

FanBot Technical Design

Peter Brier (*peter@kekkeTek.nl*)
Marieke Peelen (*mpeelen@kekkeTek.nl*)
KekkeTek.nl
2013 – CC-BY-SA-NC-3.0

Revision history

Revision	Date	Author	Notes
0.1	3-2013	M. Peelen	Initial version, info copied from various documents
0.4	4-2013	P. Brier	Added protocol information
0.6	4-2013	M. Peelen	Added use cases
0.7	19-04-2013	P. Brier	Included rev 0.3 fanbot brain schematics and added layout considerations
0.8	21-04-2013	P. Brier	Updated to rev 0.4 fanbot schematics
0.9	30-4-2013	M.Peelen	HUB picture updated, software additions
1.0	30-4-2013	P. Brier	Added info on USB communication
1.1	20-5-2013	P. Brier	Added more info on HUB protocol
1.2	10-6-2013	P. Bruer	Added VID / PID

TODO:

1. VID/PID aanvragen voor fanbot & hub
VID: 0x1FC9 (NXP) , PID: 0x8066 (Fanbot) PID: 0x8067 (Hub)
2. workshop server/client software beschrijven
3. Software updater functie beschrijven

Contents

1	Introduction.....	3
2	Fanbot	3
2.1	Actuator.....	4
2.2	Light.....	4
2.3	Communication.....	4
2.4	Power supply.....	4
2.5	Connectors.....	5
2.6	Microcontroller.....	5
2.7	Fanbot "Brain" PCB.....	5
2.8	Fanbot HUB PCB.....	6
2.9	Power supply.....	6
3	Communication.....	7
3.1	HUB Communication.....	7
3.2	PC to HUB Communication protocol.....	8
3.3	Fanbot positions and addressing.....	9
3.4	Communication between HUB and robot.....	10
3.5	Communication Protocol between HUB and robot.....	10
3.6	Communication between PC and Fanbot.....	11
3.6.1	MSD mode.....	11
3.7	HID mode communication.....	12
3.8	Communication between Control application and proxy.....	13
4	Software.....	14
4.1	Control application.....	14
4.2	Control application use case.....	14
4.3	Fanbot program.....	15
4.4	Workshop programming tool.....	16
4.4.1	Fanbot HID interface program.....	17
5	Stand.....	19
6	Schematics & Bills of materials.....	20
6.1	Fanbot Electrical schematics (obsolete by last production designs).....	20
6.2	Fanbot Bill of materials (obsolete by last production designs).....	21
6.3	Fanbot PCB layout.....	22
6.3.1	Mechanical and layout considerations.....	22
6.3.2	Silkscreen Front.....	22
6.3.3	Silkscreen Back.....	23
6.4	Fanbot HUB Electrical schematics.....	24
6.5	Fanbot HUB Bill of materials.....	24
6.6	Tinkering materials	25

1 Introduction

For WK Robocup 2013 in Eindhoven a workshop is developed for the young visitors (age ± 8 - 15 years). Main purpose is to involve youngsters in the event and robotics technology. The goal is build a stand with robots cheering on the soccer robots. Approx 1300 robots are expected to be build, the stand will be able to accommodate at least 1000 robots. 1500 robots kits are manufactured to allow 200 kits to be used in pilot workshops.

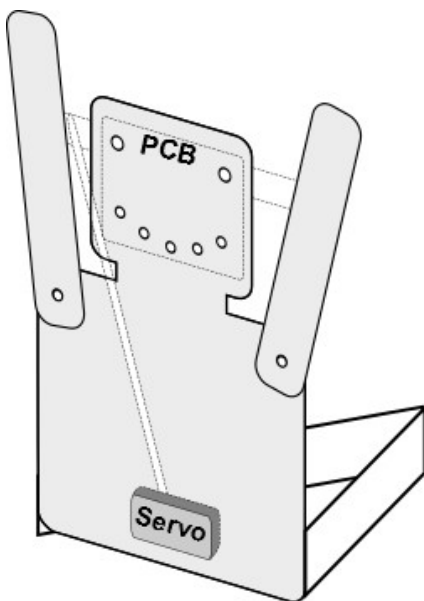
For this event following will be developed and realized:

- Fanbot (cheering robot) kit.
- Tribune with power and control.
- Workshop Fanbot building

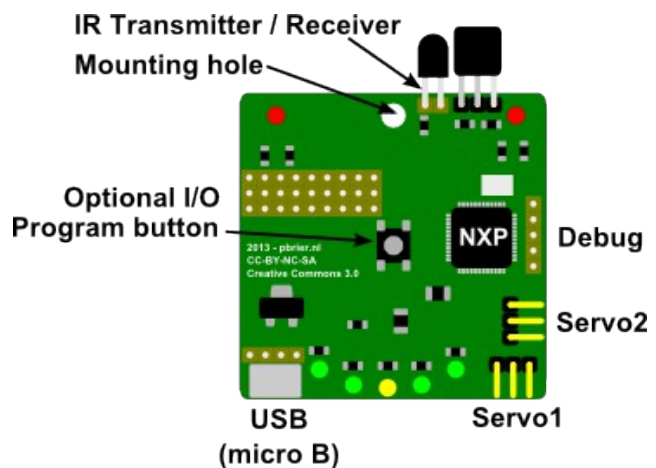
2 Fanbot

The Fanbot is a small cardboard robot that can wave (using a small RC servo) and light up (using 5 to 7 LEDs in its face).

