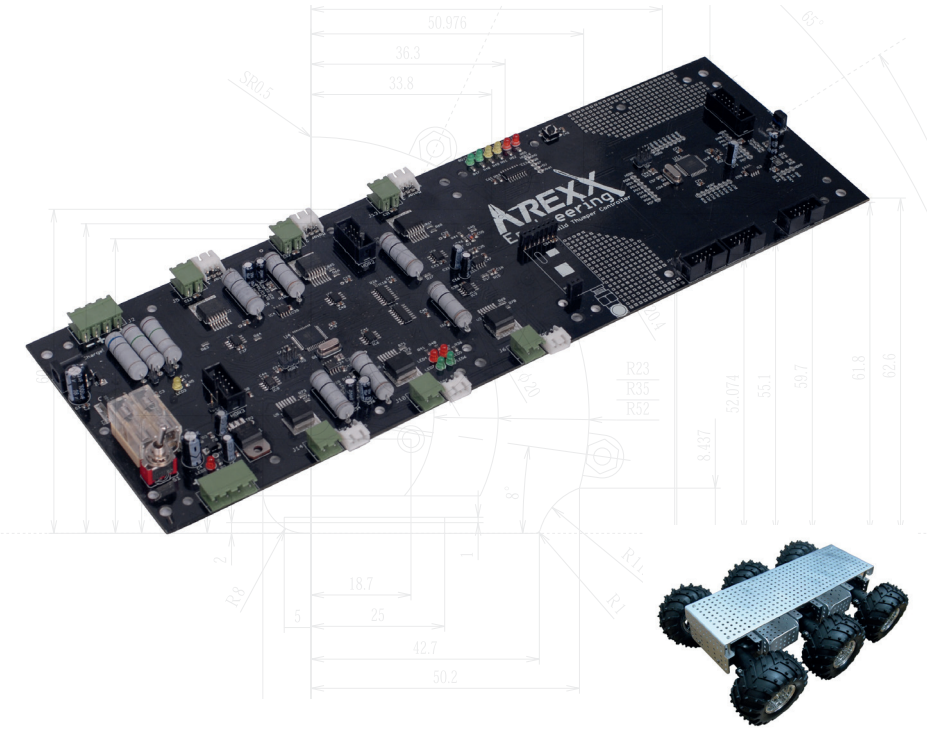
AREXX Engineering

EDUCATIONAL ROBOT
Wild Thumper Controller

Manual: Model WTR-CK1



Contents

AREXX Engineering

Manufacturer:
AREXX Engineering

European Importer:
AREXX Engineering
ZWOLLE The Netherlands

Technical support:

WWW.AREXX.COM
WWW.ROBOTERNETZ.DE

AREXX Engineering

Impressum

©2016 AREXX Engineering
Nervistraat 16
8013 RS Zwolle
The Netherlands

Tel.: +31 (0) 38 454 2028
Fax.: +31 (0) 38 452 4482

E-Mail: Info@arexx.nl

Information about limited warranty and responsibility

The warranty granted by AREXX Engineering is limited to the replacement or repair of the Wild Thumper Robot and its accessories within the legal warranty period if the default has arisen from production errors such as mechanical damage or missing or wrong assembly of electronic components except for all components that are connected via plugs/sockets. The warranty does not apply directly or indirectly to damages due to the use of the Wild Thumper. This excludes claims that fall under the legal prescription of product responsibility.

The warranty does not apply in case of irreversible changes (such as soldering of other components, drilling of holes, etc.) of the Wild Thumper KIT or its accessories or if the Wild Thumper is damaged due to the disrespect of this manual.

The warranty is not applicable in case of disrespect of this manual! In addition, AREXX Engineering is not responsible for damages of all kinds resulting from the disrespect of this manual! Please adhere above all to the „Safety recommendations" in the Wild Thumper manuals.

Please note the relevant license agreements on the CD-ROM!

IMPORTANT
Prior to using this Wild Thumper for the first time, please read this manual thoroughly up to the end. It explains the correct use and informs you about potential dangers. Moreover it contains important information that might not be obvious for all users.

Important safety recommendation

This module is equipped with highly sensitive components. Electronic components are very sensitive to static electricity discharge. Only touch the module by the edges and avoid direct contact with the components on the circuit board.

Symbols

This manual provides the following symbols:

	*The "Attention!" Symbol is used to mark important details. Neglecting these precautions may damage or destroy the module and/or additional components and additionally you may risk your own health or the health of other persons!*
	*The "Information" Symbol is used to mark useful tips and tricks or background information. In this case the information is to be considered as "useful, but not necessary".*

Safety recommendations

- Always check the polarity of the power supply and batteries.
- Keep all products dry, when the product gets wet remove the power directly.
- Remove the power when you are not using the product for a longer period.
- Before taking the module into operation, always check it and its cables for damage.
- If you have reason to believe that the device can no longer be operated safely, disconnect it immediately and make sure it is not unintentionally operated.
- Consult an expert if you are unsure about the function, safety or connection to the module.
- Do not operate the module in unfavourable conditions.
- This module is equipped with highly sensitive components. Electronic components are very sensitive to static electricity discharge. Only touch the module by the edges and avoid direct contact with the components on the circuit board.

Normal use

This product was developed as an electronic robot kit for indoors use only. The normal use of this product is to learn programming in C language and to get knowledge about microprocessors and electronics.

Any use other than that described above can lead to damage to the product and may involve additional risks such as short circuits, fire, electrical shock etc.

Please read all the safety instructions of this manual.

The electronic components of this product must not get damped or wet. Also be careful with condense when you take it from a cold to a warm room, give it time to adapt to the new conditions before you use it.

1. PRODUCT DECRIPTION WTR-CK1

The Wild Thumper controller is equipped with all the necesarry electronics to program and control the Wild Thumper "WILD THUMPER CHASSIS JSR-6WD".

The controller is based on two ATMEL microprocessors which are communicating with eachother. One of the microprocessors handles all communication between the two processors and the many sensors, which are included in this kit. The second processor is responsible for the six H-bridges which are used as motor drivers.

The kit includes two APC-220 modules. These are 433MHz transceivers which are used for the wireless communication between a PC and the Wild Thumper. Also included is a USB programmer and the (up)loader-Software, with this you can program the two processors in a very simple way. The Wild Thumper can only work with wireless control software.

The control programms should be written in C-language with the help of the freeware WINAVR. This software compiles and makes a for the micro-process sor usable HEX file after you have written your program.

Like all AREXX-Robotsystems we offer many connections for your own Wild Thumper applications like free In-/Outputs and a I2C-Bus.

Scope of delivery:

1 Pc.	Main PCB
1 Pc.	Front PCB with switches
1 Pc.	USB Programmer
2 Pcs.	APC-220 Module
1 Pc.	Motor encoder frequency divider
1 Pc.	CD with manual and software
	Assembly parts, wires and connectors

Warnings

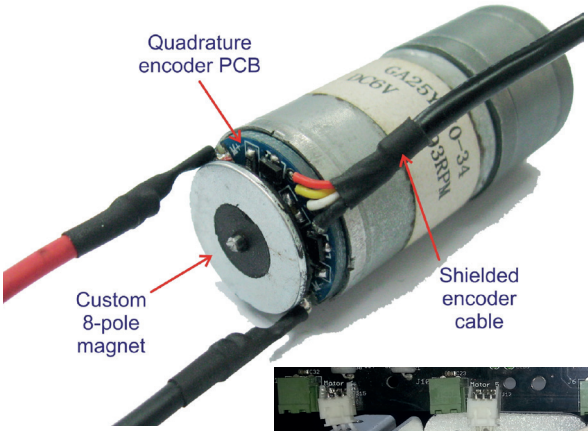
* The right of return does not apply after opening the plastic bags containing parts and components.
* Read the manual thoroughly prior to assembling the unit.
* Be careful when handling tools.
* Do not assemble the robot in presence of small children. They can get hurt with the tools or swallow small components and parts.
* Check the correct polarity of the batteries.
* Make sure that batteries and holder remain dry always. If the Controller gets wet, remove the batteries and dry all parts as thoroughly as possible.
* Remove the batteries if the Controller will not be used for more than one week.

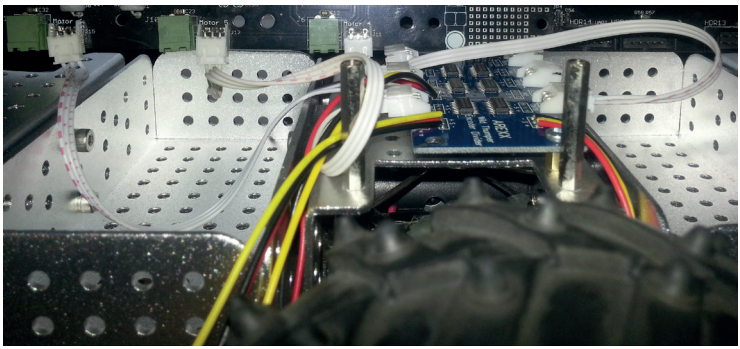
1.2. Specifications:

Power: 7,2 Volt sub C batterie pack (not included)
2 Pcs. ATMEGA 644 Processors
Odometrie sensors for all 6 motors
Motor-currentsensors for all 6 motors
Voltage sensor
Light sensor
IR-Sensors
Collision switches
Many free I/O's and I2C connectors
12 LEDS
Main PCB with APC-220 wireless transceiver module
USB-Programmer with APC-220 wireless transceiver module

Both processors are free programmable in "C" language.
The programms can be uploaded to the processor with the USB programming interface and the AREXX Robotloader Software.

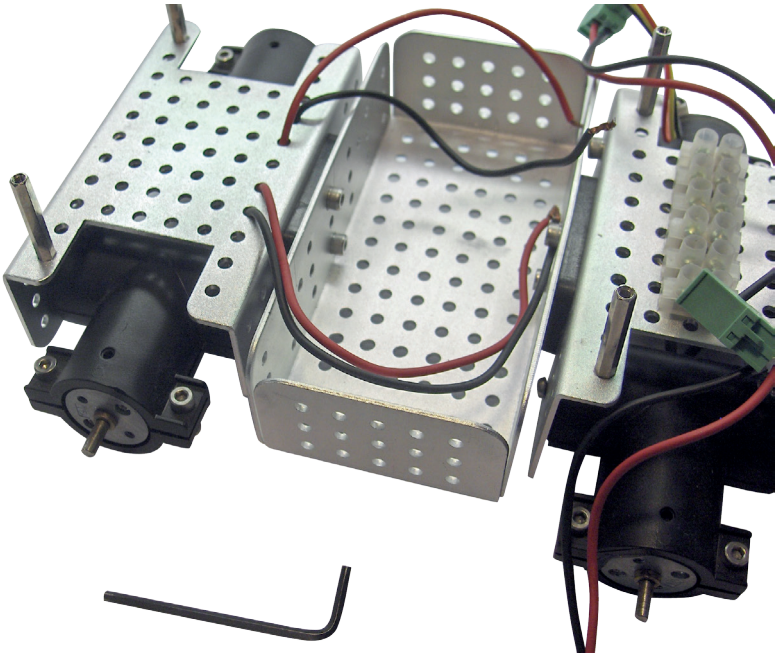
Dimensions: (LxB) 360 x 120 (mm)
Notice: Max 2 pcs. 7,2V sub C battery packs can be connected



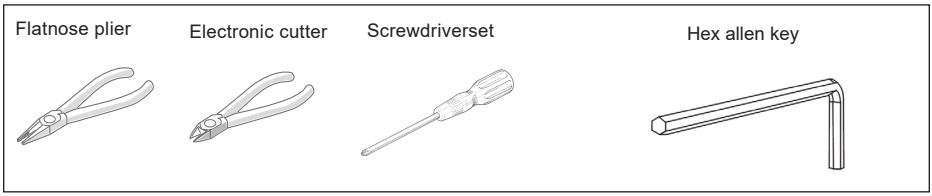


1.3. WILD THUMPER

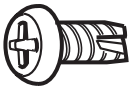
- Programmable in C with WINAVR
- Example and new programs can be transferred easily into the Wild Thumper controller microprocessors.
- The Wild Thumper can only be controlled wireless with a PC and our WTC Software.
- Connect your own wireless camera and include the pcitures in our WTC PC Software.
- The Wild Thumper can be extended with our extension modules so it will be more sensitive in its reaction to its environment.
- With I2C bus you communicate with many different modules.
- Artificial intelligence makes selflearning software so the Wild Thumper will be learning how to impove itself.

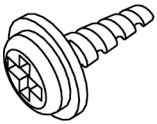


2. Necesary tools

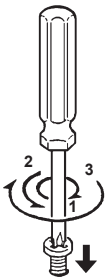


Self-tapping screws (Parker)





A self-tapping screw looks similar to a wood screw. When you screw it in a hole, it can cut the threads at the same time. Never try to screw it down all the way for a first time, because it may easily become stuck or you will damage its head.



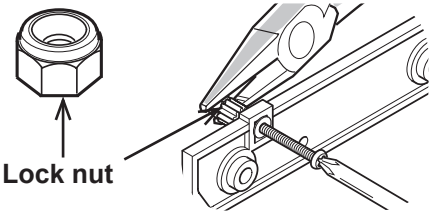
Tapping screws always have a sharp point sometimes with a small carve. The best way is to screw it in and out a bit.

1. Screw in
2. Screw out a bit
3. Screw in further and continue step 1 and 2

Do not screw a tapping screw in and out to often because the screw hole may become enlarged and the screw will loose all grip and proper function.

Lock nut

Lock nut fixation



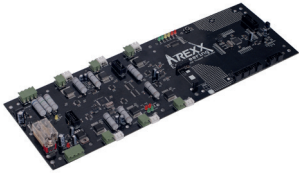
Spanner

To lock the lock nut in a proper way, use a plier or spanner.

See drawings!

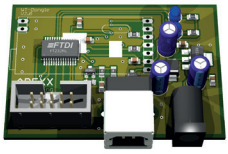
3. Partlist

Main PCB



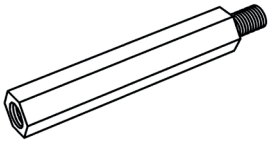
O 1x

USB Programmer



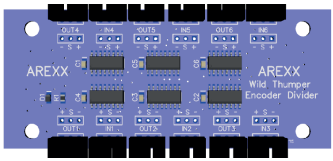
O 1x

Spacer
M3 (metal or plastic)



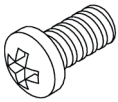
O 12x M3x25
O 4x M3x7

Motor encoder frequency divider



O 1x

Roundhead
screw M3x6



O 16x

Nut M3

O 16x

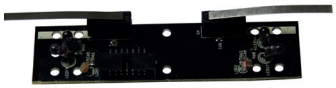
Flat cable

O 1x 10 Pol 30cm
USB Programmer
O 1x 14 Pol 12cm
Front PCB

Divider cable

O 6x

Front PCB



O 1x

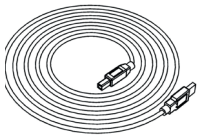
Power connector

O 2x

Motor connector

O 6x

USB cable



O 1x

APC-220

O 2x

Preparing the Wild Thumper chassis:

For this preparation you need

1x WT Chassis
1x Devider board
6x Motor connector
6x Motor wires
8x M3x25 Distance holder
4x M3x7 Distance holder
4x M3 Screw
4x M3 Nut

Remove WT Chassis top.

Connect the 6 white divider wires between main PCB and divider PCB

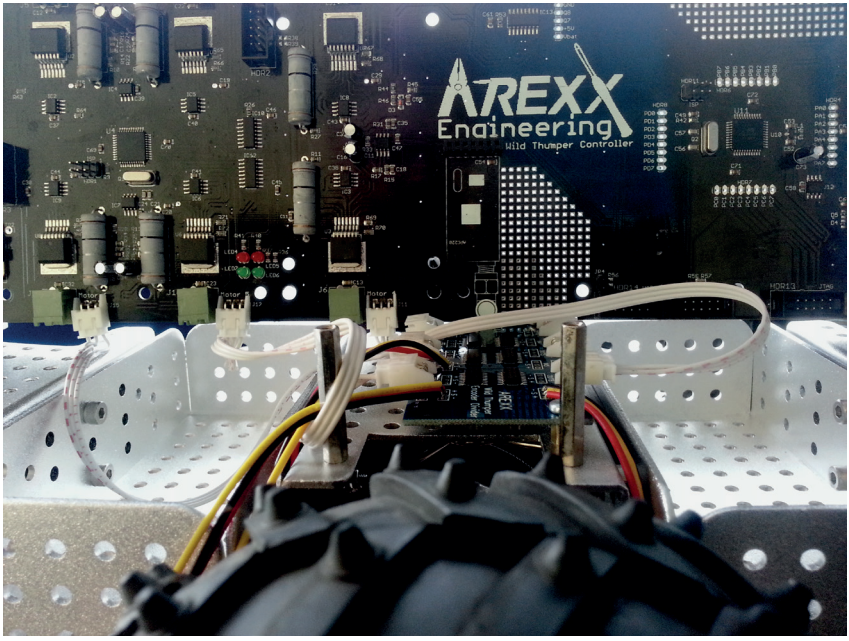
Solder the motor encoder wires to the divider PCB

CHECK THE POLARITY !

