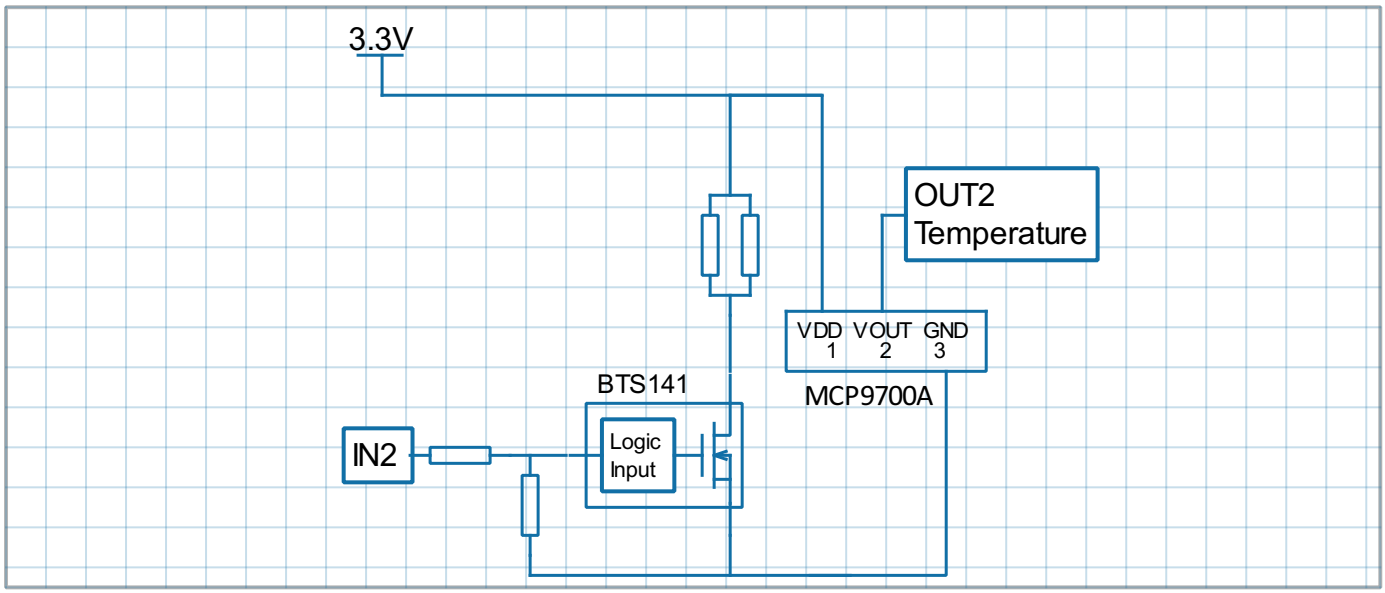
Temperature Control using LabJack and Virtument

This document illustrates how the LabJack T7 device can be used to control ambient temperatures by switching the electronic circuit on and off. When the circuit is switched on, the current through the resistors heat up. Temperature measurement is performed by using MCP9701A temperature sensors. The LabJack T7 device is used to switch the circuit on and off using a digital output (DIO) and to measure the temperature using an analog input, AIN. Virtument software, a complementary part of MatDeck, is used to set up the electronic circuits and to read the temperature values using a virtual thermometer.

Schematics of the electronic circuits

The schematics of the described system for temperature control are displayed below. It should be pointed out that the schematics are created in MatDeck, which is suitable for various professional drawings.



The description of circuit is as follows:
Functionality

- IN2 and OUT2 demonstrate PID temperature control

Demo board schematic pin descriptions

- IN2 - input for PWM driver to heat up the resistor
- OUT2 - output from the temperature sensor in mV

Connection to ADC unit

- IN2 is connected to the PWM output
- OUT2 is connected to Analog inputs 2

Parts:

- For the power driver, BTS 141 is used which is a logic level low side driver.
- Temperature sensor - MCP9700A-E/TO

LabJack T7 device uses

In this experiment, the LabJack T7 unit is used to produce a digital PWM output which is connected to the IN2 signal on the schematic diagram above At the same time, the T7 is used to measure the temperature by collecting OUT2 signals at the AIN2 channel. MatDeck supports LabJack functions which can be used

directly inside MatDeck script to configure LabJack devices and/or generate and acquire signals from the electronic circuits as described above. Here, details about the configuration of the selected features in this experiment are explained.

