MICROCOMPUTER BASED DEVELOPMENT TOOL FOR AIDING THE IMPLEMENTATION OF POWER ELECTRONIC CONVERTER PROTOTYPES

Sérgio Daher, Fernando Antunes, Edilson Mineiro and Cícero Tavares.
Departamento de Engenharia Elétrica
Universidade Federal do Ceará
Caixa Postal 6001 - 60455-760 - Campus do Pici - Fortaleza - Ceará - Brazil
sdaher@secrel.com.br, fantunes@dee.ufc.br, edilson@dee.ufc.br, cicero@dee.ufc.br

Abstract – This paper describes a development tool capable to implement digital controllers. The present system is based on a personal microcomputer (PC) and can run complex controller algorithms at speeds up to 40kHz. Isolated interface circuits for acquiring analog signals and isolated output MOS drives guarantee high flexibility and safety operation for the implementation of power converter controllers. Discussion about basic operation principles is done and the implemented circuits are presented.

KEYWORDS

Development tool, control system, power electronics, personal microcomputer, converters.

I. INTRODUCTION

Nowadays time is precious and it is important to have efficiently development tools. After the conception of a power electronic circuit by the human intelligence, simulation tools have been used as a second step of a converter design process [1]. Then the third step is the practical implementation of the prototype. Although the power circuits can not be analyzed without take into account the control strategy, at the moment of implementation there are two distinct tasks: implementation of power circuits and implementation of drivers and controller.

If one is concentrated in the design of a new converter topology, it does not matter what technology will be used to implement the controller of the prototype or the power switch drivers. To get a commercial product from a prototype is another task (in this case, other factors, as cost and size, are important to be considered).

Also when complex control strategies are necessary or when the control circuits must incorporate special features, a microcontroller or a DSP can be used as the main controller instead of analog implementations. In general, to develop and debug complex control strategies are a hard task.

For example, the proposed system is suitable to aid practical implementations of multilevel converters in which a large number of isolated gate drive signals must be generated [2], [3].

Other typical applications are the development of digital controllers and modern complex controllers [4], [5], [6]. In these cases, such digital and/or complex control algorithms can be implemented more faster and easily in a PC then in a dedicated hardware.

In close, if isolated switch drivers are ready for use and the controller can be implemented safely as a software running in a common PC, then the designer has a friendly tool, easy to program and debug. In addition, all control signals and some measurements can be acquired and stored in the PC memory, so detailed analysis can be carried out later. Therefore development time can be reduced.

II. GENERAL DESCRIPTION

Fig. 1 shows a block diagram of the proposed system. As can be seen, the entire system if composed by the following main components:

a) one IBM/PC compatible microcomputer;
b) one microcontroller;
c) one Programmable Logic Device (PLD);
d) isolated analog interfaces;
e) isolated MOS drivers;
f) power suppliers (isolated and non-isolated);
g) one press button (stop button).

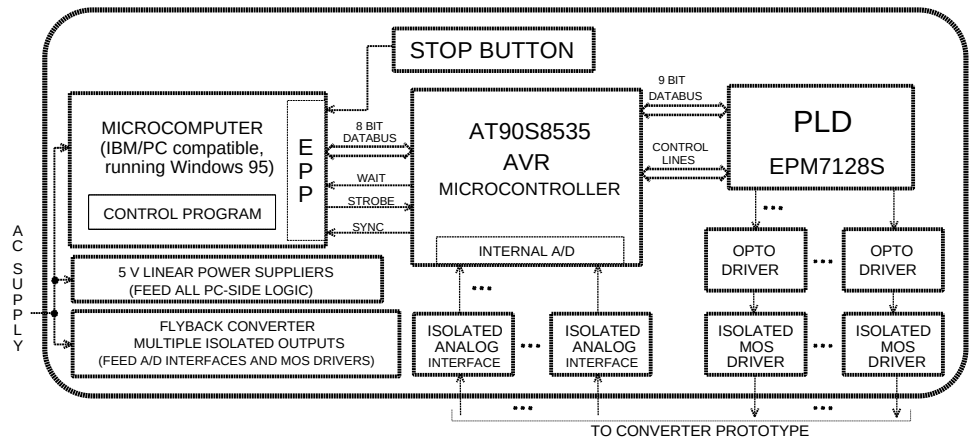

Figure 1 - Block Diagram of the Complete System.

