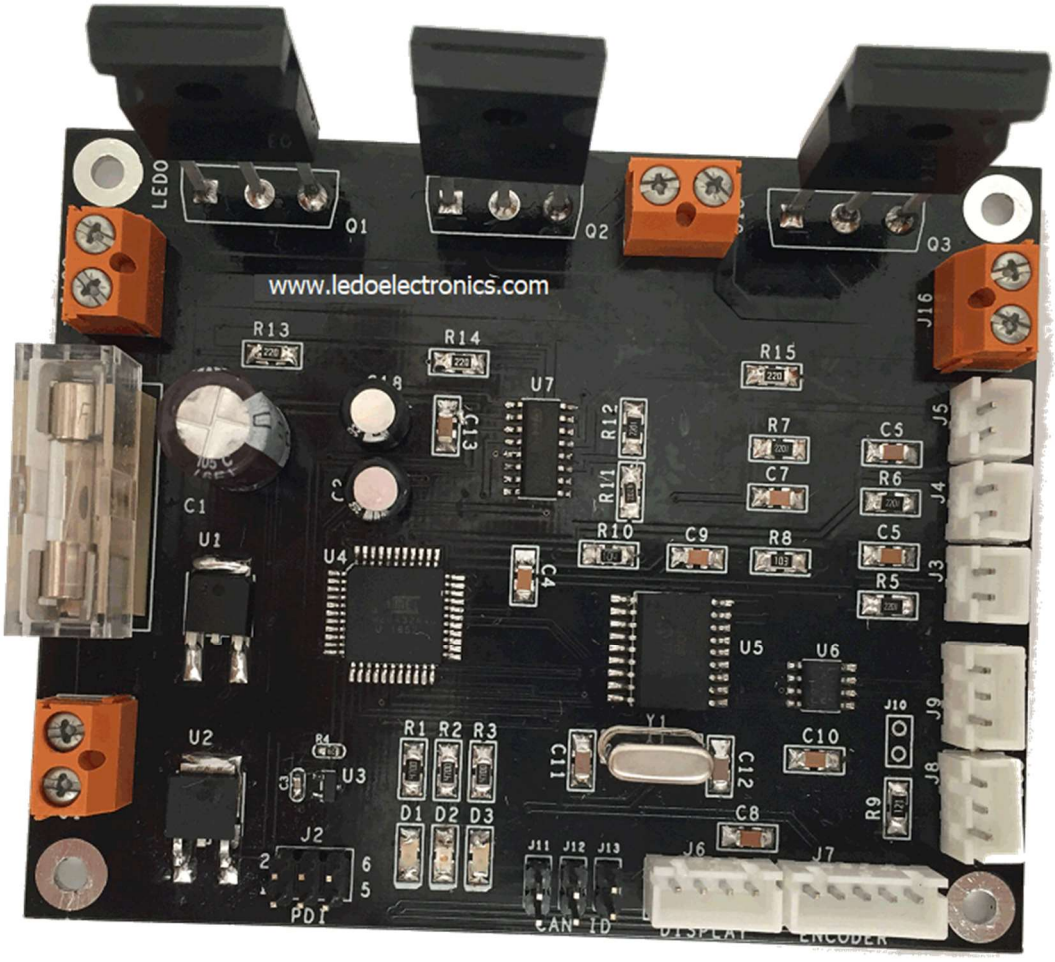
CAN bus PID Temperature Controllers



- **Supply voltage: 12 ... 24V**
- **Three PID channels of 12V / 24V 6A DC**
- **NTC as temperature sensors**
- **Connector for Rotary Encoder**
- **Connector for I2C Display**
- **CAN Bus interface (Can Open compatible)**
- **Xmega32a4u programmable in system**

The module has the Can MCP2515 controller compatible with the CAN V2.0B protocol.

The Xmega32a4u controller is responsible for generating the PWM pulses for the control of the heating elements / chillers, according to the PID control algorithm, and also contains the communication protocol. Jumpers J7, J8, J9 allow to fix an address

Ing. José Emilio Ledo Galano

on the bus for compatibility with the Can Open protocol. The LEDs D1, D2 and D3 serve as status indication.

The control can be done remotely via CAN bus, or locally using an encoder and an I2C display.

An Oled I2C screen can be connected to the J6 connector for the purpose of diagnosis and visualization of variables.

