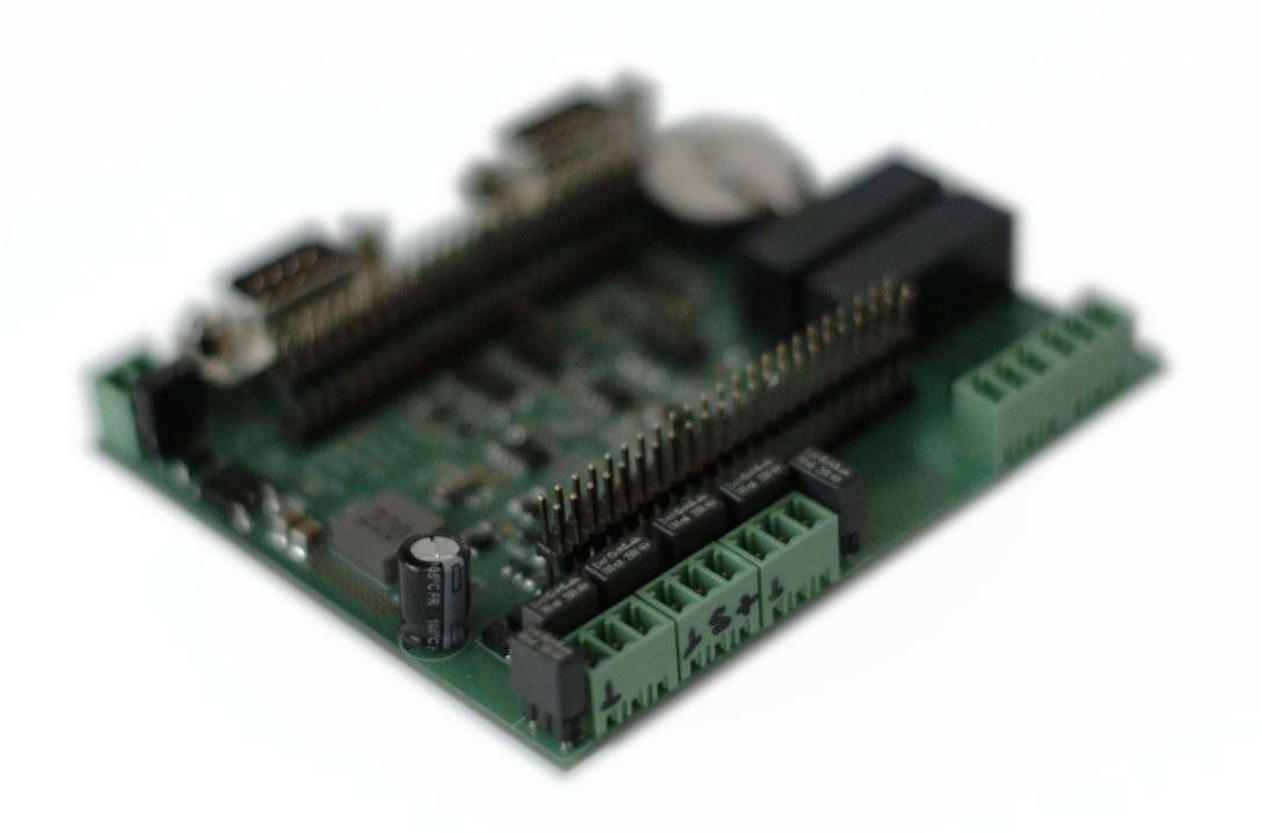
FotobotHW DUO

User's guide



HOBRASOFT

Content

Introduction

Compatibility

The FotobotHW-DUO board is compatible with BeagleBone computers:

- BeagleBone Black
- BeagleBone Industrial
- not tested with BeagleBone Wifi or BeagleBone Green

The FotobotHW-DUO board do not collide with other peripheral placed in BeagleBone Black (HDMI, eEEMC).