Of the 4 encoder wires see page 13.

Only 3 are soldered to the divider PCB pulse out white and yellow will both work

Assemble the Moter Encoder divider on the chassis use the M3x7 distanceholders, M3 screws and nuts



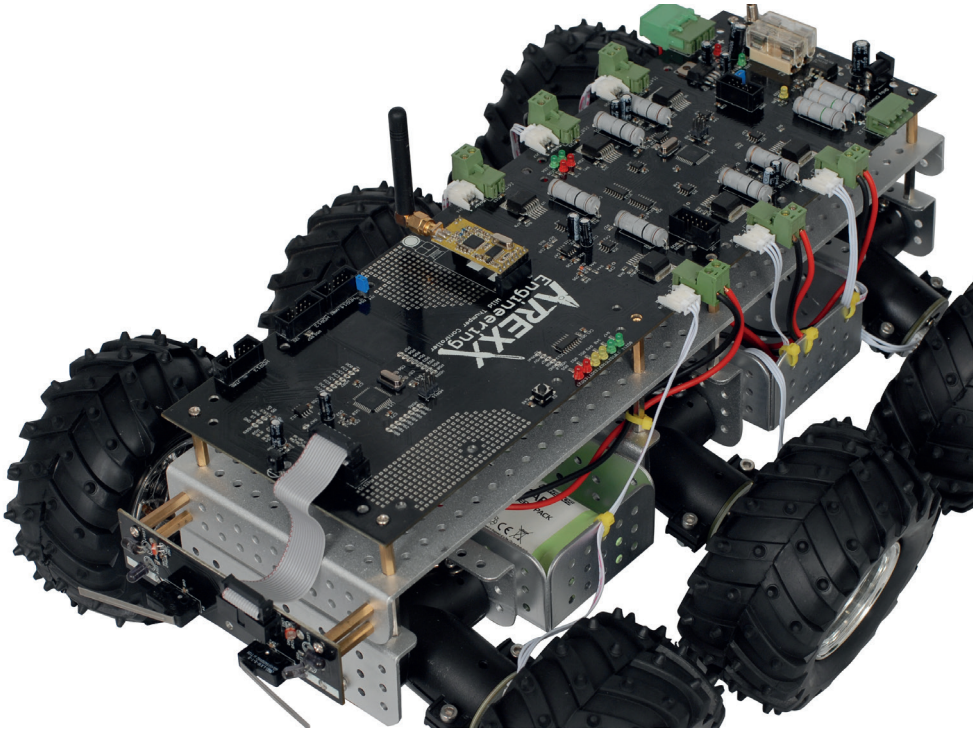
Assemble the 6 motor connectors to the motor wires.

CHECK THE POLARITY!

4. ASSEMBLY OF THE WILD THUMPER

We advise to assemble all parts in the following order;

- **Remove top**
- **Assemble the spacers for the PCB's**
- **Connect and solder all wires to the frequency divider PCB**
- **Assemble the motor encoder frequency divider PCB on the M3x7 spacers**
- **Assemble the top again with the extra M3x25 distance holders**
- **Install main PCB on the top with the M3 screws**
- **Install the axes and the wheels**
- **Assemble the front PCB)**
- **Connect all wires**
- **Install the APC module on the PCB and on the USB program adapter**
- **Program the 2 Processors with the USB program adapter**
- **Do the Selftest, this will only will work WIRELESS! not over USB cable!**

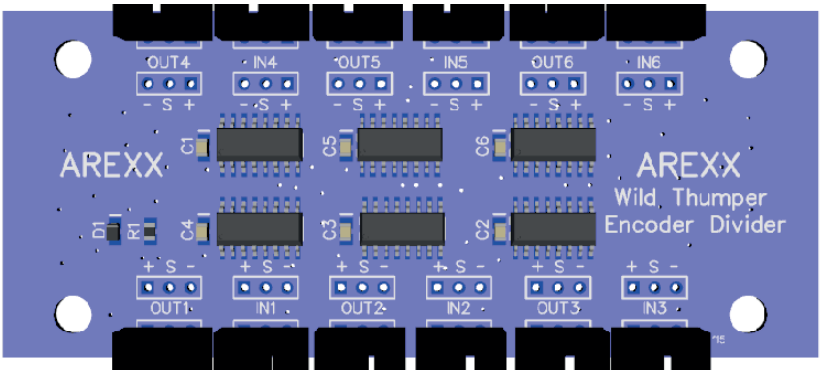


Important encoder information

Motor encoder frequency divider

The newest revisions of the Wild Thumper chassis feature encoders build right into the motor housing.

This means more sturdy encoders, no more hassle for the user in terms of assembling the encoder board and so on. The encoder board is only for Wild Thumper chassis without the encoder on the outside near the wheel. This new encoder has a much higher operating frequency since the gearbox normally reduces the frequency by 34 or 75 times. The Wild Thumper controller can't handle signals with such a high frequency directly, it needs an interface in the form of a frequency divider. This divides the frequency by 32, creating a usable signal for the Wild Thumper controller.



Assembly of the motor encoder frequency divider

Connect first the white divider wires between the main PCB encoder connector and the OUT on the Wild Thumper encoder frequency divider PCB.

Connect (solder or use a connecteor) the +, - and pulse out from the motor encoder to the IN on the Wild Thumper encoder frequency divider PCB.

You can use pulse out 1 or 2 this makes no difference!

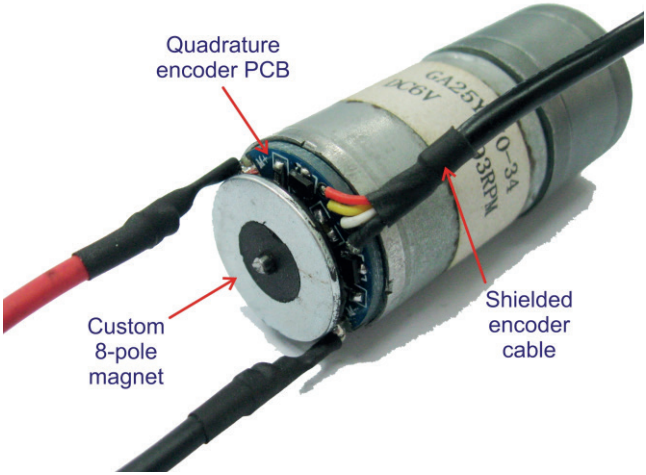
Assemble the PCB to the chassis with the M3x7 spacers and the M3 screws and nuts.

The motor encoder frequency divider PCB has 12 connectors in total, 6 inputs and 6 outputs. On each side of the board you have 3 input connectors and 3 output connectors. It is recommended to mount the board so that the connectors face the side of the Wild Thumper, it can be mounted below the main PCB on the lower deck.

It is highly recommended to mount it in the middle of the Wild Thumper. The connectors with the names IN1 to IN6 are the high frequency side of the divider, the connectors with the names OUT1 to OUT6 are the low frequency side of the connector. The OUT connectors can connected through a small cable to the Wild Thumper controller. The cables that come out of the encoder should be connected to the IN connectors. The color codes of the encoder cables is as follows:

- Red = + (3.3/5V)
- Black = - (Ground)
- White = Pulse out (1)
- Yellow = Pulse out (2)

The red cable has to be connected to the pin with the “+” marking, the black cable has to be connected to the pin with the “-” marking. The yellow or the white cable has to be connected to the pin with the “S” marking. It makes absolutely no difference if you connect the white or the yellow cable, for this use there is no difference which pulse out you are using.



Quadrature encoders

Due to popular demand the 4WD and 6WD Wild Thumper chassis's now have the option of quadrature encoders. These encoders can operate with both 3.3V and 5V logic systems and have reverse polarity protection built into the sensors.

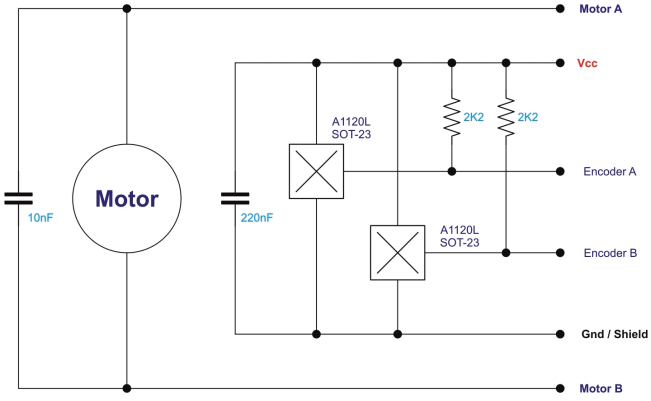
6WD chassis's will have a left and right encoder on the middle wheels. 4WD chassis's will have a left and right encoder on the front wheels. Upon request we can produce chassis's with encoders on every wheel if required.

Our encoders use custom made 8-pole magnets and 2 hall-effect sensors to provide 2 square waves, 90° out of phase with a total of 16 state changes per motor revolution.

The output resolution will be 544 state changes per revolution for the 34:1 gear-box and 1200 state changes per revolution for the 75:1 gearbox.

Specifications:

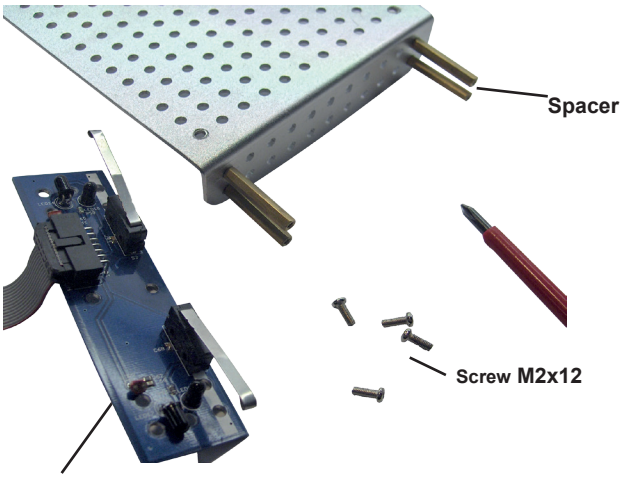
Motor voltage:	6V (7.5V maximum)
Motor stall current:	5.5A @ 7.5V (typical)
Output torque:	4Kg/cm with 34:1 ratio
	8.8Kg/cm with 75:1 gearbox
Encoder Vcc:	3V – 24V
Encoder output A:	square wave: Gnd-Vcc
Encoder output B:	square wave: Gnd-Vcc
Encoder resolution:	16 state changes per revolution



Assembly of the front-PCB

The following parts will be needed:

1x Front PCB
4x Spacer M3x25
4x Screw M2x12

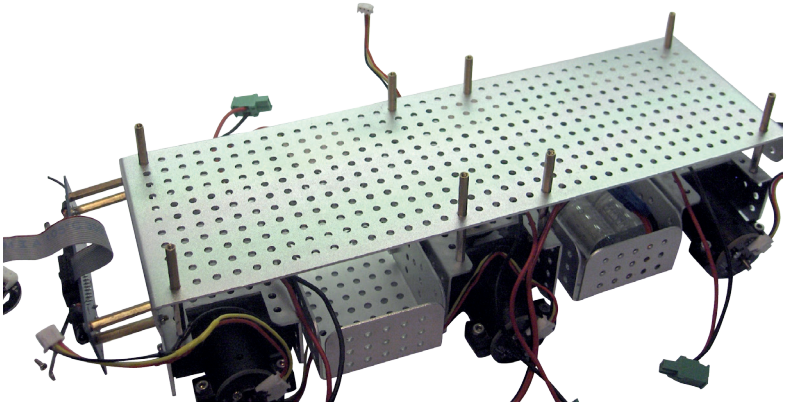


Attach the front-PCB to the front of the Wild Thumper‘s chassis according to the details in the sketch.

Assembling the spacer for the main PCB

The following parts will be needed:

1x WT Chassis
1x Main PCB
8x Spacers M3

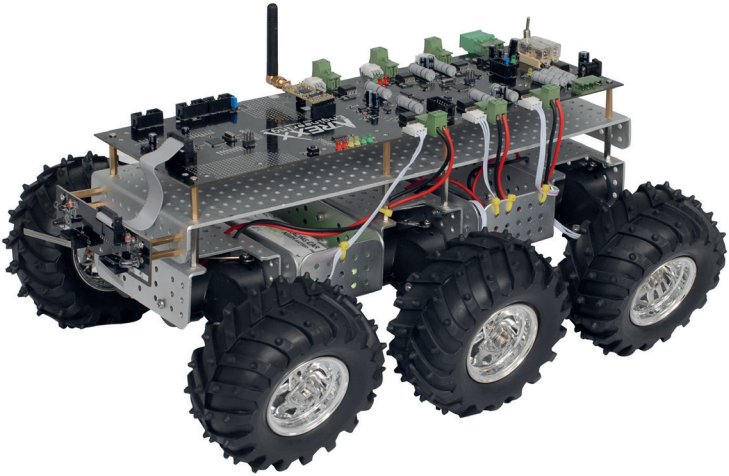


Mount the upper PCB to the WT‘s chassis and attach the module with the 8 spacers. as shown on the pciture.

Final assembly:

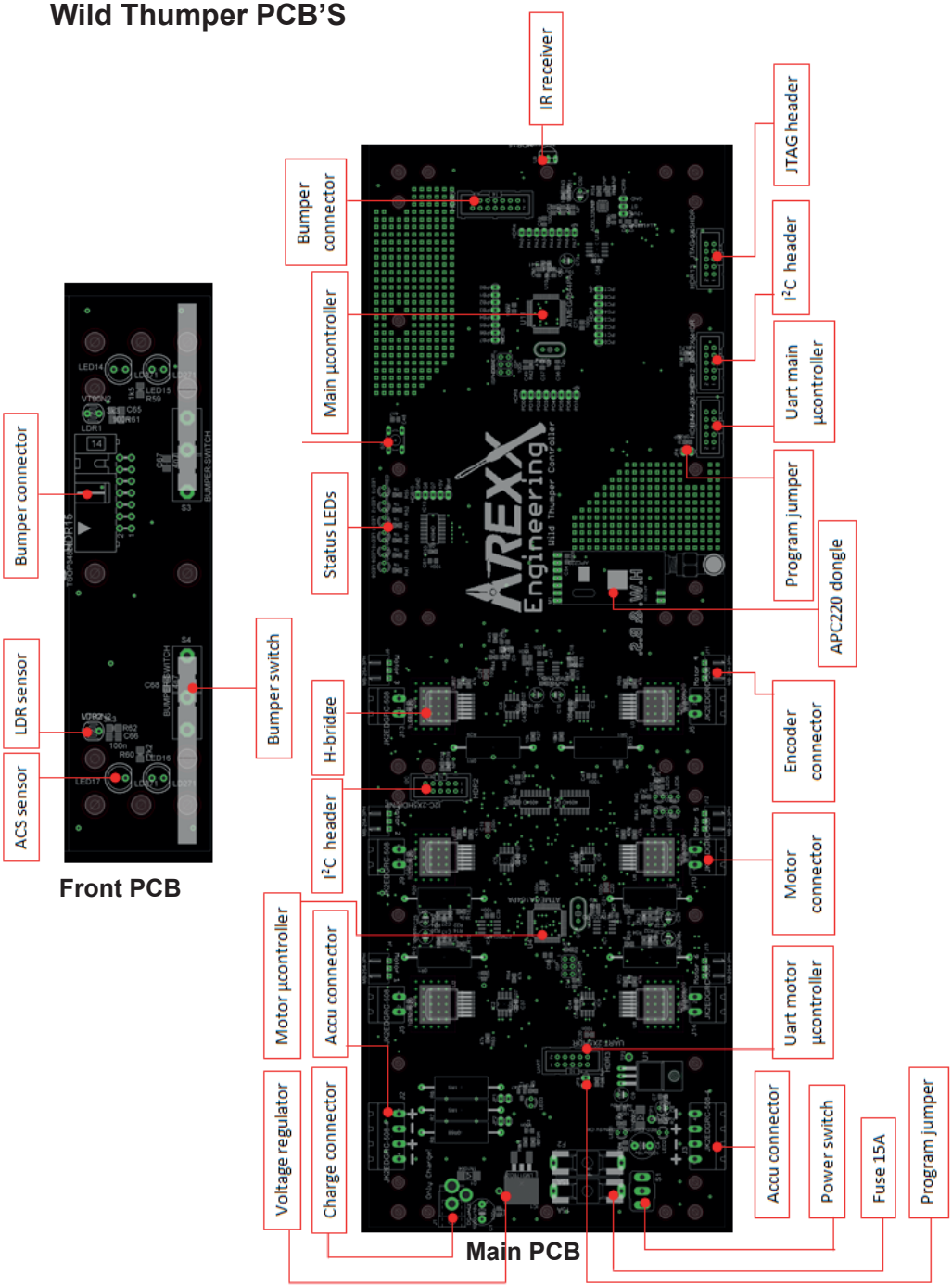
The following parts will be needed:

1x WT Chassis
1x Main PCB
1x APC-220
1x Flat cable
(for the Front PCB)



Mount the PCB at the top plate. Connect all cabling to the PCB and attach the wheels to the axes. Finally insert the APC-220 into its socket as it has been illustrated in the sketch.

