CAMBADA, Hardware Description

J. L. Azevedo, B. Cunha, A. J. R. Neves, R. Dias, P. Fonseca,
N. Lau, E. Pedrosa, A. Pereira, A. Trifan and J. Silva

1. Introduction

The CAMBADA robots were designed and completely built in-house. Each robot is built upon a triangular aluminum chassis, which supports three independent motors (allowing for omnidirectional motion), an electromagnetic kicking device, three, 4 cells, LiPo batteries and the motor controller modules. The remaining parts of the robot are placed in three higher layers, namely: the first layer upon the chassis is used to place the kicking mechanism, the ball handler and related electronic modules; the third layer contains the PC (currently a 12" notebook based on an Intel Core i5 processor); finally on the top of the robots stands an omnidirectional vision system based on a hyperbolic mirror (AIS Fraunhofer-Gesellschaft) and the IMU module.

The mechanical structure of the robot is highly modular and was designed to facilitate maintenance. The main two sections of the robot can be easily separated from each other, allowing an easy access both to the main mechanical components and to the electronic modules.

2. General Architecture of the Robots

The general architecture of the CAMBADA robots has been described in [1][2][3]. Basically, the robots architecture is centered on a main processing unit (a PC running the Linux operating system) that is responsible for the higher-level behavior coordination, i.e. the coordination layer. This main processing processes visual information gathered from the vision system, executes high-level control functions and handles external communication with the other robots. This unit also receives sensing information and sends actuating commands to control the robot behaviour by means of a distributed low level sensing/actuating system. The communication among team robots uses an adaptive TDMA transmission control protocol [5], on top of IEEE 802.11b, that reduces the probability of transmission collisions between team mates thus reducing the communication latency.

The low-level sensing/actuation system (Figure 1) is implemented through a set of microcontrollers interconnected by means of a CAN network (Controller Area Network (CAN) [6]). The main blocks of the low-level sensing/actuation system are: motion control, odometry computation, compass, kicking control and system monitoring. The motion control block is composed of three independent motor control boards each of them receiving a velocity setpoint from the high-level holonomic motion controller. The odometry block combines the encoder readings from the 3 motors and provides coherent robot displacement information that is periodically sent to the high level coordination layer. The compass block reads the compass sensor and sends periodically to the high-level the corresponding read value. The kicking control block includes the control of an electromagnetic kicker and of a ball handler to dribble the ball. Finally, the system monitor, which is in fact a distributed function, monitors the robot batteries as well as the state of all nodes in the low-level layer.

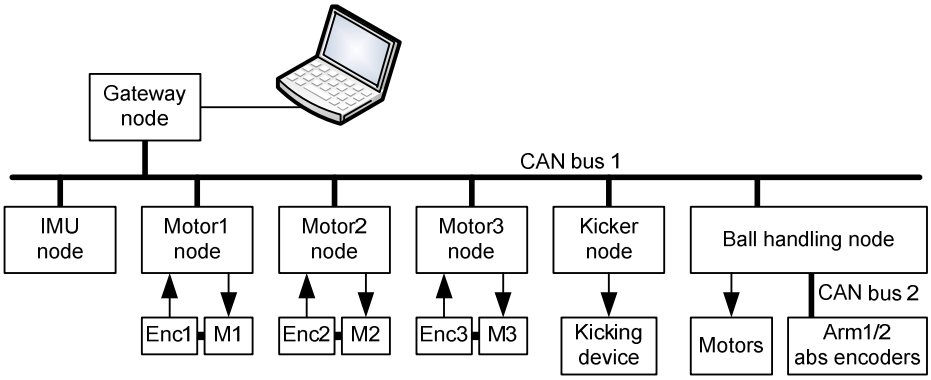

Figure 1. The CAMBADA low-level hardware architecture.

The low-level control layer connects to the coordination layer through a gateway (see **Error! Reference source not found.** for the electrical drawing of this module), which filters interactions within both layers, passing through the information that is relevant across the layers, only.

3. Low-level hardware description

The low-level layer nodes are interconnected with a CAN network operating at a bit rate of 250Kbps. A gateway interconnects the CAN network to the PC at the high-level layer either through a serial port or a USB port, operating at 115 Kbps in any case. All modules (except for the motor control modules) are based on the same underlying hardware, e.g. a PIC18Fxx8 Microchip [4] microcontroller (@40MHz, i.e., 10 MIPS) which, along with a set of useful peripherals, such as timers, PWM generators, analog to digital converter and serial communications, also integrates a CAN controller. The basic structure of every module includes the CAN port to connect to the network and also includes a 115 Kbps RS232 serial port, which is useful both to program the module firmware (through a boot-loader) and for debugging purposes.

One important characteristic of the CAMBADA hardware design is the galvanic decoupling between the "logic" blocks (e.g. microcontrollers, PC, cameras) and the "power" blocks (e.g. motors, kicker) carried out through optocouplers and isolation amplifiers. Along with improved reliability of the whole system it prevents serious damages in expensive equipment (such as the notebook in the high-level layer) whenever any electric problem occurs in the "power" block. The drawback of this solution is the need of an extra battery for the "logic" part of the system. Thus, the low-level hardware modules are powered through 3, 4 cells 5000 mAh Lipo batteries.

