2014 5th IEEE RAS & EMBS International Conference on
Biomedical Robotics and Biomechatronics (BioRob)
August 12-15, 2014. São Paulo, Brazil

Power Wheelchair Open Platform*

Marcelo A. José, *Member, IEEE*, Alexandre A. G. Martinazzo, Leandro C. Biazon, Irene K. Ficheman, Roseli D. Lopes, *Member, IEEE* and Marcelo K. Zuffo, *Member, IEEE*

***Abstract* — Power wheelchairs have high costs in developing countries, making its access difficult to the physically impaired, its main target audience, and to groups that develop assistive technologies, such as researchers, independent developers and entrepreneurs. Free software and open hardware have been successfully used to bridge access gap in areas where cost was a problem, democratizing the innovation process. This paper presents an open hardware and software platform developed for power wheelchairs, which aims to contribute to research and development in this area and stimulate local industry to implement affordable assistive technologies. We believe this work can be used as model for future research on open electronics platforms.**

I. Introduction

Open hardware and free software for power wheelchair (PWC) operation can trigger research and development of electronic controls and interfaces. The results of such research can help industry develop inexpensive solutions and make PWCs affordable. In Brazil, wheelchairs are manufactured locally but PWCs are assembled with imported electronic parts. Therefore, we believe that open hardware technology can support local electronic controls production, reduce costs and reach large populations especially in developing countries like Brazil.

"Free software" is a term defined by the Free Software Foundation [1], the main sponsor of the GNU Operating System (most commonly distributed as GNU/Linux systems), that refers to the freedom one should have with a particular program. More precisely, one may run the program for any purpose, study and freely modify the program and distribute copies of the program, modified or not.

In order to study, modify, and distribute modified versions of the program, one must have access to the source code. It is often referred to as a social movement, due to its "multi-project ecosystem" that shares many software development artifacts and ideological beliefs [2].

Similarly, "open hardware" applies to physical artifacts which design documentation is publicly available for those who want to make, modify or distribute it [3]. The items required to be available may vary from a project to another, but typical design documentation includes printed circuit board (PCB) schematic and layout files, parts list, assembly instructions and micro-controller firmware (embedded software) [4].

In the scientific research context, the freedom proposed by "free software" and "open hardware" concepts provides means of collaboration among research groups. Access to all artifacts of a particular project allows contributions to every part. Thus, the openness can result in collaboration among different people and institutions interested, such as seen in [5], [6], [7] and [8].

Researchers have reported developing and sharing robotic platforms among different research groups as a strategy to trigger research and make quicker advances in a specific field. In robotic surgery, for example, researchers have chosen the open-source model to develop Raven-II, a common collaborative research platform used by seven universities to conduct surgical robotics research [6]. Along with the platform, the group created user interaction and dissemination mechanisms to increase and support users. Another research group developed a humanoid platform called Poppy, designed to study biped locomotion and physical interaction [7]. Poppy was conceived and optimized to be accessible, low cost and easy to assemble, as the goal was to make the hardware and software platform available for academics.

Another example is the iCub robot, a common platform developed for researchers interested in the study of artificial embodied cognitive systems that has been adopted by 20 labs worldwide [8]. The iCub project encompasses various research areas, such as posture and locomotion, attention and gazing, social interaction and imitation.

Open hardware development culture has not reached the PWC research yet. In many cases, researchers have attempted to design and develop inexpensive power wheelchairs [9] and intelligent power wheelchairs [9][10][11] to assist physically impaired individuals who encounter difficulty in driving regular power wheelchairs. One example focused on low cost hardware and open-source software used to create smart PWCs that users control with a keyboard, a web cam and a microphone [9]. There are examples focused on automatic control systems that empower physically disabled individuals and reduces their effort to steer their chairs [10][12]. An "open electric network standard" that can be applied in PWCs is proposed in [13].

In addition, researchers that intend to develop human machine interfaces (HMI) for PWCs need to use commercial controls, such as the HMC EasyRider control system [14] or the PG Drives Technology VR2 motor controllers [15].

*Research supported by the Secretaria de Estado dos Direitos da Pessoa com Deficiência do Estado de São Paulo.