Wild Thumper PCB'S



5. WILD THUMPER ELECTRONICS DOCUMENTATION

5.1. Power supply

The Wild Thumper is to be controlled by wireless signals and in order to allow a wireless control we will have to provide the system with batteries, which have been designed to deliver 7.2 Volts.

The electronic circuitry however has been designed for a maximal voltage of 5V and therefore we must reduce the power supply‘s voltage to a lower value. For optimal voltage control we chose a L4940V5 regulator circuit, which has been designed by STMicroelectronics for a voltage drop of merely 0.5V.

The power supply has been protected by a diode (D2) against bad polarity of terminal connections. The diode simply refuses the current flow at a reversal of the power supply‘s voltage. Interferences by RF-signals will be blocked by RF-suppression capacitors (C7 and C8). We also provided the circuit with two buffer capacitors (C6 and C9) at the input and output circuitry.

5.2. Charger circuit

Inserted batteries may be charged without removing the devices from the system. In order to provide this feature the circuit has been equipped with a current source, controlled by a LM317 regulator. In charging mode the regulator will be used as a current source.

As soon as the charging mode has been completed the current source will be deactivated and transistor T1 switches the circuit from current source to voltage stabilization.

5.3. Motor control

The motor control circuitry has been designed to use six identical modules, each of which are controlling one singular motor. Each module has been provided with an H-bridge, a logic circuit (containing two NAND-gates) to allow a multiple usage of the microprocessor‘s gates and a current measurement circuit. The circuitry allows the engines to run in a left and right direction including a braking mode. The H-bridge has been designed to apply Pulse Width Modulation (PWM) for speed control.

5.4. Motor current measurement

Each motor current sensor module has been equipped with a power resistor. Ohm‘s law states that the power resistor‘s current will be proportional to the voltage drop. In order to minimize the voltage drop we choose a minimal value of only 0.1 Ohm for the resistor. At such values the voltage drop is very low (for instance 0.1V at a current of 1A) and must be amplified in an „operational amplifier“, abbreviated as: „opamp“).

U_in=I_motor×R_3

U_out=A_((amplification))×U_in

A_((amplification))=1+R_1/R_2

Dimensioning formulas:

R_3=0,1Ω
I_(motor_max)=5,5A
U_(out_max)=4V at a power supply voltage of 5V.

U_(in_max)=I_(motor_max)×R_3= 5,5A*0,1Ω=0,550V
A=U_(out_max)/U_(in_max) =4V/0,550V=7,27
R_2/R_1 =A-1=7,27-1=6.27
R_2/R_1 =6.27

Based on this formula we may calculate the resistor values for R1 and R2. For economical reasons we applied resistors from the E12-, respectively E24-series. For an optimal amplification value we chose:

R1 = 6200 Ohm
R2 = 39000 Ohm
A=1+R_1/R_2 =1+39000/6200=7,29

These design values reduce the deviation between the calculated and designed amplification value to a minimal value (7,29 – 7,27 = 0,02), which is a tolerable error.

5.5. Encoder

For speed control each individual wheel has been provided with its own encoder circuitry. The encoder contains a magnet ring and a Hall-sensor. The magnet ring is equipped with 6 magnetic pairs of poles, which in a rotational movement generate an alternating magnetic field.

As soon as the applied Hall-Sensor detects a south pole the sensor‘s output will generate 5V and for a north pole 0V. A pulse width analysis of the generated pulses allows us to derive the speed value for each individual wheel.

5.6. Synchronization of the motors

Tolerances in the motor specifications may vary between - 10% and +10%. For this reason each motor is provided wit its own control circuitry to guarantee an identical speed for all motors.

The speed control has been designed according to the following specification: The required speed will be communicated (by I²C) by the central control unit and transformed into a control parameter for the PID-controller for the motor. According to this value the system calculates a PWM-(pulse width modulation) value, which is used for the motor‘s control. Then the motor‘s speed will be registered again, which will result in a new control value.

5.7. Microcontroller

As listed in robot‘s specification the system uses two micro-controllers, which have been designed by Atmel. For simplicity we did choose the same type Atmega644 for these controllers, which have been equipped with I²C for internal communication. A system overview has been documented in fig. 16.

Abb. 16 : Schematic overview of motor-data current

5.7.1. Main controller

The main controller manages all jobs in the Wild Thumper‘s system and the decisions, which may lead to actions, involving:

- management of incoming PC-data
- synthesis of reports and reporting the messages to PC-systems if reporting has been activated
- command management for the individual motor control systems.
- management of the I²C communication and interface.
- regular updates of measurement parameter for the motor control system.
- updates for all sensor data.

The main controller has been equipped with two interfaces: an extra UART-interface respectively a JTAG-interface. The UART Interface is to be used for communications to a third system. The JTAG interface has been dedicated to the debugging procedure of the main controller.

5.7.2. Motor controller

The motor controller has been designed to manage the commands, which have been sent from the main controller and are arriving by I²C-interface. An analysis of incoming data and calculation process in a software-based PID-controller results in control parameters for each individual motor. These calculated parameters guarantee identical rotational speeds for each motor.

Apart from these jobs arriving from the main controller the motor controller regularly checks the current and speed. The actually valid currents and speed parameters may be read by the main controller at any time.

5.7.3. Communication between main and motor controllers (I2C)

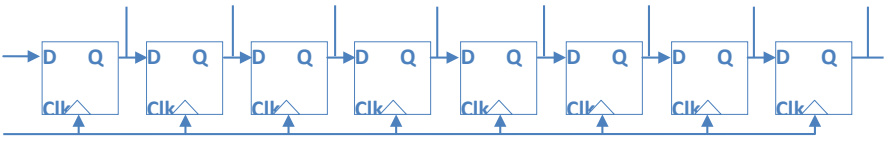
The communication between both controllers is applying the serial I²C-bus system, which is to be understood as a two-wire system-bus, using one wire (SDA) for data and another wire (SCL) for the System Clock Signal.

Both signal lines have been connected to the power supply voltage by pull-up resistors. In a passive mode the signal level is H=high. As soon as the micro-controller is ready to transmit data a start sequence is to be generated, to be followed by 8 bits for the address and additionally the information for the relevant IC. Each 8bit sequence will be completed by an acknowledge-bit, which is needed for the system to check the integrity of the received data. Having transferred all data the system generates a stop signal, completing the transfer of the communication data.

The motor controller uses two registers to read data. These registers contain measurement data and the motor controller‘s status. Additionally we provided the system with some commands to generate control commands for the Wild Thumper.

5.8. Shift register

The limited number of input and output ports of the available micro-processors required an extra device to expand the number of output ports by external devices, which have been implemented in shift registers. A shift register has been designed by concatenating a number of D-flipflops, which allow us to transform serial data into parallel data. In a D-flipflop the D-data signal will be transferred to the output at each positive flank of the clock signal. A series of eight D-flipflops allows us to convert a serial signal into a parallel bit pattern, which is presented at the parallel output terminals of the shift register.



In this project we apply shift registers of an IC-type number HEF4094, which provides us with a serial interface using three signals:

- Data
- Clock signal
- Latch signal

The additional latch-signal is to be used for the transfer of data to the output register. Internally data will be stored as long as the latch signal is 0V, but the result will not be transferred to the output terminals. The next drawing illustrates the transfer of data, in which output terminal six is to be activated.

In order to control a shift register from a micro-controller we apply a so-called SPI-bus, which is equipped with three data lines:

- MOSI (Master out, Slave in)
- MISO (Master in, Slave out)
- CLOCK (clock signal)

For communication between two systems the SPI-bus is a popular communication system. The SPI-bus uses two data lines: each system is using its own line. In analogy to the shift register data will be transferred at the positive flank of the clock signal.

The shift registers do not return any messages and therefore one data-line will be sufficient for controlling the shift register. Therefore the signals MOSI and CLOCK are all we need to completely control the shift register.

5.9. The RF-module (→ the Radio Frequency Module)

The RF-module has been connected to the main controller by a serial interface (UART). In order to setup the inboard-parameters the SET-signal line of the transceiver also has been connected to the micro-controller.

5.10. Sensors

The sensors have been located at two printed circuit boards: some sensors are to be found at the front bumper of the Wild Thumper, the others have been installed at the PCB with the remaining electronic circuitry. The Wild Thumper has been equipped with the following sensors:

5.10.1. LDR

At the front bumper of the Wild Thumper we may identify an LDR at the left and at the right side. An LDR is a Light Dependent Resistor, in which the resistance will decrease with the intensity of the incoming light. In analogy to the battery‘s voltage-sensor we will apply a voltage splitter, which now splits up the 5V-voltage instead of the battery‘s voltage.

The voltage-splitter uses a fixed resistor and an LDR, which allows us to determine the light intensity according to the included schematic. The output voltage of the splitter will be interpreted at the ADC-input terminal of the micro-controller.

5.10.2. Infrared collisions-detector

The infrared LEDs for the collisions-detector have been located at the front-PCB of the Wild Thumper. These LEDs send data packages, which will be reflected by obstacles. The reflected signals are to be detected in infrared sensors at the main PCB. As soon as an obstacle has been detected the reflected data packages are received and processed to trigger the robot‘s reactions to these events.

5.10.3. The bumper

We designed a tiny PCB, which is to be mounted at the front-side of the Wild thumper. This PCB contains two micro-switches, which are activated as soon as the robot touches obstacles and may trigger the robot‘s reactions to these events.

6. THE INITIAL WT-TEST

1. Start by assembling the mechanical and electronic modules of the Wild Thumper according to the manual.
2. Connect the battery pack (or alternatively the power supply)
3. Switch on the system by activating the main switch.

Connect the power supply

Power supply may be managed by two alternatives.
The simplest solution is a power supply in the range 8V up to 12 Volt / 2 Ampere at the DC-plug. This connector directly leads the voltage line to the input terminal of the voltage stabilizer circuit.

Batteries
The second alternative is a rechargeable battery package, connected to the battery plug. This solution directly connects the battery power line behind the output terminal of the voltage stabilizer circuit. Therefore this voltage always MUST be kept below the 7.2 Volt level!!

WARNING!

The maximum Accu- or Battery voltage for the Wild Thumper controller is 7.2 V!

As soon as the robot has been connected to the power supply you may activate the robot by closing the main power switch. The power on status will be indicated by an activated yellow LED (LED1).

Well, this has been easy for a first test and it looks like a final stage of the experiments. This however is only the first job in a number of experiments …. We are not ready yet!

Now we will proceed with chapter 7, in which the software is to be installed.

The Wildthumper must be programmed with the USB program adapter! It will be controlled with the Wild Thumper controller software (also self-test) wireless!

During battery charching the yellow LED (LED 3) wil be on.

Continue with chapter 7, where we will start with the Software installation.

7. Software Installation

Let's do the software installation now. A properly installed software is of paramount importance for all following chapters.

As you need administrator rights, you have to log into your system as an administrator!

We recommend to read the whole chapter thoroughly first and then start with the installation step by step.

The user must have basic knowledge of Windows or Linux based computers and be familiar with current programs such as file managers, web browsers, text editors, file compression software (WinZip, WinRAR, unzip and others) and eventually Linux shell etc.! If your computer knowledge is very limited, you should learn more about systems before you start using the Robot Arm. This manual is not intended as an introduction to computers which would go much too far! It is only aimed at the Robot Arm, its programming and the specific software required.

The Robot Arm CD-ROM
You have probably already inserted the CD-ROM into your computer drive - if not, please do it now! In Windows, the CD menu should appear shortly afterwards per autostart. If not, you can open the file "start.htm" with a web browser as e.g. Firefox in the main directory of the CD through file manager. By the way, the installation files for Firefox are also on the CD in the folder

<CD-ROM drive>:\Software\Firefox

if ever you haven't installed an updated web browser (it should be at least Firefox 1.x or Internet Explorer 6 ...)

After the language selection you will find in the CD menu, in addition to this manual (that you can also download from our home page), information, data sheets and pictures, also the menu item "Software". It contains all software tools, USB drivers and example programs with source code for the Robot Arm.

Depending on the safety settings of your web browser, you can start the installation programs directly from the CD!

If the safety settings of your web browser don't allow a direct installation from the CD-ROM, you have to copy the files first into a directory on your hard disc and start the installation from there. For more details please refer to the software page in the CD menu. Alternatively, you can also switch to the CD drive through a file manager and install the software from the CD. The names of the directories are self-explanatory so that you can allocate them easily to the corresponding software packages and operating systems.

WinAVR - for Windows

We will start with the installation of WinAVR. WinAVR is - as the name says - **only available for Windows!**

Linux users can skip to the next section.

WinAVR (pronounce like the word "whenever") is a collection of many useful and necessary programs for the software development for AVR micro controllers in C language. In addition to the GCC for AVR (designated by the term "AVR-GCC", more details later) WinAVR includes the convenient source text editor "Programmers Notepad 2" that we will also use for the program development of the Robot Arm.

WinAVR is a private project that is not supported by a company. It is available for free in the internet. You will find updated versions and more information at:

http://winavr.sourceforge.net/

In the meantime the project gets the official support from ATMEL and the AVRGCC is available for AVRStudio, the development environment for AVR's from ATMEL. However we will not describe it in this manual as Programmers Notepad is much better suited for our purpose.
The WinAVR installation file is on the CD in the folder:

<CD-ROM drive>:\Software\AVR-GCC\Windows\WinAVR\

The installation of WinAVR is simple and self-explanatory. Normally you don't need to change any settings. So, just click on "Continue"!

If you use Windows Vista or Windows 7, you must install the latest version of WinAVR! It should also work perfectly with Windows 2K and XP. If not, you can try one of the older versions that are also on the CD (before you make a new installation of WinAVR, you have to uninstall the existing version first!). Officially Win x64 is not yet supported but the CD contains a patch for Win x64 systems if a problem arises. You will find more information on the software page of the CD menu.

AVR-GCC, avr-libc uad avr-binutils - for Linux

(Windows users can skip this section!)

Linux might require more effort. Some distributions already contain the required packages but they are mostly obsolete versions. Therefore you need to compile and install newer versions. It is impossible to describe in detail the numerous Linux distributions as SuSE, Ubuntu, RedHat/Fedora, Debian, Gentoo, Slackware, Mandriva etc. that exist in many versions with their own particularities and we will keep here only to the general lines.

The same applies to all other Linux sections in this chapter!