The following sections describe in more detail each node of the low-level sensing/actuation system.

3.1. Motion control

The robot holonomic motion is obtained combining the speed of 3 DC motors (Maxon 24V-150W), each with its own speed controller. Each of these controllers is a distinct module of the whole distributed architecture implementing a PI closed loop speed control. It takes as inputs the motor shaft displacement, obtained through a quadrature 500 P/R incremental optical shaft encoder coupled to the main axis of the motor, and the speed setpoint. The hardware of these modules has two main blocks: 1) the "logic"

block (based on a Microchip dsPIC) that interfaces to the rest of the system and generates the required control signals, and 2) the "power" block which is essentially an NMOS H-Bridge, with two high-side drivers, to actually drive the motor. The output of the *logic* block is a set of two 20 KHz PWM signals implementing a modified lock anti-phase drive. In this drive mode the motor is energized only during the on-time, in contrast with the standard lock anti-phase where the motor is energized in reverse direction during the off-time. That is, when the motor is stopped (duty-cycle of the PWM signals is 50%) the current is zero. This implementation leads to a significant gain in battery autonomy, whenever the motor is not rotating at its maximum speed. Figure 3 and Figure 4 present the complete electrical drawing for this module.

The odometry function of the robot is accomplished through the combination of the readings of each one of the 3 encoders to generate coherent robot displacement information (Δx, Δy, $\Delta\theta$). The reading of each encoder is naturally allocated to each motor control module, using the same readings as those used by the speed feedback control. The combination of the readings is carried out in a process running on the laptop PC which receives, via CAN messages, the encoder readings from the motor control modules.

3.2. IMU module

The IMU module (see Figure 7 for the electrical drawing) is based on a Microchip dsPIC30F4013 microcontroller. The IMU module includes a 3-axis compass (a Honeywell HMC5883L) and a MPU-6000, an integrated circuit which put together a 3-axis MEMS gyroscope and a 3-axis MEMS accelerometer.

The connection between the microcontroller and the peripheral sensors is accomplished through the I2C serial bus. The IMU module communicates with the high-level decision layer through CAN.

3.3. Ball handler module

The ball handler module (see Figure 8 and Figure 9 for the electrical drawing) is based on a Microchip PIC32 microcontroller. The ball handler module includes two h-bridges that drive the two motors of the ball handling mechanism and a CAN interface that allows the connection of this module to the remaining low-level sensing/actuating system.

The ball handling mechanism is composed of two independent arms. The position of each of these two arms is sensed through a 12-bit resolution absolute encoder. These two encoders connect to the ball handler module through a second (independent) CAN bus, working with a baudrate of 500Kbps.