Inputs

- 2× potential-free inputs for buttons, relays contact etc.
- 2× analog inputs 0-10 V or 0-20 mA
- One-wire bus for DS18B20 thermometers

Outputs

- 2× open collector output
- 2× switching relay

Serial communication

- 2× RS232, only RXD and TXD signals
- Output voltage typical ±9 V, minimal ±5.5 V with load impedance 3 kΩ

Power supply

- 7-24 V DC, max. 1.25 A

Other

- Real time clock with battery backup

Installation

The installation process expects that Debian 9 with Linux kernel 4.4 and uBoot overlays are used:

```
https://beagleboard.org/latest-images
```

Installation using Debian package

Download debian package for the FotobotHW-DUO board:

```
https://www.hobrasoft.cz/en/hardware/duo
```

Install the package:

dpkg -i hobrasoft-duo-1.0.0-armhf.deb

Then reboot your BeagleBone. After reboot the board should be detected and initialized.

Manual installation

Download firmware for the FotobotHW-DUO board:

```
https://www.hobrasoft.cz/en/hardware/duo
```

Compile the firmware in your BeagleBone using dtc and copy the dtbo file to /usr/firmware:

```
dtc -q -O dtb -@ -o HOBRASOFT-DUO-00A0.dtbo HOBRASOFT-DUO-00A0.dts
cp HOBRASOFT-DUO-00A0.dtbo /usr/firmware
```

Warnings *Node /fragment@0 has a unit name, but no reg property* can be ignored safely.

Edit the /boot/uEnv.txt file, check that the u-Boot overlays are enabled:

```
enable_uboot_overlays=1
```

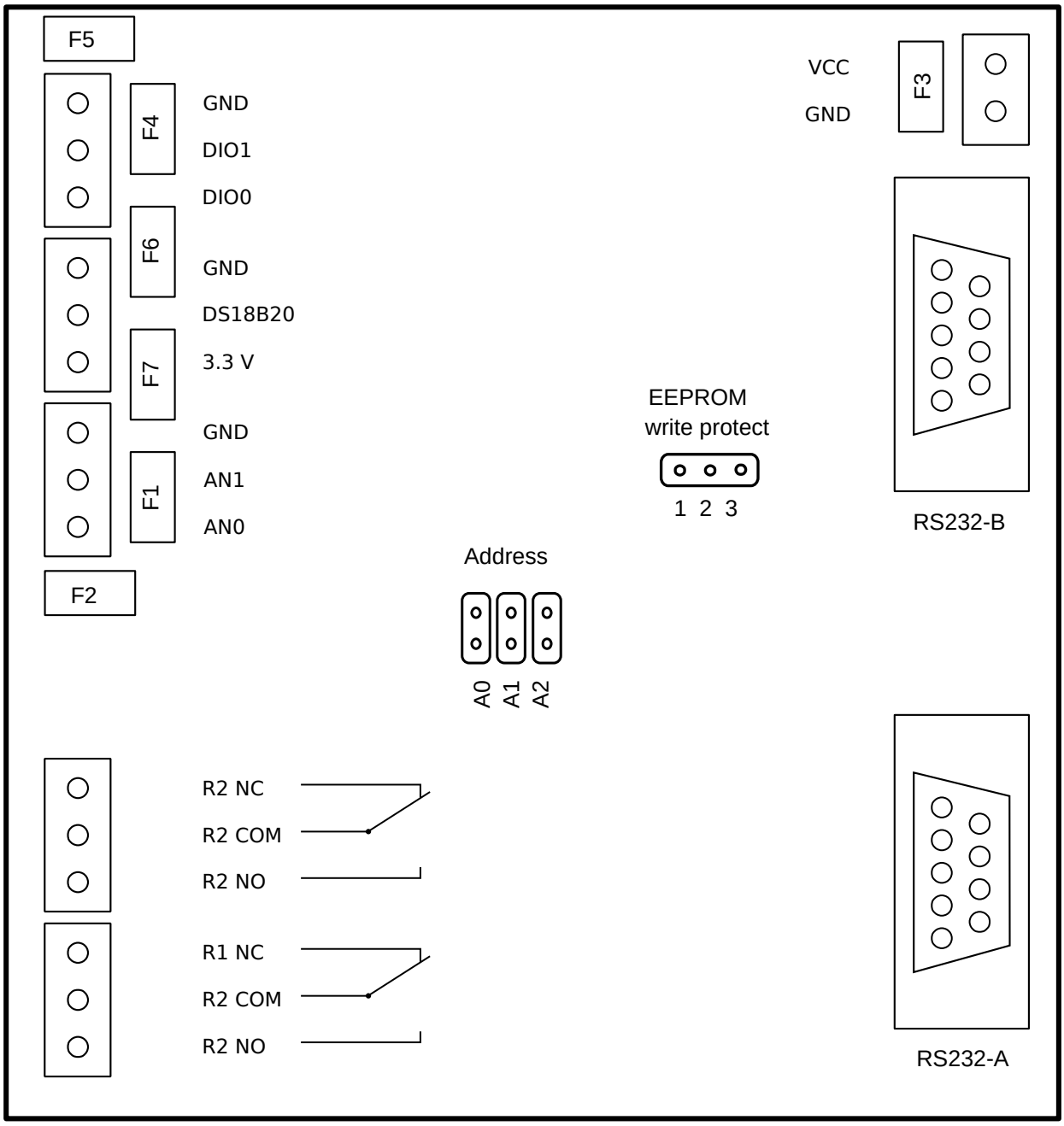
Then reboot the system. After reboot the board should be detected and initialized.