Authors are researchers of the Centro Interdisciplinar em Tecnologias Interativas and of the Laboratório de Sistemas Integráveis da Escola Politécnica da Universidade de São Paulo, São Paulo, Brazil (5511- 3091-4248). E-mails: archanjo@lsi.usp.br, am@lsi.usp.br, biazon@lsi.usp.br, irene@lsi.usp.br, roseli@lsi.usp.br, mkzuffo@lsi.usp.br.

One problem in research and development of intelligent PWCs when using proprietary platforms and available power electronic development kits is that they usually are not capable of supplying enough current to wheelchair motors.

Publishing and sharing knowledge by developing open source hardware should address these issues and boost incremental research in this field.

In this paper we present the research and development of an open platform for PWCs. We first present the platform architecture, detailing hardware and software. We then discuss strategies for its diffusion and community engagement, as well as the expected impacts.

The circuits were developed to control standard commercial PWC with DC motors within a power range from 330W to 500W each. The developed modules performed similarly to commercial electronics.

II. Architecture

This section describes the hardware and software architecture designed for the power wheelchair open platform.

A. Hardware

We designed the hardware architecture divided in two main blocks: Control Module and the Power Module. The Power Module is responsible for controlling the motors. It processes the data from a human-machine interface (HMI) directly connected to it or from the Control Module and uses the resulting information to command the motors through high-current circuits. The Control Module is responsible for advanced functionalities, such as Bluetooth, infrared (IR) and another HMI. The Power Module also supplies energy to the Control Module. The hardware architecture is presented in Figure 1.

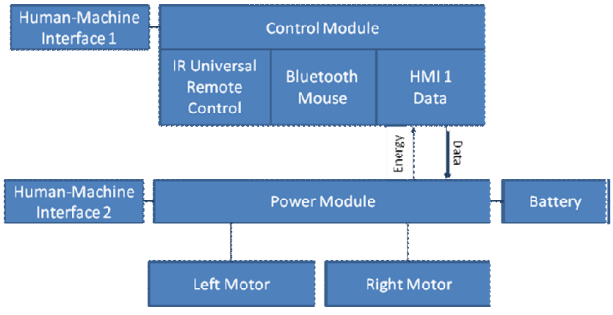

Figure 1: Hardware Architecture

We designed the hardware architecture so that Power Module can be used separately with a HMI. If the user needs advanced functionalities mentioned above, then both Control Module and Power Module should be connected.

The Control Module has a display in order to inform the user about the selected functionality. It has three operating modes: (1) command a computer mouse through Bluetooth, (2) operate as a universal remote control (via IR signal, to control electronic equipment such as TVs), and (3) control the Power Module.

The Power Module is able to drive two standard PWC 600W DC motors (with electromagnetic brakes).

Both modules are able to handle analog and digital HMIs, such as joysticks or five switches interfaces (commercially known as Penta Switchs).

We also developed three open digital HMIs: a five switches interface (penta), a touch interface and a sip-and-puff interface.

There are two different usage scenarios: complete and "seat-and-drive". In the complete one, the main HMI is connected to the Control Module, which communicates with the Power Module to drive the PWC. In this scenario, an optional secondary HMI can be connected directly to the Power Module to allow the caregiver to control the chair. In the "seat-and-drive" scenario, the main HMI is connected directly to the Power Module and the Control Module is not used, consequently the advanced functionalities are not available in this situation, just the PWC movement control.

B. Physical Connection

This section describes modules' connections. These details are important either to create modules and HMIs or to understand physical connections and signals of commercial devices.

PWCs are commonly powered with 24V using two 12V batteries in series. The batteries are connected to the Power Module that is responsible to control the DC motors.

Each DC motor has four wires, two for the brake and two high current wires for the motor itself. The Power Module controls rotation direction and rotation speed with PWM in a MOSFET h-bridge.