4. Electrical drawings

4.1. Gateway

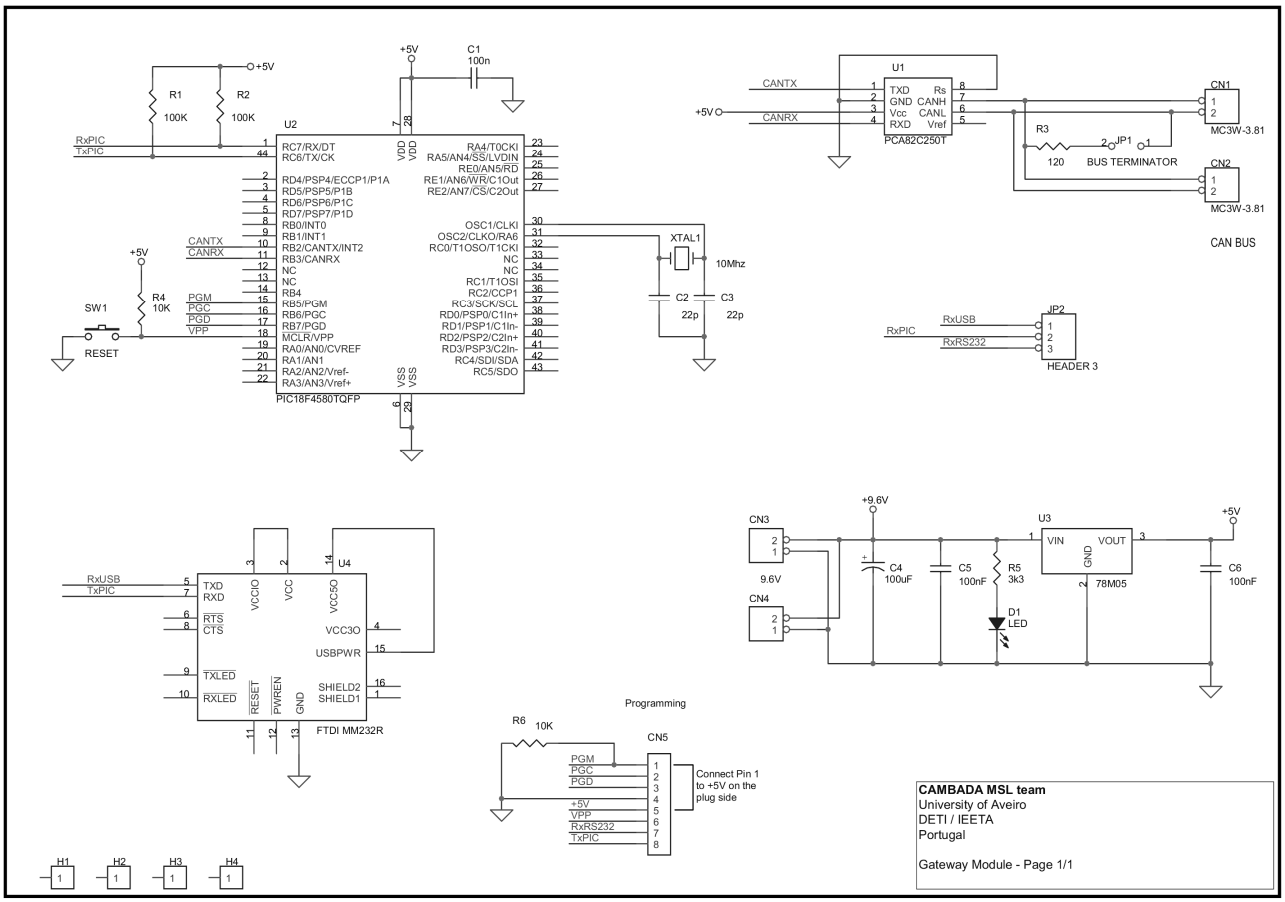

Figure 2. Gateway module.

4.2. Motor control

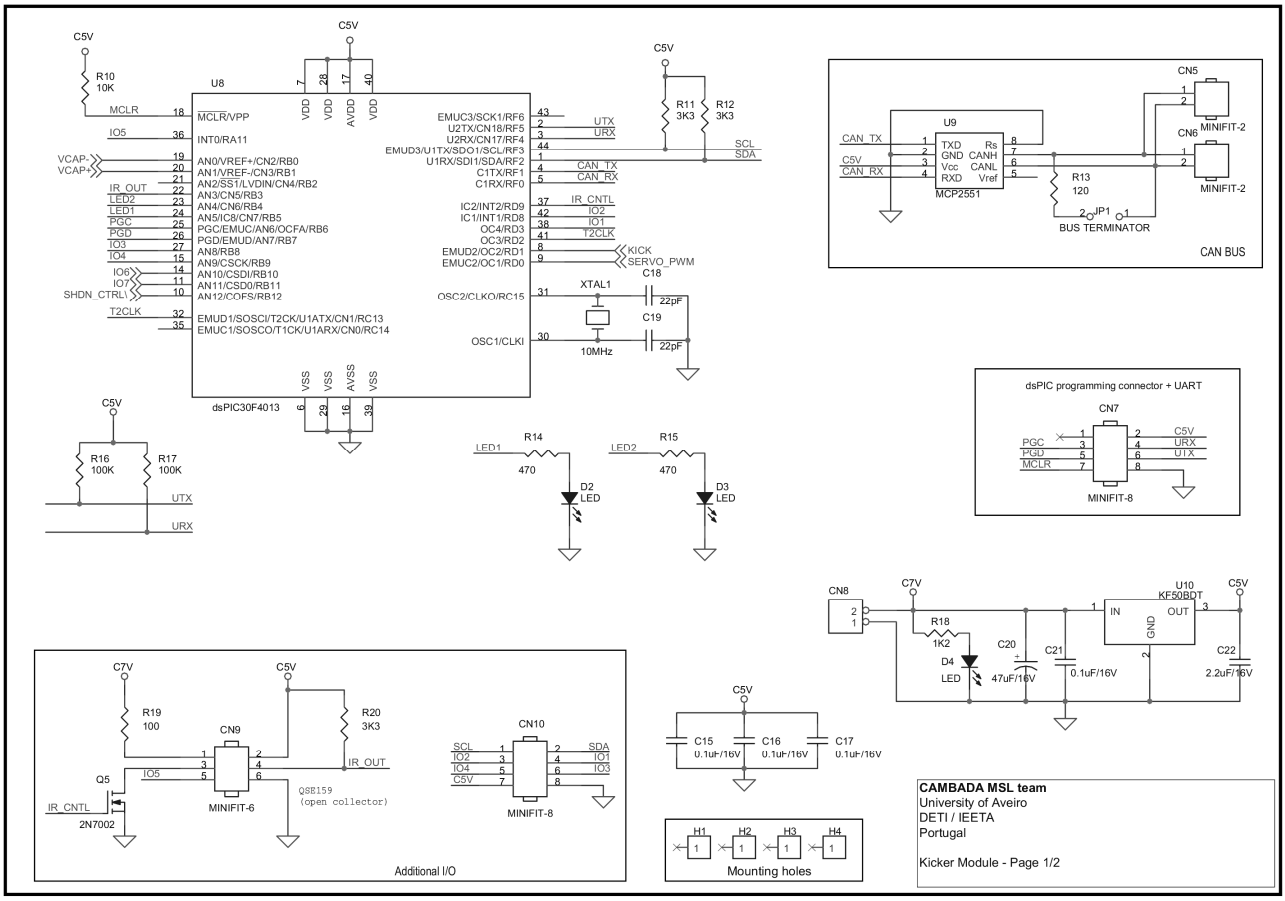

Figure 3. Motor control module (part 1).