The procedure described here must not necessarily work for you. It is often helpful to search in the internet e.g. for “<LinuxDistribution> avr gcc” or similar. (Try different spellings). The same applies to all other Linux sections - of course with the suitable keywords! If you encounter problems with the installation of the AVR-GCC, you can also take a look in our robot network forum or in one of the numerous Linux forums. First of all, you have to uninstall already installed versions of the avr-gcc, the avr-binutils and the avr-libc because, as said, these are mostly obsolete. You can do that via the package manager of your distribution by searching for “avr” start up and uninstall the three above mentioned packages - as far as they exist in your computer. You can find out easily if the avr-gcc has already been installed or not via a console as e.g.

> which avr-gcc

If a path is displayed, a version is already installed. So just enter:

> avr-gcc --version

and look at the output. If the displayed version is smaller than 3.4.6, you have to uninstall in any case this obsolete version.

If the version number lies between 3.4.6 and 4.1.0, you can try to compile programs (see following chapter). If it fails, you have to install the new tools. We will install hereafter the currently most updated version 4.1.1 (status March 2007) together with some important patches.

If the packages above do not appear in the package manager although an avr-gcc has definitely been installed, you need to erase manually the relevant binary files- i.e.search in all /bin, /usr/bin etc. directories for files starting with “avr” and erase these (of course ONLY these files and nothing else!). Eventually existing directories as /usr/avr or /usr/local/ avr must also be erased.

Important: You have to make sure that the normal Linux development tools as GCC, make, binutils, libc, etc. are installed prior to compiling and installing! The best way to do so is via the package manager of your distribution. Every Linux distribution should be supplied with the required packages on the installation CD or updated packages are available in the internet.

Make sure that the “texinfo” program is installed. If not, please install the relevant package before you continue - otherwise it will not work!

Having done that, you can start with the installation itself.

Now you have tree options: either you do everything manually or you use a very simple to use installation script and as last you can use an alrewady compiled .dep install package.

We recommend to try the installation script first. If this doesn’t work, you can still install the compiler manually.

Latest versions see; http://www.wrightflyer.co.uk/avr-gcc/

Attention:
You should have enough free disk space on your hard disk! Temporarily more than 400Mb are required. Over 300Mb can be erased after the installation but during the installation, you need all the space.

Many of the following installation steps require **ROOT RIGHTS**, so please log in with “su” as root or execute the critical commands with “sudo” or something similiar as you have to do it in Ubuntu e.g. (the installation script, mkdir in /usr/local directories and make install require root rights).

Please note in the following the EXACT spelling of all commands!
Every sign is important and even if some commands look a bit strange, it is all correct and not a typing mistake! (<CD-ROM-drive> has of course to be replaced by the path of the CD-ROM drive!)

The folder on the CD:

<CD-ROM drive>:\Software\avr-gcc\Linux

contains all relevant installation files for the avr-gcc, avr-libc and binutils.
First of all, you have to copy all installation files in a directory on your hard disk - **this applies for both installation methods!** We will use the Home directory (usual abbreviation for the current home directory is the tilde: „~“):

```
> mkdir ~/Robot Arm
> cd <CD-ROM drive>/Software/avr-gcc/Linux
> cp * ~/Robot Arm
```

After the successful installation you can erase the files to save space!

Automatic Installation Script

Once you have made the script executable via chmod, you can start immediately:

> cd ~/Robot Arm
> chmod -x avrgcc_build_and_install.sh
> ./avrgcc_build_and_install.sh

Answer “y” to the question if you want to install with this configuration or not.

PLEASE NOTE: The compilation and installation will take some time depending on the computing power of your system (e.g about 15 min. on a 2GHz Core Duo Notebook. Slower systems will need longer).

The script will include also some patches. These are all the .diff files in the directory.
If the installation was successful, following message will be displayed:

(./avrgcc_build_and_install.sh)
(./avrgcc_build_and_install.sh) installation of avr GNU tools complete
(./avrgcc_build_and_install.sh) add /usr/local/avr/bin to your path to use the avr GNU tools
(./avrgcc_build_and_install.sh) you might want to run the following to save disk space:
(./avrgcc_build_and_install.sh)
(./avrgcc_build_and_install.sh) rm -rf /usr/local/avr/source /usr/local/avr/build

As suggested, you can execute
rm -rf /usr/local/avr/source /usr/local/avr/build
This erases all temporary files that you will not need anymore.

You can skip the next paragraph and set the path to the avr tools.

If the execution of the script failed, you have to look attentively to the error message (scroll the console up if necessary). In most cases it is just a matter of missing programs that should have been installed earlier (as e.g. the before mentioned texinfo file). Before you continue after an error, it is recommended to erase the already generated files in the standard installation directory “/usr/local/avr“ – preferably the whole directory.

If you don’t know exactly what has gone wrong, please save all command line outputs in a file and contact the technical support. Please join always as much information as possible. This makes it easier to help you.

GCC for AVR

The GCC is patched, compiled and installed a bit like the binutils:

```
> cd ~/Robot Arm> bunzip2 -c gcc-4.1.1.tar.bz2 | tar xf -
> cd gcc-4.1.1
> patch -p0 < ../gcc-patch-0b-constants.diff
> patch -p0 < ../gcc-patch-attribute_alias.diff
> patch -p0 < ../gcc-patch-bug25672.diff
> patch -p0 < ../gcc-patch-dwarf.diff
> patch -p0 < ../gcc-patch-libiberty-Makefile.in.diff
> patch -p0 < ../gcc-patch-newdevices.diff
> patch -p0 < ../gcc-patch-zz-atmega256x.diff
> mkdir obj-avr
> cd obj-avr
> ../configure --prefix=$PREFIX --target=avr --enable-languages=c,c++ \
--disable-nls --disable-libssp –with-dwarf2
> make
> make install
```

After the \ just press Enter and continue to write. This way the command can be spread over several lines, but you can also just drop it.

AVR Libc

And last but not least the AVR libc:

```
> cd ~/Robot Arm
> bunzip2 -c avr-libc-1.4.5.tar.bz2 | tar xf -
> cd avr-libc-1.4.5
> ./configure --prefix=$PREFIX --build=`./config.guess` --host=avr
> make
> make install
```

Important: at –build=`./config.guess` make sure to put a backtick ` (à <-- the grave accent on the a!) and not a normal apostrophy or quotation marks as this wouldn't work.

Set the Path

You must make sure now that the directory /usr/local/avr/bin is registered in the path variable otherwise it will be impossible to retrieve the avr-gcc from the console or from the makefiles. To that end, you have to enter the path in the file /etc/profile or /etc/environment or similiar (varies from one distribution to another) – separated by a colon ":" from the other already existing entries. It could look in the file like:

PATH="/usr/local/bin:/usr/bin:/bin:/usr/X11R6/bin:/usr/local/avr/bin"

Now enter in a console "avr-gcc —version" as described above. If it works, the installation was successfull!

NEWER VERSIONS

There are regular newer versions on internet, please always check if you can find a newer version as we put on the CD now.

Manual Installation

If you prefer to install the compiler manually or the installation via th script failed, you can follow the instructions below.

The description is based on following article:

http://www.nongnu.org/avr-libc/user-manual/install_tools.html

that is also included on the CD in PDF format in the AVR Libc documentation:

<CD-ROM drive>:\Software\Documentation\avr-libc-user-manual-1.4.5.pdf

Our description here is much shorter but includes a few important patches. Without these, some tings will not work properly.

First of all we have to create a directory in which we wil install all tools. That should be /usr/local/avr.
Also enter in a console AS A ROOT:

> mkdir /usr/local/avr
> mkdir /usr/local/avr/bin

It must not necessarily be this directory. We just create the variable $PREFIX for this directory:

> PREFIX=/usr/local/avr
> export PREFIX

This must be added into the PATH variable:

> PATH=$PATH:$PREFIX/bin
> export PATH

NEWER VERSIONS
There are regular newer versions on internet, please always check if you can find a newer version as we put on the CD now.

Binutils for AVR

Now you must unpack the sourcecode of the binutils and add a few patches. We suppose in our example that you have copied everything into the home directory ~/Robot Arm:

> cd ~/Robot Arm
> bunzip2 -c binutils-2.17.tar.bz2 | tar xf -
> cd binutils-2.17
> patch -p0 < ../binutils-patch-aa.diff
> patch -p0 < ../binutils-patch-atmega256x.diff
> patch -p0 < ../binutils-patch-coff-avr.diff
> patch -p0 < ../binutils-patch-newdevices.diff
> patch -p0 < ../binutils-patch-avr-size.diff
> mkdir obj-avr
> cd obj-avr

Now execute the configure script:

> ../configure --prefix=$PREFIX --target=avr --disable-nls

This script detects what is available in your system and generates suitable makefiles. Now the binutils can be compiled and installed:

> make
> make install

Depending on the computing power of your system, this can take a few minutes. That applies also to the next two sections, especially to the GCC!

Java 6

The RobotLoader (see Info below) has been developed for the Java platform and is suitable for Windows and Linux (theoretically also for operating systems like OS X but AREXX Engineering is unfortunately not yet in a position to give official support). To make it work, you need to install an updated Java Runtime Environment (JRA). It is often already installed on the computer but it must be at least version 1.6 (= Java 6)! If you have no JRE or JDK installed, you must install the supplied JRE 1.6 from SUN Microsystems or alternatively download a newer version from http://www.java.com or http://java.sun.com.

Windows

The JRE 1.6 for Windows is in following folder:

<CD-ROM drive>:\Software\Java\JRE6\Windows\

Under Windows the installation of Java is very simple. You just have to start the setup and follow the instructions on the screen - that's it. You can skip the next paragraph.

Linux

Under Linux the installation doesn't present any major problems although some distributions require some manual work.

In the folder:

<CD-ROM drive>:\Software\Java\JRE6\

you will find the JRE1.6 as an RPM (SuSE, RedHat etc.) and as a self-extracting archive ".bin". Under Linux it is advisable to look for Java packages in the package manager of your distribution (keywords e.g. „java", „sun", „jre", „java6" ...) and use the packages of your distribution rather than those on the CD-ROM! However make sure to install Java 6 (=1.6) or a newer version but definitely not an older one!

Under Ubuntu or Debian, the RPM archive doesn't work directly. You will have to use the package manager of your distribution to find a suitable installation package. The RPM should however work well with many other distributions like RedHat/Fedora and SuSE. If not, you always have the solution to unpack the JRE (e.g. to /usr/lib/Java6) from the self-extracting archive (.bin) and set manually the paths to the JRE (PATH and JAVA_HOME etc.).

Please refer to the installation instructions from Sun that you will find also in the above mentioned directory and on the Java website (see above).

Windows

The JRE 1.6 for Windows is in following folder:

<CD-ROM drive>:\Software\Java\JRE6\Windows\

Under Windows the installation of Java is very simple. You just have to start the setup and follow the instructions on the screen - that's it. You can skip the next paragraph.

Linux

Under Linux the installation doesn't present any major problems although some distributions require some manual work.

In the folder:

<CD-ROM drive>:\Software\Java\JRE6\

you will find the JRE1.6 as an RPM (SuSE, RedHat etc.) and as a self-extracting archive ".bin". Under Linux it is advisable to look for Java packages in the package manager of your distribution (keywords e.g. „java", „sun", „jre", „java6" ...) and use the packages of your distribution rather than those on the CD-ROM! However make sure to install Java 6 (=1.6) or a newer version but definitely not an older one!

Under Ubuntu or Debian, the RPM archive doesn't work directly. You will have to use the package manager of your distribution to find a suitable installation package. The RPM should however work well with many other distributions like RedHat/Fedora and SuSE. If not, you always have the solution to unpack the JRE (e.g. to /usr/lib/Java6) from the self-extracting archive (.bin) and set manually the paths to the JRE (PATH and JAVA_HOME etc.).

Please refer to the installation instructions from Sun that you will find also in the above mentioned directory and on the Java website (see above).

You can check if Java has been correctly installed by entering the command “java-version” in a console. The output should be approximately as follows:

```
java version “1.6.0”
Java(TM) SE Runtime Environment (build 1.6.0-b105)
Java HotSpot(TM) Client VM (build 1.6.0-b105, mixed mode, sharing)
```

If the output is totally different, you have either installed the wrong version or there is another Java VM installed in your system.

Robot Loader

The Robot Loader has been developed to load easily new programs and all extension modules into the Robot Arm (as long as the modules are fitted with a compatible bootloader). Moreover it contains a few useful extra functions as e.g. a simple terminal program.

It is not necessary to install the RobotLoader. Just copy the program somewhere in a new folder on the hard disk.

```
<CD-ROM drive>:\Software\RobotLoader\RobotLoader.zip
```

Unpack the program somewhere on your hard disk e.g. in a new folder C:\Programme\RobotLoader (or similiar). This folder contains the RobotLoader.exe file that you can start with a double-click.

The Robot Loader program itself is in the Java archive (JAR) RobotLoader_lib.jar. Alternatively you can start this via the command line:

Under Windows:

```
java -Djava.library.path=”.\lib” -jar RobotLoader_lib.jar
```

Linux:

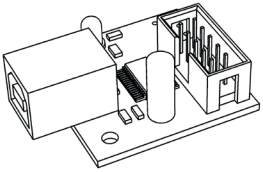
```
java -Djava.library.path=”./lib” -jar RobotLoader_lib.jar
```

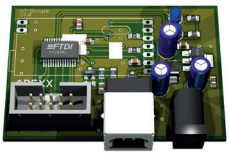
7. Programmer and Loader

To load a HEX Robot Arm program from the PC into the Robot Arm, we will use the USB programming adaptor and our RobotLoader software.

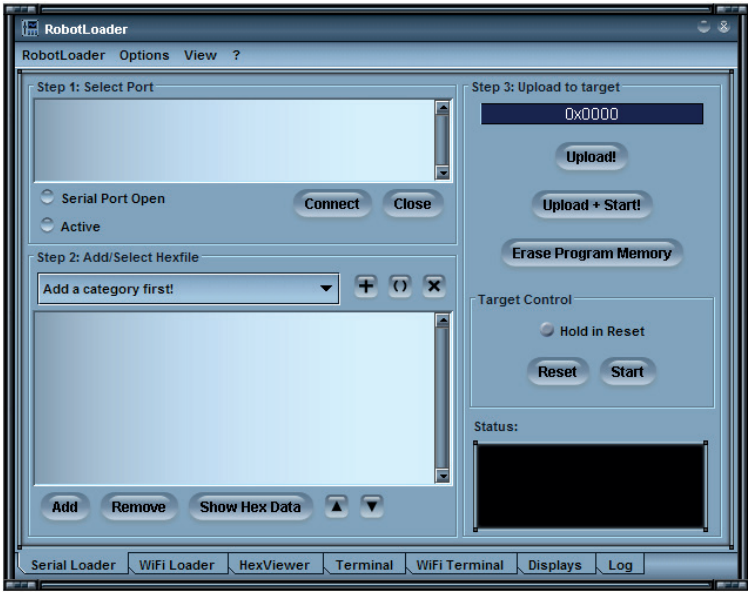
The loose USB port adaptor transmitter/receiver (transceiver) included in the package must be connected on one side to a USB port of the computer and on the other side to the Prog/UART port of the Robot Arm PCB.

The program upload into the Robot Arm erases automatically the previously existing program.





USB Programming adaptor



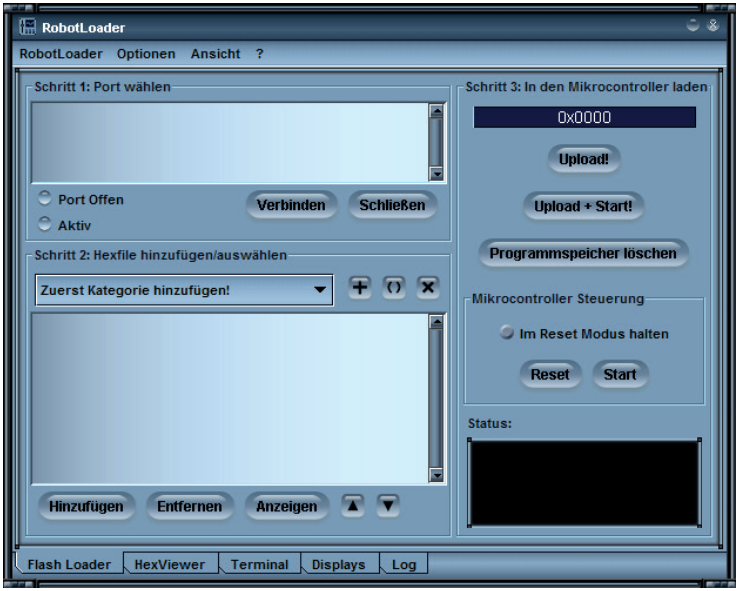
RobotLoader software

7.1. Robot Loader

As said, the RobotLoader has been developed to upload easily new programs into the Robot Arm and into all our robots (provided that they contain a compatible bootloader).

