DESIGN AND IMPLEMENTATION OF RAPID CONTROL PROTOTYPING SYSTEM WITH MATLAB/SIMULINK

Nikolaj Semenič[1], Andrej Sarjaš[2], Rajko Svečko[2], Amor Chowdhury[1]

[1]Margento R&D, Maribor, Slovenia
[2]FERI, Maribor, Slovenia

Key words: Rapid control prototyping, Real time workshop, DSP, TMS320F28335, Matlab, Simulink, State space control, inverted pendulum

Abstract: This paper presents the development and implementation of a Rapid Control Prototyping (RCP) System. Key features of RCP are working with simulink block sets, automation of code generation and automatic downloading of executable code to the target DSP. In such manner, control algorithms can be rapidly integrated or adapted in no time. Simulink integrated with DSP TMS320F28335 provides a useful development tool for design verification of control algorithms, suitable for different plants. Through the use of RTW (Real Time Workshop) functionality, different modules on custom made DSP board are developed, which are needed to create an environment that is suitable for processing execution. Software has been developed with the use of Simulink block sets. Some of them have been provided from TIC2000 library. More complex blocks have been developed with the use of S-functions and TLC, target language compiler.

Design and realization of regulation algorithms for stabilization of rotary inverted pendulum are presented.

Načrtovanje in implementacija RCP sistema v okolju Matlab/Simulink

Kjučne besede: RCP, RTW, DSP, TMS320F28335, Matlab, Simulink, Prostor stanj, inverzno nihalo

Izvleček: V delu predstavljamo razvoj in implementacijo sistema, ki omogoča hitro in zanesljivo integracijo in verifikacijo algoritmov vodenja na realnih objektih. Ena izmed bistvenih prednosti takšnega sistema je, da ga zgradimo s pomočjo blokovnih objektov v Simulinku. Ti pa se samodejno prevedejo v ustrezno izvršno datoteko, ki se naloži na DSP. V takšnem smislu sistem omogoča enostavnejšo implementacijo algoritmov in hkrati njihovo lažjo prilagodljivost. Z uporabo Simulinkove knjižnice RTW je moč razviti različne module za podporo različnih funkcionalnosti na digitalnem signalem procesorju. Sistem smo razvili s pomočjo osnovnih blokov, ki jih najdemo v knjižnici TIC2000. Kompleksnejše funkcionalnosti pa smo zagotovili z razvojem lastnih, naprednejših blokov, s pomočjo izgradnje S-funkcij in z ustrezno korekcijo prevajalnika TLC.

Predstavljamo r*azvoj algoritmov vodenja za stabilizacijo rotacijskega inverznega nihala.*

1. Introduction

With the fast development of digital signal processors, they not only became less expensive but also faster and more complex. Tools for planning and development of embeded-systems are developing in a similar fashion as well. Development of embeded systems is such a complex task that it demands from an engineer a precise knowledge of the DSP architecture and the periphery, as well as programing tools for planning of process systems. The knowledge of suitable program languages is necessary as well. In spite of the fact that many manufacturers of developing program equipment invest a lot of energy in the development of efficient compilers, sometimes it is neccessary to use the assambly language to optimize specific processes in DSP.,. Even more with the complex systems, the probability of an error is high, despite the experience and appropriate expertise of the developer. Thus, an industrial manufacturing leans towards a smaller number of errors during the production as well as to decrease the time needed to develop the product and all the related costs. With that intent, the rapid control prototyping (RCP) has started to develop. In that way, various algorithms can be tested in real time on real plants through use of the circuit boards, which have been especially developed for that purpose. That way, there is no need to develop any hardware parts and it is not necessary for the designer to deal with possible mistakes that would come along. It would be logical that the developer focuses only on the development of algorithms. A way of programing is set in a way of blocks that assemble in complex block models. A way of rapid prototying is offered through the MathWorks' Matlab or Simulink. With the built in real time workshop (RTW) library or precisely TIC2000, Simulink offers the possibility to develop the embeded block model system along with blocks that are offered by other libraries. Through RTW mechanism Simulink block model is converted to the project in the DSP software developing environment Code Composer Studio (CCS), in a form of files (.c and .h), it generates an executable file (.out) and loads it onto the DSP. TIC2000 library enables the development of the Texas instruments' DSPs from C2000 family. The RTW offers support for custum circuit boards as well.It si also posible to implement circuit board with DSPs which are not found in the library. Although this procedure is a bit more complex.

In previous work, with the use of Simulink, the researchers have been involved with rapid prototyping in the control systems. Some of them have only used the basic blocks, which are found in the TIC2000 library /1-5/. In part /6/, the development of custom Simulink blocks is discussed with S-functions in the sense of adding the functionality in a form of a C language program . In such form there is not possible to add additional assebly commands or to support additional hardware moduls. Some of them broadened a basic block recruitment with the development of additional, more complex hardware moduls. In this case, a more efficient custom circuit board is used which has been created independently /7,8/.

Development of rapid prototyping sytem using Simulink environment and a custom circuit board based on floating point DSP TMS320F28335 is presented in this work, . To take into account that we have not used any of the expected circuit boards from the TIC2000 library, but developed it independently, it has also been necessary to develop specific blocks that have been more complex than the basic ones. The more complex modules or drivers have been developed through use of S-functions and the TLC (Target Language Compiler). Our rapid prototyping system is more usable since it has, in comparison to the circuit boards from the TIC2000 library, some extended hardware peripherals. The system is tested with the stabilization of the rotary inverted pendulum in which we use different control algorithms that are designed with the help of Matlab functions.

System for rapid prototyping designed with Matlab/Simulink is Presented in the section two. The third section describes, in detail, a circuit board and the modules that are necessary to obtain measured quantities. The fourth section describes the procedures of implementation of simple and more complex blocks. The fifth section offers a detailed description of the controled plant or the rotary inverted pendulum and the measured quantitiy. The base and the working of stabilization system of the rotary inverted pendulum is explained in the sixth section . In the seventh, design of regulators is summarized. The resaults wich proove the success of implementation of the system for rapid prototyping with the Matlab/Simulink are presented at the end.