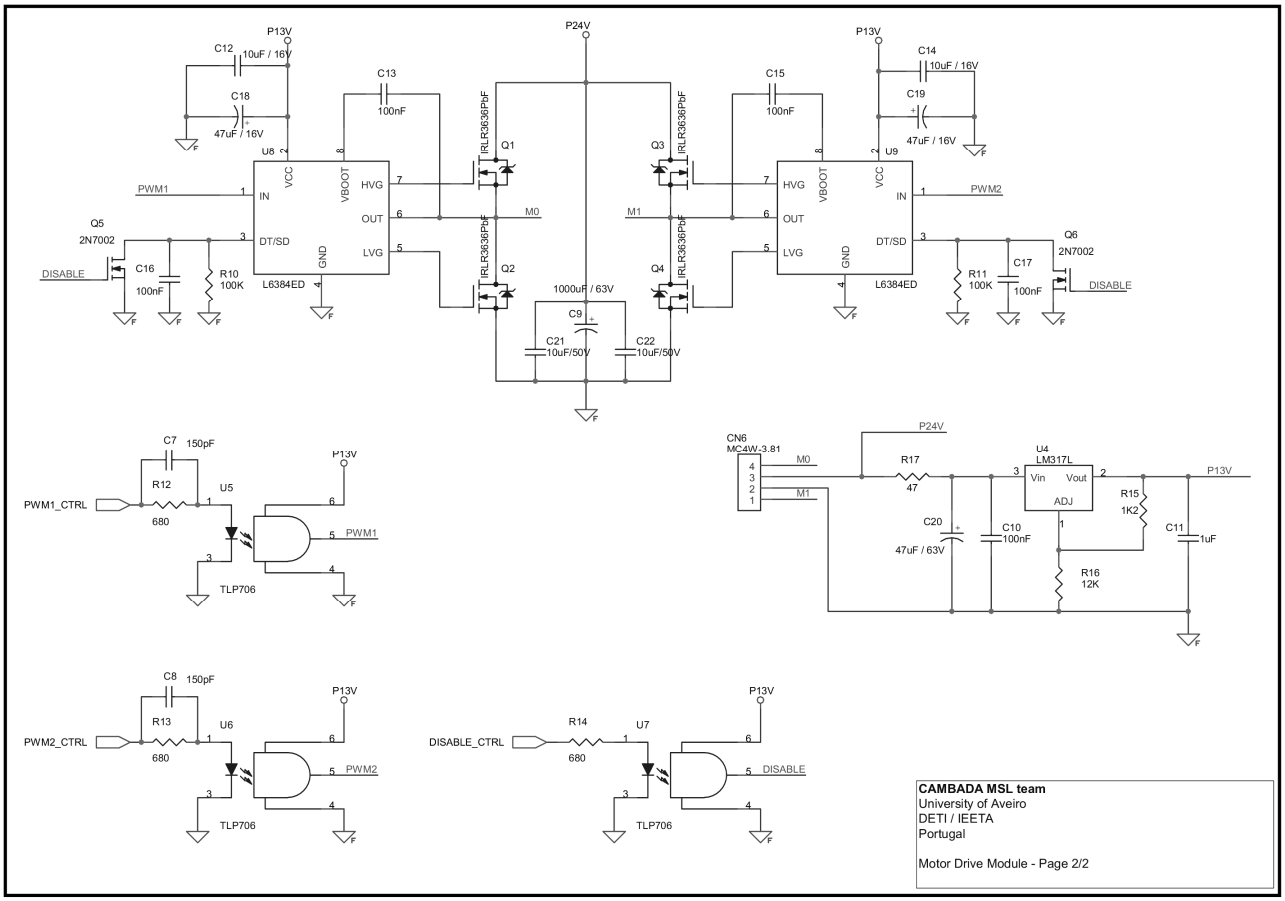

Figure 4. Motor control module (part 2).