Real time clock (RTC) uses rtc_mcp7941x kernel module. Initialization should be made during the system startup:

```
echo mcp7941x 0x6f > /sys/class/i2c-adapter/i2c-2/new_device
```

Then the RTC can be accessed using /dev/rtc1.

Connectors


F5
F4
GND
DIO1
DIO0
F6
GND
DS18B20
F7
3.3 V
GND
F1
AN1
AN0
F2
VCC
GND
F3
EEPROM
write protect
1 2 3
RS232-B
Address
A0
A1
A2
R2 NC
R2 COM
R2 NO
R1 NC
R2 COM
R2 NO
RS232-A

	Input	Output	Pins	Control
GND				
DIO1	gpio67	gpio68	P8.8, P8.11	
DIO0	gpio66	gpio45	P8.7, P8.10	
GND				
DS18B20			P8.9	
3.3 V				
GND				
AN1	AIN1		P9.39, P9.27	gpio115: 0=0-10V, 1=0-20mA
AN0	AIN0		P9.40, P9.23	gpio49: 0=0-10V, 1=0-20mA
R2 NC				
R2 COM		gpio50	P9.14	
R2 NO				
R1 NC				
R1 COM		gpio51	P9.16	
R1 NO				

Serial ports

	Pins	Device
RS232-A	P9.26, P9.24	/dev/ttyS1
RS232-B	P9.22, P9.21	/dev/ttyS2

The system device names are /dev/ttyS1 and /dev/ttyS2 usually. They depends on distribution and Linux version used.

Fuses

F1	AN1	80 mA
F2	AN0	80 mA
F3	VCC	1.25 A
F4	DIO0	80 mA
F5	DIO1	80 mA
F6	DS18B20 signal	80 mA
F7	DS18B20 VCC 3.3 V	80 mA

EEPROM write protection

All open	EEPROM protected
1 – 2	EEPROM write enabled

2 – 3	EEPROM write protection controlled by gpio3_21 (P9.25)

Note: The EEPROM is programmed during production. Do not change it‘s content unless you know what you are doing.

EEPROM address

A0	A1	A2	Address
open	open	open	57 - default
close	open	open	56
open	close	open	55
close	close	open	54

Battery

Use CR2032 battery for RTC backup.

Potential-free inputs

Potential-free inputs can be used to connect buttons, relay contacts etc. Inputs share their connector pins with outputs. The input and output can be used at the same time with limitation. If you set the digital output to 1, then the digital input will be always read as 0.