2. Rapid Control Prototyping with Matlab/Simulink

Simulink, with the RTW tool, and the TIC2000 library, that offers support to all of the DSPs from the C2000 family, with the development software Code Composer Studio, presents a basis of rapid prototyping. Before the Simulink is capable to translate and load the block model onto the DSP through the use of the CCS, it is necessary to take care of the DSP's fundamental settings. That is done through the block F28335 eZdsp from the TIC2000 library. Despite the fact that our circuit board differs from the expected F28335 eZdsp board, that is not an obstacle in assuring the basic functionality of the DSP. This block takes care for an appropriate setting of the memory map and chooses desired functionalities at the DSP pins. In the RTW system, it is also necessary to set a suitable System target file, Make command and a Template makefile. The first one is set in a relationship to the DSP, however in the case of other two, the default settings are suitable. /9/

Procedure to the executable file is a complex process (Figure 2.1). Models in the Simulink scheme are translated into objects (.rtw). The TLC translates these objectcs into appropriate files (.c and .h) and while at it, adds the system files. It can add custom files as well. Template makefile (.tmf), with the generation of the appropriate Makefile (.mk), links together a file into the executable file (.out) (Figure 2.2). The system of generating a code into the RTW can

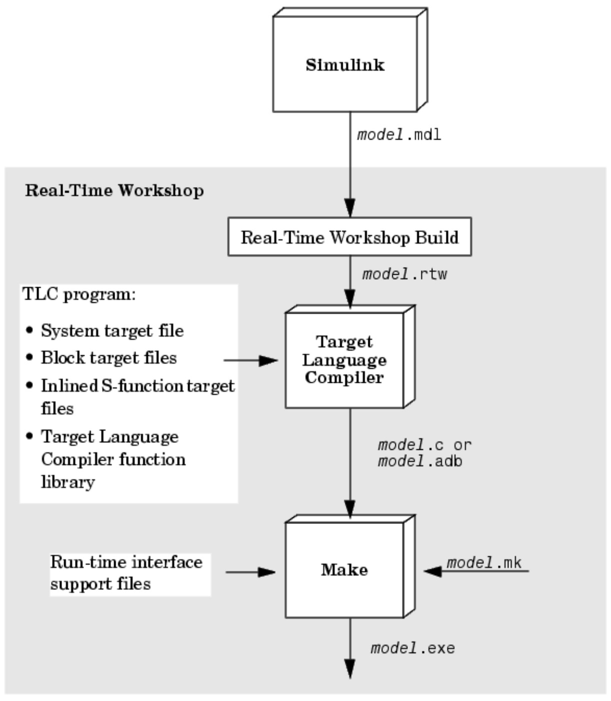


Fig. 2.1: Process of creating a code/9/

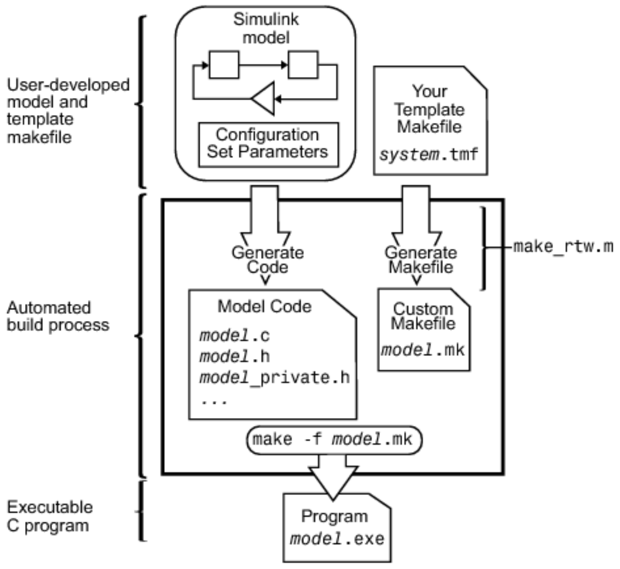


Fig. 2.2: Process of creating an exe file /9/

be adapted with the help of the RTW. With an appropriate adaptation of the TLC, inlined S-functions can be created.

Through use of TLC and wrapper or inlined S-functions we can add not only the individual functions in the form of C code, but also the adaptations in the assembly language or hardware instructions. /9/ In that way, individual drivers can be implemented to extend the functionality of the circuit board in the Simulink.

3. Hardware architecture and software implementation considerations

Circuit board, based on DSP TMS320F28335, was developed with the purpose to establish the system for verification of the control algorithms used in various control plants (Figure 3.1). Individual modules are enabled and executed on the DSP, which ensure gathering and processing of the measured quantities as well as signals for driving actuators.

DSP belongs among a 32-bit microcontrollers intended for applications in real time. DSP TMS320F28335 has a capacity to operate with the floating point. A good number of functionalities are built-in, which can be used in different applications. /10/

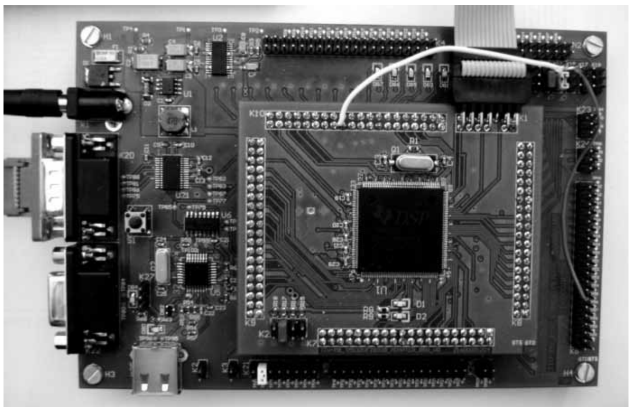


Fig. 3.1: Custom made circuit board

Characteristics of the DSP are:

- Highly efficient 32-bit CPU (IEEE-754 Single-Precision Floating-Point Unit (FPU), 16 x 16 Dual MAC)
- Six channel DMA controller (for ADC, SARAM, ...)
- 16-bit or 23-bit external memory interface
- Integrated memory (F28335/F28235: 512KB Flash, 68 KB SARAM, 2 KB OTP ROM)
- Pins from GPIO0 to GPIO63 can be connected to one of the eight external interrapts
- Up to eight PWM outputs
- Three 32-bit CPU Timers
- Serial ports (up to two CAN modules, up to three SCI (UART) modules, up to two McBSP modules (set as SPI), one SPI module, one integrated I2C Bus)
- 12-Bit, 16 channels
- Module for an event count and a modul for encoder support
- Up to 88 multiplex GPIO pins with the input filters

In comparison to the circuit board from the TIC2000 library eZdsp F28335, our circuit board has a number of important advantages:

- On the circuit board, there is a 22-bit four channel external AD converter ADS1213,
- external flash 8Mb, M25PE80,
- external 512 kW RAM ISSI IS61LV51216,
- external EEPROM 128 kb, M95128,
- Converter from RS232 to USB for the communication with the personal computer.

External AD converter enables a higher quality gathering of measurements than the built-in one. Difficulties with lackage of memory are taken over by large external memories such as RAM, Flash and EEPROM. Because of high capacity of memory, the circuit board enables development of very complex systems as well. A similar circuit board is used in the mobile payment system /14/.

With the help of the block from the TIC2000 library and an additional block that extends the basic functionality, a modul for an encoder support was developed on the circuit board. Custom made blockfor the support of the external AD converter was developed independently, with an additional driver and SPI communication-bus.

3.1. Encoder support modul

Encoder support module (eQEP) in the DSP is designed for linear or rotational incremental encoders for measuring position, direction of rotation, and speed. The counter clockwise rotation is defined at the time when QEPA signal outpasses the QEPB signal (Figure 3.2). A frequency of QEPA and QEPB signals alters in relationship to the speed of the motor. That way, the speed is calculated with the QEPA and QEPB event count or by measuring the time between two events.

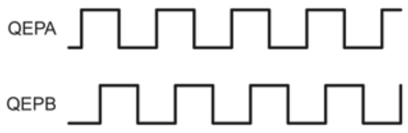


Fig. 3.2: Signals scheme from encoder

The speed can be calculated in two different ways:

$$v(k) \approx \frac{x(k)-x(k-1)}{T} = \frac{\Delta X}{T} \quad /9/ \qquad (3.1)$$

$$v(k) \approx \frac{X}{t(k)-t(k-1)} = \frac{X}{\Delta T} \quad /9/ \qquad (3.2)$$

Equation (3.1) represents a number of pulses that are perceived by the DSP in a defined period of time. DSP can be set in a way that it counts positive and negative fronts of the QEPA and QEPB signals. That way twice the amount of events is obtained in return.

A unit period is set to scan time value , in which the DSP counts the events. A unit period is set in the QUPRD registry where the end number, up to which the counter reaches, is written. The amount of one time quantum is inversely proportional with the frequency of the processor core, which is 150 MHz. Upon the expiration of the time period, the register position (QPOSLAT) is copied, with the number of events, which are perceived by the DSP in the period unit. This type of measuring and calculating the speed is suitable for higher speeds, since with the large number of counted events, the counting error is less noticable.

A more suitable way for lower speeds is offered in the equation (3.2). In that case the time between the two events is measured. Performed path between the two events presents an inverse of a maximum number of events in one turn. In that case the error of the time counter is smaller, the longer is the time duration between two events. The timer value, which measures the QCTMR time, is saved into the QCPRD register when the second event occurs.

Encoder support module was executed with the help of block from the TIC2000 library. Direction of rotation and values of registers QPOSLAT and QCPRD were chosen as output values of the module. To that block, one more was added, which was developed independetly with the help of S-function. In that module, both algorithms for speed measuring were executed. Further the system that increases the robustness of measurings was added. An algorithm (3.1) is used to measure higher speeds and an algorithm (3.2) for lower speeds.

A boundary that defines which speed has a lower error is defined with the calculation of relative speed errors. That way, there is a relatively low error in wide range of speed measuring optaned.

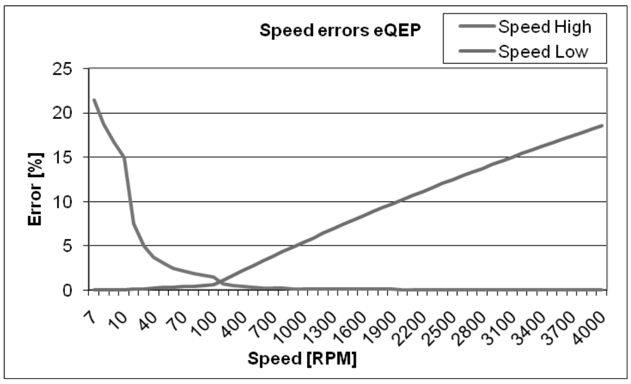


Fig. 3.3: Algorithm errors

As it is seen, (3.2) algorithm has lower error for the speeds up to 240 turns per minute. For higher speeds (3.1) algorithm is more accurate.

3.2. External AD conversion through the SPI bus

Conversion from the notebook or an analog world to a discret or digital world is one of the frequent procedures that are part of the everyday, when it comes to control systems. External AD converter ADS1213 is located on the circuit board and is a 22-bit delta sigma AD converter with the 20-bit resolution at 10Hz and a 16-bit resolution at sampling rate of 1kHz. It has four chanels at the input which are, via multiplexer, connected onto the diferential input. It also has an integrated amplifier for small signals and it has a support for SPI or I2C communication. It is connected to the DSP through the SPI communication bus and it enables gathering of data with a frequency of up to 6,25 kHz. The chip has integrated special callibration mechanisms that offer a precise set up of the measuring range and of the initial offset. It also has its own built-in micro controller that manages instructional, command and data registers as well as two callibration registers. The way something is written in these registers is through the SPI bus.