It is controlled by a microcontroller. The microcontroller can be programmed via USB (during the workshop). The robot is able to work autonomously once programmed (power applied via USB bus). A PCB has to be designed to be used in this robot. The PCB can also serve as a generic controller for educational (robotics) projects



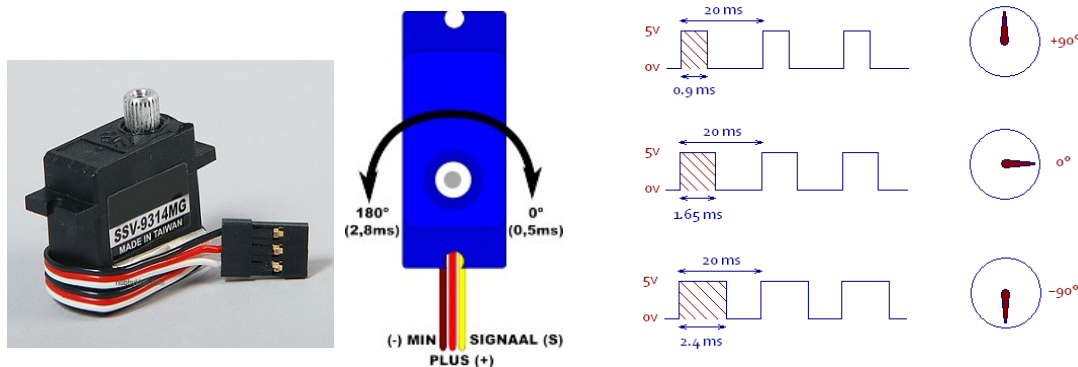
Fanbot



*Concept of the controller PCB
(IR receiver & transmitter are optional)*

2.1 Actuator

To actuate the robotic limbs, a hobby RC servo is used. The servos assume a rotor position that corresponds to the received PWM input signal. The power supply requirements are typically 3..6VDC @ 0.1 to 1A (peak). The connector is usually a 3 pin female header (2.54mm pitch) with 1=GND, 2=V+, 3=Signal. The input impedance of the signal line is typically 1 to 10kOhm.



RC servo

2.2 Light

A number of LEDs are placed on the PCB, in the shape of a smiley face, to be able to make visual patterns (see PCB design). Different color LEDs can be used (e.g red, green, yellow), 5 to 7 LEDs make up the face. The LEDs are individually programmable with the micro-controller.

2.3 Communication

During the event the robot can communicate via USB (to be programmed from a PC)

In addition to the standard USB communication via the USB connector, it should be possible to signal at low speed over the DP and DN lines of the USB connector. This way, the robot can be controlled via the USB cable without having to implement a USB HOST controller. The signalling rate should be at least 100bps in this situation. One of the lines could act as a transmitter and one as a receiver pin, or one of the lines may be used for both TX and RX (half duplex) depending on the difficulty of implementing this. The low speed signalling protocol can be RS232, I2C, a PWM signal, or whatever does the job. There should be a mechanism to determine if a normal USB connection is made, or if the low speed communication needs to be activated.

In addition a single switch is provided. This is used to activate the programming mode of the microcontroller.

2.4 Power supply

The robot is powered via the USB bus (5V, 500mA max). An onboard voltage regulator provides the voltage for the micro-controller and servo. It limits the maximum current drawn by the board and is short-circuit proof. In addition to the USB connector, a (unpopulated) 2 pin connection is provided (3.81mm pitch, through hole screw terminal) where an external battery power supply can be connected. (typically 4 to 6V for 4xAA battery cells).

2.5 Connectors

A USB micro-B type connector is used for the connection to the PC.

A 3 pin angled male header is provided for the servo motor connections.

Pin 1: GND

Pin 2: 5V

Pin 3: PWM Signal

In addition, unused microcontroller pins are connected to a (unpopulated) through hole header connector, for future use (additional IO). They are complemented by two rows of 3V3 and GND pins.