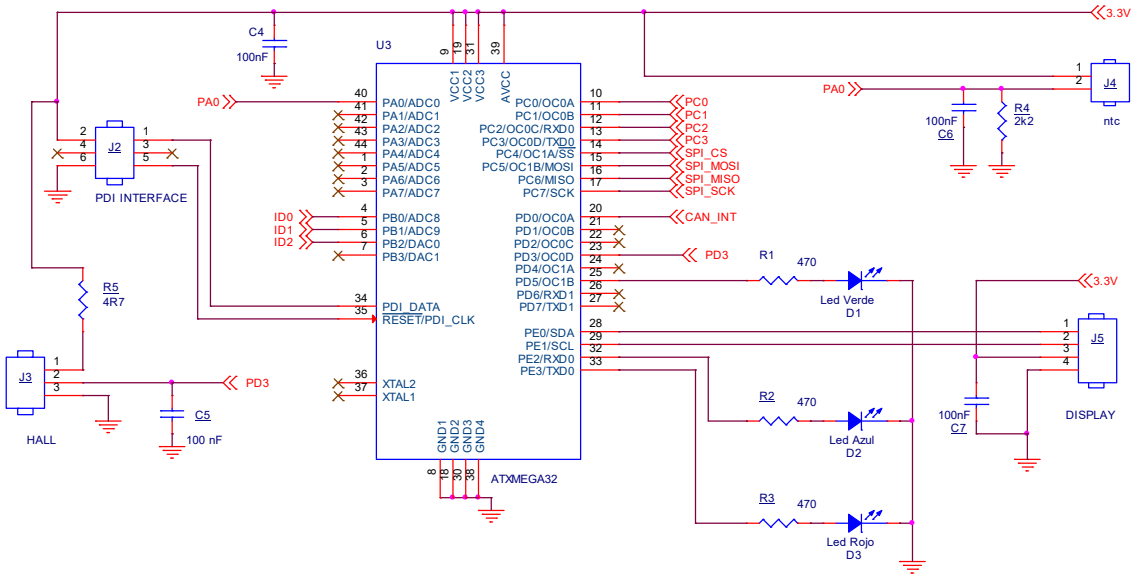


Fig.1. CPU.

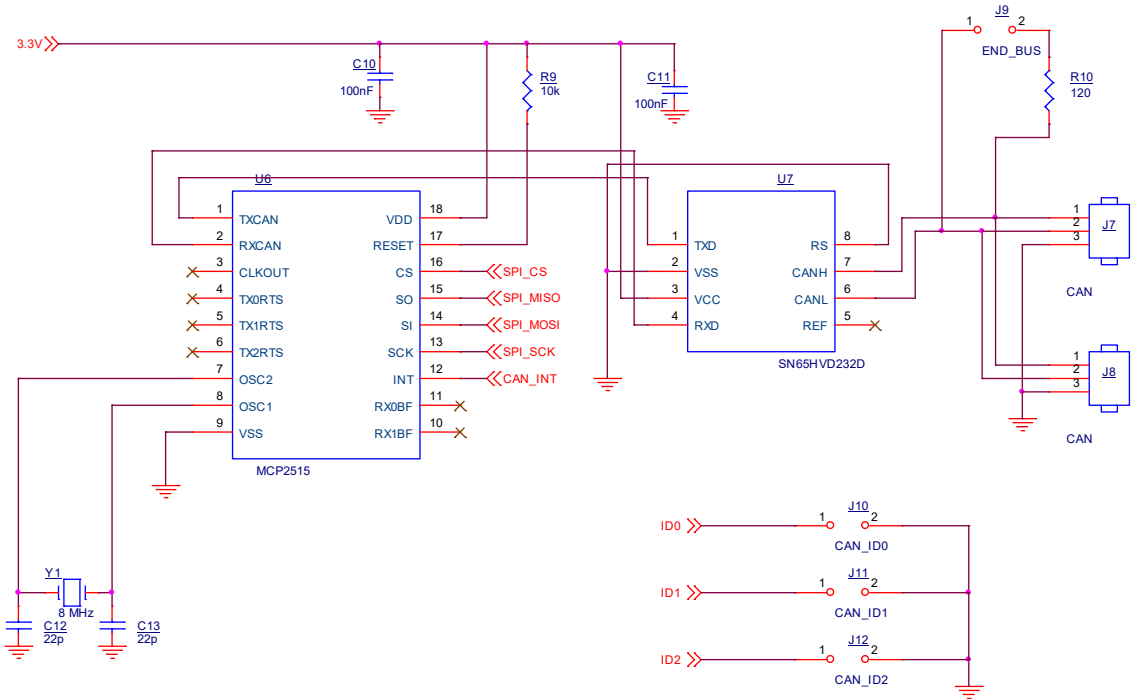


Fig.2. CAN Bus.