Export the GPIO pins for the inputs and outputs and set the pins. This should be made only once when starting your application:

```
echo 66 > /sys/class/gpio/export                        # DIO0 input
echo 67 > /sys/class/gpio/export                        # DIO1 input
echo in > /sys/class/gpio/gpio66/direction
echo in > /sys/class/gpio/gpio67/direction
echo 68 > /sys/class/gpio/export                        # DIO0 output
echo 45 > /sys/class/gpio/export                        # DIO1 output
echo out > /sys/class/gpio/gpio68/direction
echo out > /sys/class/gpio/gpio45/direction
echo 0 > /sys/class/gpio/gpio68/value                   # Open DIO0 output
echo 0 > /sys/class/gpio/gpio45/value                   # Open DIO1 output
```

You can read input 0 using:

```
cat /sys/class/gpio/gpio66/value
```

You can read input 1 using:

```
cat /sys/class/gpio/gpio67/value
```

Value 1 means that the input is opened. Value 0 means that the input is connected to ground.

Open-collector outputs

Outputs share their connector pins with outputs. The input and output can be used at the same time with limitation. If you set the digital output to 1, then the digital input will be always read as 0.

Export the GPIO pins for the outputs and set the pins. This should be made only once when starting your application:

```
echo 68 > /sys/class/gpio/export
echo 45 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio68/direction
echo out > /sys/class/gpio/gpio45/direction
```

You can open the DIO0 collector output using:

```
echo 0 > /sys/class/gpio/gpio68/value
```

You can close the DIO0 collector output using:

```
echo 1 > /sys/class/gpio/gpio68/value
```

You can open the DIO1 collector output using:

```
echo 0 > /sys/class/gpio/gpio45/value
```

You can close the DIO1 collector output using:

```
echo 1 > /sys/class/gpio/gpio45/value
```

Relay outputs

Export the GPIO pins for the relays and set the pins. This should be made only once when starting your application:

```
echo 51 > /sys/class/gpio/export
echo 50 > /sys/class/gpio/export
echo out > /sys/class/gpio51/direction
echo out > /sys/class/gpio50/direction
```

Then you can control relays writing 1 or 0 to value file.

Set the relay 1 on and off:

```
echo 1 > /sys/class/gpio51/value
echo 0 > /sys/class/gpio51/value
```

Set the relay 2 on and off:

```
echo 1 > /sys/class/gpio50/value
echo 0 > /sys/class/gpio50/value
```

Analog inputs

Analog inputs can measure voltage 0-10 V and current 0-20 mA. The range of each analog input can be set independently.

Export the control GPIO pins. This should be made only once when starting your application:

```
echo 115 > /sys/class/gpio/export                       # AIN0
echo 49 > /sys/class/gpio/export                        # AIN1
echo out > /sys/class/gpio/gpio49/direction
echo out > /sys/class/gpio/gpio115/direction
```

Voltage 0-10 V

After initialization switch the analog input to voltage mode:

```
echo 0 > /sys/class/gpio/gpio115/value                  # AIN0 0-10 V
echo 0 > /sys/class/gpio/gpio49/value                   # AIN1 0-10 V
```

You can read input 0 using:

```
cat /sys/bus/iio/devices/iio:device0/in_voltage0_raw
```

You can read input 1 using:

```
cat /sys/bus/iio/devices/iio:device0/in_voltage1_raw
```

The value read has to be converted to voltage:

```
voltage = value / 4096 * 10                             [V]
```

Current 0-20 mA

After initialization switch the analog input to current mode:

```
echo 1 > /sys/class/gpio/gpio115/value                  # AIN0 0-20 mA
echo 1 > /sys/class/gpio/gpio49/value                   # AIN1 0-20 mA
```

You can read input 0 using:

```
cat /sys/bus/iio/devices/iio:device0/in_voltage0_raw
```

You can read input 1 using:

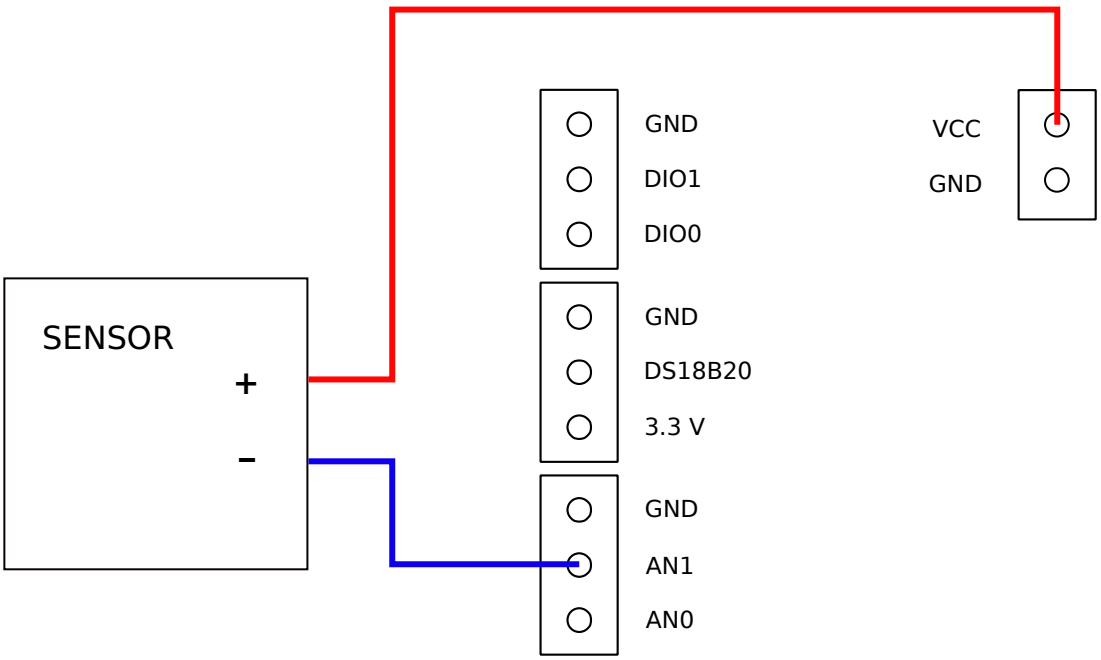
```
cat /sys/bus/iio/devices/iio:device0/in_voltage1_raw
```

The value read has to be converted to current:

```
current = value / 4096 * 20                                    [mA]
```

Sensor connection

When using current loop 0-20 mA, connect the *plus* wire of the sesor to VCC and *minus* wire of the sensor to AN1 or AN0 input.



Real Time Clock

Real time clock (RTC) uses rtc_mcp7941x kernel module. The RTC chip is located on i2c bus 0x6f address.

Initialization should be made during the system startup:

```
echo mcp7941x 0x6f > /sys/class/i2c-adapter/i2c-2/new_device
```

Write current system date and time to RTC:

```
hwclock -wu -f /dev/rtc0
```

Read RTC and set current system date and time:

```
hwclock -s -u -f /dev/rtc1
```

When you change the CR2032 battery, you should write current system date and time to RTC again, otherwise you would not be able to read any data from RTC.