One of the main advantages of this converter is the built-in digital filter. Its function is decimation of the current measurements that are offered by the delta sigma modulator. A number of resaults that the filter decimates depends on the decimation ratio that is set in the command registry. Turbo mode influences the frequency of the modulator. This means that those two factors indirectly influence the calculation speed and the delivery of data into the data register./11/

The speed of output data is determined by the following relation:

$$f_{podatki} = \frac{f_{XIN} \cdot Turbo\ Mode}{128 \cdot (Decimation\ Ratio + 1)} \qquad (3.3)$$

Where:

$f_{podatki}$ - frequency of output data

f_{XIN} - frequency of the input oscilator is 2 MHz

Because we want an even more precise measurings, that way we want an even higher decimation ration. That can be reached with the highest setting of the turbo mode.

It is necessary to enter Turbo Mode, Decimation Ration and many other parameters in appropriate addreses in the command registry. The measuring range of the input signl is from -5 to +5 V, with the selection of inside amplification.

The communication between the AD converter and the DSP was realized through the SPI, serial peripheral interface. The type of communication is in a form of master to slave, in which the DSP is the master and the AD converter is a slave. This means that the AD converter only responses to a demand of the DSP. Communication is functions in a way that it is necessary to send the content of the instructional registry to the AD converter first, with which AD receives the name of the appropriate registry and the information if DSP will write out of it or read from it. Since DSP manages the SPI clock, there is a need to detect when the data is ready to be received from the AD converter. AD reports that through an external pin. External pin triggers external interrupt in which DSP processes received information.

The external AD converter is used because of the higher quality of measurements in the system. As we have seen, it has a built-in digital filter that averages in our case 2499 measurements. One more advantage is that the AD converter is completelly autonomous and works independently from the DSP. Therefore a good amount of processing time is saved for the DSP.

Modul that enables gathering of measurements from the external AD converter was developed in Simulink, independently, in a form of a block. Due to the complexity, the communication system was not able to be built with the basic blocks from the TIC2000 library.

3.3. Connection to the computer

Serial Communication Interface (SCI), also known as UART, is a two-wire asynchronous bus that is used for communication with other DSPs or devices. It is made out of seperate units for sending and receiving which enables half-duplex or full-duplex transfer of data.

UART is used for the communication with personal computers. In such way the measurements are received from the DSP. The reverse communication, from personal computer to the DSP, was also realized.

Communication of data was realized according to the idle-line,, because there are not more than two participants in the communication. Measuring resaults are sent in the core of the main programe without disturbing time-deterministic programe interrupts.

That model was also developed on our part. In such way it was simpler to realize the measuring of time necessary to send the data to the personal computer. The time duration of programe routines were measured with CPU-timer by its activation at the beginning of selected programe routine and its determination at the end of programe routine.

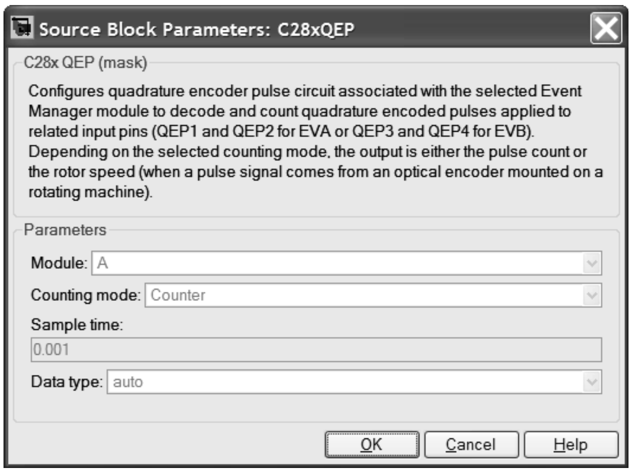
3.4. Pulse width modulation

The DSP supports six PWM modules that enable, on each of the external pins, two different pulse width modulated signals. Every modul enables an optional selection of the most different properties at the exit of the generated signal. PWM signals have a wide use in the telecommunication, to manage different elektromechanical sections, as well as to supply systems in the alternative circuit.

The necessary moduls have been enabled with the help of blocks from the Simulink library.

4. Implementation of program units into the Matlab/Simulink

Individual programe units on the circuit board were developed with the Simulink programe, real time workshop (RTW) tool. Matlab 7.4, respectively, the Simulnk TIC2000 library with the DSP support TMS320F28335 was used for the implementation of modules. The library supports all basic models that are provided by the DSP. In comparison with the previous versions of Matlab, this library is wealthier. Modules are more comprehensive and enable more precise registery settings. For example, a comparison between an encoder modul for TMS320F2812 and TMS320F28335 is noticable. (Figure 4.1 and Figure 4.2). As it can be seen, the difference is considerable. We are very limited with the older version. In the mean time, with the newer version, we can reach for more demanding settings of the DSP hardware modul, which enables a more demanding and more universal use of the modules. Matlab 7.4, in comparison to its previous versions is reacher for newer platforms. A functionality to control interrupt routines in a form of Hardware interrupt and Software interrupt blocks was added.



Source Block Parameters: C28xQEP

C28x QEP (mask)

Configures quadrature encoder pulse circuit associated with the selected Event Manager module to decode and count quadrature encoded pulses applied to related input pins (QEP1 and QEP2 for EVA or QEP3 and QEP4 for EVB). Depending on the selected counting mode, the output is either the pulse count or the rotor speed (when a pulse signal comes from an optical encoder mounted on a rotating machine).