2.6 Microcontroller

The micro-controller needs to have sufficient I/O and memory to control:

- 2 servos (PWM)
- 7 LEDs (or more)
- Serial communication (USB and low speed)

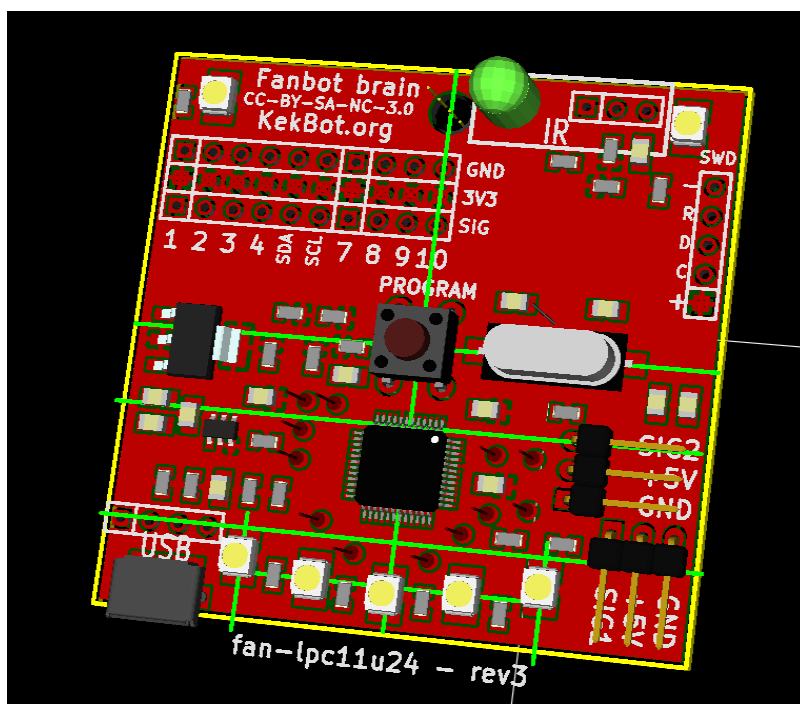
Ideally it has a ROM bootloader that is activated by the Program switch that is capable of receiving a new program via the USB as Mass Storage Device.

The device should be be programmable using a freely available C compiler (e.g. GCC).

2.7 Fanbot "Brain" PCB

The controller is made using a 2 layer PCB, with all components on one side. The PCB size is approx 50x50mm (size of the robot head). It is intended to be used in vertical orientation, with USB connector on the bottom edge and servo connectors on two sides (using 90degrees angled connectors). The program switch is located at the enter of the board.

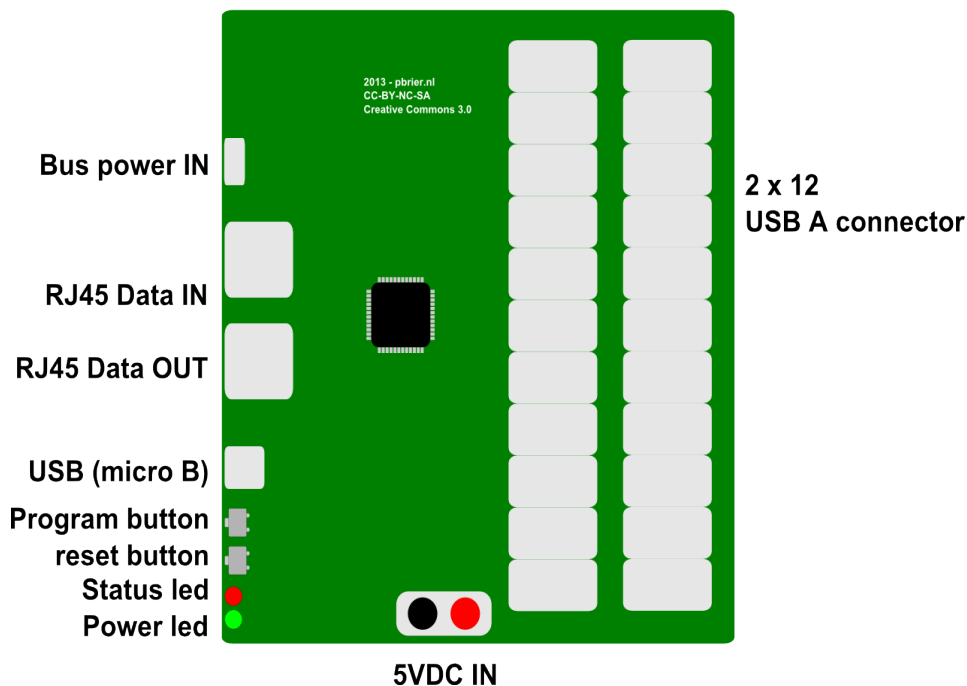
One mounting hole (3.5 mm) is present to be able to mount the board. At the top-center of the board.



2.8 Fanbot HUB PCB

The task of connecting and powering the Fanbots is handled by the Fanbot hubs. They provide power to the devices and are capable to address all devices individually. This document describes the technical requirements and design of these hubs. The hub is called the master and the Fanbots are slaves. One of the hubs is connected to a PC. This hub is the main controller.