4.3. Kicking module

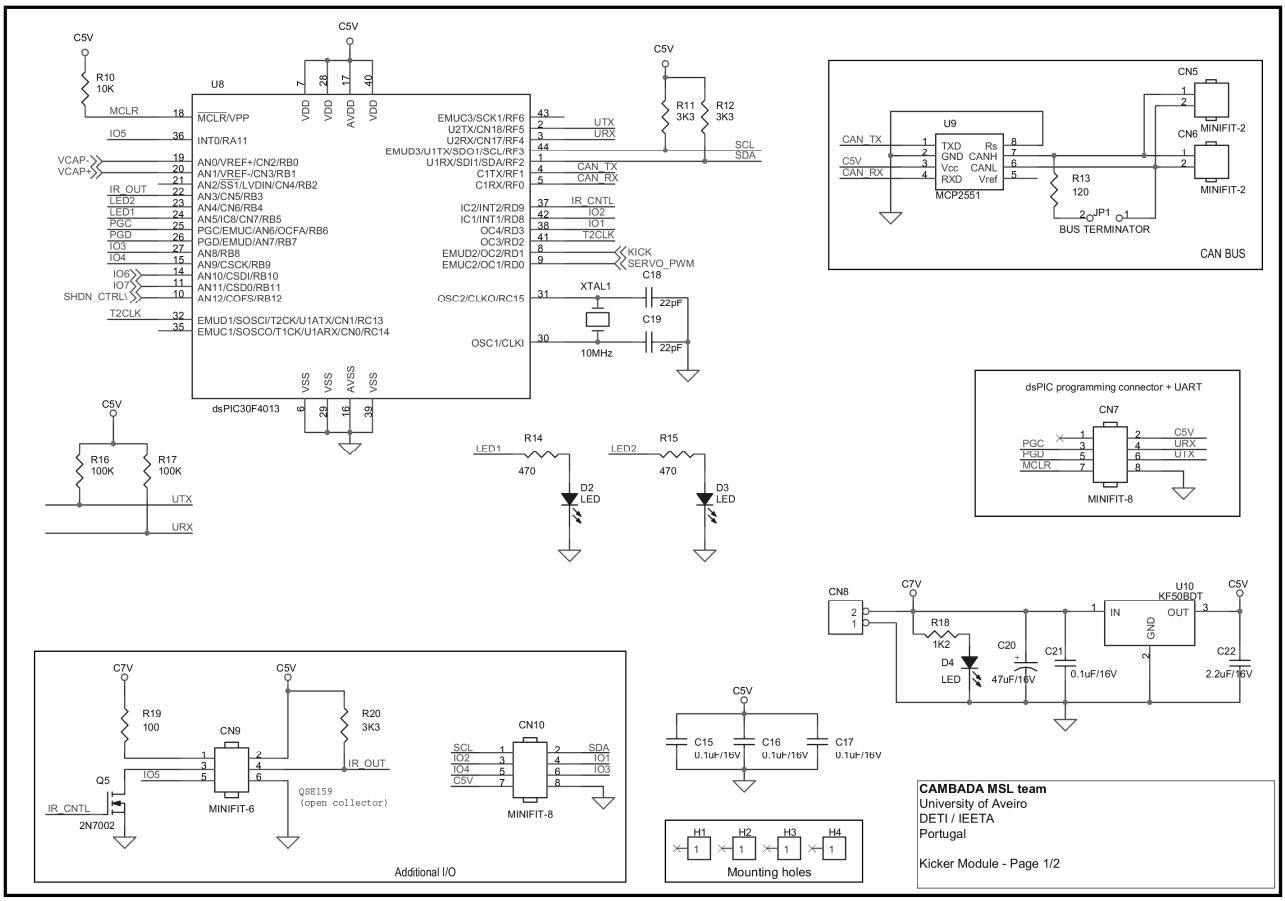

Figure 5. Kicking control module (part 1).