II.a. *Microcomputer*

The PC is the heart of the system and it is connected to the microcontroller thought its parallel port. It has been used a general purpose IBM/PC compatible personal microcomputer running on Windows'95. Notice that even an "old" computer (for example, 100 MHz Pentium microprocessor based PC) can be used as it is able to run a great number of control algorithms faster enough. In general, I/O operations for communication between the PC and other components (in this case, the microcontroller) could spend much more time then execution time of the control algorithm. It is even important to note that communication time (through the parallel port) is not so influenced by the microprocessor speed because standard I/O operations have standard cycle times which must be independent of microprocessor speed (so the newest PC parallel port is still compatible with ISA standard parallel port of older PC´s). Considering this fact, making use of a high performance PC will not allow to obtain very short control periods because I/O communication time will not be so reduced. Make use of a faster PC is only justified if the control algorithm is quite complex so computation time could become important.

Now it is clear that the proposed system is limited in speed because it uses the PC parallel port. Of course other high speed I/O interfaces, like PCI I/O cards or USB ports, could also be used but they could increase system cost and complexity. The adopted solution is cost effective, simple to implement, makes use of easily founded components and also guarantees short enough control periods for a large number of control applications.

In order to operate the parallel port at high speed it was configured in the enhanced parallel port (EPP) mode. This operation mode allows higher communication speed when compared to standard operation mode (EPP mode uses hardware handshake).

II.b. *Microcontroller*

The microcontroller (C) used is the AT90S8535, made by ATMEL [7]. It is a fast 10 MIPs C with internal A/D and it receives from the PC all driver´s command information and also sends A/D conversions results to the PC.

The use of a C between the PC and other hardware components is a key point for achieve safe operation. This happens because the C validates the received data and only update driver´s commands if all received information seems to be correct. In any case, this conception increases system reliability in fault conditions. In practice, data corruption can occur if the cable which connects the PC to the C is too long for example.

Also the C incorporates flexibility to the whole system since its program is flexible and the communication protocol between the PC and C can be easily adjusted to each specific application. For example, the C could receive from the PC just one PWM information as well as it can receive three or four PWM informations. The amount of information exchanged by the PC and C is defined by the programs running in these both components. Of course the control period is increased as the amount of exchanged information increases. In this way, maximum controller operation frequency, of about 40 kHz, is achieved if the C is configured to receive only one PWM information and send only one A/D conversion result.

In the implemented prototype, it has been used the internal 10 bit resolution A/D converter of the AT90S8535 C but an external A/D converter with higher resolution and speed could be used also. In fact, for some applications, an external A/D converter could be required and only small adaptations should be made in the presented system.

II.c *PLD*

The third main logic component used is a EPM7128S PLD from ALTERA Corporation [8]. It is responsible for the generation of high frequency PWM signals at frequencies up to 100 kHz or more (limitation factors are PLD main clock and desired PWM resolution).

Like the C, the PLD is flexible (it has in circuit programming feature) and can be programmed to generate multiple PWM outputs or it can be used as a simple port expansion for the C so the system can command a large number of switches at low frequency (useful for develop multilevel converters).

In addition, the use of a PLD improves system capabilities as dead-times can be easily implemented, the number of PWM outputs and resolution can be modified and even hardware protections can be added.

At this point it is important to note that although all three logic control sub-systems (i.e., PC, C and PLD) are easily programmed, it is more interesting to have "standard" firmwares for the these components so the system can really be used as a ready for use tool.

II.d. *Isolated Analog Interfaces*

These analog interfaces provide galvanic isolation between the development system and signals from the prototype under test. This guarantees safe operation for the designer as well as protects the PC and other control logic from high voltage levels or noise.