FanBot Hub PCB concept



Fanbot 44xHUB

2.9 Power supply

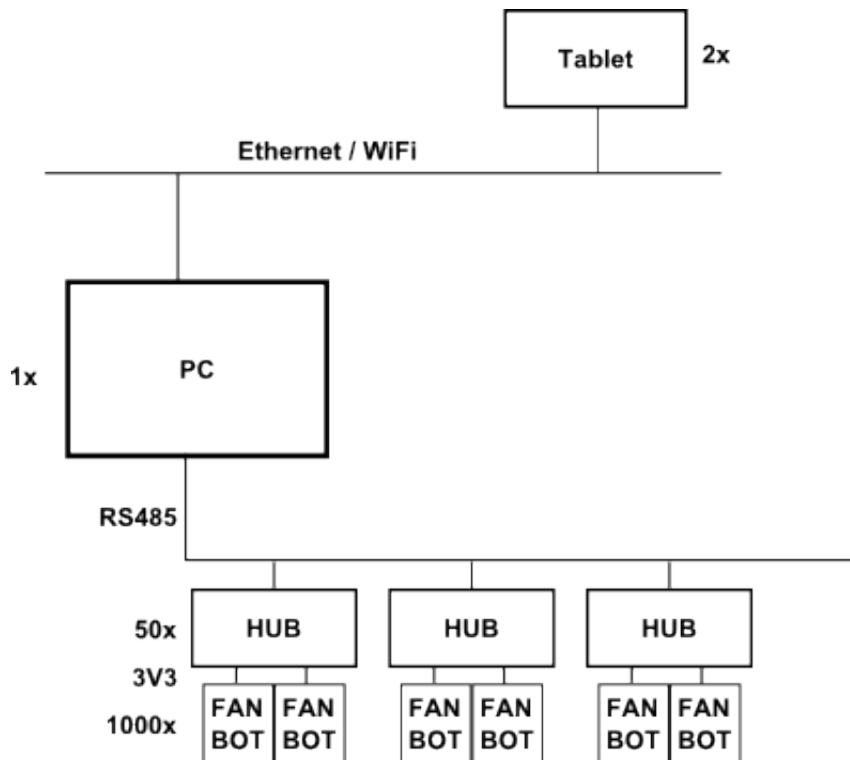
The board is powered by a 5VDC power supply (150W). The bus is powered by a 12VDC voltage (current per node <50mA).

3 Communication

3.1 HUB Communication

The Hub receives commands via an isolated RS485 link.

Physical connection	RJ45, 4 wire (D+/D-, GND, 12V)
Electrical signalling	RS485 half duplex
Link protocol	Asynchronous, 115200 bps, 8 bits, no parity, 1 stop bit
Application Protocol	"DMX like" with additional functions



The network communication is performed via a serial link from the PC to the hubs. The PC can issue a number of commands.

3.2 PC to HUB Communication protocol

From PC to HUBS via RS485 bus

Message format	
Byte	Meaning
0 .. 1	Start of message: Always '#' + '#'
2 .. 3	Opcode 16 bit opcode. (LSB first)
4 .. 5	16 bit Length (LSB first) of subsequent data (excluding checksum)
6 .. [length + 6]	8 bit binary data (can be zero length)
[Length + 6] .. [Length + 7]	16 bit checksum (LSB first)

All messages that do not have a valid CRC will be ignored. A message needs to be sent as a continuous stream of data. If a message transmission is interrupted for more than half a second, the message is ignored. Subsequent data is ignored until a "start of message" and valid opcode.

Opcodes			
NR	Name	Direction	Data
1	REQUEST_ID	PC → HUB	All nodes that are not tagged will send their ID on the bus after a random time [0 to 1 sec]
2	TAG_ID	PC → HUB	4 bytes (UID). The hub that is tagged (its UID corresponds to the UID data) will not respond to any REQUEST_ID commands.
3	PLAY_FRAME	PC → HUB	128 bytes (1024 bits) with play status Send a bitmapped block of data to all hubs. Each hub will decode the appropriate bits and send their state to the robots via the hub outputs
4	LED_FRAME	PC → HUB	128 bytes (1024 bits) with led status (ON/OFF)
5	POS_FRAME	PC → HUB	256 bytes (2048 bits) with servo position status (2 bits / port, 0=OFF, 1=LEFT, 2=MIDDLE, 3=RIGHT)
6	REQUEST_STATUS	PC → HUB	4 bytes (UID). The addressed slave will respond with their unique ID and status information in the HUB_STATUS_REPORT
7	CONFIG_FRAME	PC → HUB	4 + 48 bytes (8 bytes UID and 24 x 16bits indexes for all outputs)
128	ID_REPORT	HUB → PC	4 bytes (UID of HUB)
129	STATUS_REPORT	HUB → PC	4 bytes (UID of HUB) + 128 bytes additional status information containing the 24 x 4 bytes UID of the connected robots. Followed by the 4 byte UID of the hub, and 4 byte software revision code of the hub
0xDEAD	RESET	PC → HUB	Cause a reset of the hubs

Opcode 1: Send data frame

The data frame on the bus will be read by all slaves. Each slave decodes the appropriate bits, as configured using the config frame and send their state to the outputs.