An analog joystick commonly has five wires (12V, ground – GND, X axis, Y axis and reference). The reference provides a voltage of 6V (half of 12V), and X and Y axes provide the same voltage of the reference when the joystick stands in the neutral position. If the stick is turned to the maximum position in one direction, the correspondent axis will provide the 6V plus 1.2V (10% of 12V) or minus 1.2V according to the direction. The maximum diagonal position of the stick will produce the change in the voltage in both axes.

A five-switch digital HMI has four switches for direction commands and one selection command. Each switch has one pin connected to the ground (0V) and the other connected to a 3.3k Ohms resistor and then to the correspondent pin.

We developed a touch HMI using five charge-transfer touch keys. Those keys are connected to an electronic circuit that processes the signal to make it compatible with a conventional mechanical HMI. The charge-transfer switches are arranged as four directional arrows and a separate selection command, as shown in the Figure 2.

In addition, we developed a sip-and-puff digital HMI that also operates like a conventional five switch interface. The sip-and-puff device, presented in Figure 3, has a pressure sensor able to handle positive and negative pressure, a microcontroller and additional circuitry. The operation

Figure 2: Touch HMI device

demands prior calibration based on user's lung capacity. Intermediary levels of sip and puff are used as basis for pressure analysis: strong puff means forward, weak puff means right, strong sip means backward, weak sip means left and a brief puff followed by a sip means selection command.

Figure 3: Sip-and-puff digital HMI

Every HMI device has a cable with a DB9 female connector, as shown in Table 1. The detect pin is expected to be 0V (ground or GND), but for security reasons this connection must be done in the interface side (not in the connector), since if the cable is accidentally cut the correspondent module will detect an error and stop the PWC.

Table 1: HMI cable DB9 female pin out

Pin	Analog HMI	Digital HMI
1	Y Axis (forward/backward)	Forward
2	X Axis (left/right)	Backward
3	Reference	Left
4	Not Connected	Right
5	Detect	Detect
6	Not Connected	Selection switch
7	12V	12V
8	GND	GND
9	12V	12V

The connection of the Power Module and the Control Module is done with four wires, two for I2C (Inter-Integrated Circuit) communication and two for energy.

C. *Software*

In this section we describe the embedded software we developed for the Control and Power Modules.

Software was designed to provide the functionalities commonly available in commercial PWCs, namely infrared (universal remote control) and Bluetooth (allowing mouse capability in a PC). It also includes driving capability and driving profiles.

The development was based on the Arduino environment (http://www.arduino.cc/), using some of its code structure and libraries. It was chosen because of its large availability and low cost. The basic Arduino integrated development environment (IDE) does not provide enough flexibility to properly organize the code (like separating code in different files), so we decided to customize the scripts the Arduino IDE would use. The Arduino environment uses scripts on top of free software tools (such as avr-gcc and avrdude) to compile and send the binary file to the board.

We made a customization to choose the micro-controller target at the compilation scripts (known as makefiles) and set the libraries and variables to our development setup. We used the Ubuntu distribution as operating system and git as version control tool for code management.

Software development was tied to hardware evolution, and we adopted Scrum to manage the process. Weekly meetings helped organize what features would be implemented in a particular sprint.

Figure 4 shows how we have structured the embedded software.

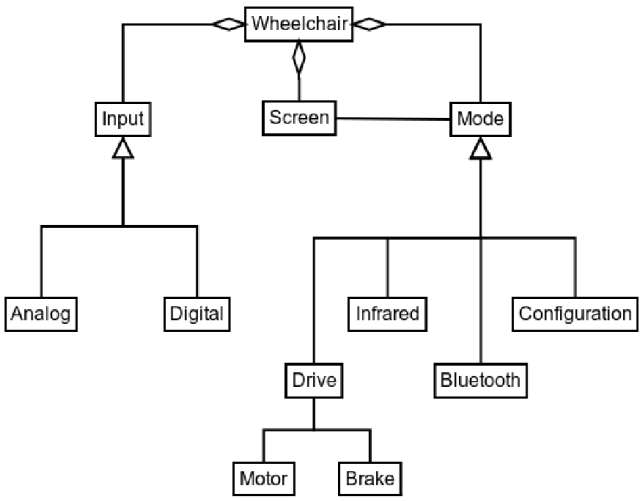

Figure 4: Open platform for PWC class model

The Wheelchair class has an Input, a Screen and a Mode. The Input class represents an input device, such as a joystick (analog) or a penta switch (digital). The Screen class is the interface to the display connected to the module and the information it shows. The Mode class generalizes the active

resources. As shown in Figure 4, the Wheelchair can operate in Drive, Infrared, Bluetooth and Configuration modes, with only one of those running at time. Mode can also access Screen to show the relevant information to a particular operating mode.