II.e. *Isolated MOS Drivers*

These circuits provides appropriated gate drive signals to MOSFETs/IGBTs. They are also isolated for the same reasons already explained.

II.f. *Power Suppliers*

To operate the isolated MOS drivers and isolated A/D interfaces a large number of isolated DC suppliers is required. These suppliers are generated from a multiple output flyback converter specially designed to reach the system requirements (in the implemented system prototype there are thirteen 20 Vdc and eight 10 Vdc outputs with high isolation between each other). The 20 Vdc are used to feed the MOS drivers and each pair of 10 Vdc is used to implement an symmetrical $\pm$ 5 Vdc supply for feed each isolated A/D interface.

II.g. *Stop Button*

The stop button is a simple normally closed press button. It provides a way to stop the control program execution, as will be explained later.

III. INTERFACE CIRCUITS DESCRIPTION

In this section a detailed descriptions of the analog interface circuit and MOS driver circuit is carried out.

Fig. 2 shows the circuit of the analog interface. In order to protect this circuit from over-voltages and to avoid high frequency noise a front-end passive filter/(voltage divider) is included and it is implemented by R_1, R_2, R_3, C_1, Z_1 and Z_2. Resistors R_1, R_2, R_3 as well as capacitor C_1 must be designed to match each specific requirements of input voltage level and frequency bandwidth. A dual operational amplifier (TL082 – A1:1, A1:2) is used to amplify the input voltage and also to close the analog optocoupler feedback loop. If the input voltage range excursion is less than $\pm$ 2,5V then a voltage gain can be obtained by choosing appropriate values for R_4 and R_5. In fact, for achieving best accuracy, the output of A1:1 must have full excursion of $\pm$ 2,5V for full bipolar input voltage range (or 0 to +2,5V for unipolar input voltage range).

In order to obtain high speed response, good linearity and low thermal drift an IL300 linear optocoupler was used. The IL300 achieve these characteristics by incorporating a feedback photodiode so a closed loop can be formed through A1:2 and Q_1. At the output of the IL300 another TL082 is used to adjust output voltage gain and offset. The output of this interface can vary between +2,5 V and -2,5 V. It is also possible to have an bipolar input and a unipolar output so this circuit is compatible with most A/D converters input.

Fig. 3 shows the implemented circuit of the isolated MOS driver. This circuit is optically isolated by a 6N136 optocoupler and its output is implemented by a MC34151 (dual high speed and current MOS driver). The MC34151 allows to reduce board size and component count. It also provides two outputs so it is possible to command two switches (if they have same reference).

Each driver was implemented in a single small board and has it's own 15 V regulator in order to improve driver reliability. Because of this and considering that 6N136 internal emitter diode is current fed then it is expected to achieve higher noise immunity for this driver circuit.

In addiion, this driver board make available an second reference point (at 2,4 V), so the output can vary from –2,4V to 12 V.

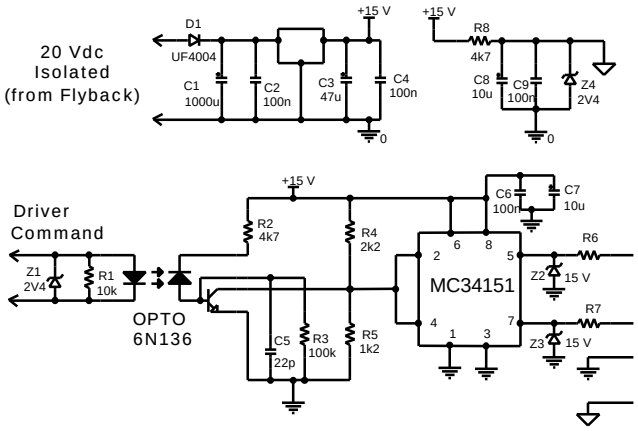

Figure 3 - Isolated MOS Driver Circuit - Board Schematic.

