A System Approach*," San Francisco, California: Morgan Kaufman Publishers, Inc. 1996.

[33] "Fiber-Optic Solutions for 125 MBd Data communication Application at Copper Wire Prices," Application Note 1066, Hewlett Packard, 1995.

[34] “Simultaneous Sampling Dual 250 kSPS 12-bit ADC - AD7862,” Data Sheet, Analog Devices, Inc., 1996.

[35] “Motorola MC33153 Gate Driver,” Data Sheet, Motorola 1999.

[36] “LEM Model LA 205-S,” Data Sheet, LEM U.S.A., Inc, 1999.

[38] “LEM Model LV 25-P,” Data Sheet, LEM U.S.A., Inc, 1999.

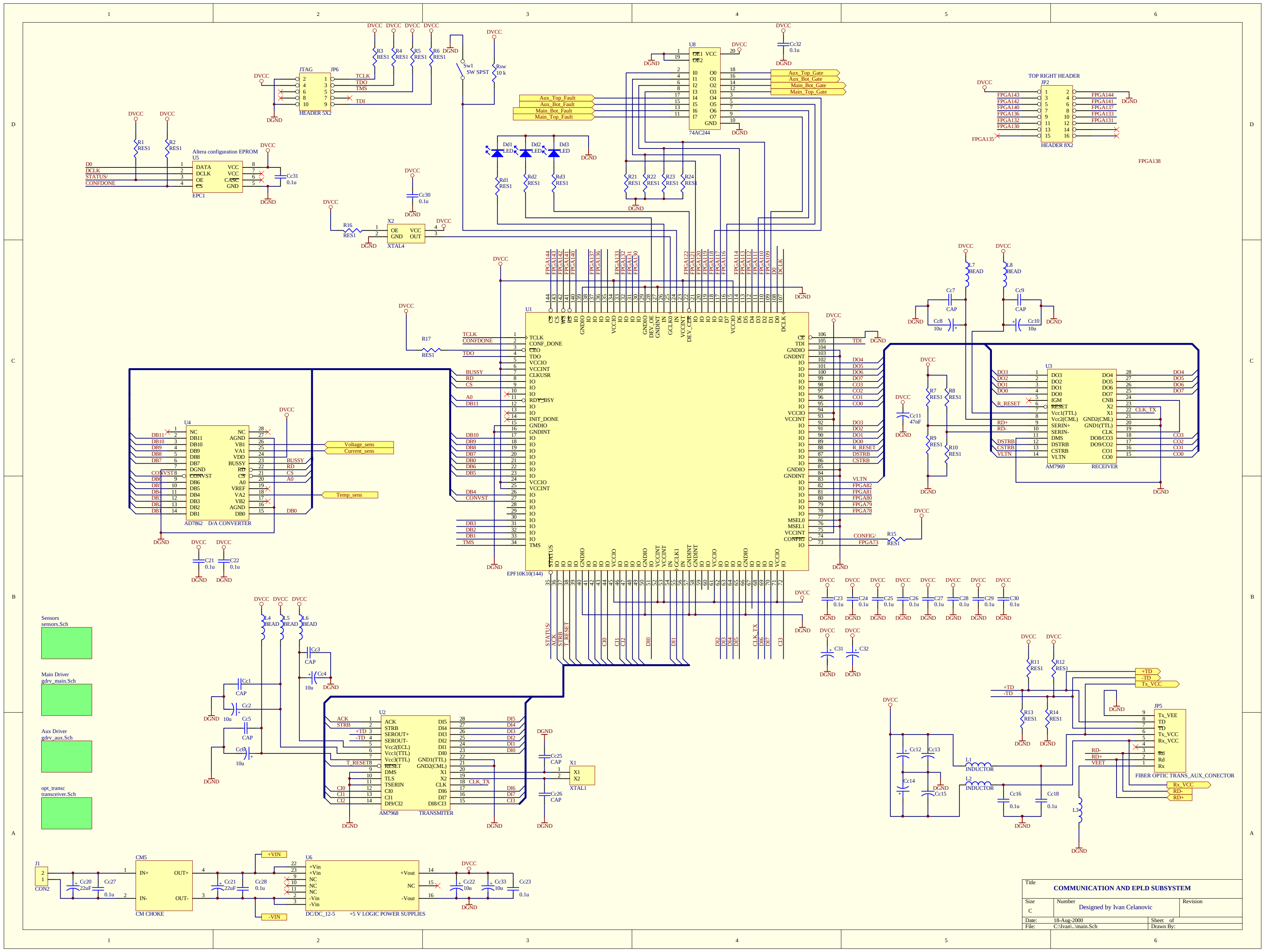
[39] A. Crane and T. J. McCoy, “Electromagnetic Compatibility Design for a 19 MW PWM Motor Drive.”