Parameters

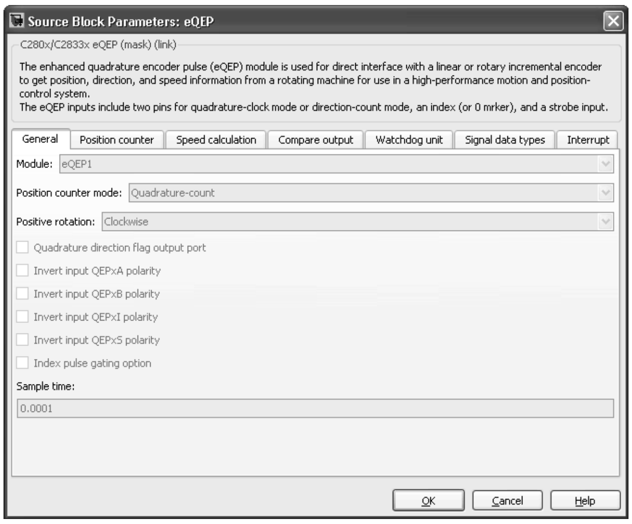
Module: A

Counting mode: Counter

Sample time: 0.001

Data type: auto

Fig. 4.1: Block for encoder support in Matlab 7.1



Source Block Parameters: eQEP

C280x/C2833x eQEP (mask) (link)

The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system.
The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 mrker), and a strobe input.

Module: eQEP1

Position counter mode: Quadrature-count

Positive rotation: Clockwise

☐ Quadrature direction flag output port
☐ Invert input QEPxA polarity
☐ Invert input QEPxB polarity
☐ Invert input QEPxI polarity
☐ Invert input QEPxS polarity
☐ Index pulse gating option

Sample time: 0.0001

Fig. 4.2: Block for encoder support in Matlab 7.4

4.1. Set of Moduls in TIC2000

In the TIC2000 library, which is designed for hardware support of the texas' DSPs, the following blocks can be found; block for AD converter support, digital inputs and outputs, PWM block, encoder block, block for an event count, Watchdog, support for SCI, SPI, CAN, I2C communication, idle task block as well as a support block for hardware interrupts and software interrupts. In the library, next to those, are also blocks To RTDX and From RTDX. Those blocks are used to reestablish the connection between CCS and Matlab programe, which enables transfer of data in real time.

Out of those blocks, folowing blocks were used; for the actuator PWM block drive, encoder support block, block for hardware interrupt as well as the idle task block. The hardware interrupt block enables control over triggering hardware interrupt service routines for external AD converter and encoder. The idle task block was used for independent functioning of communication system with the personal computer in a way that it does not distrub carrying out of time deterministic interrupt routines (Figure 5.1). Blocks To and From RTDX were used to reestablish the connection between CCS and Matlab software. That way, the executable file can be downloaded and run onto the DSP out of Matlab's workspace. This tool is also suitable for exchange of optional variables between the DSP and Matlab's workspace. In spite of the fact that we were able to establish a connection through a single channel and to qualify the transfer of one variable, we noticed large irregularities by the transfer of more variables through more channels. That kind of working could be expected due to a poor J-Tag equipment. By the work J-Tag XDS510 was used. Better J-Tags (XDS560) promise more robust transfer of data and higher data rates (up to 2 MB/s) through the use of the Hi Speed RTDX functionality. Because of unespacted functioning of RTDX, the transfer of data was conducted via serial bus SCI, respectively UART.

The basic framework programe, built by the Simulink in the CCS tool programe, is built in the form of a CpuTimer0 timer interrupt routine in a way that the blocks in the basic Simulink scheme are carryed out with a specified sampling time. Therefore we have to be careful by choosing the sampling time. In order to assure a time deterministic system, the sampling time has to be longer than the carrying out of a system in the preemption. In case we want a certain part of the system to be carryed out in the background, so it does not disturb high priority interrupts, an idle task has to be used. In idle task block can be specified whether the preemption is preemptible or non-preemptible.

4.2. Development of additional blocks with S-function

S-function is a tool that enables integration of custom functionality in the form of a program language C or Matlab's script language. That way a custom block, with a specific functionality, can be created. There are three different ways to integrate program codes in Simulink, Noninlined S-functions, Wrapper S-functions and Fully Inlined S-functions. Noninlined S-function does not require any of the additional tasks such as code adaptation in TLC. They can be created with tools that enable an automatic creation of S-functions or they can be written individually. In the last case, we have to be deeply engaged in the building of a function as well as to master the syntax. Taking into an account that the syntax of S-functions is unique, this work, specialy with complex functions, may take a good amount of our valuable time. That type of building of S-functions does not prove to be the best in the case when we want to integrate a code into Simulink. For example, in the program language C, which is a widespread way of programming especially among embeded systems programers. The integration of the code can be done with the Wrapper S-function. In that case it is not necessary to integrate an entire code manually in the S-function, but only a call to the assigned routine can be added. Ony the basic model of S-function can be used, therefore we do not have to occupy ourselves with the in-depth knowledge of S-functions. The above mentioned procedures are only suitable for the integration of a C programe language and that way they are not suitable for the development of drives, that require hardware instructions syntax for access to registers and the assignment of their evaluations. Fully Inlined S-function means that the algorithm is integrated with the TLC functionality. In that case, a more precise knowledge of TLC syntax is necessary. This case of S-function is suitable for development of drivers, since an optional code, or hardware instructions, can be added into the TLC file. Or appropriate instructions can be added to the TLC, which can add suitable hardware instruction files (.c or .h) to the CCS project.

Matlab 7.4 version is reacher for quite appliable tool, Code Legacy Tool /9/, which provides automated creation of Wrapper or Fully Inlined S-functions. Matlab 7.1 does not support that tool. That way, an integration of the programe code written in the C language is now faster and a lot simpler. A code that we want to implement in the form of S-function has to be written in the form of a function based on C language. It is needed to remodel input and output function variables. With mentioned tool, we were able to develop custom blocks in the C language. Also, through this procedure an outline for development of drives was also developed.

4.3. Increase of encoder robustness

A block for encoder support, from the TIC2000 library is used to gather position and rotation speed measurements. The block is adjusted to the appropriate settings. On the output of the block, an information on the rotation direction and the value on the event count in sampling time and the number of time quantums between two events can be obtained. With this information, the position and the rotation speed in every sampling time can be determined. During testing of functionality of the modul, was found that in specific situations it came to an irregular calculation of values. A block or S-function, which eliminates the occured difficulties, was added to increase the robustness of operating of the modul.