Since each PLD pin has limited output capability an interface circuit must be inserted between the PLD pins and 6N136 inputs (each 6N136 input is driven by approximately 30 mA). This problem was solved by using the circuit shown in Fig. 4.

As can be seen in Fig. 4, two cascaded ULN2003 (transistor arrays) are used to connect 7 pins of the PLD to 7 MOS drivers. This configuration has been used because it provides inverted logic and if one input is left open then the respective output will not be activated.

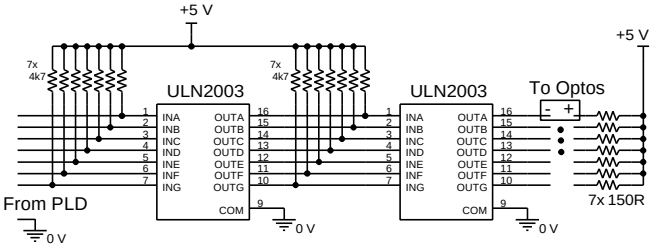

Figure 4 - Circuit to Drive the Optocoupler's Fotodiodes from PLD Outputs.

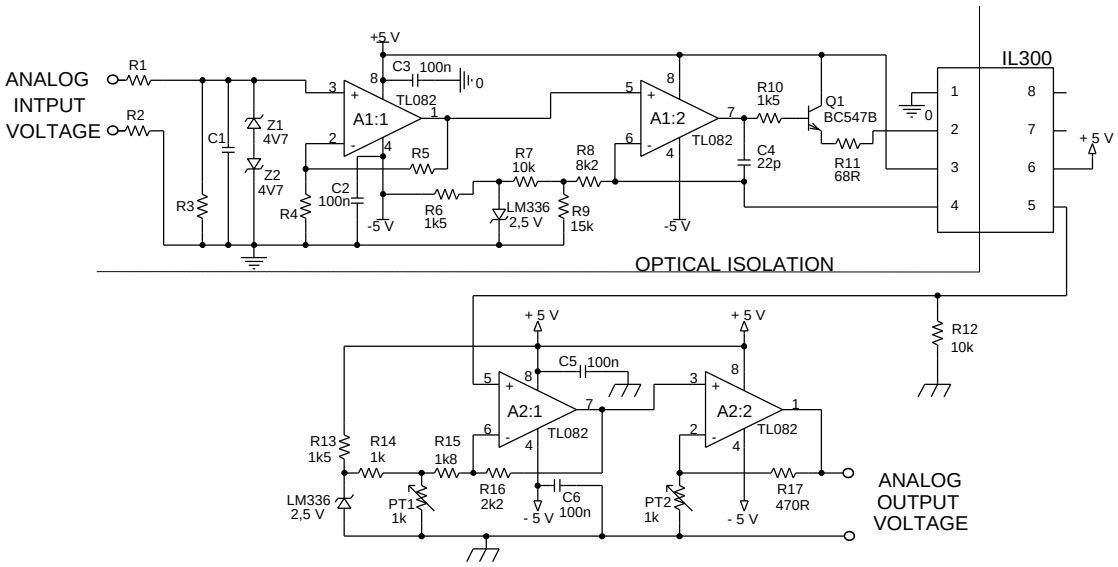

Fig. 2 - Isolated A/D Interface Circuit.

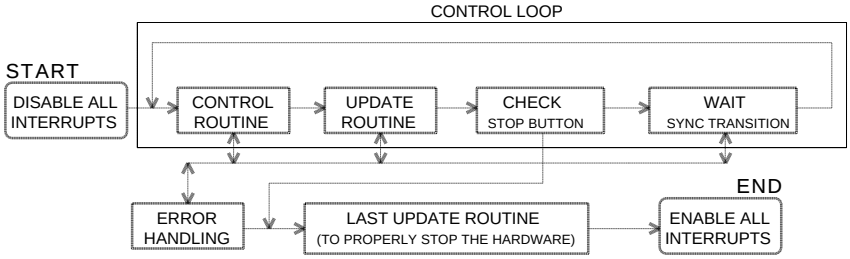

Fig. 5 - Block Diagram of the Control Program.