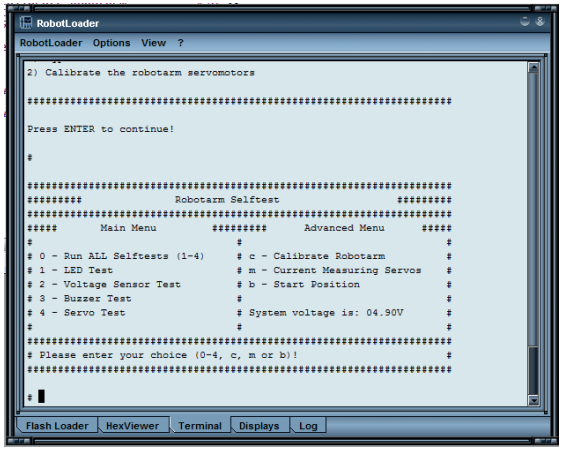
RobotLoader

If the voltage drops below < 6.7 V, a warning is displayed.



There are a few very usefull additional functions integrated like; a simple terminal program.

Terminal screen



The RobotLoader itself does not need any installation – just copy it someware in an file or put it on your harddisk.

If you are in this situation, a dialogue appears (under Windows) to install the new driver. You have to indicate the path to the system where it can find the driver. Under Windows 2k/XP you need to select first the manual installation and not to look for a web service. On our CD the driver is in the above mentioned directories.

So, just indicate the directory for your Windows version and eventually a few other files that the system doesn't find automatically (they are all in the directories mentioned below!) ...

Under Windows XP and later versions there is often a message that the FTDI drivers are not signed/verified by Microsoft (normally not here as the FTDI drivers are signed). This is irrelevant and can be confirmed without any problem.

Operation

For **32 and 64 Bit Windows 7, XP, Vista, Server 2003 and 2000 systems:**

<CD-ROM drive>:\Software\USB_DRIVER\Win2k_XP\FTDI_CDM2\

For older **Windows 98SE/Me systems**:

<CD-ROM drive>:\Software\USB_DRIVER\Win98SE_ME\FTDI_D2XX\

After the installation of the driver a re-start of the computer may be necessary with older versions like Win98SE! PLEASE NOTE: Under Win98/Me only one of both drivers is working: Either Virtual Comport or the D2XX driver from FTDI! Unfortunately there is no driver that offers both functions. Normally there is no virtual comport available as the RP6Loader under Windows uses as a standard the D2XX drivers (you can change this - please contact our support team!).

Check the Connection of the Device

To check if the device has been correctly installed you can use the device manager as an alternative to the RobotLoader under Windows XP, 2003 and 2000: Right click on My Computer --> Properties --> Hardware --> Device manager

OR alternatively: Start --> Settings --> Control panel --> Performance and Maintenance --> System --> Hardware --> Device manager and check there in the tree view under "Connections (COM and LPT)" if you find a "USB-Serial Port (COMX)" - the X replacing the port number, or look under "USB serial bus controller" for a "USB Serial Converter" !

If you wish to uninstall the driver some day

If ever you wish to uninstall the driver (no, not now - this is just a hint if you need this some day): If you have used the CD ROM installation program, you can uninstall it directly via Start --> Settings --> Control panel --> Software. In the displayed list you will find an item "FTDI USB Serial Converter Drivers" – select it and click on "uninstall".

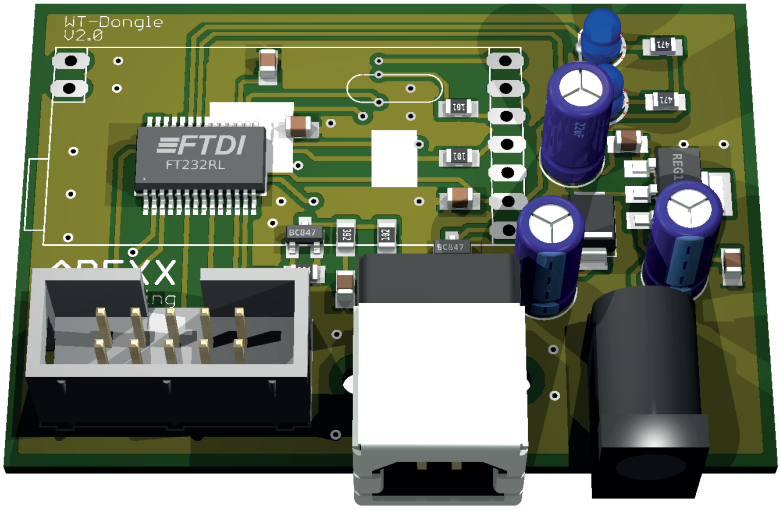
If you have installed the driver manually, you can execute the program ""FTUNIN.exe" in the directory dedicated to the USB driver for your system! Warning: USB-->RS232 adaptors with FTDI chip set often also use this driver!

ODER alternativ: Start --> Einstellungen --> Systemsteuerung --> Leistung und Wartung --> System --> Hardware --> Gerätemanager und dort in der Baumansicht unter “Anschlüsse (COM und LPT)” nachsehen ob ein “USB-Serial Port (COMX)” zu sehen ist - wobei das X für die Portnummer steht oder unter „USB-Controller“ nach einem „USB Serial Converter“ suchen!

Treiber später wieder Deinstallieren

Sollten Sie den Treiber jemals wieder deinstallieren wollen (Nein, das tun Sie jetzt bitte nicht - ist nur ein Hinweis falls Sie das jemals brauchen sollten): Wenn Sie das CDM Installationsprogramm verwendet haben, können Sie das direkt über Start-->Einstellungen--> Systemsteuerung-->Software tun. In der dortigen Liste finden Sie einen Eintrag des „FTDI USB Serial Converter Drivers“ – diesen auswählen und dort dann auf deinstallieren klicken!

Wenn Sie den Treiber von Hand installiert haben, können Sie das Programm “FTUNIN.exe” im Verzeichnis des jeweiligen USB Treibers für Ihr System ausführen! Achtung: USB-->RS232 Adapter mit FTDI Chipsatz verwenden meist ebenfalls diesen Treiber!



7.3. Connection of the USB Interface – Linux

Windows users can skip this section!
Linux systems with kernel 2.4.20 or higher already include the required driver (at least for the compatible previous model FT232BM of the chip on our USB interface, the FT232R). The hardware is automatically recognized and you have nothing else to do. In case of a problem, you can get Linux drivers (and support and maybe also newer drivers) directly from FTDI:

http://www.ftdichip.com/

Once the hardware has been connected, you can check under Linux via:

cat /proc/tty/driver/usbserial

if the USB serial port has been correctly installed. This is normally all you have to do.

It is worth to mention that the Robot Loader uses under Windows D2XX drivers and the full USB designations appear in the port list (e.g. "USB0 | Robot USB Interface | serialNumber"). Whereas under Linux the virtual comport designations appear such as /dev/ttyUSB0, /dev/ttyUSB1 etc.. The normal com ports are equally displayed as "dev/ttyS0" etc.. In this case you have to try which port is the correct one!

Unfortunately Linux doesn't have such a convenient driver that does both. Therefore it made more sense to use the Virtual Comport drivers that are included in the kernel anyway. The installation of a D2XX driver would require quite a lot of manual work....

Finalization of Software Installation

Now the installation of the software and the USB interfaces is completed! You just need to copy the most important files from the CD on a hard disk (especially the complete "Documentation" folder and, if it hasn't been done yet, the example programs). This avoids to look constantly for the CD if you need these files. The folders on the CD are all named in such a way that they can be easily allocated to the relevant software packages or documentation!

If you "loose" the CD one day, you can download the most important files as this manual, the RobotLoader and the example programs from the AREXX home page. You will find there also the links to the other software packages that you require.

7.4. Testing the USB Interface and starting the RobotLoader

The next step is a test of the program upload via the USB interface. Connect the USB interface to the PC (always connect the PC first!) and the other end of the 10-pin ribbon cable to the "PROG/UART" connector on the Robot Arm. (Robot Arm MUST BE SWITCHED OFF!) The 10-pin ribbon cable is mechanically protected against polarity inversion. As long as it is not forced, it can't be connected the wrong way round.

Then start the RobotLoader.
Depending on which language you have selected, the menus might have a bit different names. The screen shots show the English version. Via the menu item "Options->Preferences" you can select under "Language /Sprache" the required language (English or German) and then click on OK.

Once you have selected your language, you have to re-start the Robot Loader to validate the changes!

Open a port - Windows

Select the USB port. As long as no other USB->Serial Adaptor with FTDI controller is connected to the PC, you will see only one single entry that you have to select.

If more ports exist, you can identify the port via the name "Robot USB Interface" (or „FT232R USB UART"). Behind the port name the programmed serial number is displayed.

If no ports are displayed, you can refresh the port list via the menu item "RobotLoader-->Refresh Portlist" !

WARNING!

If the voltage drops below < 6.7V, a warning is displayed.

7.5. Open a port – Linux

Linux handles the USB serial adaptor like a normal comport. The installation of the D2XX driver from FTDI would not be as simple as that under Linux and the normal virtual comport (VCP) drivers are included anyway in the current Linux kernels. It works almost the same as under Windows. You just need to find out the name of the Robot Arm USB interface and make sure that the USB port is not unplugged from the PC as long as the connection is open (otherwise you might have to re-start the RobotLoader to re-connect). Under Linux the names of the virtual comports are "/dev/ttyUSBx", x being a number e.g. "/dev/ttyUSB0" or "/dev/ttyUSB1". The names of the normal comports under Linux are "/dev/ttyS0", „/dev/tty- S1" etc.. They also show up in the port list as far as they exist.

The RobotLoader remembers - if there are several ports - which port you have used last time and selects this port automatically when you start the program (in general, most of the settings and selections are maintained).

Now you can click on the button "Connect"! The RobotLoader will open the port and test if the communication with the bootloader on the robot is working. The black field "Status" on the bottom should show the message
"Connected to: Robot Arm ..." or similiar together with an information about the currently measured voltage. If not, just try again! If it still doesn't work, there is a mistake! Switch the robot off immediately and start searching for the error.

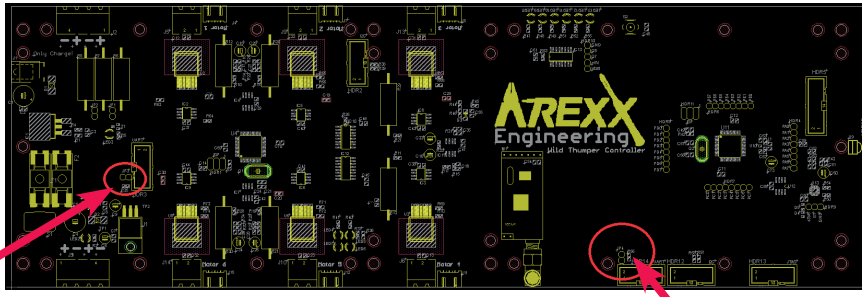
If the voltage is too low, a warning is displayed. You should immediately charge the accumulators (preferably even earlier when the voltage drops below 6,70V)!

WARNING!

If the voltage drops below < 6.7 V, a warning is displayed.

8. SELFTEST

When you connect the power and switch on the Wild Thumper, LED (1) will light green.



When there is no program loaded LED 13 will blink.
With a loaded program LED 13 will be on.

Programming the main processor, use UART HDR 14 and Jumper JP4 is connected! Programming the motor processor, use UART HDR 3 and Jumper JP3 is connected!

The status-LED 8, 9 and 10 blink when you upload a HEX file into the main processor. The Status-LED 6 and 7 blink when you upload a HEX file in the motor processor.

When you start to run a program, all Status LEDS will blink like a running light.

When this all works you can run the selftest program which we supply on the CD. Now you can check all functionalities of the robot. Click in the Robot Loader screen on the button „Add“ and select the file Wild Thumper Examples: **„WTExample_Selftest\WT_Main.hex“ UND „WT_Motor.hex“** in the example files.

Before you run the selftest first read all chapters about the software and robot loader, see chapter 9. The software manual is in chapter 10.

Also install the APC-220 when you want to control the Wild Thumper or do a selftest it works ONLY wireless!

IMPORTANT!
You need to program TWO processors in the Wild Thumper controller!

9.0. Programming the Robot Arm

Now we are gradually coming to the programming of the robot.

Setting up the source text editor
First of all, we need to set up a little development environment. The so-called “source text” (also called “sourcecode”) for our C program must be fed into our computer one way or the other!

To this end, we will definitely not use programs like OpenOffice or Word! As this might not be obvious for everybody, we stress it here explicitly. They are ideally suited to write manuals like this one, but they are totally inappropriate for programming purposes. Source text is pure text without any formatting. The compiler is not interested in font size and colour...

For a human being, it is of course much clearer if some keywords or kinds of text are automatically highlighted by colours. These functions and some more are contained in Programmers Notepad 2 (abbeviated hereafter by “PN2”) that is the source text editor that we will use (ATTENTION: Under Linux you need to use another editor that offers about the same functions as PN2. Usually, several editors are pre-installed! (e.g. kate, gedit, exmacs or similiar)). In addition to the highlighting of keywords and others (called “syntax highlighting”) it offers also a rudimentary project management. This allows to organise several source text files in projects and to display in a list all files related to a project. Moreover you can easily retrieve programs like the AVR-GCC in PN2 and get the programs conveniently compiled via a menu item. Normally the AVR-GCC is a pure command line program without graphic interface...

You will find more recent versions of Programmers Notepad on the project homepage: ***http://www.pnotepad.org/***

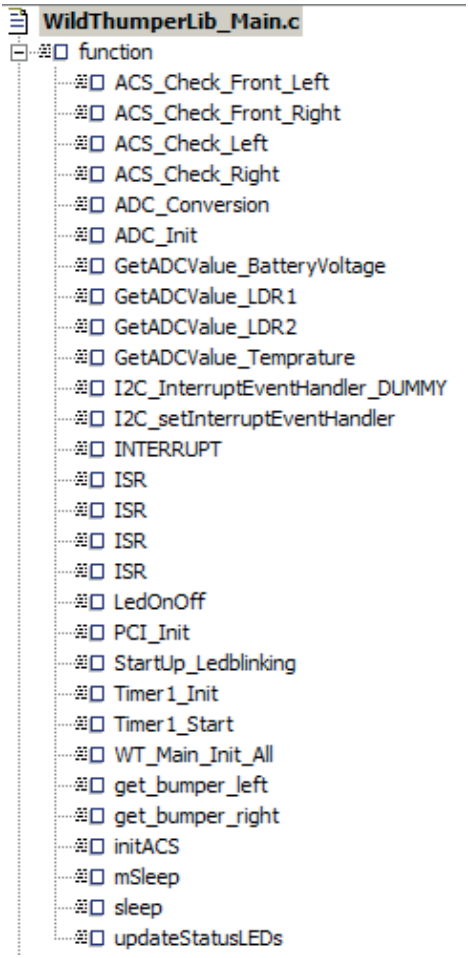
The newest versions of WINAVR don’t require the setting up of menu items anymore!

PLEASE NOTE:
In this section we no longer describe how you have to set up menu items in PN2 as the newest WINAVR versions have done this already for you!

See on page 49 “Open and compile an example project” how you can open an example project!

If you have opened an example project, it should look a bit like this on the PN2 screen:

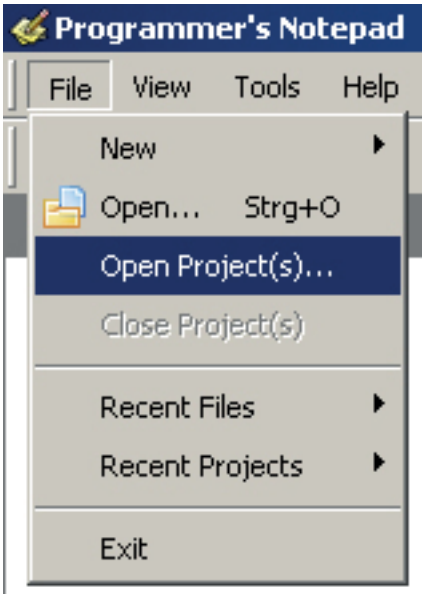
Datei „Wild Thumper Examples.ppg“.
This is a project group for PN2 that uploads all example programs plus the Robot Arm Library into the project list (“Projects“) .