D. Communication

This section presents the software communication strategy adopted to send and receive messages along the data bus.

We implemented a master-slave multiprocessor architecture where processing units exchange messages. The master processor is responsible for displaying information, reading inputs and setting the operating mode (in other words, controlling which functionalities should be available from a user point of view). This architecture was implemented with 3 micro-controllers that provide the functionalities described before. Those micro-controllers are ATMega 328, the same found in Arduino Uno boards. They are connected through an I2C bus.

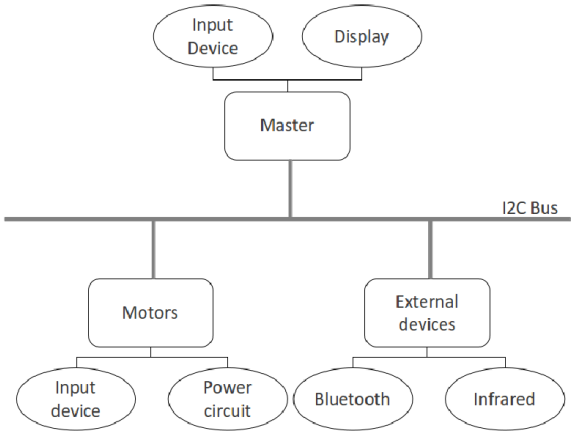

Figure 5: Communication architecture

Processing units are identified according to the function or the functionalities they control, as shown in Figure 5. Hereafter we detail master unit, motors unit and external devices unit. Master unit and external devices unit run on the Control Module, and motors unit runs on the Power Module.

The master unit processor controls the communication along the I2C bus. It is also responsible for reading the input and controlling the display. Input devices can be either analog or digital, and the software is able to automatically recognize how to process its values. Input is internally represented as a pair of 10 bits values, one to represent direction and the other to represent speed, referred to as x and y respectively. When an analog input device is detected, it also tries to minimize the voltage drifting by calculating the average to represent the x and y pair.

The motors unit processor deals with the power circuits. Its role is to assure the brake and the H-bridges are properly activated. It also provides PWM signals to the power circuit proportionally to the input values and driving profile. Motors unit also processes HMI signals when an input device is connected to the Power Module. When a Control Module is connected to the Power Module, motors unit will process the HMI data from master unit.

The external devices unit processor controls the Bluetooth module and the infrared circuit. When in mouse mode, a Bluetooth connection to a PC is established and the input device behaves like a pointer to the PC operating system. When in infrared mode, the user is able to record the signals from different remote controls and later use them to activate electronic equipments.

The master unit has to provide the destination address and the data when sending messages. Sent data vary according to the operating mode, but it must start with the current mode code. Following that information, a structure is created in order to accommodate the data a particular mode would require. For example, Bluetooth and Drive modes need the values from y and x from the input device while Infrared mode only needs a reference to a command list. Slave units (motors unit and external devices unit) reply with an error/success message after processing received data. The Configuration mode is the only mode that needs no message exchanging, because the master will just show the configuration menu on the display.

III. DISTRIBUTION

Choosing an adequate model of distribution is one of the most important aspects of an open-source project. A model of distribution must provide a) a set of design files to be shared, b) a distribution platform, and c) a license for each project asset.

As discussed before, there are general guidelines that define what an open hardware is but there is no clear statement of what files should be distributed [4]. Considering some of the most popular open projects, we decided to include the following design files for each module: circuit board CAD files, i.e. schematics and printed-circuit board (layout), in both editable and printable formats; parts list; manufacturing instructions (Gerber files), firmware source-code and firmware binaries (hex files).