Opcode 2: REQUEST_HUB_STATUS

This message is similar triggers the hub with a UID that matches the supplied UID to send its UID and status information. The hub is required to respond as soon as possible and WITHIN 0.1 sec to this message. There is a special situation when the UID is equal to 'everyone'. In that case all slaves will wait a random time (0.1 to 5 seconds) and send their information. This can be done to identify all slaves on a bus, when it is unknown what their UID's are.

Opcode 3: SEND_CONFIG_FRAME

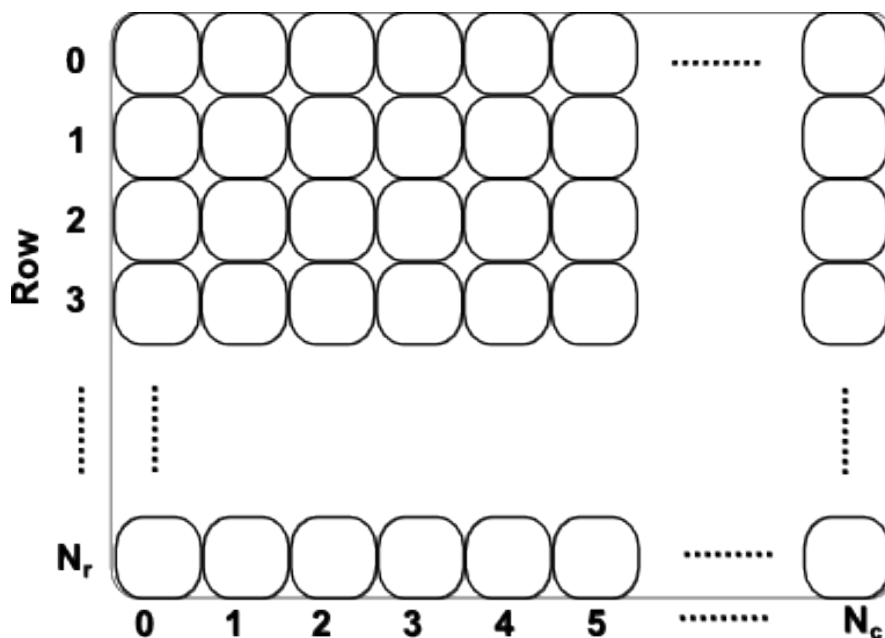
Upon reception of this frame by the slaves, they will check if the first 8 bytes match their ID and if so it will decode the rest of the data. Each 16 bit data item identifies a bit number in the data frame to decode for a given output. The first 16 bit corresponds to output "0" (the first output).

Opcode 4: HUB_ID_RESPONSE

This message is sent by the hub, in response to an IDENTIFY_HUBS or REQUEST_HUB_STATUS

3.3 Fanbot positions and addressing

The slaves are arranged in a matrix according to the following layout.



Each hub drives up to 24 cells the matrix. The matrix has 50 columns and 20 rows.

The hubs are configured to respond to the appropriate bits in the data frame, via their configuration.

3.4 Communication between HUB and robot

The communication between the hub and the robot is performed using an asynchronous serial protocol. 9600Bps, 8 bits, no parity, 1 stopbit.

Physical connection	USB cable/connector 4 wire (DP/DM, GND, 5V)
Electrical signalling	3V3, RS232 half duplex
Link protocol	Asynchronous, 9600 bps, 8 bits, no parity, 1 stop bit
Application Protocol	Specific

3.5 Communication Protocol between HUB and robot

The HUB can send a number of opcodes to the Fanbot brain board.

Opcodes			
NR	Name	Direction	Data
1	PLAY [p]	HUB → FAN	N.A.
2	STOP [s]	HUB → FAN	N.A.
3	SET_LEDS [L][#]	HUB → FAN	1 byte (8 bit LED status)
4	SET_SERVO1 [A][#]	HUB → FAN	1 byte (8 bit servo position)
5	SET_SERVO2 [B][#]	HUB → FAN	1 byte (8 bit servo position)
6	GET_NUMBER [n]	HUB → FAN	N.A.
7	GET_NAME[N]	HUB → FAN	N.A.
	NAME	FAN → HUB	32 bytes
	ID	FAN → HUB	8 HEX digits [0..F]

Opcode 1: PLAY

Program number 1 is executed immediately. It is executed once. If a new PLAY opcode is received while a program is running, the program will be interrupted and started again at the start. At the end of the program, the final state of the LEDs will remain.

Opcode 2: STOP

The running program is suspended immediately. Movement of the servo is stopped and all leds are turned off.