[40] “Sharc ADSP-21062 EZ-LAB Development System Manual,” Hardware Rev. 3, BittWare Research Systems, Inc. 1996

[41] V. H. Prasad, S. Dubovsky, N. Celanovic, R. Zhang and D. Borojevic, “DSP Based Implementation of a Power Electronics Control System,” *Proceedings of VPEC Seminar*, Blacksburg, Virginia, 1995.

Appendix A

Schematic diagrams of phase-leg based hardware manager.



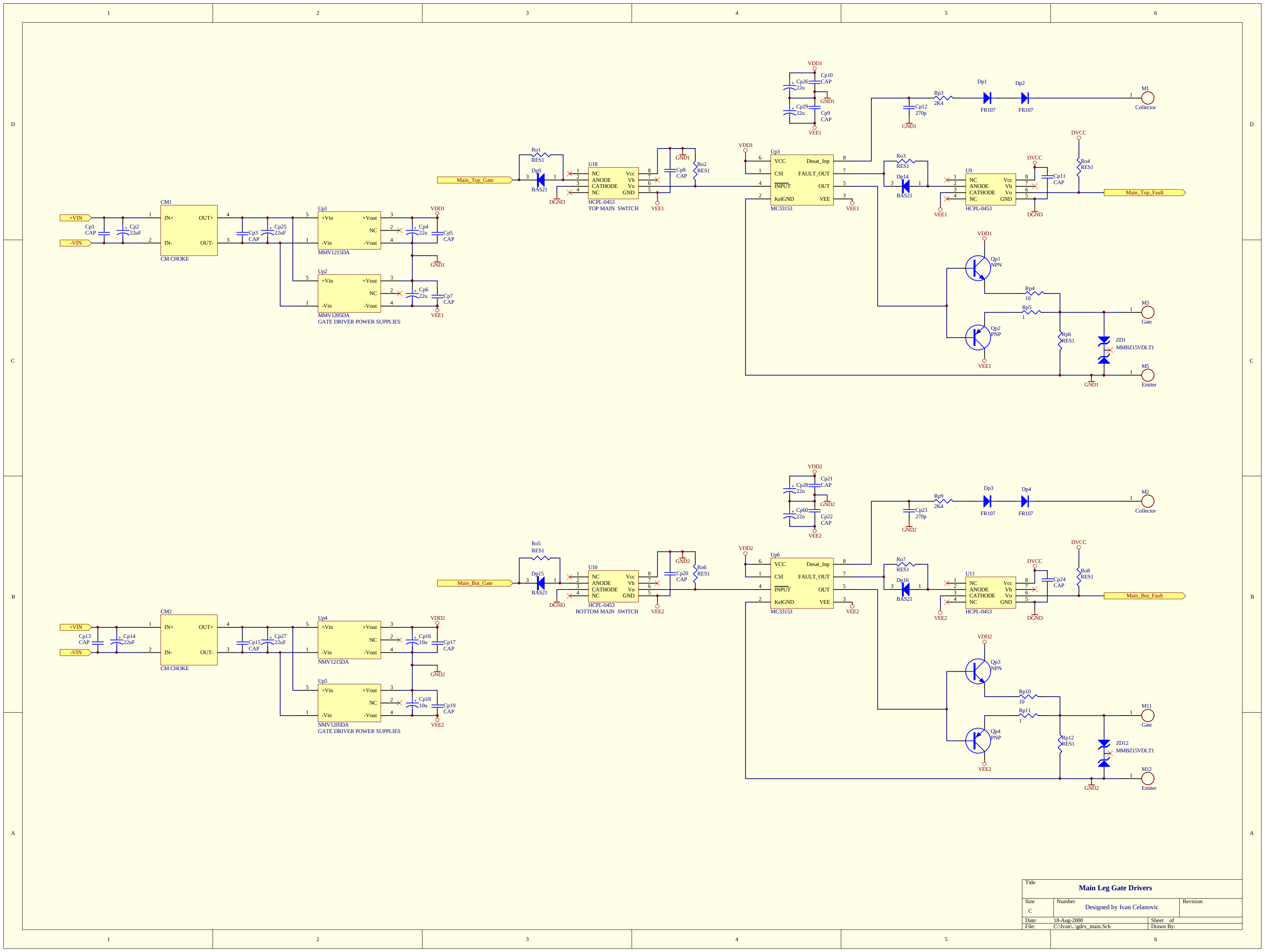
Title

COMMUNICATION AND EPLD SUBSYSTEM

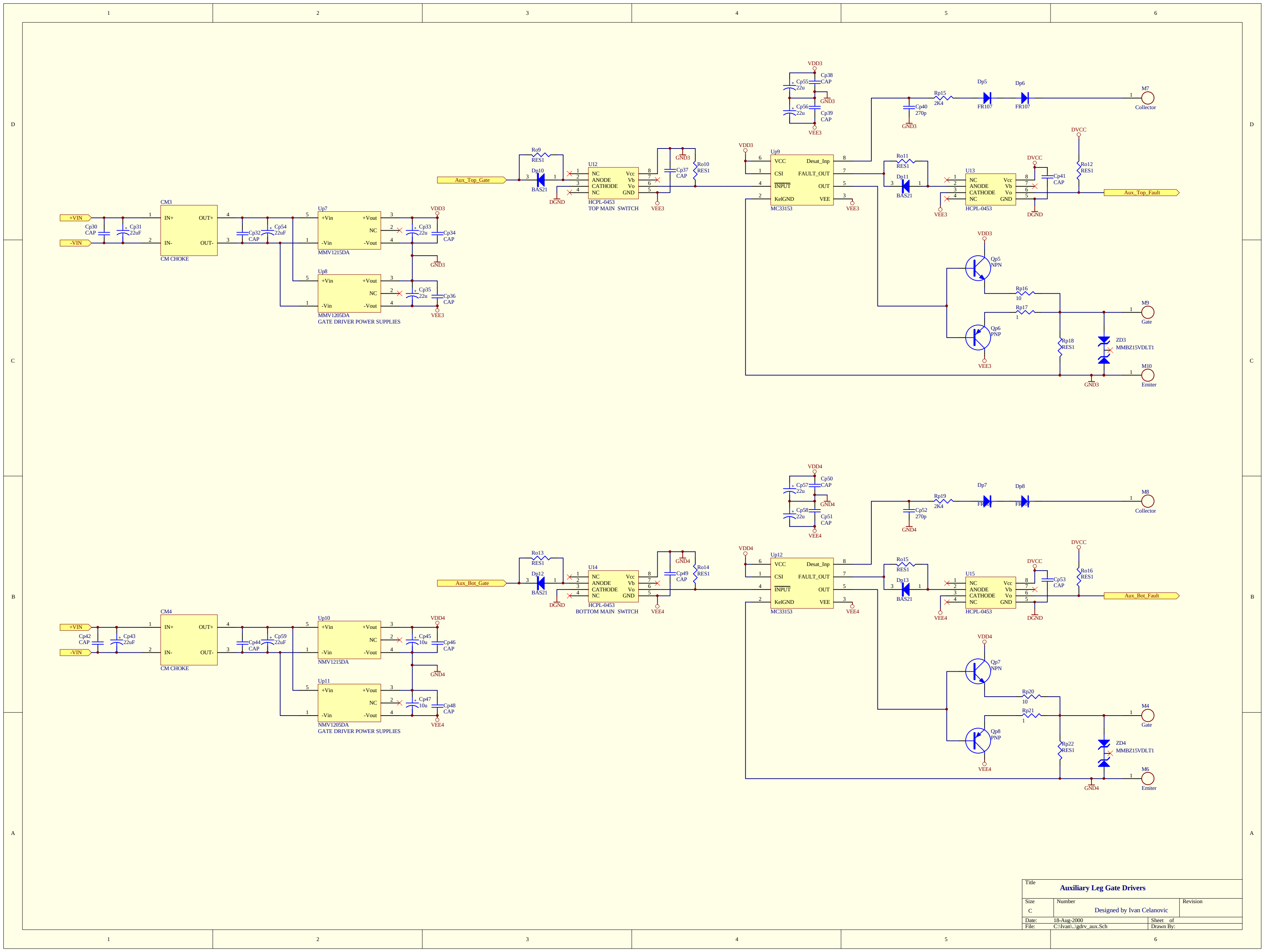
Size	Number	Revision
C	Designed by Ivan Celanovic	