IV- CONTROL PROGRAMM DESCRIPTION

The implemented prototype uses C language to implement both PC and C programs. The designed controller is described by a routine (control routine) which is inserted in the general control program. The development software used is the C++ Builder 1.0 C-compiler.

As can been seen in Fig. 5, the control program initiates disabling all PC microprocessor interrupts: this implies that all processor time is allocated only by the control program and no other tasks will interrupt its execution. So, when the PC is executing the control program, it will not respond to any mouse or keyboard command. This is why an external stop button is necessary and there is no other way to arbitrary stop program execution after it started to run.

The control loop is divided in four main tasks:

1. Control routine: this is the specific application control routine which is programmed by the user in order to implement the designed digital controller. The user can determine abnormal conditions that requires program termination by calling error handling routine.
2. Update routine: this routine establish communication between the PC and C so the latter receives information to update outputs and also send last A/D conversions results to the former. If a communication error occur then this routine can cause program termination by calling error handling routine.
3. Check stop button routine: at this routine the PC reads the parallel port status register in order to determine the actual state of the stop button (in the status register there is a bit associated with a parallel port pin in which the stop button is connected). If the stop button is pressed then the last update routine is called and execution gets out of control loop thus PC control returns to be under WINDOWS´95 normal operation.
4. Wait Synchronism routine: this forth control loop task is a key point to achieve accurate and stable control period: (independent of PC processor speed). This is done by waiting a periodic transition of a very stable frequency signal generated by the C. This signal is applied to a parallel port status pin so its state can be checked by reading the status register. If an out of synchronism state is detected then this routine causes program termination by calling error handling routine.

Regarding error handling routine it should be noted that it can stop or not program execution. For example, if an out of synchronism is detected only once in millions of cycles then it could not be desirable to stop the system immediately. In this case the error handling routine could keep program execution but it could also stop execution if this error occurs frequently. This gives more flexibility to system operation and also can help to debug hardware problems.

Regarding last update routine, it is responsible to properly generate a safe condition (for example, it can disable PWM outputs and/or can generate a signal to turn off converter's power supply) so the control program can be stopped safely.

The described program runs at the PC but there is also other program running at the C side concurrently. Fig. 6 shows how both programs are executed along the time. It is important to note that while the PC is executing the control routine the C is updating output signals and reading A/D results. So, looking at C side, the minimum control period must be more then $T_1 + T_3$. Looking at PC side, the minimum control period must be more then $T_1 + T_2$. Therefore the the minimum control period must be more then both latter results.

While the PC waits for a SYNC transition after execution of the control routine the C waits a signal transition (in STROBE signal: see Fig. 1) after it concluded output updates and A/D result reads. The STROBE signal generates a hardware interrupt and the associated subroutine is responsible for the communication with the PC. In this way all devices still in synchronism with the SYNC signal so a stable control period is guaranteed.

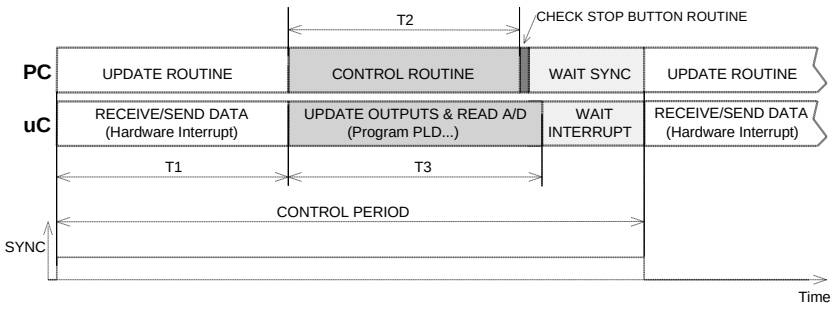

Fig. 6 - Program Execution Time Diagram.