All the files and instructions are hosted at an online repository (http://git.febrace.org.br/nate/openpwe), with wiki and version control support, enabling users to browse the project history and compare different revisions of the files. Figure 6 shows the main project page from one of the HMI devices.

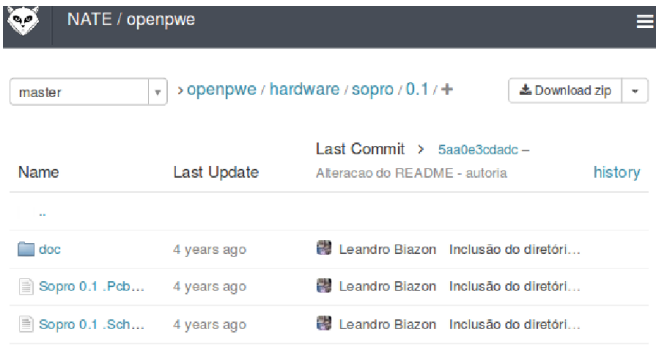

Figure 6: Online project repository sample page.

The project uses two different licenses, one for software and the other for design files and documentation. Firmware source-code is licensed under the version 2 of the Gnu Public License (GPLv2). Design files and documentation use the Creative Commons Attribution, Share-Alike (CC-BY-SA). Both of them are copyleft (viral) licenses, which mean that derivative work is allowed as long as it uses the original work's license. The consequence of this licensing scheme is that commercial products can be developed as derivative work, but they must share their source-code and design files.

IV. Discussion

One of the reasons for adopting an open source model to manage a project is community engagement, which can contribute and greatly improve the initial project [5][6][7][8]. In this section, we discuss some challenges that emerge from working on projects aimed at open hardware communities and present some recommendations based on our experience.

Documentation is a key issue in open hardware projects; its depth, organization and translations need to be considered with care. Thus, open project teams should produce documentation artifacts so that different community members are able to understand, manufacture and modify the firmware, design files or documentation itself, depending on their abilities. There is no established rule for that. To address this matter, we decided to write instructions detailing how to operate and assemble each board in order to facilitate project adoption.

Also, circuit design tools should not be expensive or restricted to specific components, so the community can easily improve design files of projects they participate in. Open or cost-free tools are preferred, if they can handle circuit size. In our project, we used Mentor Graphics PADs PCB design tool, which is proprietary software, because of previous team experience. This decision limits potential contributions from the community since obtaining a license is required to edit design files.

Modularity is also an important feature for open hardware projects. Design teams should consider developing modular architectures, fragmenting circuits in blocks whenever possible, with clear functions and interfaces. Each block should be reusable and independent of the whole project, whether it is a separate circuit board or a functional group of components on a board.

For example, as an advanced feature, we initially planned to integrate the sip-and-puff HMI and the Control Module. To attend a modular architecture, during the project development we decided to design the sip-and-puff HMI as a separate circuit board. As a consequence, the resulting HMI device can be used both in our Control and Power Modules and in commercial PWC controls. In addition, it can be used as a resource in human-computer interaction research.

Another challenge is simplifying the manufacturing process of circuit boards. Its components should be chosen based on local market availability to encourage community engagement. In our PWC open platform, we designed the circuits using components available at local electronics vendors. To keep the manufacturing processes simple, we kept layouts either single or double layered in the PCB design. We also selected through-hole technology components instead of surface mount technology (SMT) ones. These design choices were made in order to make the project reproducible with basic tools such as those available at hobbyist electronics spaces.

V. Conclusion

In this paper we presented the reasons we believe open source hardware is a development model suited to academia research that can also enhance technology transfer to society. Having identified the lack of this kind of initiative in the PWC research field, we decided to develop a new platform applying the principles and methods identified in open hardware communities.

The discussion provided in community engagement was based on our previous experience as users and researchers of open projects. The PWC Open Platform has been recently released and has yet to gather an audience of users and developers. In the next months, we intend to track user interaction with the available content to better understand their engagement and improve the project artifacts based on community feedback.