When the encoder modul counts a full event, it means that the encoder modul measured the change of the position exactly for one turn. The encoder modul is designed in a way that when it counts a full event, the encoder counter is set to 0. In case of the opposite rotation, when it counts to 0-th events, the values are set to the maximum number of events in one turn. Most of the time, the difficulties with the measuring had occured exactly at the turn of 0., the maximum number of events and when the rotation direction suddenly had changed. Difficulties had appeared when the position had to be determined and with the calculation of speed through an algorithm for higher speeds (Equation 3.1). From an equation 3.1, it follows that the position and the rotation speed are usually measured with the difference between the current number of events and the number of previous events. Scheme on how the position and speed are determined is presented in Figure 4.3. The beginning of the arrow represents the number of previous events, the end of the arrow the number of current events. Arrows are designed to schematically illustrate posible occasions of two successive events and they do not proportionaly represent events of real values. In a real example, the proportions among values of previous and current events are diminishing, they depend on the rotation speed.

In example A, the difference between the number of current and previous number of events is higher than the half of the maximum number of events (Half-event). A difference upon which the new position and speed can be calculated is defined with:

difference = -(*full-event* - (current_eve -previous_eve)); abs_position += difference; where the position is reduced with an actual difference, which is negative

In the example B, difference is defined with:

difference = current_eve - previous_eve;

abs _position += difference; where the position is increased for an actual difference.

In the example C, difference is defined with:

difference = (*full-event* + current_eve - previous_eve); abs _position += difference; where the position is increased for an actual difference

In the example D, difference is defined with:

difference= current_eve - previous_eve;

abs _position += difference; where the position is decreased for an actual difference

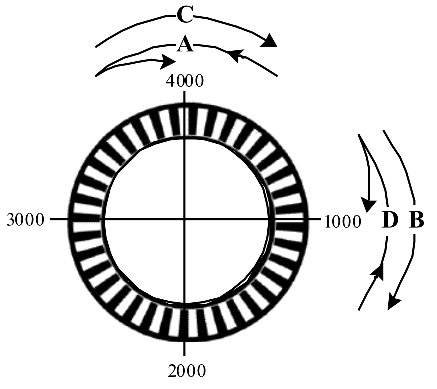


Fig. 4.3: Scheme of the encoder events and the principle calculation of speed and position

A suggested algorithm is suitable for both directions of rotation, since it considers actual shifts, regardless of direction of rotation. It also fixes incorrect calculations that occur around 0-th or full event. Operating of the module improves and the measurements robustness increases.

4.4. Realization of drivers in the Simulink

As it has been already mentioned, the Code Legacy Tool enables an automated procedure of code integration written in C. In this way, an outline of more complex blocks were created, which also include manipulations with registers and hardware instructions. A custom block was developed to receive values from the external AD converter through the SPI communication bus and a block for communication with the personal computer. For the development of complex blocks, that also contain drivers, we used the following algorithms:

- A compiler with the command mex(‚-setup') is set in Matlab
- In the custom C function, input and output values are appropriately adapted just like Code Legacy Tool requires. Input and output function values represent inputs and outputs of the block. The function can also contain instructions. It can also contain calls of other functions from other files as well as macroes or lists.
- In the custom C function it is necessary to coment all calls to other functions, macroes, hardware instructions, registry manipulations. The purpose of the Code Legacy Tool is only the development of C-MEX S-function, which can be integrated as Simulink's S-function block.
- C-MEX S-function and the TLC file is created with the use of Code Legacy Tool.
- To the TLC file, required source or header files (.c or .h) containing hardware instructions are added. Source files (.c) can be added with the instruction %<LibAddToModelSources(„SPI")>, header files (.h) can be added with the instruction %<LibAddToCom monIncludes(„Device.h")>. Instructions have to be added in the core of BlockTypeSetup function in TLC file. Calls to initialization routines of hardware moduls can be added to the function start. In the function outputs, function call to the custom C function can be noticed, which is a result of previous C-MEX S-function generation.
- The model is loaded in Simulink. A project with source and header files in CCS is generated. We look for a custom C function file in the project and uncomment the commented parts of code.
- Added part of the code can be debuged as well and in that way the success of operation of the added module can be prooved in the CCS

After the mentioned algorithm, a modul that receives values from an external AD converter and a modul for communication with a personal computer were added. Modul for AD conversion was developed, since that kind of system was not possible to realize with basic blocks, mainly because of the complexity of communication through the SPI bus. Modul for communication with a personal computer enables an easier supervision over the sent and received data, since a duration time needed for the execution of the specific part of code can be exactly measured with the CPU timer in the DSP.

5. Example of test plant control

A test plant consists of a mechanical framework, pendulum, electrical motor, feeder and a processor with the adapted electrical circuit.

The concept of stabilization of an invert pendulum presents a mounted arm on an electrical motor, where at the end, on the bearings, a pendulum is mounted. With a suitable control of the electrical motor, we can affect the position of the arm and indirectly so, the position of the pendulum.

State space controller have been designed to achieve stabilization. That way we need the information on all of the internal states od the system. These are the speed and angle of the pendulum and the speed and position of the arm. The speed of pendulum is calculated on the basis of pendulum angle change. Position and speed of the arm is measured with the encoder.

Potentiometer is set behind the bearing of the pendulum and it is supplied by voltage of 5 V. The signal in the frame of 0-5 V presents the position of the pendulum 0-360 °. Due to the nonlinearity of the potentiometer, it is possible to measure in a range of 0-340 ° with ± 3 ° of tolerance.

Encoder modul enables measuring of the position and the speed of the arm with event signals received from encoder. Event represents every positive or negative front of the encoder output QEPA or QEPB signals, that are dislocated among each other by 90 ° (Figure 3.1). Encoder has 1000 lines. Taking that in consideration, due the possibility to register positive and negative fronts, it can be registered 4000 events in one turn.