Ing. José Emilio Ledo Galano

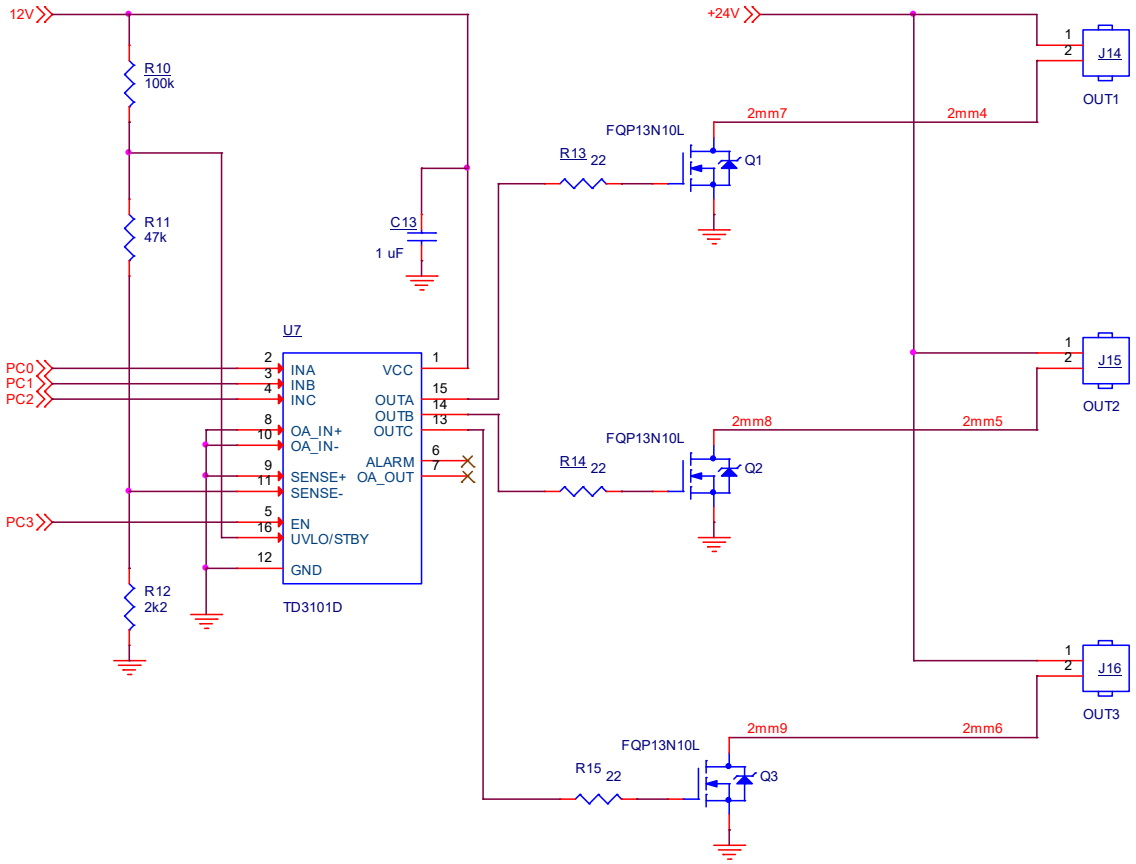


Fig.3. Power Drivers.

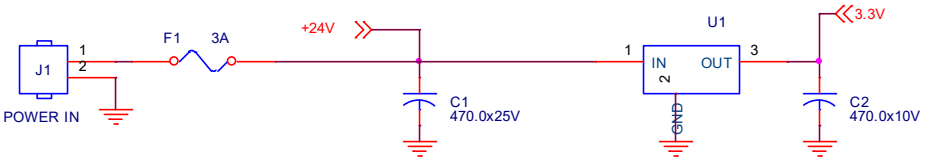


Fig.4.Power supply.

A Buck-type regulator is used to minimize power losses, given the large difference between the input voltage and the output voltage.

CAN BUS PROGRAMMING REFERENCE

Only messages addressed to the node Id are processed, which can be modified with the jumpers present on the plate according with the formula:

CAN_ID = 1536 + ((PB2 << 2) | (PB1 << 1) | PB0);

The jumpers act in the formula, when they are open.