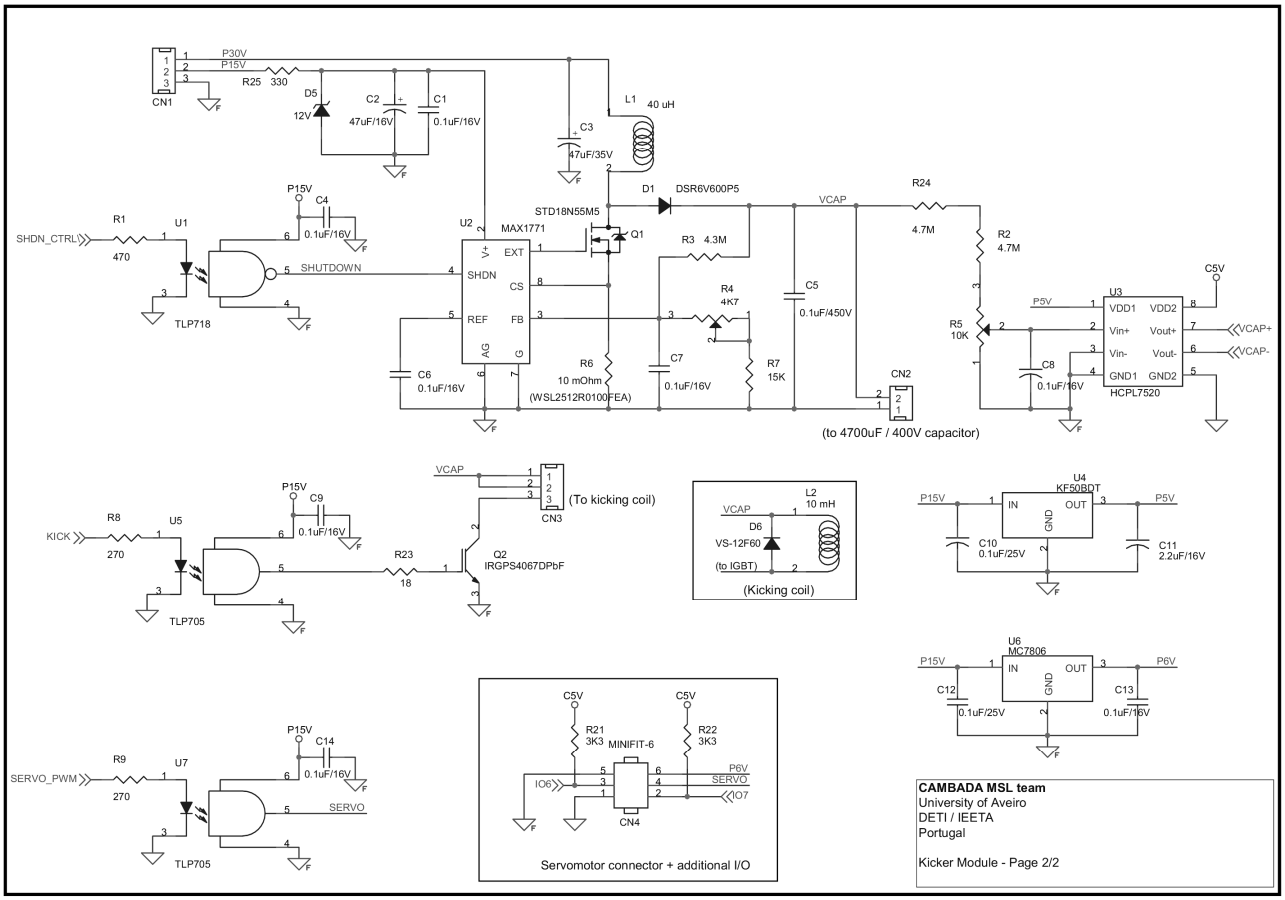

Figure 6. Kicking control module (part 2).