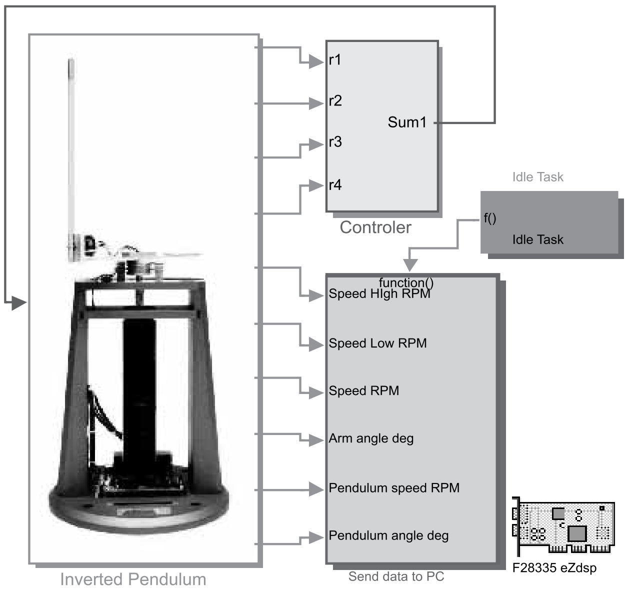


Fig. 5.1: RCP for inverse rotation pendulum

An electrical circuit is located on the external circuit board, which adapts levels of voltage supply between the processor and the motor as well as between the processor and the sensores.

On the ecternal circuit board, there is a H-bridge that provides up to 6 A of current for the motor. H-bridge is controled by the processor by the principle of pulse width modulation (PWM).

6. Design of the core program for stabilization of the inverted pendulum

A row of hardware interrupts were realized in the DSP. Every interrupt has its own specific time deterministic task. In the center portion of the program core, which is presented by an infinite while loop, functions are carryed out that take care of sending the measuring resaults to the personal computer.

Synchronization of the external AD converter with the DSP, is carryed out through an external interrupt. When AD has the current measured data available in the memory data registry, the external pin DRDY falls to 0. This happens exactly evey 0,01 s, which is the sampling time. This event triggers the hardware interrupt in which, through the SPI bus, value of the data register of AD converter is obtained. In that interrupt, the value of the register QUTMR of encoder module is adapted as well. That is the encoder modul timer which is responsible for triggering of the encoder interrupt,. that is also triggered every 0,01 s, respectively the sampling time. With the adaptation of the encoder timer, the synchronization of the encoder interrupt routine with the interrupt of the AD converter is reached. In the interrupt of the AD converter, with the appropriate setting of CpuTimer0 value, the synchronization with the system interrupt is also performed. Through the proper synchronization encoder interrupt occures just before the interrupt of AD converter. System interrupt of Simulink follows both interrupts. The position and the speed of the arm is obtained during the interrupt of the encoder. In the interrupt of the AD converter, the information on the position of a pendulum is obtained, the speed of the pendulum is calculated on the basis of the change of the pendulum position. In the system interrupt, the chosen regulator is calculated and an adequate PWM exit is set (Figure 5.1)

The system is designed so that the interrupt of AD converter has a priority over the encoder interrupt.

7. Designing the state space controllers

For the control system, inverted pendulum two type of state space controllers were designed. The first type of state space controller was designed through pole placement technique (PPT) and the second type with optimization of the quadratic continuous-time cost functional (LQR). A mathematical model of third and fourth order was used for design of controllers. Model was adapted from work /12/. Pole placement controller was designed with the closed-loop poles assignment and Matlab command »acker«. LQR controller was designed with the expression »lqr« and the appropriate weightening (Q and R) matrices.

Taking into consideration that the controllers were designed for a linear system and tested on a nonlinear model in Simulink, where was checked whether the solutions suit the stability criteria. With the simulations, we could also test until which point the system was stable, if pendulum had deviated.

8. Resaults

During the design of PPT controllers, a third order model was used, since the control upon the position of the arm is not significant for the stabilization of the pendulum. During the realization of the controller with the setpoint, a fourth order model and the position of the arm was used as a reference signal. LQR controller was designed with the systems of third and fourth order.

As results show, the best designed controllers are those, which are designed with the third order system. That is to say, the controllers do not demand the stabilization of pendulum in exact defined point of the position of the arm. If we compare these two controllers to the controller with the setpoint, it is established that in the maximal deviation of angle of the pendulm, it swings almost twice the amount. It is comprehensive that by the setpoint controller considerations not only about the angle of the pendulum but also the position of the arm like setpoint is taken into account.

