ARDOLORES, an Arduino based motors control system for DOLORES

Gonzalez M., Ventura H., San Juan J. and Di Fabrizio L.

Affiliation, FGG-INAF, Telescopio Nazionale Galileo, Rambla José Ana Fernández Pérez, 7, 38712 Breña Baja, TF - Spain

ABSTRACT

We present ARDOLORES a custom made motor control system for the DOLORES instrument in use at the TNG telescope. ARDOLORES replaced the original PMAC based motor control system at a fraction of the cost. The whole system is composed by one master Arduino ONE with its Ethernet shield, to handle the communications with the external world through an Ethernet socket, and by one Arduino ONE with its custom motor shield for each axis to be controlled. The communication between the master and slaves Arduinos is made possible through the I^2C bus. Also a Java web-service has been written to control the motors from an higher level and provides an external API for the scientific GUI. The system has been working since January 2012 handling the DOLORES motors and has demonstrated to be stable, reliable, and with easy maintenance in both the hardware and the software parts.

Keywords: Arduino, Automation, Motor control, Electronics

1. INTRODUCTION

DOLORES\(^1\) is an imager and low resolution spectrograph mounted on the Nasmyth B rotator interface of the Telescopio Nazionale Galileo (TNG) telescope\(^2\). The TNG is a 3.58 m telescope located at an altitude of 2387 m at the ”Roque de Los Muchachos” astronomical observatory on the San Miguel de La Palma island (Canary Islands, Spain). This is the italian astronomical community telescope inside this observatory and it is operated by INAF (Istituto Nazionale di Astrofisica ). DOLORES is, in short, a focal reducer that is interfaced with the f/11 telescope focal plane and transforms the telescope focal aperture to an f/32 one. The instrument is a multipurpose imager and low resolution spectrograph. Various optical devices can be inserted into the optical path to perform multi band photometry and polarimetry, long slit or multi slit low resolution spectroscopy, or low resolution spectropolarimetry. To make all these observing modes possible eight axes carry the optical devices in or out the optical path. In addition to these moving axes DOLORES has up to 6 spectroscopy calibrations lamps and 6 temperature sensor to be managed. A motor control system able to handle all the axes should have the capability to drive DC motors, read absolute and relative encoders, read limit switch signals, handle the motors brakes, switch lamps on and off and read temperature sensors. An observing setup is realized moving various axes to a final position. This implies that all the motors must be controlled in a parallel way to save time.

The original DOLORES motors control system was based on two PMAC–I boards\(^3\) mounted on a personal computer running the Windows NT operating system. This computer was receiving commands from the scientific GUI via a socket connection and was controlling the various DOLORES parts trough the PMAC boards connected to a power stage. This power stage was an electronic box where the motor drivers where placed to power the motors. All the system was undergoing obsolescence problems that forced us to consider to renew it. The final decision to refurbish this DOLORES part has been to use the Arduino\(^4\) platform and a custom made shield developed in house. This has been done at a fraction of the cost of the PMAC boards.

Every axis counts on an Arduino with its motor shield . Besides, a master Arduino handle the communication

\(^1\) www.tng.iac.es/instruments/lrs
\(^2\) www.tng.iac.es
\(^3\) www.deltatau.com
\(^4\) www.arduino.cc
with them through the I²C bus and with the external world through an Ethernet socket. This socket connection allows to easily communicate with the axes. In our case we decided to develop a middle level Java web-service that works as an interface between the low level Arduinos commands interface and the high level user interface. This program provides an external API for the scientific GUI and allows to have a more modular system. The system has been working since January 2012 handling the DOLORES motors and has demonstrated to be stable, reliable, and with easy maintenance in both the hardware and the software parts.

2. HARDWARE

The idea underlaying the Arduino micro-controller boards is to make custom electronic projects development easy. Arduino was born for educational purpose and the original idea was that it had to be cheap, easy to be managed and fully documented to let people look under the hood and learn from this experience. To grant these features the Arduino micro-controller board is protected by a creative common license. The board is easily programmed using a free IDE and an user friendly programming language. Arduino is a successful project and is used in as many applications as people can imagine. People is encouraged to develop its project and to share it with other people in collaborative community attitude. We were fascinated by the possibility to have an easy to maintain, expand and to program system to manage motorized axes and we wondered if this board was suited to be used to make a "serious" project as the replacement of all the motor control system of an astronomical instrument can be.

![Figure 1. Electronic schematics of the TNG motor shield.](http://scientificweb.com/images/tng-motor-shield-schematic.png)

http://creativecommons.org
2.1 Architecture

When we started the project the goal was not only to develop a new system to refurbish the DOLORES motor control system, but also to be able to handle any new automated axis to be placed at the telescope. For this reason it had to be modular, flexible and a general purpose design. Another important requirement was to include an Ethernet interface. We chose to have a master board with an Ethernet interface and single slave boards to handle each axis, all of them communicating through I²C bus. The I²C is limited to 127 devices on the same network, and the standard speed is 100 KHz. This is enough for our requirements. The master board function is to receive a command from the operator and deliver it to the specific board handling the selected axis, and also to gather telemetry from the different working boards to make it available to the operator.

![Figure 2. Picture of a multiple axis system with one master board and three slave boards.](image)

For the master board we chose an Arduino UNO coupled with an Ethernet shield or the equivalent Arduino Ethernet. For the slave boards we chose an Arduino UNO attached to a custom-made shield developed in house to handle motors, different kind of encoders, brakes, and limit switches. In case only one axis is to be controlled there’s no need for a master/slave architecture. The motor shield design allows for a simplified configuration with only one Arduino UNO plus an Ethernet shield plus a motor shield doing all the work.