Opcode 3: SET_LEDS

The forces the outputs to a certain state. If a program was running, it is immediately stopped and the provided LED state is assumed

Opcode 4: SET_SERVO1

Send a position for servo 1 to the robot. If a program was running, it is immediately stopped and the provided SERVO1 position is assumed

Opcode 4: SET_SERVO2

Ridential to SET_SERVO1, but controlling servo 2

Opcode 6: GET_NUMBER

The robot sends the 32 bit serial number as 8 hex digits

Opcode 7: GET_NAME

The robot sends the content of the name memory as 32 hex characters (null value if not entered)

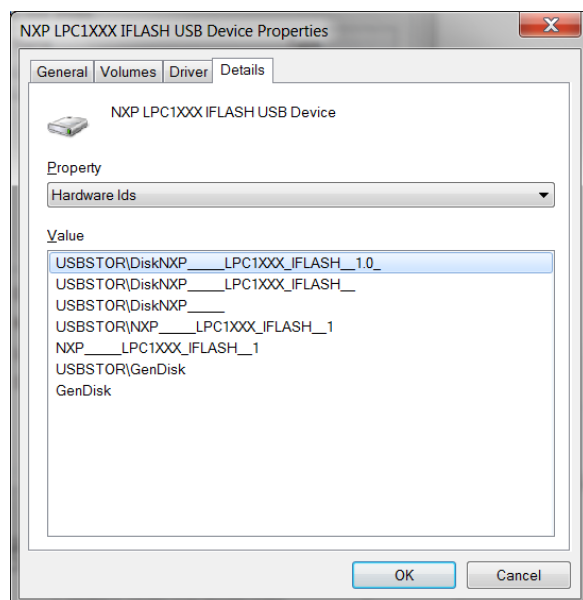
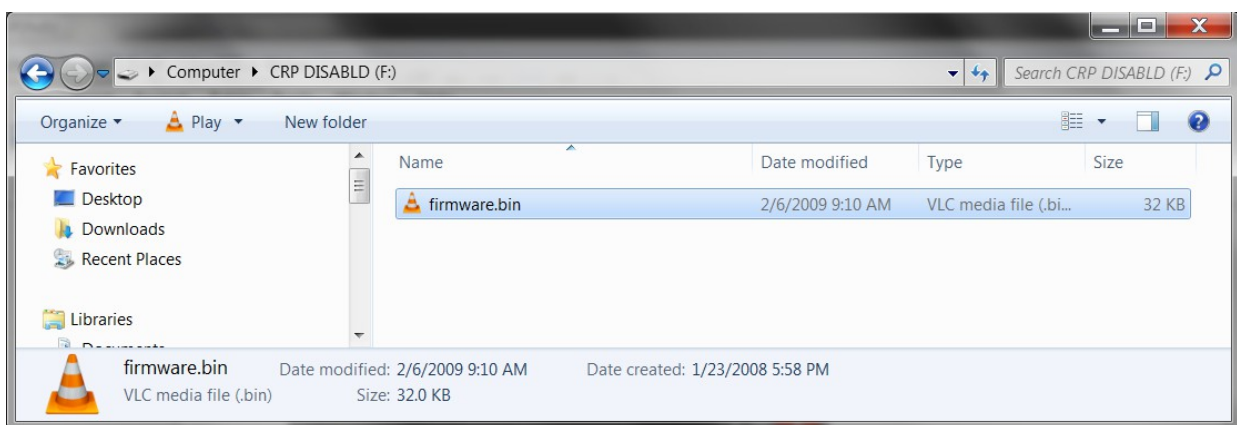
3.6 Communication between PC and Fanbot

A PC can communicate directly to a Fanbot using the USB connection. Two communication modes are available: MSD (Mass storage Device) and HID (Humane Interface Device).

Physical connection	USB cable/connector 4 wire (DP/DM, GND, 5V)
Electrical signalling	USB high speed (12MHz)
Link protocol	HID (Note: MSD is used to flash firmware)
Application Protocol	Specific
Message length	Fixed 48 byte messages
PID/VID	HID: 0x1234 / 0x1234 MSD: 0x1FC9 / 0x000B

3.6.1 MSD mode

MSD mode is activated when the program key is pressed while the USB connector is inserted, OR when no valid firmware is present on the chip. MSD is the default mode when the chip is not yet programmed.



A drive with the volume name "CRP DISABLD" and a single file "FIRMWARE.BIN" is presented to the OS. This file can be deleted or overwritten. A new firmware file can be copied over the existing file. The name of this file is not significant. The size should be smaller than the available FLASH size. The BIN file should be a valid LPC ARM executable with a valid CRC (see NXP application note). To activate the new firmware, the board needs to be reset.

3.7 HID mode communication

The HID mode is activated when the fanbot firmware is operational. The HID protocol allows sending and receiving of fixed length messages. The messages have a maximum length of 64 bytes.

Message format (PC → FAN)		
Byte	Name	Description
1	Opcode	The opcode specifies what the fan should do. See the list with opcodes for more information
2 .. 63	Data	Optional data for the opcode. If not used, the content is ignored. The data should be transmitted in each frame.