In order to configure and use the device, the LabJack T7 device should be opened in the document:

```
dev := ljdevice_open("any", "any", "any")
```

Configuration of DIO EF PWM out

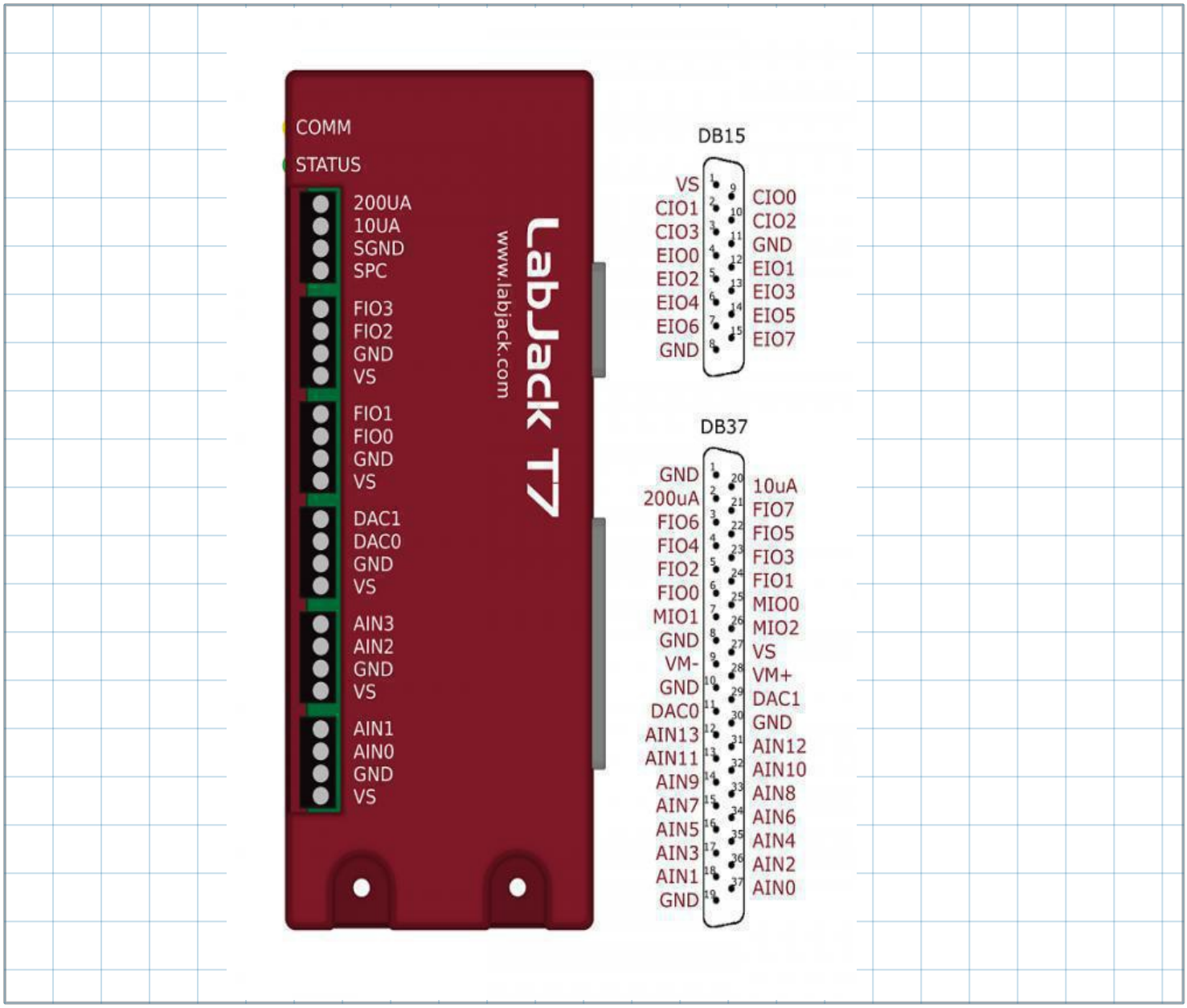
Here, DIO0 is used to produce IN2 signals. When DIO0 is high, the transistor is switched on and the current through the resistors heats the temperature sensor. If DIO0 is low, the transistor is switched off and there is no current, thus the temperature will fall.