4.4. IMU module

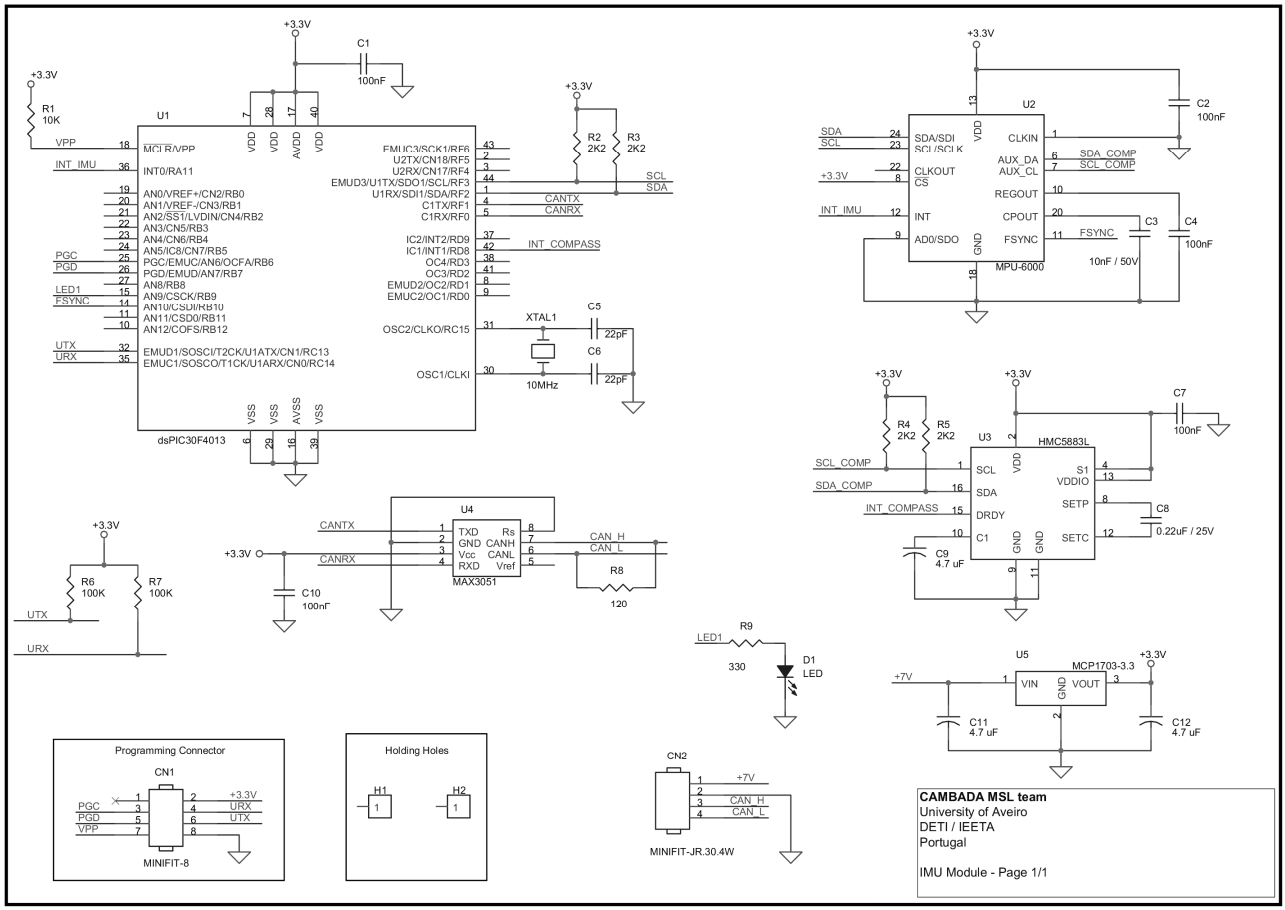

Figure 7. IMU module.

4.5. Ball handler module

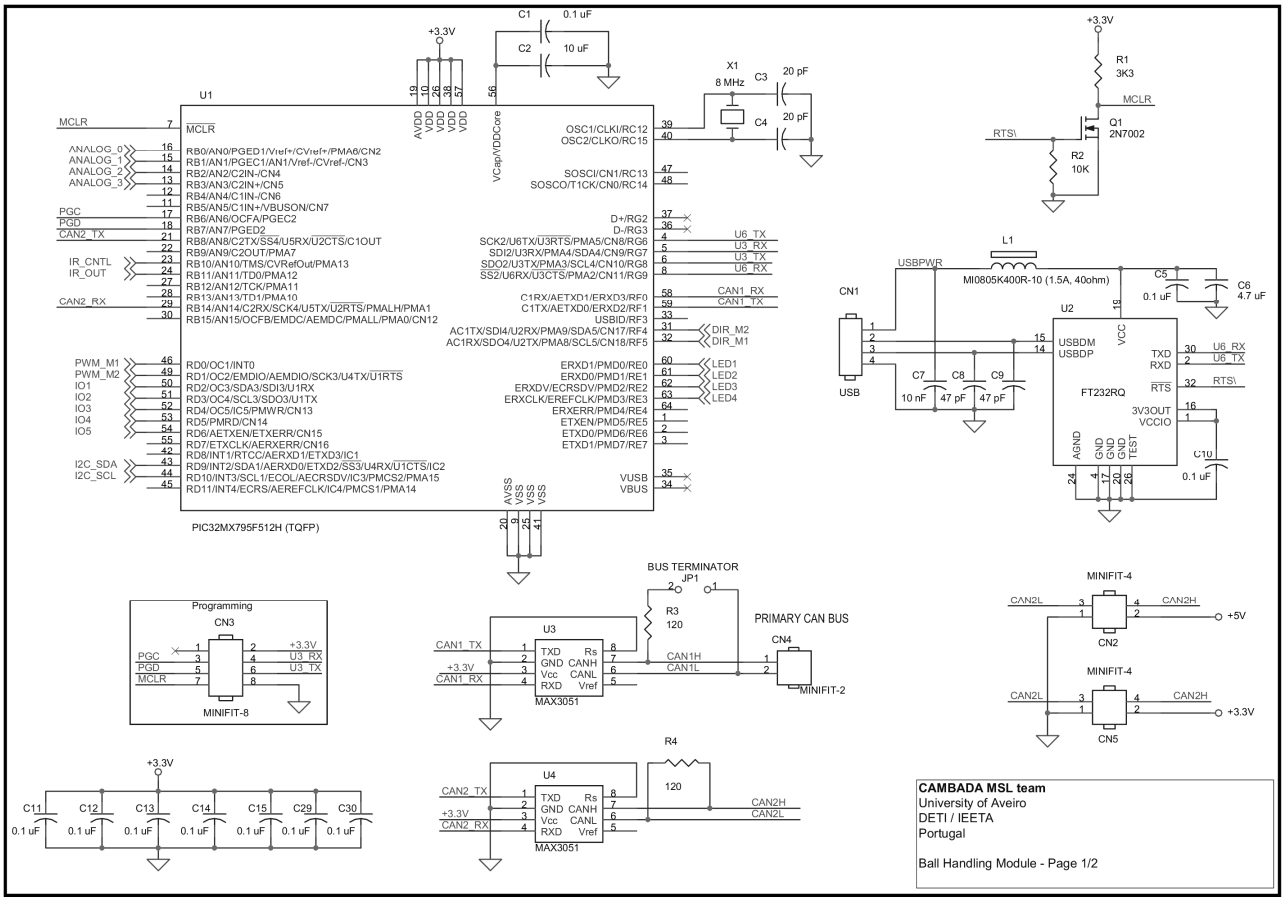

Figure 8. Ball handler module (part 1).