Serial ports

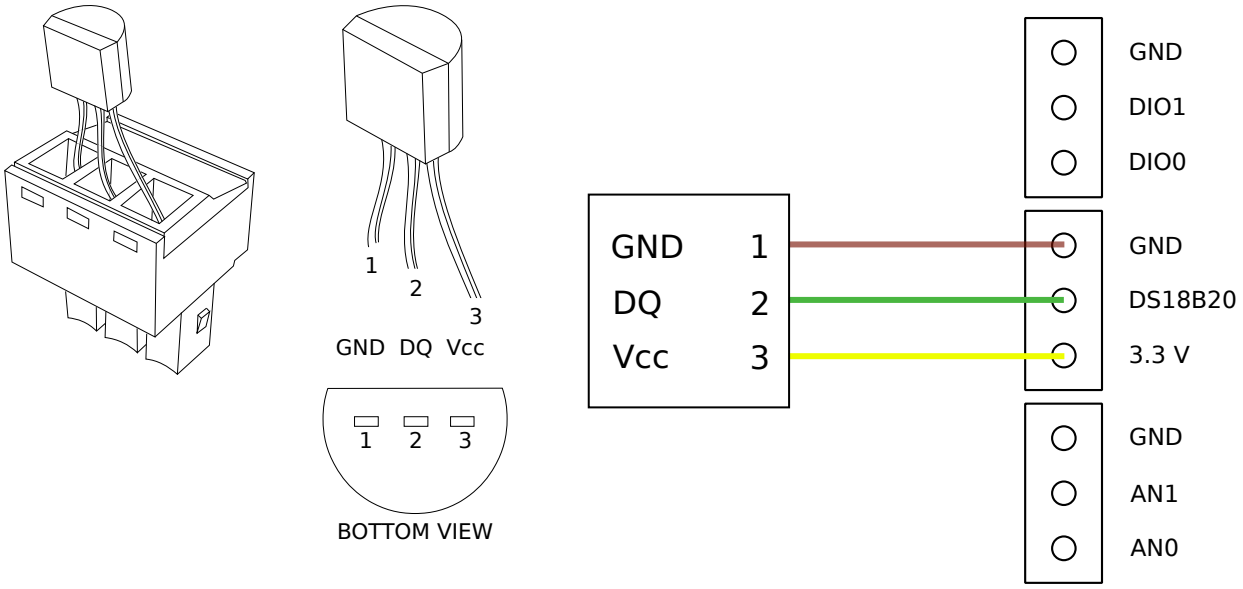
There is no need to initialize serial ports. When the system is started with proper firmware, serial ports are initialized automatically.

Ports are labeled as RS232-A and RS232-B. Actual device names depends on Linux version and distribution. Usualy the device are named like this:

	BeagleBone UART	Pins	Device
RS232-A	UART1	P9.26, P9.24	/dev/ttyS1
RS232-B	UART2	P9.22, P9.21	/dev/ttyS2

Thermometers

The bus with thermometers of DS18B20 type can be connected to the device. The number of thermometers is not limited in Linux operating system. The number of reliably working thermometers is limited with bus length and the number of thermometers.



Each thermometer is available in the /sys/bus/w1/devices directory:

```
# ls /sys/bus/w1/devices
10-00080050204b  w1_bus_master1
```

You can read temperature using:

```
cat /sys/bus/w1/devices/10-00080050204b/w1_slave

3c 00 4b 46 ff ff 09 10 2d : crc=2d YES
3c 00 4b 46 ff ff 09 10 2d t=30187
```

Temperature here is 30.187 °C.

Specification

Power supply	
Voltage	7-24 V DC
Current	max. 1.25 A
Open-collector output	
Maximum Voltage	30 V
Maximum Current	60 mA, fused
Polarity	GND minus, DIO0 or DIO1 plus, must not be changed
Digital inputs	
Maximum Voltage	30 V
Relay output	
Contact rating	8 A, 250 VAC / 24 VDC
Maximum carrying current	10 A
Maximum switching voltage	400 VAC / 150 VDC
Maximum switching power	2,000 VA / 192 W

Fuses

F1	AN1	80 mA
F2	AN0	80 mA
F3	VCC	1.25 A
F4	DIO0	80 mA
F5	DIO1	80 mA
F6	DS18B20 signal	80 mA
F7	DS18B20 VCC 3.3 V	80 mA

Revision history

2018-05-23	Added schematic of sensor connection when using current loop 0..20mA
2018-05-30	Fixed gpio addresses for AIN0 and AIN1 mode settings. Section *Thermometers* added.