PWM Out at FIO0(DIO0) requires the clock source, thus the clock is first configured. There are three parameters to select for the configuration: clock source, clock divisor and the roll value for the given clock. There are three different clocks supported by T7, the most common is clock0 whose frequency is 80MHz. The clock divisor can be any power of two from 1, 2, up to 256, in this example we select the value of 1. The roll value is determined according to the desired frequency of the PWM Out signal. For example, if the desired frequency is 1kHz, the roll value is 80Hz/Divisor/1kHz=80000. Before all the values are written to the appropriate registers, clock0 should be disabled.

```
//Disable clock0
ljdevice_write(dev,"DIO_EF_CLOCK0_ENABLE", 0)
//Setup Clock
ljdevice_write(dev,"DIO_EF_CLOCK0_DIVISOR", 1)//DIO_EF_CLOCK#_DIVISOR are 1,2,4,8,16,32,64,or 256 pre-counter to devide 80MHz
ljdevice_write(dev,"DIO_EF_CLOCK0_ROLL_VALUE", 80000)//Devide 80MHz by 8000 = 1000
```

PWM output at FIO0 (DIO0) is configured by setting 0 at DIO0_EF_INDEX. A duty cycle is set by writing the appropriate value at DIO0_EF_CONFIG_A. If the desired value of the duty cycle is 50%, then DIO0_EF_CONFIG_A will be equal to half of the roll value, which is 40000.

```
//PWM Out at DIO0
ljdevice_write(dev,"DIO0_EF_ENABLE", 0)
//DIO Index valu 0 - PWM set
ljdevice_write(dev,"DIO0_EF_INDEX", 0)
//PWM duty cycle
ljdevice_write(dev,"DIO0_EF_CONFIG_A", 40000)
```



Enable Extended Features

Configured Extended Features should be enabled, this is done by writing one into the ENABLE register:

```
//Enable and run clock DIO18 and PWM DIO0
ljdevice_write(dev,"DIO_EF_CLOCK0_ENABLE", 1)
ljdevice_write(dev,"DIO0_EF_ENABLE", 1)
a := ljdevice_last_error("s") //Check for errors in configuration
```

```
a = "LJ_SUCCESS"
```

Configuring Temperature Measurement

The temperature is measured by using the AIN2 channel of the T7, where the OUT2 signal is connected. The low power linear active thermistor circuit ,MCP9701A, is used as a temperature sensor. Here, OUT2 is voltage which depends on the ambient temperature, and should be converted into temperature using the linear function given in the data-sheet. The sensor transfer function is:

$$V_{OUT} = T_C \cdot T_A + V_{0°C}$$

Here, V_{OUT} is the sensor output voltage, T_A is ambient temperature, T_C is the temperature coefficient, and $V_{0°C}$ is the sensor output voltage at 0°C. From the MCP9701A datasheet, T_C=19.5 mV/°C and $V_{0°C}$=400mV. In order to determine the temperature from the voltage, we need to inverse the function.

$$T_A = V_{OUT}/T_C - V_{0°C}/T_C$$

Slope and offset can be determined as follows:

```
Tc := 0.0100
V0 := 0.5
Slope := 1 / Tc
Offset := -V0 / Tc
```

AIN2 is configured to use the Offset and Slope extended feature, EF_INDEX is 1, which automatically adds a slope and an offset to analog readings according to the linear function above.

```
ljdevice_write(dev, "AIN2_EF_INDEX", 1)
ljdevice_write(dev, "AIN2_EF_CONFIG_D", Slope)
ljdevice_write(dev, "AIN2_EF_CONFIG_E", Offset)
```

The temperature is automatically read using:

```
Ta :=ljdevice_read(dev, "AIN2_EF_READ_A")
```

$$Ta = 30.612 \quad C$$

Virtument

The MatDeck software has a instruments panel otherwise known as visual PC instrumentation called Virtument. The Virtument instrument panel can receive and transmit data to both the MatDeck software and documents. Both software's can connect to IP address, receive and transmit data.

Here, a vertical slider in Virtument is used to control the duty cycle of the PWM out signal. In this document, the channel is set and connected with variable duty from Virtument which will be used by the PWM. The temperature variable Ta is connected to the channel in order to display its value in Virtument using a virtual analog, digital thermometer and graph. Virtument's instrument panel ,labjackT.vir, contains the virtual instruments mentioned above.

```
duty := 50
roll_value := 80000
config_a := roll_value * duty / 100
```

In the following loop, the temperature Ta is controlled using variable duty. Both variables are connected to the virtual instruments as explained above.

```
while(true)
{
  config_a = roll_value * (duty) / 100
  ljdevice_write(dev,"DIO0_EF_CONFIG_A", config_a)
```

```
32
33    Ta = ljdevice_read(dev, "AIN2_EF_READ_A")
34    sleep(1)
35  }
```

Finally, all the extended features should be disabled and the device should be closed.

```
36  ljdevice_close(dev)
```