Opcodes (HID write)			
NR	Name	Direction	Data
0	SET OUTPUTS	PC → FAN	Byte 1: LEDs (bit mapped. Bit 0 is Left eye, bit 1 is right eye, bit 2 is left mouth, etc.) Byte 2: Servo 1 output Byte 3: Servo 2 output
1	Write EEPROM data	PC → FAN	Byte 1: Memory bank (0=program or 1=name) Byte 2..62: Data. Memory bank 0: First Program
2	PLAY	PC → FAN	Play current frame
3	NAME	PC → FAN	Set the name string (byte 1..33 is ASCII name)
4	STOP	PC → FAN	Stop playing

Message format (FAN → PC)		
Byte	Name	Description
1..8	Serial number	The serial number is read from the microcontroller
9 .. 41	Name	The name of the robot. Read from memory bank 8.

3.8 Communication between Control application and proxy

The proxy application receives information via a TCP/IP socket and forwards the data via USB (serial) to the RS485 HUB network. It can serve multiple clients.

Physical connection	Ethernet (wired/wireless)
Protocol	TCP/IP
Network configuration	Private IP range, Private ESSID wifi router with DHCP server
IP-Range	192.168.88.0/8
ESSID / PW	FANBOT / TRIBUNE10
IP address	192.168.88.100
Port	2013

On the tcp/ip port, the following protocol is used:

Opcodes			
CHAR	Name	Direction	Function
'^'	START_OF_FRAME	CLIENT → SERVER	Set cursor to (0,0), current frame is sent to the fanbots
'\n'	NEW_LINE	CLIENT → SERVER	Increment cursor y coordinate and set x to zero
'*'	SET_ON	CLIENT → SERVER	Set pixel (x,y) on, and increment x
'.'	SET_OFF	CLIENT → SERVER	Set pixel (x,y) to OFF and increment x

4 Software

There are various software components in the system.

4.1 Control application

The control application has a number of buttons that trigger these functions:

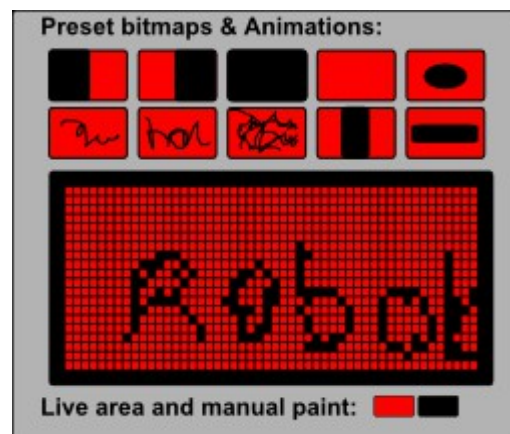
- 1) Initialize bus (reads the config file and sends the config data to all hubs)
- 2) Identify slaves (shows all identified hub UID's in a list)
- 3) Go Live (this enables the bitmap buttons and live area and starts sending data)

In live mode, there are a number of "bitmap" buttons that trigger the transmission of a single data frame or a number of dataframes (a sequence of multiple bitmaps). Each bitmap is 42 pixels wide and 24 pixels high.

In the "live area" you can see the actual data that is sent to the slaves and "paint" over the 42x24 pixel canvas with the mouse (various brushes: ON/OFF and TIME). Where the TIME brush turns on the pixels for a fixed amount of time (eg. 0.5 sec).

In live mode, data frames are sent continuously to the hubs at several frames/sec. The hubs are polled one by one to update the status on the screen.



The control application is visible on the PC, and can be controlled via a tablet using a Java applet in a browser (or native Android app?)



Tablet user interface: Allow to select preset bitmaps and animations, and manually paint on the live canvas

4.2 Control application use case

The soccer matches are hosted by one or two hosts who will comment on the game and entertain the audience during the break. The walk around with a wireless microphone and during the break they join the audience on the stands to answer their questions. One of these hosts will also control the fanbots with a tablet in the following situations:

Situation	Fanbot reaction	Program	HMI
When a team scores	The fanbots on one site cheer	Made by the kids (2-4 secs)	
Begin/end of the game	All the fanbots cheer for 5 sec	Made by the kids (2-4 secs)	
During the game	Some random robots will cheer	Made by the kids (2-4 secs)	Preset button
	A wave is done	All on	Preset button (left to right and right to left)
	(Nice:) play short gif movies	All on	Select from a list
During the breaks	The audience/host can draw a pattern of waving fanbots (stay on about 1 sec and then die out)	All on	Draw on tablet
	(Nice:) play short gif movies	All on	Select from a list (make a queue)

