ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA076

A NOVEL GENERAL PURPOSE DATA ACQUISITION BOARD WITH A DIM INTERFACE

J. Jadlovský, A. Jadlovská, S. Jadlovská, M. Oravec, D. VoščEk, M. Kopčík, J. Čabala, M. Tkáčik,
Technical University of Košice, Košice, Slovakia
P. Chochula, O. Pinazza, CERN, Geneva, Switzerland

Abstract

A new general purpose data acquisition and control board (Board51) is presented in this paper. Board51 has primarily been developed for use in the ALICE experiment at CERN, but its open design allows for a wide use in any application requiring flexible and affordable data acquisition system. It provides analog I/O functionalities and is equipped with software bundle, allowing for easy integration into the SCADA. Based on the Silicon Labs C8051F350 MCU, the board features a fully-differential 24-bit ADC that provides an ability to perform very precise DAQ at sampling rate up to 1kHz. For analog outputs two 8-bit current-mode DACs can be used. Board51 is equipped with UART to USB interface that allows communication with any computer platform. As a result the board can be controlled through the DIM system. This is provided by a program running on a computer publishing services that include measured analog values of each ADC channel and accepts commands for setting ADC readout rate and DACs voltage. Digital inputs/outputs are also accessible using the DIM communication system. These services enable any computer on a common network to read measured values and control the board.

INTRODUCTION

The main motivation for the development of the Board51 is to provide a general purpose low-cost device suitable for data acquisition of analog sensor values. The digital and analog outputs extend its application in various cyber-physical applications. The device has been developed mainly for educational purposes and will be used in several applications at the Center of Modern Control Techniques and Industrial Informatics at the Department of Cybernetics and Artificial Intelligence, FEEI, TUKE or at CERN. Two main mechanisms have been developed to monitor and control the board. The first option is the use of the MATLAB programming library, which allows for collection and processing of measured data, as well as the control of the board directly through the MATLAB functions. The second approach is based on a server application that mediates measured data and control the board using the DIM system [1]. This approach ensures compatibility with the communication architecture used in CERN experiments.

HARDWARE DESIGN OF BOARD51

The Board51 measuring board, shown in Fig. 1 is based on the C8051F350 micro-controller from Silicon Labs. Schematic design of the C8051F350 can be seen in Fig. 2. This chip has been selected for this application because its manufacturers focused on analog interfaces. It is equipped with a 24-bit 8-channel fully differential ADC with a sampling frequency of up to 1 kHz. This allows you to perform fast and very precise analog measurements, while the built-in multiplexer allows you to perform measurements between any channels or between the selected channel and the analogue ground. Since it has a large resolution, it must be provided with an accurate and constant reference voltage. This is ensured by the LM4121, which is calibrated to a 3V voltage and can be fine-tuned with a precision potentiometer mounted on the board.

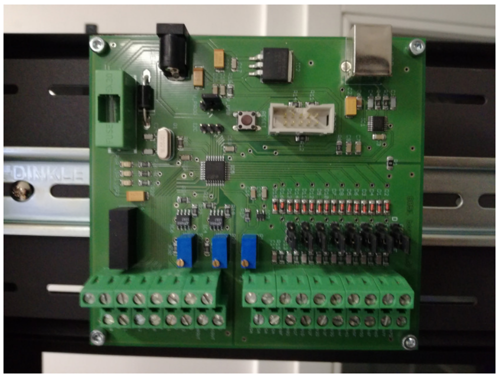


Figure 1: General purpose measuring board Board51.

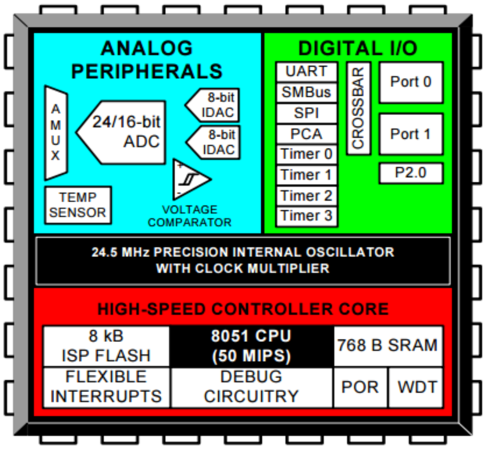


Figure 2: Schematic design of C8051F350 MCU.

Other important peripheral devices on this board are the two 8-bit current DACs that are connected via the resistors to an analogue ground, providing a voltage output instead of a current. Operational amplifiers are connected to the outputs with an adjustable amplification ratio so that the connected load does not cause a voltage change on the DACs and at the same time that the range of the output voltage can be adjusted.