Date: 18-Aug-2000 Sheet of

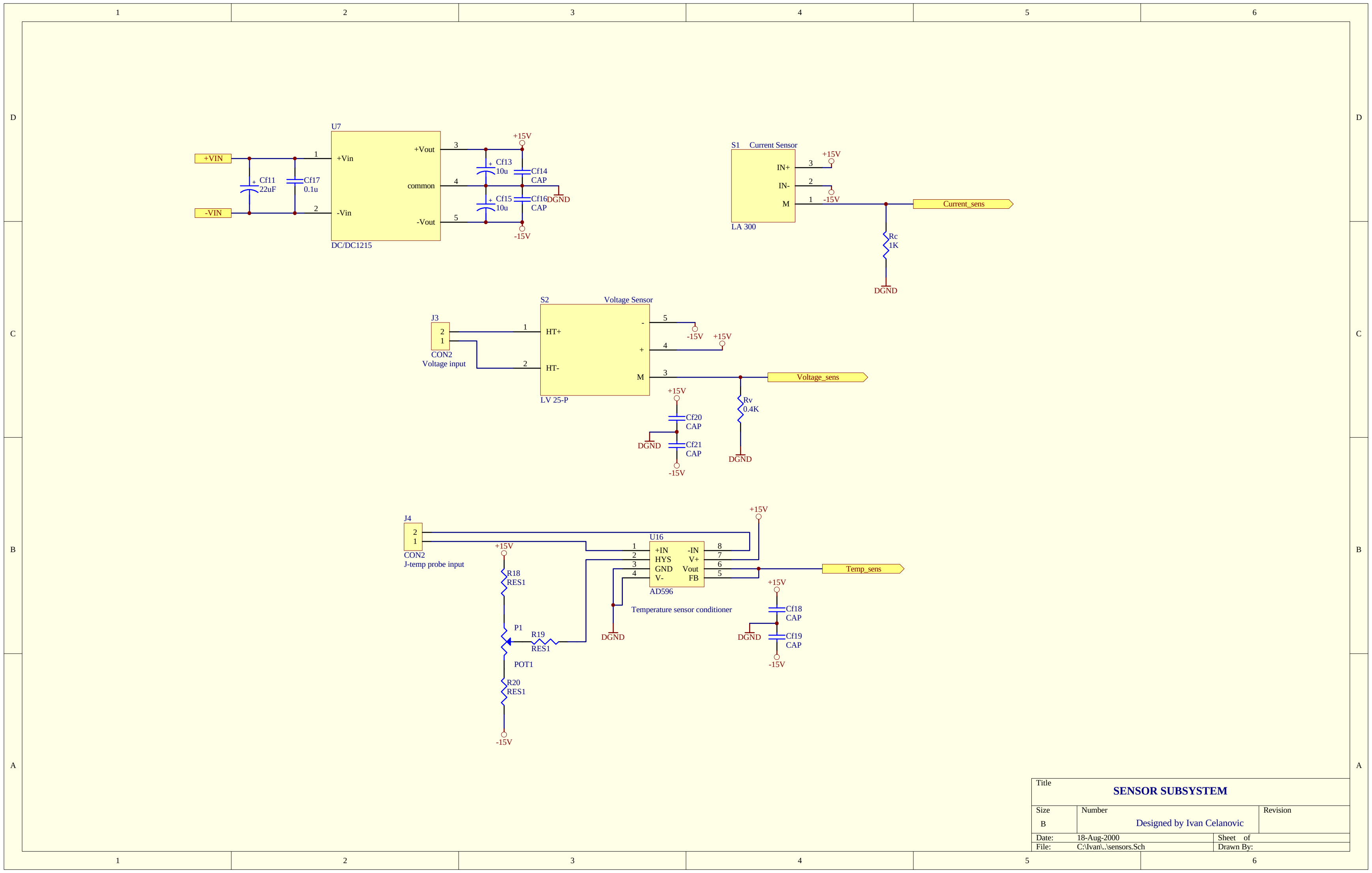
File: C:\Ivan\..\main.Sch Drawn By:



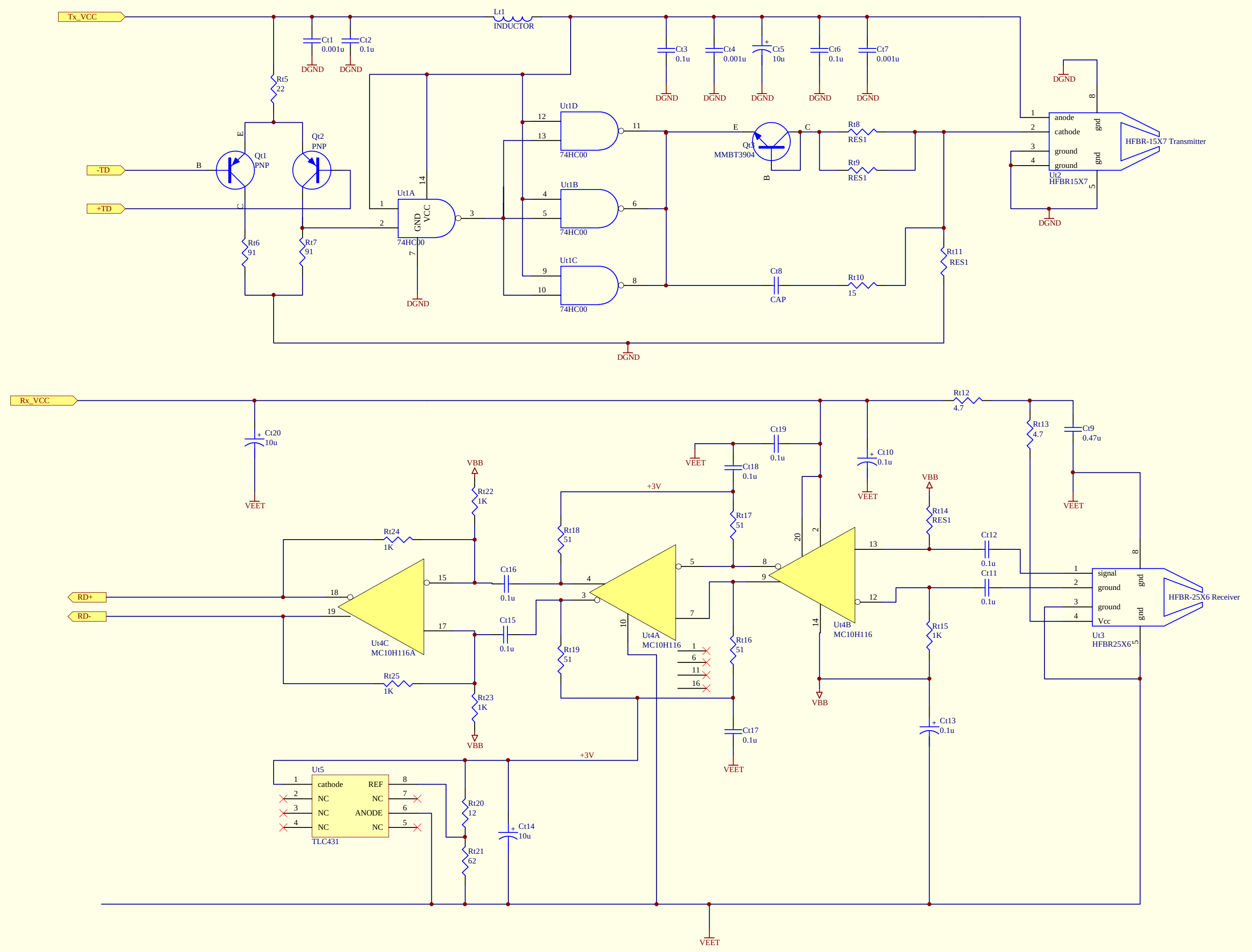
Title	Main Leg Gate Drivers	
Size C	Number: Designed by Ivan Celanovic	Revision
Date: 18-Aug-2000		Sheet of
File: C:\Ivan\..\gdrv_main.Sch		Drawn By:



Title	Auxiliary Leg Gate Drivers	
Size: C	Number: Designed by Ivan Celanovic	Revision
Date: 18-Aug-2000	Sheet of	
File: C:\Ivan\..\gdrv_aux.Sch	Drawn By:	