V. KEY POINTS

Some details enables the proposed system to achieve the described performance. Because their importance, they are emphasized by the following list:

1. There is used WINDOWS´95. This guarantees that is possible to disable all interrupts so the PC can function like a dedicated system. If other newer operational systems are used, disable interrupts could fault.
2. It has been used a stop button. This is a very simple way to properly stop program execution.
3. The PC parallel port is used in EPP mode so it is possible to obtain high operation speeds.
4. An external reference signal is used to synchronize the PC program execution (and consequently the entire system). This guarantees a stable control period that is independent on the PC speed and control routine time period (there is only limitation of its maximum time period).

VI. SYSTEM LIMITATIONS

The proposed system has some limitations regarding its capabilities, as follows:

1. Operation frequency: it is limited to approximately 40 kHz (i.e., 25 s period) which is achieved if only one 12 bit PWM is configured and only one 10 bit analog measurement is made. Of course, it is possible to have higher frequencies and short periods if lower resolutions are acceptable.
2. In cycle control: the system, as it is presented here, is not suitable to implement controllers that may require signal updates to occur within the control period. For example, it is not possible to measure a current signal within a control cycle and change an output state when this current value pass by a threshold value. Although actions like this could be implemented through the use of an analog comparator and perhaps some changes in PLD/ C firmware, these type of control is out of the scope of this work.
3. There are also limitations regarding the use of the C internal A/D: resolution is limited and also acquisition time becomes critical if more then one measurement is needed. This limitation could be overcomed by using an external A/D converter so acquisition frequency and resolution can be improved.

VII. SAFETY REMARK

The question of safety was not discussed in details. In fact, protection circuits must be included between the converter prototype and the primary power source. Also it is strongly recommended that these protections must operate independently of the development system in order to increase safety (There is some fault conditions that may occur, like computer reset or program crash).

The development system tool also may include software and hardware protections and these protections could interact with the indispensable external protection, but the latter must be able to operate at all situations despite of development tool status.

VIII. CONCLUSION

The presented development tool is a flexible tool suitable for the implementation of digital controllers. Isolated analog interfaces and MOS drivers are included so it can be applied in the development of power converters. This system is based on a PC and can be easily implemented at low cost. Although it has some limitations regarding speed and in cycle control it can be applied to a large number of controller designs. Finally, complex digital controllers and new control strategies can been implemented in a short time using the proposed system.

REFERENCES

[1] D. Maksimovic, A.M. Stankovic, V.J. Thottuvelil, G.C. Verghese, "Modeling and Simulation of Power Electronic Converters", Proceedings of the IEEE, Vol. 89, No. 6, pp. 898-912, June 2001.

[2] H. A. C. Braga and I. Barbi, “Conversores Estáticos Multiníveis - Uma Revisão”, *SBA Controle & Automação*, Vol. 11, No. 01, January-April 2000.

[3] T. A. Meynard, H. Foch, P. Thomas, J. Courault, R. Jakob and M. Nahrstaedt, “Multicell Converters: Basic Concepts”, *IEEE Transactions on Industrial Electronics*, Vol. 49, No. 5, pp. 955 - 964, October 2002.

[4] A. M. Wu, J. Xiao, D. Markovic, and S. R. Sanders, "Digital PWM Control: Application in Voltage Regulation Modules," *Proc. IEEE Power Electronics Specialists Conf.,* Charleston, SC, pp. 77-83, June 1999.

[5] A. V. Peterchev and S. R. Sanders, “Quantization Resolution and Limit Cycling in Digitally Controlled PWM Converters”, *IEEE Transactions on Power Electronics*, Vol. 18, No. 1, pp. 301-308, January, 2003.

[6] A.M. Stankovic and H. Lev-Ari, "Randomized Modulation in Power Electronic Converters", *Proceedings of the IEEE*, Vol. 90, No. 5, pp. 782-799, May 2002.

[7] ATMEL Corpotation, see website *www.atmel.com*.

[8] ALTERA Corpotation, see website *www.altera.com*.