From digital interfaces, this board mainly uses three pulse-width modulated outputs operating at 62 kHz with 8-bit resolution. In addition, three universal digital inputs/outputs can be used to control or detect the status of digital devices.

The board itself has no network interface, therefore a supervising computer is needed to connect the board to the system's communication architecture. The computer connects to a USB 2.0 interface that is emulated by the FT230XS controller. It mediates communication between the USB interface and the UART interface located in a MCU. In this way, it appears to the computer as a virtual serial device.

The board can be powered from three sources. One of the options is to power the board through a power jack connector where a voltage of 5 to 26 volts can be applied. The most common way of powering the board is through the USB connector as the USB interface is used to connect to the supervisor computer. This type of power is used in almost all applications. The least frequent power supply source is the use of a programming adapter, but since it does not make sense for the programming adapter to be connected over the entire operating time of the board, this power supply is hardly ever used. Except for analog interfaces, the whole board works at the 3.3V logic, while all analogue interfaces are calibrated to a maximum usable voltage of 3 V. If necessary, the reference voltage of the analogue interfaces can be changed by the mentioned precision potentiometers.

MCU PROGRAM

The program at the MCU level is written in C language, consisting of several modules, each of which provides control of another peripheral device. The master module initializes all other modules and then enters the empty endless loop, as all the board's functionality is ensured by interrupts from the peripherals. The interconnection of the program modules of MCU program is shown in Fig. 3. The Board51 with this MCU program is also used in the ALFRED architecture as the FEE interface [2].

The *UART* module provides all communication with the parent computer. Within a UART interrupt, the received byte is stored in the receive buffer and a byte from the send buffer is sent to the parent computer. After receiving the entire four-byte command, the module to be targeted with the current command is marked in the first character. The entire command is then routed to the corresponding module.

ADC module manages the ADC functionality. It allows for processing of three commands, the first one to execute a single ADC conversion, the other to run continuous ADC conversion and the third to stop the the current conversion. The commands also contain the channel numbers between which the measurement is to be made, as the ADC operates in differential mode. Subsequently the multiplexer is set to the desired channels before starting the conversion. The

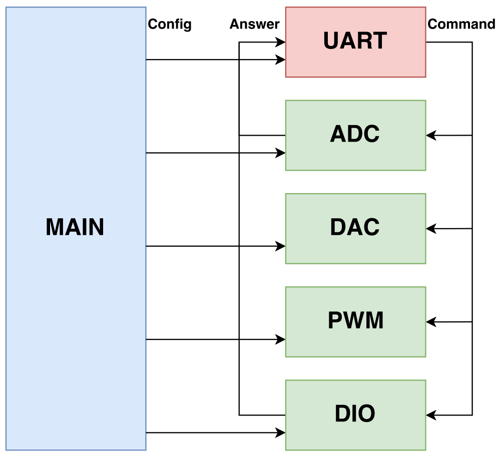


Figure 3: The logical deployment of MCU program's modules.

function itself is not waiting for the end of conversion, but after the conversion is complete, the interrupt is triggered, where the measured data is copied to the send buffer and the *UART* module will send it to the computer.

The *DAC* module serves the operation of the DAC. In the received command, the requested channel and the value representing the requested voltage are encoded. The *PWM* module ensures the operation of timers generating a PWM signal. In the received command, there is information about the selected channel and a duty which should be assigned to that channel.

The *DIO* module manages the setting of three digital inputs/outputs channels. This module provides three commands. The first is to set the mode of selected channel to the input or output. If the output mode is selected, a logic zero is provided on the output terminal. If the input mode is selected, the channel is configured to a high impedance state. Another command is to set the logic value on a given terminal, either at 0V or 3.3V if the channel is set to the output mode. The functionality of last command is to read the logic value on the channel when it is set to input mode. The readout value is written to the send buffer and the *UART* module sends it to the computer.

MATLAB LIBRARY

Due to the ease of use of this board with the MATLAB simulation environment, an interface that would ensure communication between the Board51 and MATLAB was created. This interface is realized in form of a library in the MATLAB simulation environment to enable the board to be controlled directly by the functions provided by the library.