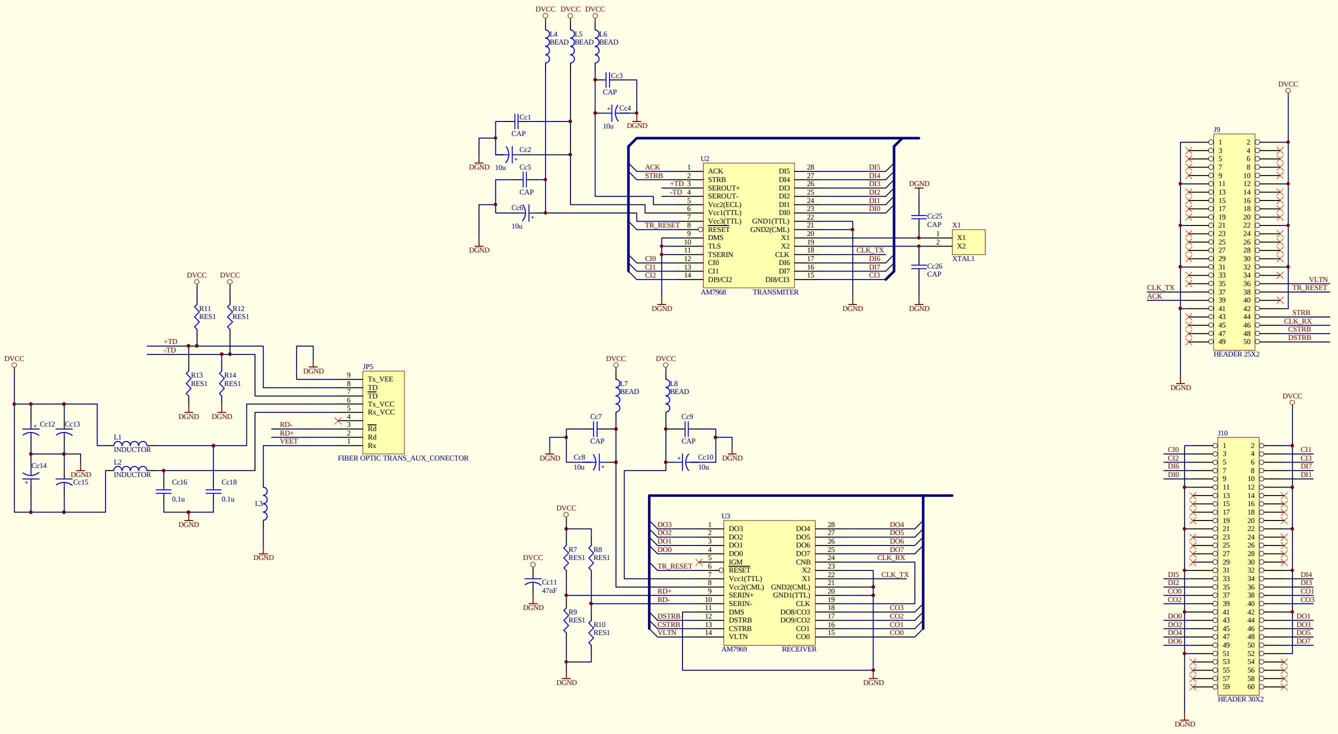
Title	**SENSOR SUBSYSTEM**	
Size B	Number: Designed by Ivan Celanovic	Revision
Date: 18-Aug-2000		Sheet of
File: C:\Ivan\..\sensors.Sch		Drawn By:



Title	HP OPTICAL TRANSCEIVER		
Size B	Number	Designed by Ivan Celanovic	Revision
Date:	18-Aug-2000	Sheet of	
File:	C:\Ivan\..\transceiver.Sch	Drawn By:	

Appendix B

Schematic diagrams of communication interface board for application manager.



Title	COMMUNICATION BOARD FOR INTERFACE WITH SHARCPACK	
Size C	Number	Revision
Date: 25-Aug-2000	Sheet of	
File: C:\Ivan_Celanovic_CPES_Aug_2000\..\comm.Sch	Drawn By:	

Vita

The author was born in Novi Sad, Yugoslavia, in 1973. He received the B.S. from the University of Novi Sad, Yugoslavia in 1998 in electrical engineering.

From January 1999 to September 2000 he was employed as a Research Assistant with Center for Power Electronics System (CPES), Virginia Polytechnic Institute and State University.

His research interests include modeling and control of power converters, digital control and communications in power electronics and DSP.

In fall 2000 the author will begin his Ph.D. studies at Massachusetts Institute of Technology.