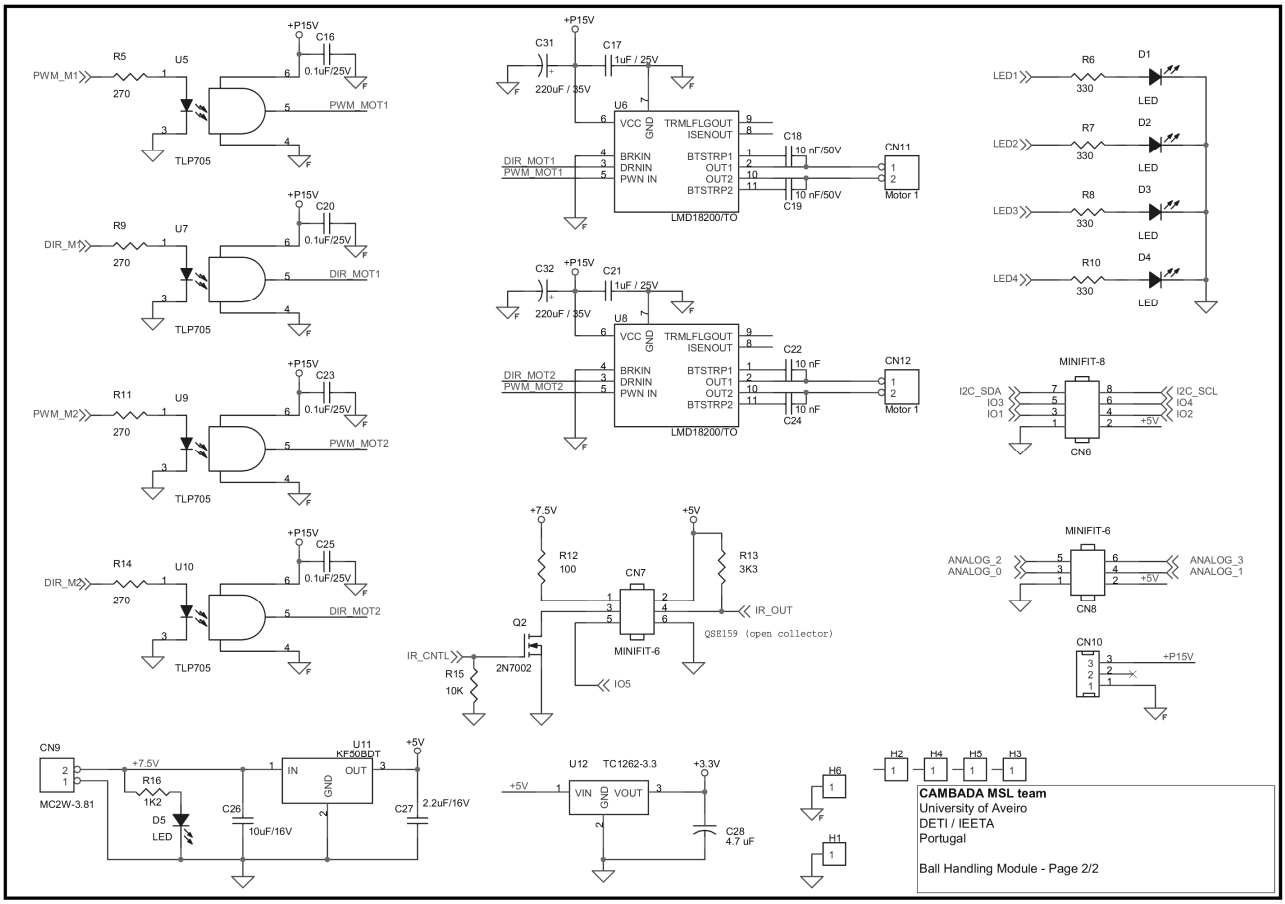

Figure 9. Ball handler module (part 2).

5. References

[1] L. Almeida, F. Santos, T. Facchinetti, P. Pedreiras, V. Silva, L.S.Lopes, "Coordinating distributed autonomous agents with a real-time database: The CAMBADA project". ISCIS'04, 19th International Symposium on Computer and Information Sciences. 27-29 October 2004, Kemer - Antalya, Turkey.

[2] V. Silva, R. Marau, L. Almeida, J. Ferreira, M. Calha, P. Pedreiras, J. Fonseca, "Implementing a distributed sensing and actuation system: The CAMBADA robots case study", IEEE ETFA 2005, Catania, Italy. September 2005.

[3] José Luís Azevedo, Manuel Bernardo Cunha, Luís Almeida (2007), Hierarchical Distributed Architectures for Autonomous Mobile Robots: a Case Study. ETFA2007- 12th IEEE Conference on Emerging Technologies and Factory Automation, pp. 973 - 980, Patras (Grece) September 25-28, 2007.

[4] Microchip website, available at www.microchip.com

[5] F. Santos, L. Almeida, P. Pedreiras, L.S.Lopes, T. Facchinnetti, "An Adaptive TDMA Protocol for Soft Real-Time Wireless Communication Among Mobile Computing Agents". WACERTS 2004, Workshop on Architectures for Cooperative Embedded Real-Time Systems (satellite of RTSS 2004). Lisboa, Portugal, 5-8 Dec. 2004.

[6] Controller Area Network - CAN2.0, Technical Specification, Robert Bosch, 1992.