The library is implemented as a Board51 class in the MATLAB simulation environment, with individual board functionalities being accessed using member functions of the class. The creation of the class object is based on the name of the serial port to which the board is attached. Subsequently, the serial line is initialized and opened, if the port could not be opened, an error message is displayed. After successfully

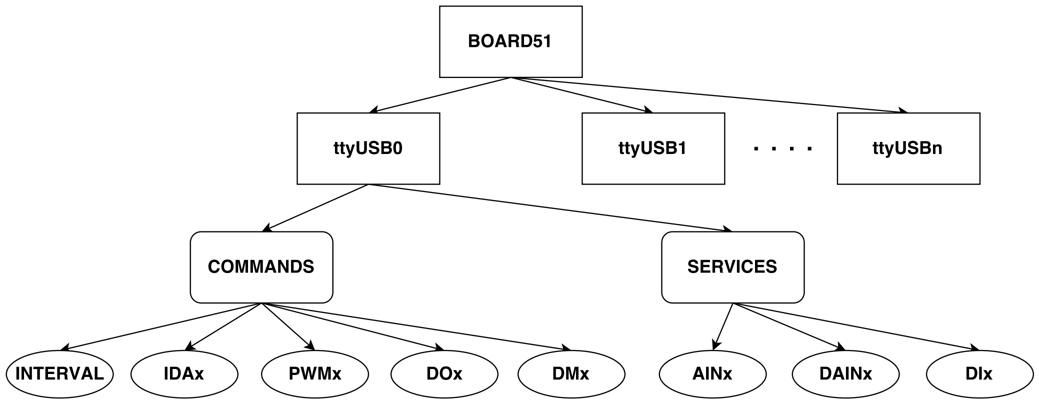


Figure 4: The hierarchy of DIM services and commands created by the program.

opening a serial line, workspace creates a class object that can then be accessed. There are three functions to work with the ADC. The *ADCSingle* function serves to execution of a single measurement using the specified ADC channels. After reading the measured value from the board, this value is converted to the voltage that is returned by the function.

In the *ADCContinuous* function, the desired time interval during which the continuous measurements are executed has to be specified. This function is especially useful for a series of measurements when measurements are required at precise time intervals, since the ADC conversions are triggered in a MCU rather than in the MATLAB simulation environment. After the desired time has elapsed, the function returns the time vector when the measurements were made and the vector of the measured voltages.

The *ADCIdle* function is used to stop the measurement that is currently being performed, or to stop the series of measurements.

The *IDAVoltage* function allows one to set the desired voltage on the selected DAC, ranging from 0 to 3 V. The *PWMDuty* function serves to set the PWM duty on a selected PWM channel in the range of 0 to 100 percent.

Work with digital inputs/outputs is provided by the three functions, *DIOMode* allows to set a specific pin to either input or output and *DIORead* serves to read the logic value of the pin if it is set to input mode. The last function *DIOWrite* allows to write a logical value to selected pin, but only if the pin has been set to output. In all functions, input parameter range control is built in, so if the user attempts to call any of the functions with a parameter that is outside the allowed range, the function will end with an error message.

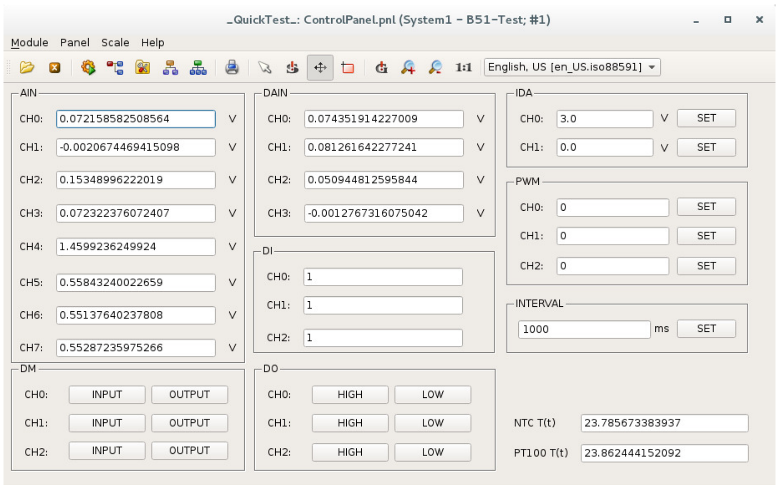
THE DIM INTERFACE

This application enables the Board51 functionality to any device over the shared computer network. The software on the parent computer level provides a number of DIM *services* and *commands* to control the digital and analog outputs of the board as well as to read data from the digital and analog inputs of the board. While it is necessary to send a command to change the value of outputs of the board, reading of the values from the digital and analog inputs is done automatically within the selected time interval.

The supervisor computer's program must be able to communicate with the board via the USB interface and at the same time provide data and control options to other computers on the network via a DIM system. The software is implemented in C++ language so that it can be compiled and run on any computer running Microsoft Windows or any Linux distribution. The program consists of two main modules, the *Control* and the *Serial*. The main thread of the program is to initialize the *Control* module and then go to the endless loop as all the functionality is initiated by the callback functions.