On the left hand side are shown all example projects, on the right hand side the source text editor (with the mentioned syntax highlighting) and at the bottom the tools output (in this case the output of the compiler).

You can convert many other things in PN2 and it offers many useful features.

Open and compile an example project



Let's test now if everything runs properly and open the example projects:

Select in the "File" menu the item "Open Project(s)".

A normal file section dialogue appears. Search the folder "Robot Arm_Examples [MINI]\" in the folder into which you have saved the example programs.

Open the "Robot ArmExamples.ppg" file. This is a project group for PN2 that uploads all example programs as well as the Robot Arm Library into the project list ("Projects").

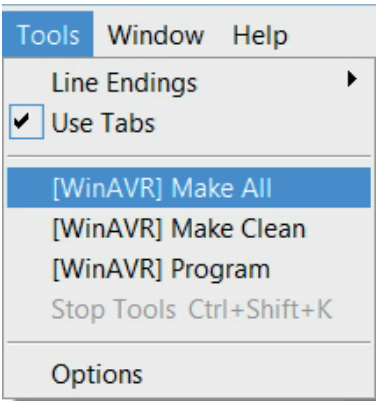
Now all example projects are conveniently at hand if you want to refer to them at the beginning or look for functions in the Robot Arm Library etc..

Open the first example program on top of the list ("01_Leds" and select file "01_Leds") that appears on the left edge of the program window! Just double-click on "01_Leds.c"! A source text editor is displayed in a window inside the program.

An output area should appear on the bottom of the program window of PN2. If not, you have to enable this area via the "View" menu --> "Enable output" OR if the area is too small, increase the size by pulling the edges with the mouse (the mouse cursor changes into a double arrow at the upper edge of the grey area marked "output" at the bottom of the program window...).

You can take a quick look at the program that you just opened with the source text editor but you don't need to understand right now what is happening exactly. However as a first info: The green text are comments that are not part of the actual program. They are only used for description/documentation purposes.

We will explan this in detail a bit further down (there is also a version of this program WITHOUT comments so that you can see how short the program is in fact. The comments inflate it a lot but are necessary for the understanding. The uncommented version is also useful to copy the code in your own programs!).



First of all we just want to test if the compilation of programs works properly.

In the Tools menu on top both freshly installed menu items (see fig.) should appear (or the [WinAVR] inputs existing as a standard in PN; whatever, it works normally with both).

Please click now on “MAKE ALL“!

PN2 retrieves now the above mentioned “make_all.bat“ batch file. This will on its turn retrieve the program “make“. More info about “make“ will follow later.

The example program will now be compiled. The generated hex file contains the program in the translated format for the microcontroller and can be uploaded and executed later. The compilation process generates a lot of temporary files (suffixes like “.o, .lss, .map, .sym, .elf, .dep“). Just ignore them. The newly set up tool “make clean” will erase them all. Only the hex file is of interest for us. By the way, the function “make clean” will not erase this file.

After the activation of the menu item MAKE ALL, following output should display (below in a considerably shortened version! Some lines may look of course a bit different):

```
> „make.exe“ all

avr-gcc  -mmcu=atmega644pa -Wall -gdwarf-2 -Os -std=gnu99 -funsigned-
char -funsigned-bitfields -fpack-struct -fshort-enums -MD -MP -MT WildT-
humperLib_Main.o -MF dep/WildThumperLib_Main.o.d  -c  ../WildThumper-
Lib_Main.c

avr-gcc  -mmcu=atmega644pa -Wall -gdwarf-2 -Os -std=gnu99 -funsigned-
char -funsigned-bitfields -fpack-struct -fshort-enums -MD -MP -MT WT_Led.o
-MF dep/WT_Led.o.d  -c  ../WT_Led.c

avr-gcc -mmcu=atmega644pa -Wl,-Map=WT_LED.map WildThumperLib_
Main.o WT_Led.o     -o WT_LED.elf

avr-objcopy -O ihex -R .eeprom -R .fuse -R .lock -R .signature  WT_LED.elf
WT_LED.hex

avr-objcopy -j .eeprom --set-section-flags=.eeprom=“alloc,load“ --change-
section-lma .eeprom=0 --no-change-warnings -O ihex WT_LED.elf WT_LED.
eep || exit 0

avr-objdump -h -S WT_LED.elf > WT_LED.lss

AVR Memory Usage
----------------
Device: atmega644pa

Program:    5514 bytes (8.4% Full)
(.text + .data + .bootloader)

Data:        296 bytes (7.2% Full)
(.data + .bss + .noinit)



> Process Exit Code: 0
> Time Taken: 00:02
```

The “Process Exit Code: 0“ at the end is most important. It means that no error occurred during compilation. If another code appears there, the sourcecode contains an error that must be corrected before it will work. In this case, the compiler will output various error messages that give some more information.

Please note however that the “Process Exit Code: 0“ is not a guarantee of a fully error-free program! The compiler will not find flawed thinking in your program and it can’t prevent the robot from running into a wall ;-)

IMPORTANT: You might find warnings and other messages further above. These are often very helpful and always indicate important problems! That’s why these always need to be solved. PN2 highlights warnings and errors by colours to make the identification easier. Even the line number is indicated that the compiler is criticizing. If you click on the coloured error message, PN2 skips in the relevant editor directly to the faulty line.

The indication at the end “AVR Memory Usage“ is also very useful.

```
Size after:
AVR Memory Usage
----------------
AVR Memory Usage
----------------
Device: atmega644pa

Program:    5514 bytes (8.4% Full)
(.text + .data + .bootloader)

Data:        296 bytes (7.2% Full)
(.data + .bss + .noinit)
```

This means for the Atmega64 processor that our program has a size of 3074 bytes and that 68 bytes of RAM are reserved for static variables (you have to add to this the dynamic ranges for heap and stack but this would go too far... just keep always at least a few hundred bytes of memory free). We dispose in total of 64kb (65536 bytes) of Flash ROM and 2kb (2028 bytes) of RAM. On the 64kb, 2k are occupied by the bootloader - so we can only use 62kb. Make always sure that the program fits into the available memory space!
(The RobotLoader doesn’t transfer the program if it is too big!)

This means that the example programs above leave 60414 bytes of free space. The relatively short example program Example_01_Leds.c is only so big because the Robot ArmBaseLibrary is included! So, don't worry, there is enough space for your programs and so small programs usually don't need so much memory space. The function library on its own needs several kb of Flash memory but makes your job much easier and therefore your own programs will generally be quite small compared to the Robot ArmBaseLibrary.

THE WILD THUMPER USES TWO PROCESSORS!
DO NOT FORGET TO PROGRAM BOTH PROCESSORS!!

The just compiled program can now be uploaded via the RobotLoader into the robot. To do that, you have to add the newly generated hex file into the list in the RobotLoader via the button "Add", select it and click on the "Upload" button exactly as you did for the selftest program. After that you can switch back to the terminal and look at the output of the program. Of course you need to launch the execution of the program. The easiest way to do it in the terminal is to press the key combination [STRG]+[S] on the keyboard or to use the menu (or just to send an "s" - after a reset you have to wait a little bit though until the message "[READY]" is displayed in the terminal!). The key combination [STRG]+ [Y] is also very convenient as the currently selected program is uploaded into the Robot Arm and immediately started. This avoids to click on the "Flash Loader" tab in the terminal or to use the menu.

The example program is very simple and is only composed of a small LED running light and some text output.
Das eben kompilierte Programm kann nun mit dem RobotLoader in den Roboter geladen werden.

For wireless use, please install both APC-220!

-

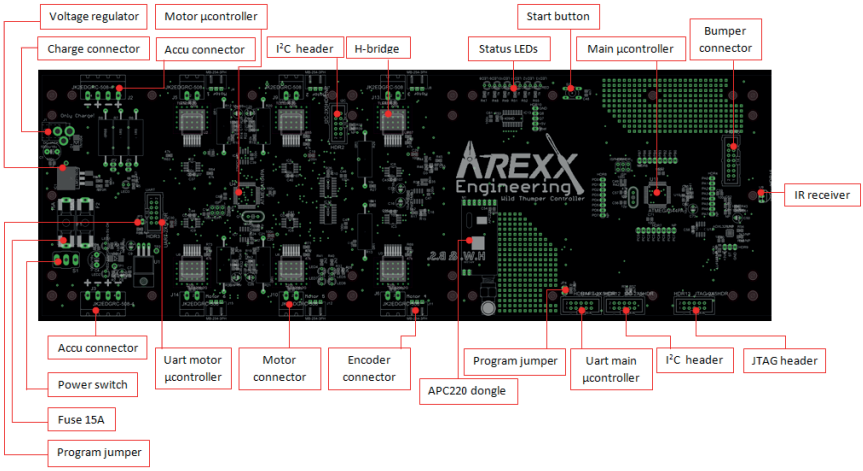
10. Activate the WT‘s wireless control

The CD has been provided with various pre-compiled programs (HEX Files) which directly may be loaded tot the WT‘s processor by RobotLoader.

ATTENTION !
THE WILD THUMPER USES TWO PROCESSORS!
Do not forget the JP3 and JP4 program jumpers
they are absolutely necessary during programming!

As an example you may transfer the HEX-file for wireless control to the main processor (Wild_Thumper_Main.HEX) and motorprocessor (Wild_Thumper_Motor.hex) DO NOT FORGET JP3 and JP4!.
Start the Wild Thumper Software aplication and see chapter 10.

To upload the previously generated HEX-file you must enter the file-name to the RobotLoader‘s list by clicking “Add”, select the file and the click the „Upload!“- key. The procedure is similar to the one you applied at the self-test program.



After completion you may switch to the terminal window and inspect the program‘s output. Of course the program execution has to be restarted. The easiest method to restart is pressing [CTRL]+[S] at the keyboard or to use the menu (or simply send an “s” - after a reset however you will have to wait some time for the system to display „[READY]“ at the terminal!).

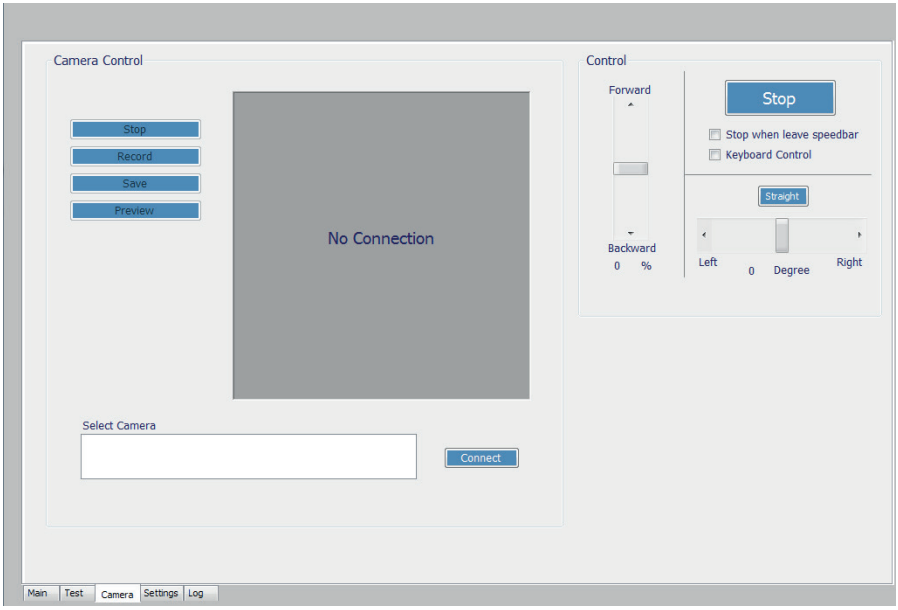
The [CTRL]+[Y] is a helpful key-combination as well and allows you to load and subsequently start a selected file into the Wild Thumper! The method avoids a few strokes like switching from terminal to the „Flash Loader“-tab or activating the menu.

10.1. Installing the wireless control

Chapter 10 contains the complete documentation for the wireless control software. In order to use this package you must install the software to your PC. Please copy the WT-software to your hard disk or install the software directly from the CD.

Start the application

If you plan to use a wireless camera you may add the application by opening the camera-tab and selecting your PC‘s camera.

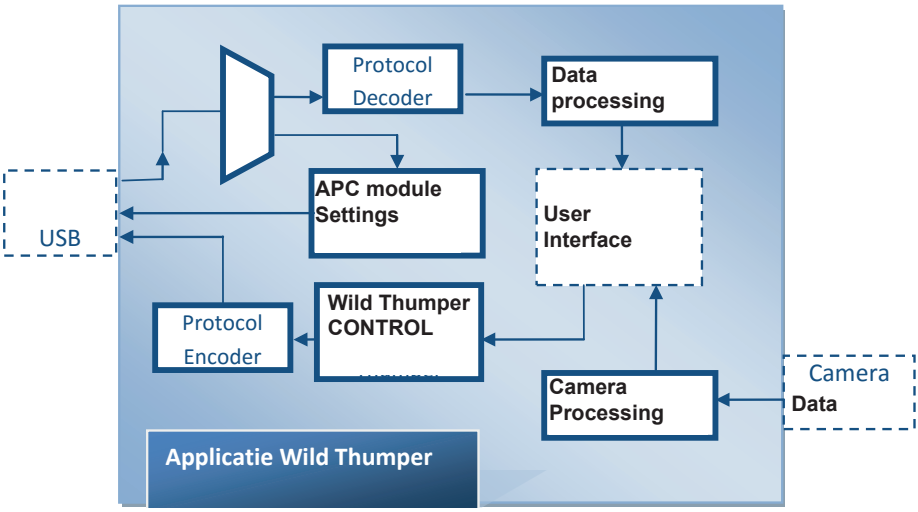


Controlling and selftest WORKS ONLY WIRELESS!

11. WT WIRELESS CONTROL

10.1. Overview of the wireless control system

The application may serve two outward bound connections: the serial USB connection link with an APC-220 transceiver and the communication link to the camera module. The application consists of various functions and a user interface, which is to be illustrated in the following block diagram.



In analogy to the various blocks the application‘s control window has been equipped with a number of tabs. Each tab will be serving a number of functions, which have been clearly arranged.

Before testing the wireless control you will have to transfer the required HEX.file to the main processor by the Robot-Loader (see chapter 9.1 as a guide for this job).

Additionally both APC-220 modules have to be inserted.
1 item at the Wild Thumper‘s main board
1 item at the programming adapter.

You also will have to connect the programming adapter to a 5 Volt source.

The following tabs have been prepared:

The ‚Main' tab:
The 'main' tab serves the following functionality:

- The communication link to the APC Dongle (USB)
- The Wild Thumper's control
- The presentation of measurement data (amperage, motor speed, etc..)

The ‚Test' tab:
This tab has been designed to test all functionality of the Wild Thumper.
The following tests have been implemented:

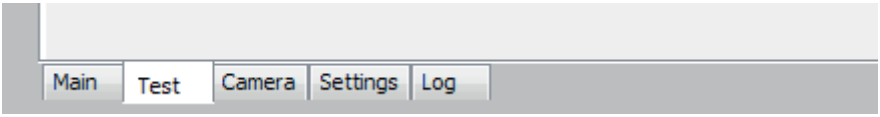
- Each motor may be controlled individually. The amperage and motor speed of this motor is displayed.
- Individual test-function for each LED.
- Test-routine for the I²C-connection between main-controller and motor-controller.
- Test-routine for the RF-link.

The ‚Camera' tab:
If a wireless camera has been connected to the PC and installed at the WT you may display these views from the WT-mounted camera at the ‚camera'-tab. The tab-application allows you to save the camera's images at a PC-located storage. The system also has been designed to control the Wild Thumper from this tab-application while you are observing the camera's view.

The ‚Settings' tab:
The ‚Settings' tab allows you to alter all settings, which may be listed as:

- Parameters for the APC-220 modules
- Parameter for the application

The ‚Log' tab:
In order to trace important events we defined the tab-functionality „LOG", in which the log-file may be inspected.