We expect this research to have three major impacts: 1) as a platform for developing new products in industry, 2) as a platform for research and education for power electronics and HCI in academia and 3) as a model for other researchers that wish to develop and publish open source hardware projects. The first one was a main concern in the research, as we believe the adoption of an open platform by the industry can reduce their research and development costs and boost the production of affordable power wheelchairs.

References

[1] Free Software Foundation, "What is free software? - GNU Project - Free Software Foundation." [Online]. Available: http://www.gnu.org/philosophy/free-sw.en.html. [Accessed: 27-Feb-2014].

[2] W. Scacchi, "Free/open source software development," in *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, Dubrovnik, Croatia, 2007, pp. 459–468.

[3] Tucson Amateur Packet Radio Corp, "The TAPR Open Hardware License." [Online]. Available: http://www.tapr.org/TAPR_Open_Hardware_License_v1.0.txt. [Accessed: 27-Feb-2014].

[4] C. Harnett, "Open source hardware for instrumentation and measurement," *IEEE Instrumentation Measurement Magazine*, vol. 14, no. 3, pp. 34–38, Jun. 2011.

[5] B. Gati, "Open source autopilot for academic research - The Paparazzi system," in *American Control Conference (ACC), 2013*, 2013, pp. 1478–1481.

[6] B. Hannaford; J. Rosen; D. W. Friedman; H. King; P. Roan; Lei Cheng; D. Glozman; Ji Ma; S. N. Kosari; L. White, "Raven-II: An Open Platform for Surgical Robotics Research," *Biomedical*

Engineering, IEEE Transactions on , vol.60, no.4, pp.954,959, April 2013.

[7] M. Lapeyre; P. Rouanet; P. -Y. Oudeyer, "The poppy humanoid robot: Leg design for biped locomotion," *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on* , vol., no., pp.349,356, 3-7 Nov. 2013.

[8] G. Metta; L. Natale; F. Nori; G. Sandini, "The iCub project: An open source platform for research in embodied cognition," *Advanced Robotics and its Social Impacts (ARSO), 2011 IEEE Workshop on* , vol., no., pp.24,26, 2-4 Oct. 2011.

[9] A. R. Trivedi, A. K. Singh, S. T. Digumarti, D. Fulwani, and S. Kumar, "Design and Implementation of a Smart Wheelchair," in *Proceedings of Conference on Advances In Robotics - AIR '13*, 2013, pp. 1–6.

[10] C. M. Chipaila; D. Grigore; G. Marin; R. Solea; D. C. Cernega, "Hardware and software solutions for a conventional electric-powered wheelchair," *System Theory, Control and Computing (ICSTCC), 2012 16th International Conference on* , vol., no., pp.1,6, 12-14 Oct. 2012.

[11] M. R. Petry; A. P. Moreira; B. M. Faria; L. P. Reis, "IntellWheels: Intelligent wheelchair with user-centered design," e-Health Networking, Applications & Services (Healthcom), *2013 IEEE 15th International Conference on* , vol., no., pp.414,418, 9-12 Oct. 2013.

[12] R. K. Megalingam; A. P. Rajendran; D. Dileepkumar; A. T. Soloman, "LARN: Indoor navigation for elderly and physically challenged," *Global Humanitarian Technology Conference (GHTC), 2013 IEEE* , vol., no., pp.326,330, 20-23 Oct. 2013

[13] S. Linnman, "M3S: the local network for electric wheelchairs and rehabilitation equipment", *IEEE Transactions on Rehabilitation Engineering*, v. 4, no. 3, pp. 188–192, Sep. 1996.

[14] M. E. Lund, H. V. Christiensen, H. A. Caltenco, E. R. Lontis, B. Bentsen, and L. N. S. Andreasen Struijk, "Inductive tongue control of powered wheelchairs.," *Conf. Proc. IEEE Eng. Med. Biol. Soc*., vol. 2010, no. Fig 1, pp. 3361–4, Jan. 2010.

[15] X. Huo and M. Ghovanloo, "Using unconstrained tongue motion as an alternative control mechanism for wheeled mobility.," *IEEE Trans. Biomed. Eng*., vol. 56, no. 6, pp. 1719–26, Jun. 2009.