The *Serial* module is used to forward data between the board itself and the Control Module. When initializing, it will connect to the selected virtual serial port that was created after connecting the board to the computer. Within initialization the transfer rate, the number of stop bits, the parity bit, and the required transmission timeouts are set. If the connection to the serial port has not been successful, the program will end with an error message. After finalizing the initialization, the *Serial* module provides functionality to read, write or close the serial connection to the *Control* module. The write function waits for the data to be sent, if it fails, it will cause an error message. The read function waits until it receives the required number of bytes specified by the *Control* module. In case of unsuccessful receipt of the expected data it will also cause an error message.

The *Control* module includes all the other functionality. After its startup, the *Serial* module initializes the connection to the board. Subsequently, DIM creates services that publish the values obtained from the board. There are eight services (*AIN*x) providing the voltages measured between the ADC's channels and analogue ground, four services (*DAIN*x) providing voltages between adjacent converter channels and three services (*DI*x) providing logical levels on digital I/O pins. The generated DIM commands allow the client computers to control the board though the network. It is a command (*INTERVAL*) allowing to change the period of analogue and digital readings from the board, two commands (*IDA*x) to set the voltage on the DACs, two commands (*PWM*x) to set



English, US [en_US.iso88591]

AIN
CH0: 0.072158582508564 V
CH1: -0.0020674469415098 V
CH2: 0.15348996222019 V
CH3: 0.072322376072407 V
CH4: 1.4599236249924 V
CH5: 0.55843240022659 V
CH6: 0.55137640237808 V
CH7: 0.55287235975266 V

DAIN
CH0: 0.074351914227009 V
CH1: 0.081261642277241 V
CH2: 0.050944812595844 V
CH3: -0.0012767316075042 V

DI
CH0: 1
CH1: 1
CH2: 1

IDA
CH0: 3.0 V
CH1: 0.0 V

PWM
CH0: 0
CH1: 0
CH2: 0

INTERVAL
1000 ms

DM
CH0:
CH1:
CH2:

DO
CH0:
CH1:
CH2:

NTC T(t) 23.785673383937
PT100 T(t) 23.862444152092

Figure 5: The WinCC OA visualization window.

the duty of PWM channels, three commands (*DO*x) to set the digital channels levels and three commands (*DM*x) to set the digital channels modes. The hierarchy of DIM services and commands created by the program for the supervisor computer is shown in Figure 4.

After accepting the *INTERVAL* command, the software timer is set in a program whose callback execution period is based on a value received in the command. After calling the callback function, all selected values are measured using the ADC on the board and then published to corresponding service. Logical values are also measured on the individual digital inputs and are sent to the appropriate service. After receiving some of the other commands, the corresponding callback function is called, in which the selected operation is performed to set the appropriate periphery on the board.

WinCC OA Visualisation

In order to test the correct operation of the software for measuring and distributing data using the DIM system, a simple visualization has been developed in the WinCC OA SCADA/HMI system. This visualization shows the measured voltages on all channels including voltages on differential channels. The measuring board itself is connected to the Raspberry Pi single-board computer, where the aforementioned program was used to mediate board's options via the DIM channel.

After execution, the visualization program connects to the DIM server *BOARD51/ttyUSB0* and searches for all available services and commands registered at the DIM Name Server, those are mapped to the Data-Points in WinCC OA. Subsequently, it is possible to set the interval in which the individual measurements are performed, after which all the measured voltages as well as the logical values on the digital inputs of the board are displayed. With the visualization, it is also possible to set the voltage on DAC's, duty on PWM channels, or logical values on digital outputs. The window of the visualization is shown in Fig. 5.

CONCLUSION

The purpose of this paper is to present a new general purpose data acquisition and control board named Board51, together with the software for connecting the board to the MATLAB simulation environment as well as the application for data distribution and control of the board through the DIM system. This application makes it possible to connect the board with an existing communication architecture used in the detector control system of the ALICE experiment including the direct connection to the WinCC OA SCADA/HMI system.

REFERENCES

[1] C. Gaspar, M. Dönszelmann and Ph. Charpentier, "DIM, a portable, light weight package for information publishing, data transfer and inter-process communication", Computer Physics Communications, CERN, Geneva, Switzerland, 2001, `https://dim.web.cern.ch/dim/papers/CHEP/DIM.PDF`

[2] J. Jadlovský, A. Jadlovská, S. Jadlovská, M. Oravec, D. Voščcek, M. Kopčík, J. Čabala, M. Tkáčik, P. Chochula and O. Pinazza, "Communication architecture of the Detector Control System for the Inner Tracking System", ICALEPCS 2017, Barcelona, Spain, 2017