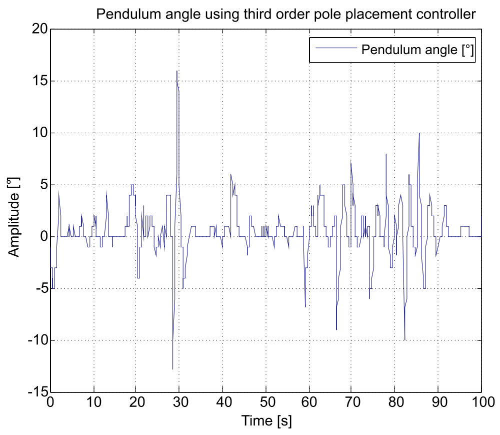


Fig. 8.1: Pole placement controller

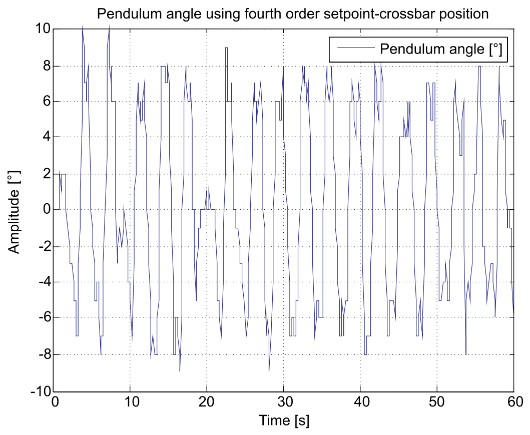


Fig. 8.2: Controller with the setpoint-crossbar position

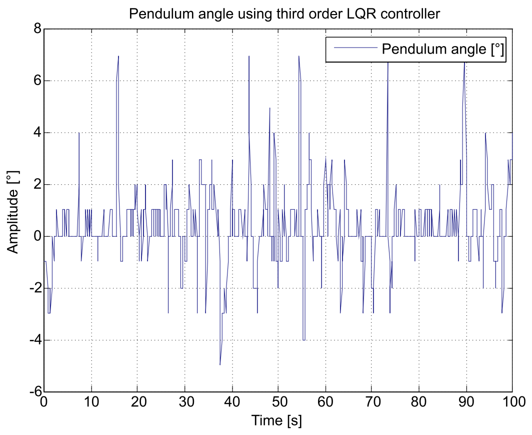


Fig. 8.3: Third order LQR controller

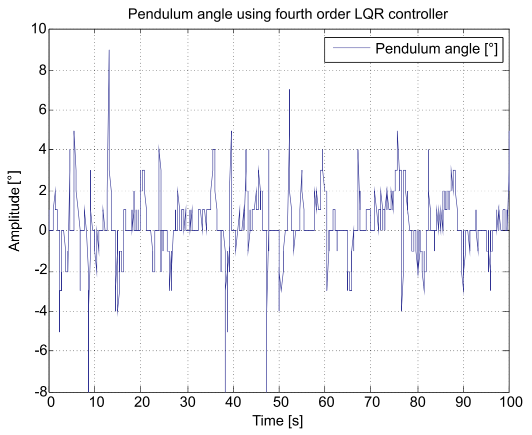


Fig. 8.4: Fourth order LQR controller

Two linear quadratic controllers were calculated. A third order system was used to design the first one. With the second one, a fourth order system and the position of the arm as a reference position were used. If we compare the measurements with previously mentioned controllers, it is visible at first sight that both positions of the pendulum were steadier. If we compare the strenghtening of both regulators with the ones before, it is stated that they are much more sensitive to the position of the pendulum.

9. Conclusion

The goal of our work was to carry out the system for rapid prototyping in the Matlab/Simulink environment. It is confirmed that the newer version of Matlab has numerous advantages with the development of Code Legacy Tool modules and the expanded set of TIC2000 library. Individual basic moduls in the library are more complex and enable more precise hardware settings, that are very welcomed since they enable improved functionality. With that, an in-depth knowledge of hardware DSP moduls is neccessary. The system of rapid prototyping, with rich library collection in Simulink, presents a simpler and faster way of adaptation of algorithms that are in the form of block models and a simplified algorithm testing on real plants. The system of rapid prototyping was successfully tested with the stabilization of the rotary inverted pendulum.

10. Acknowledgment

Operation part financed by the European Union, European Social Fund.

References

/1/ L. Baotai, Z. Rongxiang, Rapid Control Prototype Design of the DC/DC Converter for Fuel Cell Electric Vehicles, Power and Energy Engineering Conference, 2009. APPEEC 2009. Asia-Pacific.

/2/ R. Duma, P. Dobra, M. Abrudean, M.Dobra, Rapid Prototyping of Control Systems using Embedded Target for TI C2000 DSP, Mediterranean Conference on Control and Automation, July 27-29, 2007, Athens.

/3/ W.S. Gan, Y.K. Chong, W. Gong, W.T. Tan, Rapid Prototyping System for Teaching Real-Time Digital Signal Processing, IEEE transactions on education, vol. 43, no. 1, february 2000.

/4/ A. Rubaai, M.J. Castro-Sitiriche, A.R. Ofoli, Design and Implementation of Parallel Fuzzy PID Controller for High-Performance Brushless Motor Drives: An Integrated Environment for Rapid Control Prototyping, IEEE transactions on industry applications, vol.44, no.4, July/August, 2008.

/5/ S.H. Seo, S.W. Lee, S.H. Hwang, J.W. Jeon, Development of Platform for Rapid Control Prototyping Technique, SICE-ICASE International Joint Conference, October 18-21, Busan, 2006.

/6/ K.H. Hong, W.S. Gan, Y.K. Chong, K.K. Chew, C.M. Lee, T.Y. Koh, An Integrated environment for rapid prototyping of DSP Algorithms using MATLAB and Texas instruments' TMS320C30, Microprocessors and Microsystems 24, 349-363, 2000.

/7/ Z. Fang, Y. Qi, Q. Zhang, T. Chai, Design and Implementation of a Low Cost DSP Based Rapid Control Prototyping System, Industrial Electronics and Applications, ICIEA 2009.

/8/ Z. Fang, N. Song, L. Wang, Design and Implementation of a Novel Fuzzy Controller with DSP for Rotary Inverted Pendulum, Control and Decision Conference, 2009.

/9/ Real-Time Workshop, User's Guide Version 7.4, 2009.

/10/ Texas Instruments, Incorporated, *Reference Guide, TMS320x2833x, 2823x Enhanced Quadrature Encoder Pusle (eQEP) Module*, SPRUG05A, avgust 2008.

/11/ Texas Instrumets, Incorporated, *Data Manual*, *TMS320F28335, TMS320F28334, TMS320F28332, TMS320F28235, TMS320F28234, TMS320F28232, Digital Signal Controllers (DSCs)*, SPRS439F, junij 2007.

/12/ Burr Brown Corporation, *ADS1212*, *ADS1213*, *22Bit analog to digital converter,* USA, april 1998.

/13/ Tan Kok Chye, Teo Chun Sang, *Rotary inverted pendulum*, Scool of electrical and electronic engineering, 1999.

/14/ Zdenko Mezgec, Andrej Medved, Amor Chowdhury, Rajko Svečko,. Mobile payments - design of new terminal = Mobilno plačevanje - razvoj sodobnega terminala. Inf. MIDEM, mar. 2008, letn. 38, št. 1(125), str. 53-60.

Nikolaj Semenič[1], Andrej Sarjaš[2], Rajko Svečko[2], Amor Chowdhury[1]

[1]Margento R&D, Gosposvetska c. 84, 2000 Maribor
[2]FERI, Smetanova ul. 17, 2000 Maribor
E-pošta: nikolaj.semenic@margento.com

Prispelo: 01.07.2010 *Sprejeto: 24.06.2011*