COMMANDS OF THE CAN_COOLER_BOARD PLATE

COMMAND (BYTE0)	DATA BYTES	DESCRIPTION
1 SET KpA	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
2 SET KiA	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
3 SET KdA	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
4 ENABLE/DISABLE A	BYTE1	HIGH/LOW
5 SETPOINT A	BYTE1=LOW BYTE2=HIGH	ºc X 100
6 SET KpB	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
7 SET KiB	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
8 SET KdB	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
9 ENABLE/DISABLE B	BYTE1	HIGH/LOW
10 SETPOINT B	BYTE1=LOW BYTE2=HIGH	ºc X 100
11 SET KpC	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
12 SET KiC	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
13 SET KdC	BYTE1=LOW BYTE2=HIGH	ENTERO 16 BITS
14 ENABLE/DISABLE C	BYTE1	HIGH/LOW
15 SETPOINT C	BYTE1=LOW BYTE2=HIGH	ºc X 100
16 ENABLE ALL	NO	
17 DISABLE ALL	NO	
18 LEER TEMPERATURAS	NO	
19 LEER SETPOINTS	NO	

To all the commands from 1 to 17, the node responds with the following data structure:

Byte 0: Command received

Byte 1: Modified parameter (Low byte)

Byte 2: Modified parameter (high byte)

When command 18 (Read temperatures), the node responds with the following data structure:

Byte[0] = 18

Byte[1] = TA_LOW

Byte[2] = TA_HIGH

Byte[3] = TB_LOW

Byte[4] = TB_HIGH

Byte[5] = TC_LOW

Byte[6] = TC_HIGH

The temperature in ° C is calculated as:

T = (T_HIGH * 256 + T_LOW) / 100

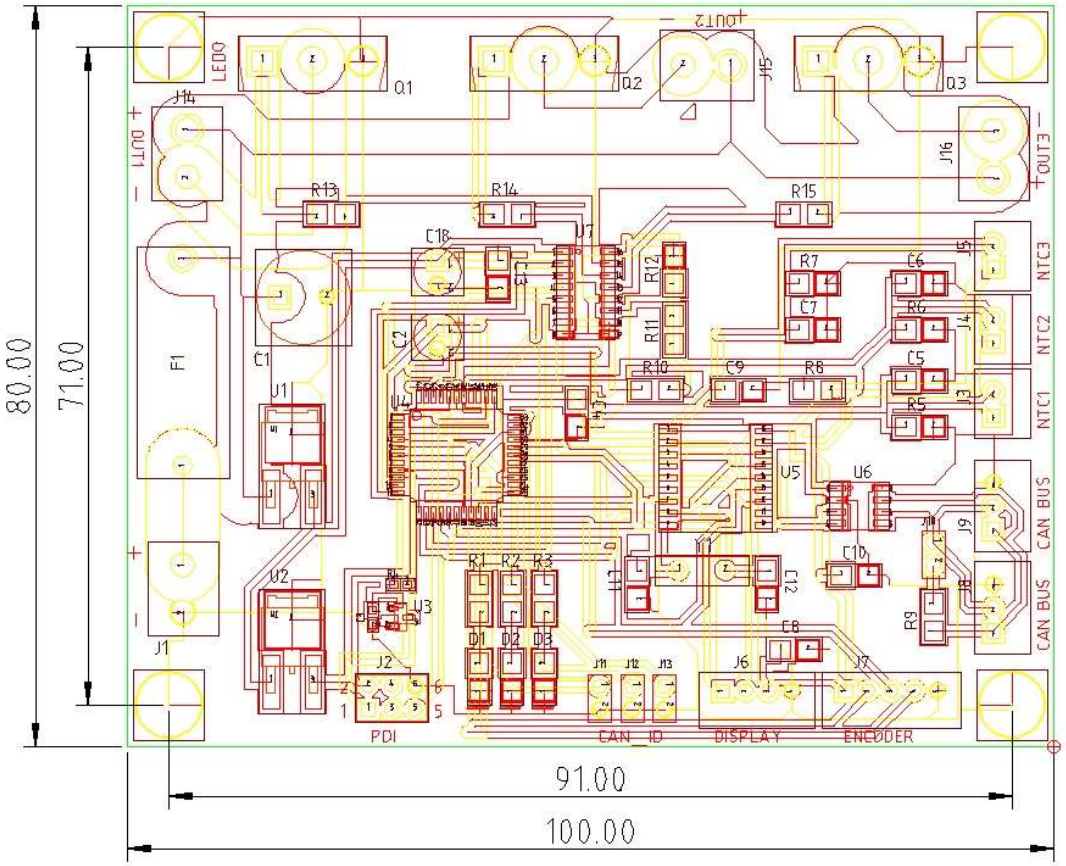


Fig.5. Board outline.