2.2 The TNG MOTORS shield

The TNG MOTOR shield is the kernel of this project. This is a PCB electronic board on which the user has to solder some electronic components. The shield can control absolute or relative encoders so the final board configuration depends upon which kind of encoder the user wants to control. Once realized, this board can be coupled directly to an Arduino ONE (or Arduino Ethernet) as a common Arduino shield. The requirement for this board were:

- To be able to manage incremental (single ended or differential) or absolute (serial based) encoder telemetry
- To be able to handle brakes signals

**In Arduino slang, a shield is an electronic board that can be coupled to an Arduino to perform specific functions, i.e. Ethernet interface."
To be able to manage limit switch signals

To be able to communicate with other Arduino boards via the I²C bus and to be coupled directly with an Arduino Ethernet board

2.2.1 driver

The driver stage is based on a LMD18200. This is an H-bridge that can deliver up to 3A in continuous current and has a number of features that has worked very well for us in other projects (thermal shut down, internal clamp diodes, shorted load protection, current sensing). Due to board space constrains, PCB traces regarding motor power are limited to 2A. The user that needs the full 3A power output will need to add an appropriate wire or to add soldering material to the PCB trace.

2.2.2 encoder

To satisfy the requirement to manage incremental encoders (both single ended and differentially transmitted) as well as absolute encoders (Heidenhain SSI absolute in our case) with the same electronic base board we placed all the necessary circuit on the PCB board. The user can decide to mount only the needed parts related to the encoder that will be used. Our chose to handle differential transmission signals for incremental encoders has been the SN75175 integrate circuit (similar to 26LS32) and SN75179 from Texas Instrument for absolute ones.

2.2.3 limit switches

To handle the limit switches signals (inductive in our case) the board has an optocoupler followed by a Schmidt trigger comparator to enhance the signal waveform (74LS14)
2.2.4 brakes

The brakes are handled by an NPN transistor. An input optocoupler channel is provided to have telemetry from the brake if needed.

2.2.5 caveats

Each board has place to solder the required I²C pull-up resistors. If more than one axis is to be implemented, and the Arduino stages have a common power supply, a diode is required to insulate the axis from each others and to be able to program each axis by an USB computer connection without removing the board from the system.

Due to the limited space on the board the following not so frequently used signals: board led, encoder index channel and the brake signal for the motor driver, have been routed to the same Arduino input. The user can choose which one need to use by short-circuiting the corresponding jumper headers.

A dedicated web page with all the necessary instructions and electronic schematics is under construction. By now if the reader is interested in the detail of this project can contact the authors of this paper.

3. ARDUINO SOFTWARE

A complete Arduino software library has been developed using the standard Arduino IDE. For the Master Ethernet and I²C communications we have benefit from the standards Arduino libraries available. The added custom code is needed to interpret the commands and to control the loops in the slave boards.

The slave stage software is basically a state machine with the the following states:

0 STAND BY: idle

1 START: perform some necessary calculations and initialize related variables before moving.

2 MOVING: motion control loop

3 STOP: reset some variables needed in other states

4 HOMING: move until limit switch is reached. This is needed to initialize the incremental encoders.

5 GETOUT: move until limit switch is not longer active. This is needed to set the zero position of incremental encoders.

In the special case of DOLORES all the axes are on the rotator telescope interface. Each axis stays on a plane parallel to each other and parallel to plane containing the gravity vector. During the telescope operation the angle between each axis and the gravity vector changes and this implies that the effective momentum changes. In this challenging situation a normal position Proportional-Integral-Derivative (PID) loop may not work fine. To solve it we implemented a double PI loop, one controlling the position and another one controlling the speed, and a trapezoidal speed pattern movement, making the system more reliable and adaptable to variations in axis momentum.

For the command interpreter, we use a character by character recognition schema rather than one based on full string comparison, trying to speed up this part of the code and allowing more processing power to the control loop.

Limit switches and overheat alarms are polled continuously. The Arduino UNO can handle only two hardware interrupt inputs and currently they are being used for the encoder inputs.
4. THE JAVA WEB SERVICE

To provide a mechanism of abstraction between the hardware and the software at high level we have developed a Java-based webservice, which has kept the old scientific interface without making any changes on it, providing also standard control methods that enables interoperability with other systems of the telescope. This webservice has been developed under JavaEE specifications using Glassfish Open Source Application Server, and from which has been implemented an XML-based API that allows querying the telemetry of the axes and sending commands to the motors. By enabling the interoperability of multiple systems on the same hardware, execution queues has been implemented in addition with the parallelization of different commands, where possible, allowing a great efficiency when initializing the system or make changes on the configuration of the axes. The parameters of each of the axes are stored in a configuration file which is loaded each time the service is initialized, this allows making changes in the configuration a simple way. Once that the web-service has been deployed, a maintenance interface has been developed for testing and system adjustments using Java and Swing technologies, also a web maintenance interface are planned in the near future.

![Diagram of ARDOLORES high level software](image)

Figure 4. ARDOLORES high level software diagram.

5. CONCLUSIONS

We have presented an Arduino based motor control system. This has replaced the DOLORES obsolete original one based on PMACs commercial boards at a fraction of the cost. The whole project has been realized at a cost of less than 3500 $. The resulting motor control system is working since the early 2012 at the TNG telescope and has resulted to be stable, reliable and easy to be modified or maintained. The modularity and flexibility of the resulting system has permitted us to set this automation custom made system as the preferred one at the TNG telescope and at the moment is used to automate various movable telescope parts as the HARPS-N shutter on the rotator interface and the future HARPS-N polarimeter optics, as an example.

REFERENCES