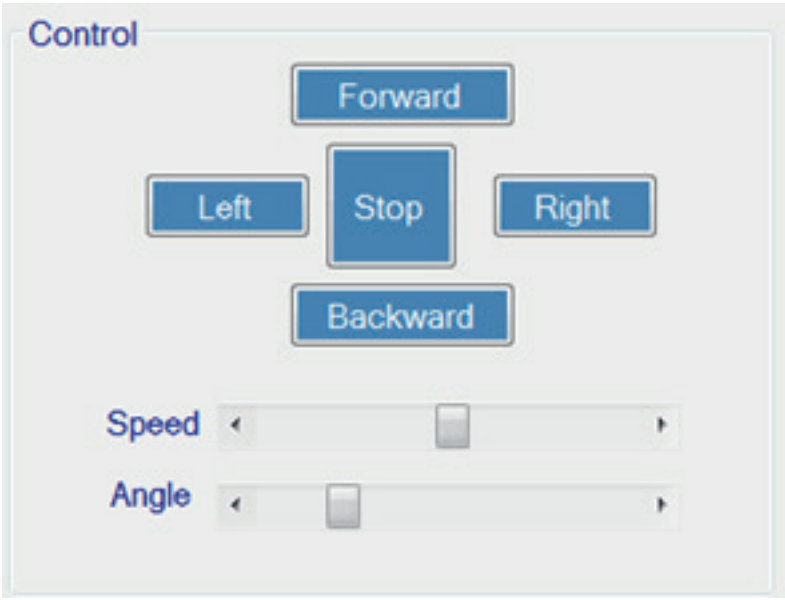


10.2. USB in Visual Basic

In order to communicate with the application the system‘s APC220 must be connected to the PC. In order to provide a communication link we modify the serial UART-port of the APC220‘s FTDI-Chip in a USB-port. The chip‘s design is quite advantageous because the newly created „USB-device“ is treated as a device which is connected to a virtual COM-port. As soon as we have activated the serial port we may quite comfortably configure the port-settings.

10.3. The Wild Thumper‘s control

As has been described in the previous chapter the Wild Thumper may be controlled from the tab-applications ‘Main’and ‘Camera’. The control system has been designed with the following control desk:



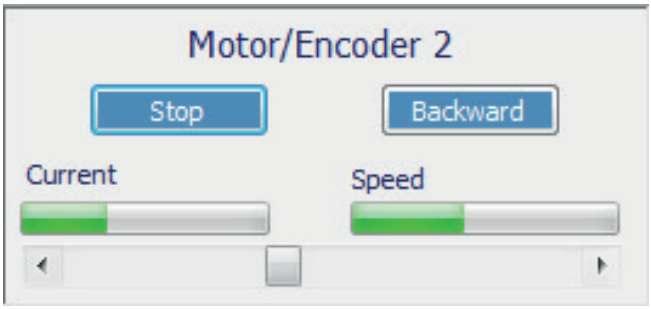
In the previous image you may identify five buttons and two slide controls. The buttons will be used to control the robot‘s directions. The slide controls are used to adjust the speed and the angle control for the Wild Thumper‘s movements.

10.5. Testing

In order to test the Wild Thumper we designed a test desk, which is accessible at the ‚test'-tab and provides the functionality you need to test all functions and parts of the robot's system. The following parts of the Wild Thumper may be tested.

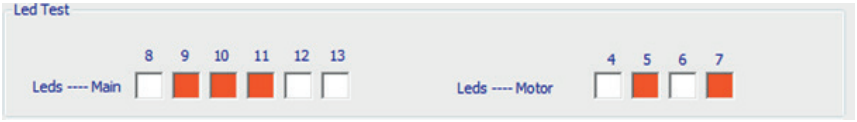
Motortest:

Each motor may be tested individually. The test result displays the amperage and speed for the motor in test. You may switch the motor in forward or reversed direction and control the speed.



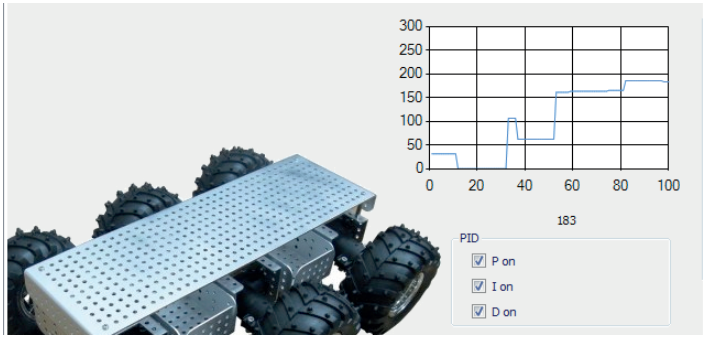
LED-Test:

All LEDs may be tested. Switching LEDs on or off will be followed by a transfer of the command by WT-protocoll to the Wild Thumper, in which the addressed LED is to be activated respectively deactivated.



PID controller:

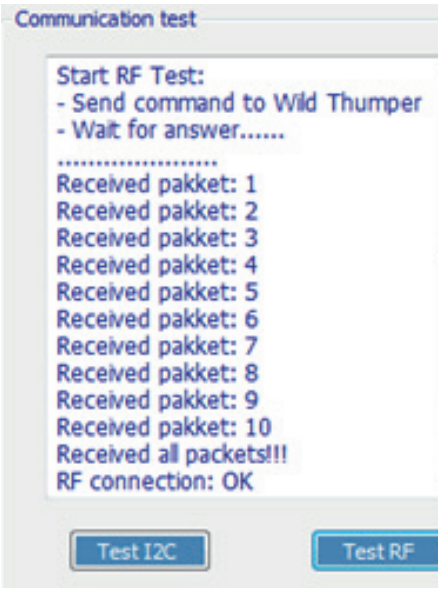
You can define the different functions of the PID controller yourself.
This is only for test and example purpose.

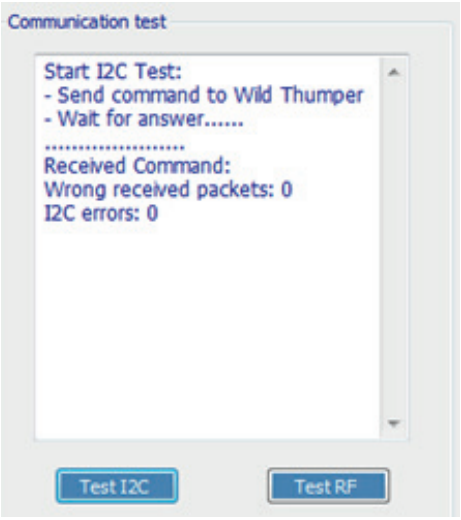


Sensortest:
The status of various sensors will be displayed in the tab-section “Test”, from where you as a user are enabled to comfortably check all sensors in an overview.

Communicationstest:
Two communication link tests have been designed: an RF-test and a I²C- test. The RF-test is used to test the wireless communication link and transmits a data set to the Wild Thumper in order to generate an RF-test. The data set orders the robot to transmit ten data packages separated by 100ms interval periods. As soon as all bytes have been received in good order the test will be registered as OK.

In the I²C-test the system starts by transferring a wireless command to perform an I²C-test (see the protocol‘s documentation). This start procedure is to be followed by a byte transfer of a number of bytes for motor control and for reading a few bytes. The test is completed by a I²C-test-report, which will be retransmitted to the application for display.





10.6. Setups for APPCON APC220

A dedicated application window allows you to setup the APC220-module and/ or to read the module‘s parameters. For this feature the system uses the following interface:

As a conclusion

We hope that our robots have guided you on your way into the world of robots. We share the conviction of our Japanese friends that robots will become the next technological revolution after computers and mobile phones. This revolution will trigger new economical impulses.

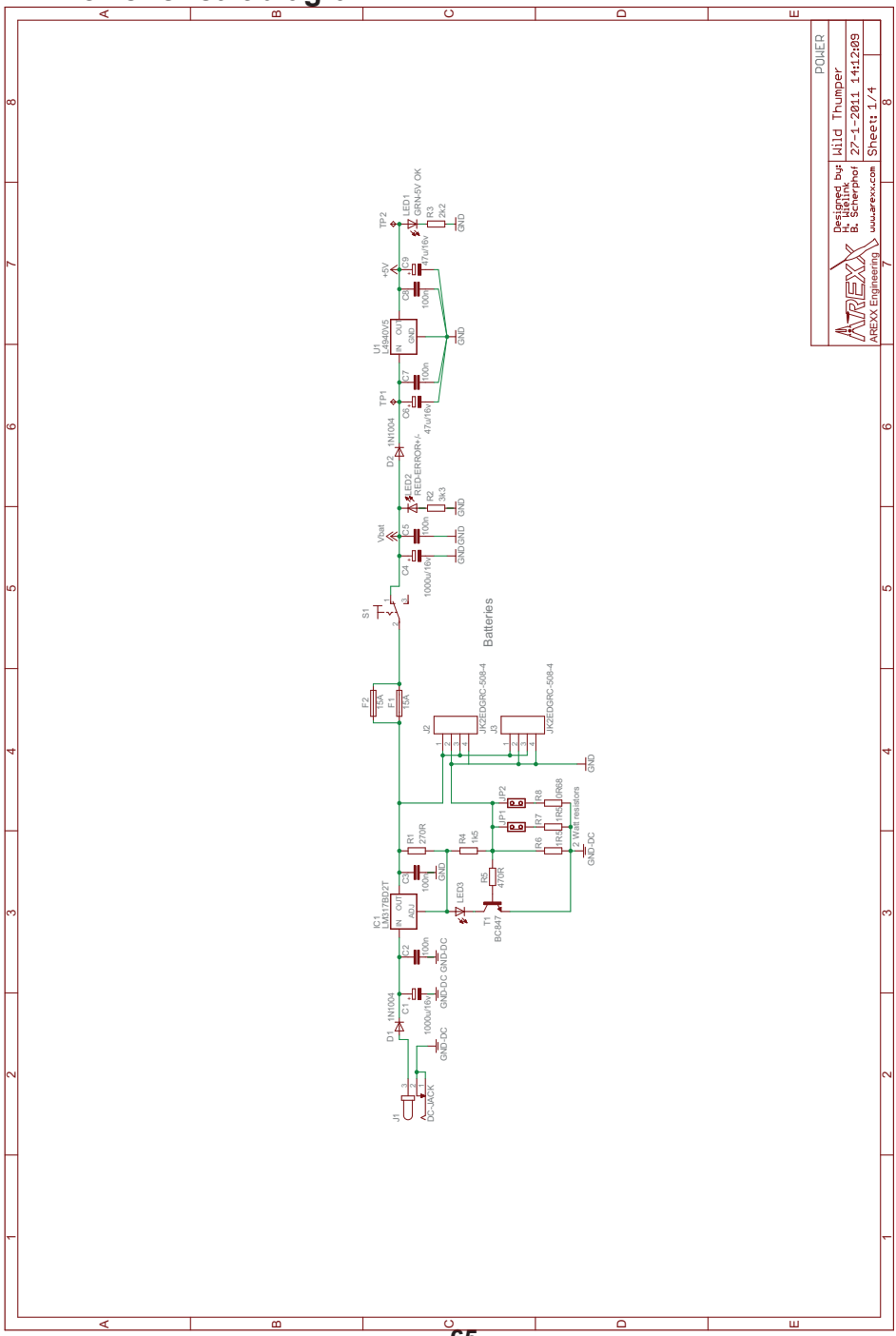
Unfortunately Japan, other Far East countries and also the USA have largely overtaken Europe in this field. Unlike Europe, technical courses start in Far East already in the primary school and are an important part of the education.

Our target in the development of our robots ASURO, YETI, ARDUINO, CATERPILLAR, ROBOT ARM and WILD THUMPER is therefore:

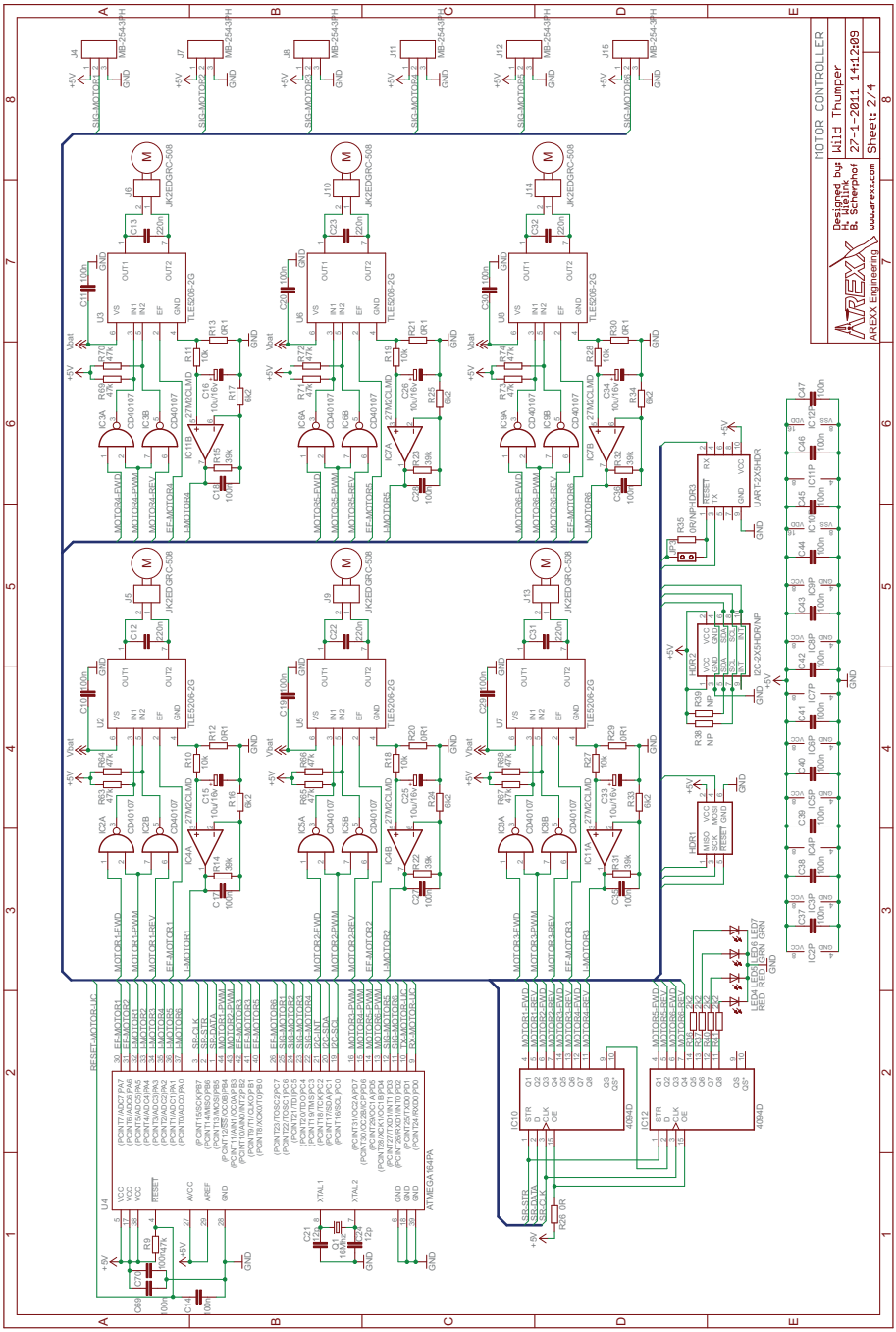
TO TRAIN A SCIENTIFIC MIND

APPENDIX

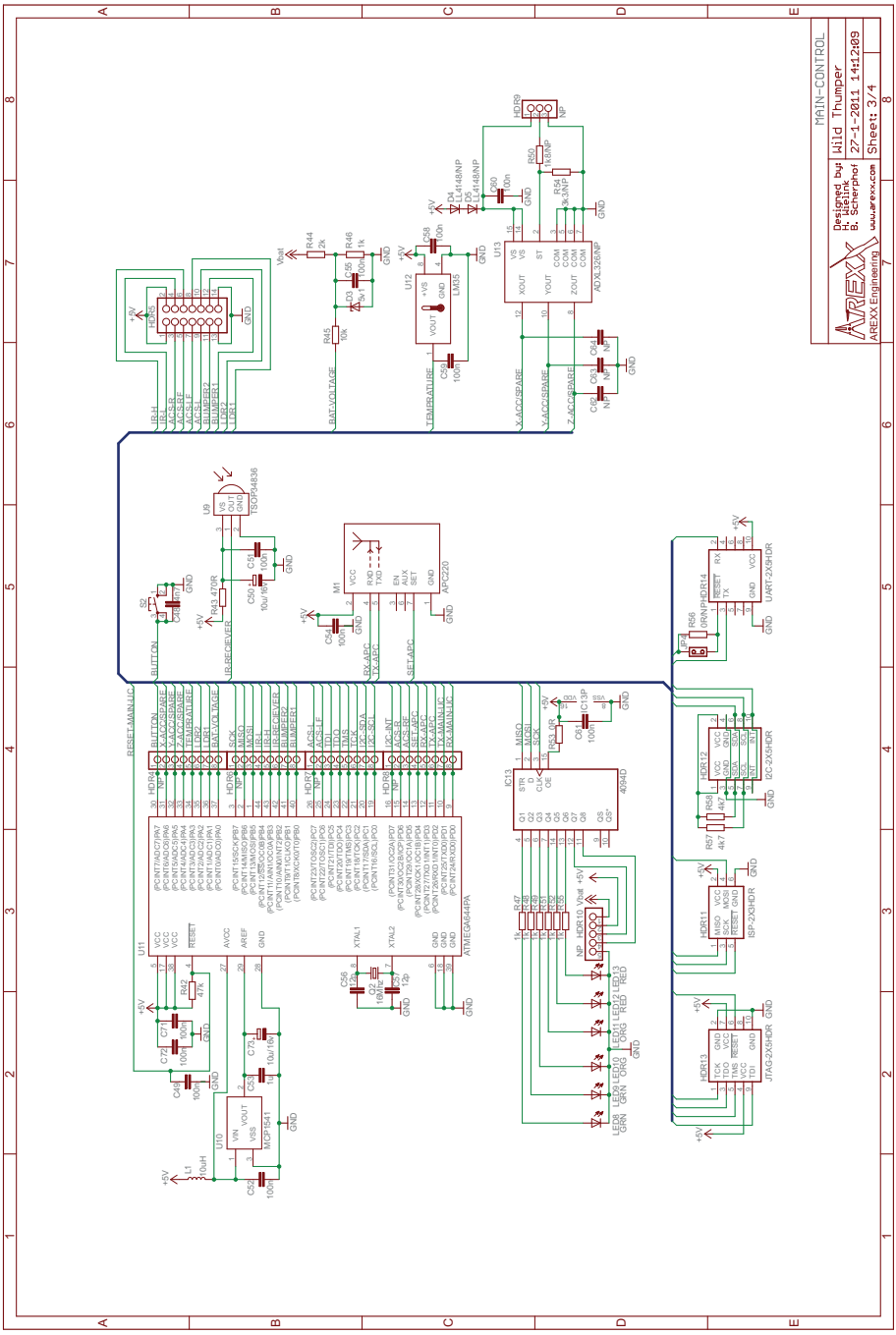
A. Power circuit diagram



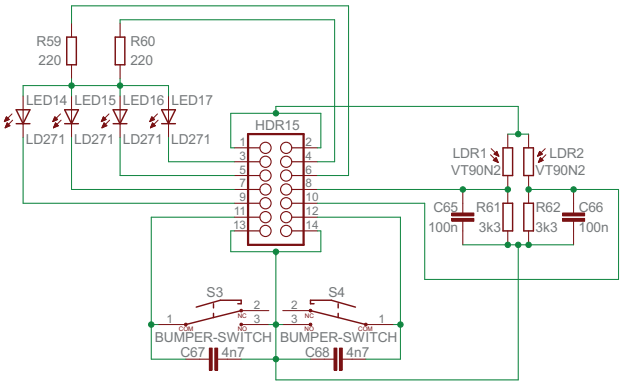
B. Motor control circuit



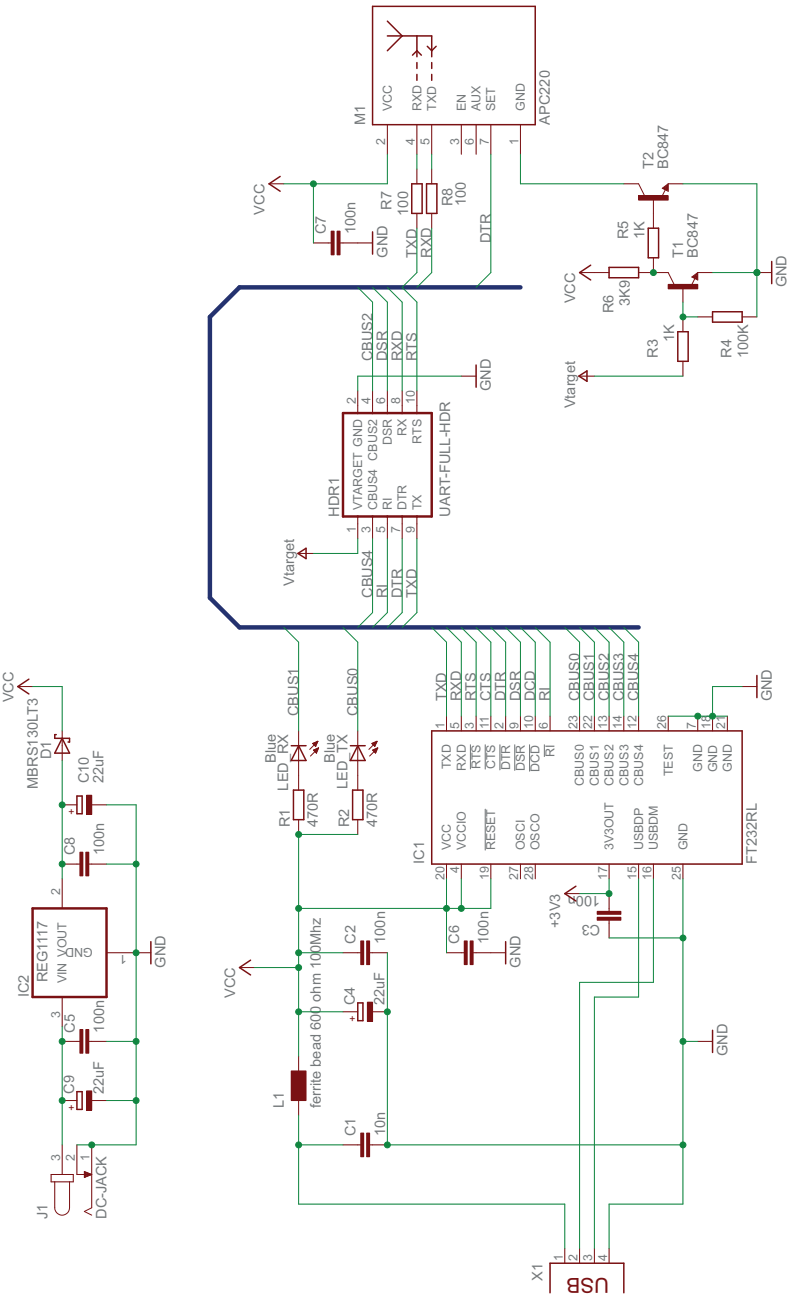
C. Main processor circuit

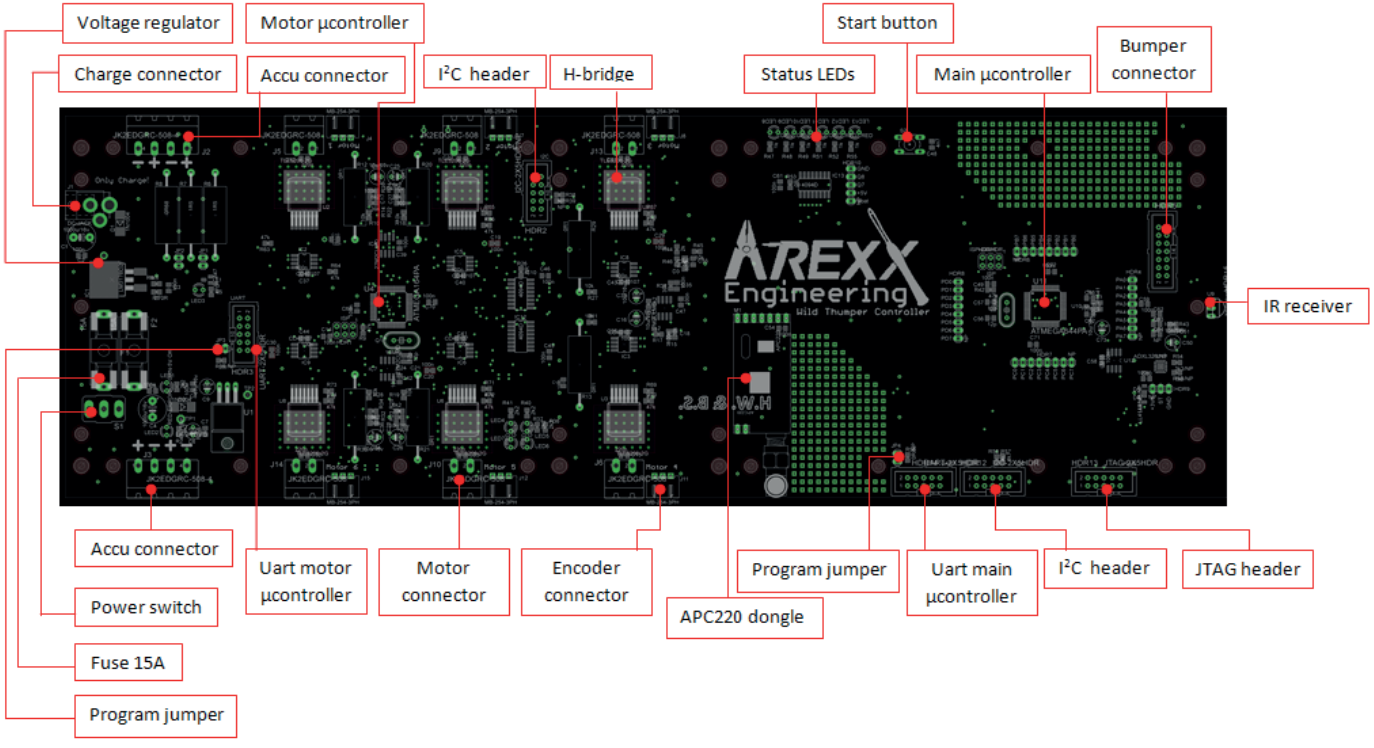


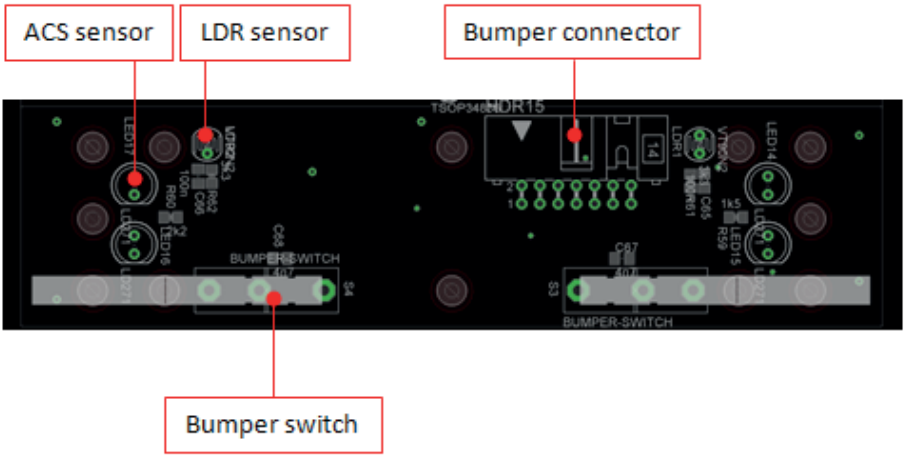
D. Bumper circuit (Front PCB)

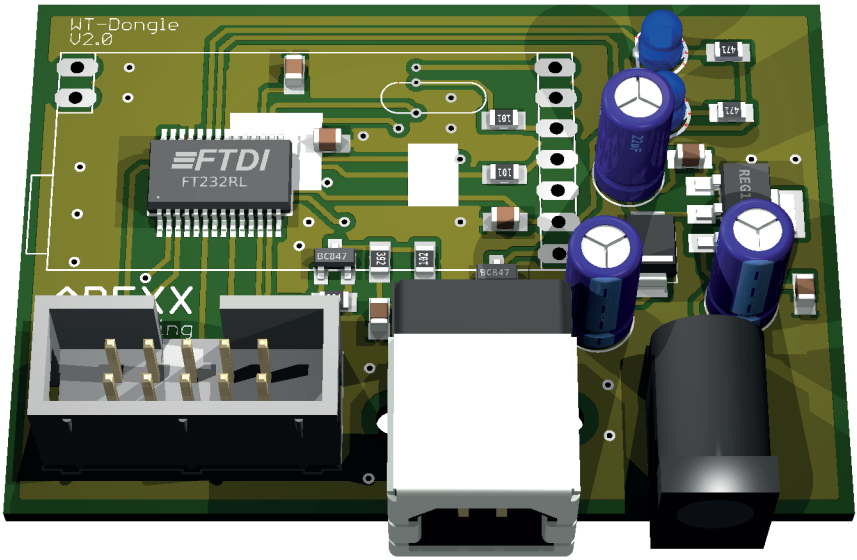


E. USB programmer circuit




Voltage regulator
Motor µcontroller
Charge connector
Accu connector
I²C header
H-bridge
Start button
Status LEDs
Main µcontroller
Bumper connector
IR receiver
Accu connector
Uart motor µcontroller
Motor connector
Encoder connector
Program jumper
Uart main µcontroller
I²C header
JTAG header
Power switch
Fuse 15A
Program jumper
APC220 dongle
AREXX
Engineering
Wild Thumper Controller


ACS sensor
LDR sensor
Bumper connector
Bumper switch
BUMPER-SWITCH
BUMPER-SWITCH


WT-Dongle
V2.0
FTDI
FT232RL

Partlist

Part	Value
C1,C4	1000u/16v
C2,C3,C5,C7,C8,C10,C11,C14,C17,C18,C27,C28,C35,C36,C37,C38,C39,C40,C41,C42,C43,C45,C46,C47,C49,C51,C52,C54,C55,C58,C59,C60,C61,C65,C66,C69,C70,C70,C72,C71	100n
C6,C9	47u/16v
C12,C13,C22,C23,C31,C32	220n
C15,C16,C25,C26,C33,C34,C50	10u/16v
C19,C20,C29,C30	100n
C21,C24,C56,C57	12p
C48,C67,C68	4n7
C53	1u
C74,C75,C76,C77,C78,C79	330u/16v
D1,D2	LL4004
D3	zener 5v1
F1	33A
HDR1	ISP-2X3HDR
HDR3	UART-2X5HDR
HDR5	ML14
HDR11	ISP-2X3HDR
HDR12	I2C-2X5HDR
HDR13	JTAG-2X5HDR
HDR14	UART-2X5HDR
HDR15	ML14/90
IC1	LM317
IC2,IC3,IC5,IC6,IC8,IC9	CD40107
IC4,IC7,IC11	27M2C
IC10,IC12,IC13	4094D
J1	DC-JACK
J2,J3	JIEKE JK2EDGRC-508-4
J4,J7,J8,J11,J12,J15	MB-254-3PH
J5,J6,J9,J10,J13,J14	JIEKE JK2EDGRC-508
JP1,JP2,JP3,JP4	JUMPER-2P
L1	10uH
LDR1,LDR2	VT90N2
LED1,LED6,LED7,LED8,LED9	GREEN
LED2,LED4,LED5,LED12,LED13	RED
LED3,LED10,LED11	ORANGE
LED14,LED15,LED16,LED17	LD271 IR
M1	APC220
Q1,Q2	20Mhz
R1	270R
R2	3k3
R3,R36,R37,R40,R41,R60	2k2
R4,R59	1k5
R5,R43	470R
R6,R7	1R5
R8	0R68
R9,R42,R63,R64,R65,R66,R67,R68,R69,R70,R71,R72,R73,R74	47k
R10,R11,R18,R19,R27,R28,R45,R61,R62	10k
R12,R13,R20,R21,R29,R30	0R1
R14,R15,R22,R23,R31,R32	39k
R16,R17,R24,R25,R33,R34	6k2
R26,R53	0R