4.3 Fanbot program

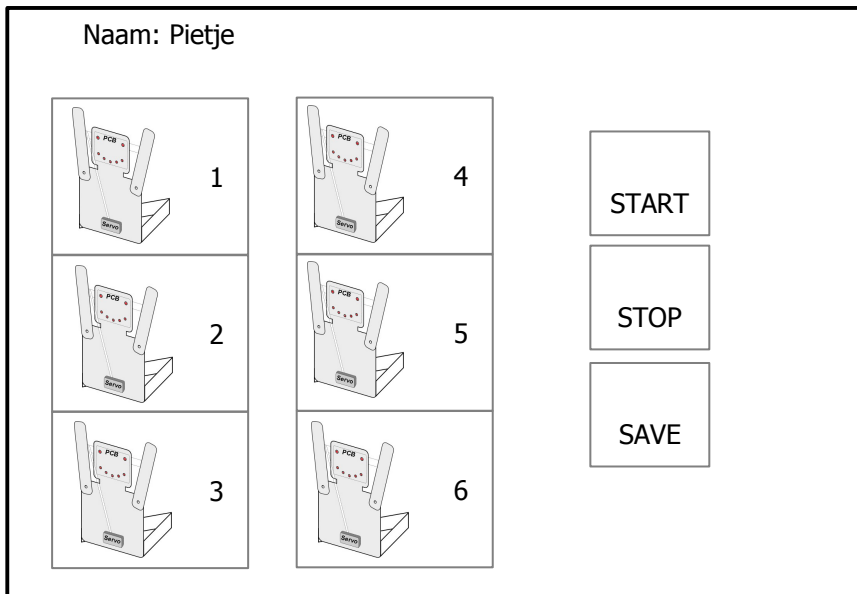
The fanbot should be able to wave (servo movement) and flash (7 LEDs).

The movement must be fixed between two maximum values (180° is too much, approx 60°-90°)

For the wave and gif animation the fanbots must have a second programm with 4 LEDs on and all LEDs on (movements?)

4.4 Workshop programming tool

In the workshop, the participants can program the tool, using a simple user interface. It allows the user to make "Moves" and switch the LEDs ON and OFF. Programming is done in the form of a "comic strip".



- LEDS can be programmed individually
- Sequence of 10-20 steps for LEDs
- Servo movement are set with a slider for each step in the program.
- When a LED is toggled on the screen of the slider position is changed, the data is sent directly to the connected hardware to update the state in real-time.
- When the "RUN" button is pressed, the sequence is programmed in the EEPROM of the robot, and the sequence is executed. The state is updated in real-time on the screen of the computer.

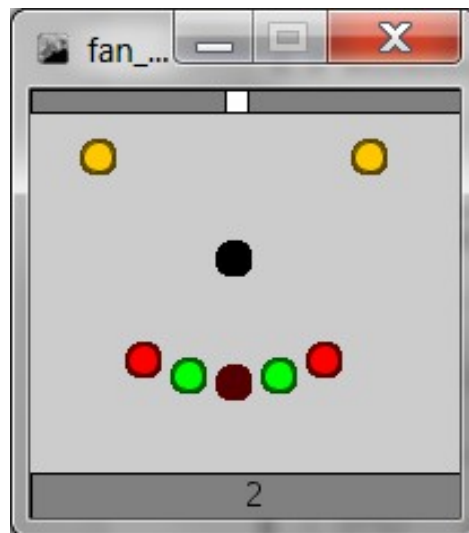


Illustration 1: Example of fanbot user interface element. The LEDs can be toggled on/off. The top slider can move from left to right to change the arm position. Pressing the centre (nose) will store the sequence and start the playback. The step number can be selected using the bottom slider.

Additional functions for the workshop tool

The program should copy a specified file when a drive is detected with a volume name "CRP DISABLED" that contains a file FIRMWARE.BIN. The FIRMWARE.BIN file should be overwritten with the new file content. Once copied, the file content should be verified

A pictogram should indicate the state of the program.

State	condition	Pictogram
1 CONNECT ROBOT	No robot is detected (no MSD or HID communication)	Robot with cross
2 COPY FIRMWARE.BIN	The FIRMWARE.BIN file is being copied and verified	Robot busy with file
3 FIRMWARE OK	The written firmware file matches the required firmware.	Disconnect robot
4 ROBOT READY	HID communication active	Robot with serial number and name
5 ROBOT ERROR	Unspecified error	Robot with question mark

4.4.1 Fanbot HID interface program

The Fanbot HID interface program is a standalone executable that can be started from the human interface. It will handle all communication with the fanbot. It is controlled via the Standard input and reports state information via the standard output using simple ASCII commands and event.

When the program is started it will wait for a FanBot to be connected to the computer.

State #	State	INPUT	OUTPUT
<i>The program is started</i>			
1	Wait for robot to connect	-	-
2	Robot connected		CONNECT [serial#]
			PROGRAM [l1] [s1] .. [n1] [sn]
		SET [leds] [servo]	
		PROGRAM [l1] [s1] .. [ln] [sn]	
		NAME [ASCII NAME]	
		PLAY	
3	Disconnected		DISCONNECT
<i>The program will terminate with exit code 0</i>			

The controlling application will monitor the existence of a "firmware.bin" file on a pre-configured path. If this file appears, it will be overwritten with the pre-defined firmware file.

The HID communication application will be continuously restarted from the controlling application. After starting, it will wait for the "CONNECT" message.

The controlling application should save the name in the robot using the NAME command each time the name is changed on the screen.

The user is now able to create program steps and download the program. Each time the program is downloaded it is also played.

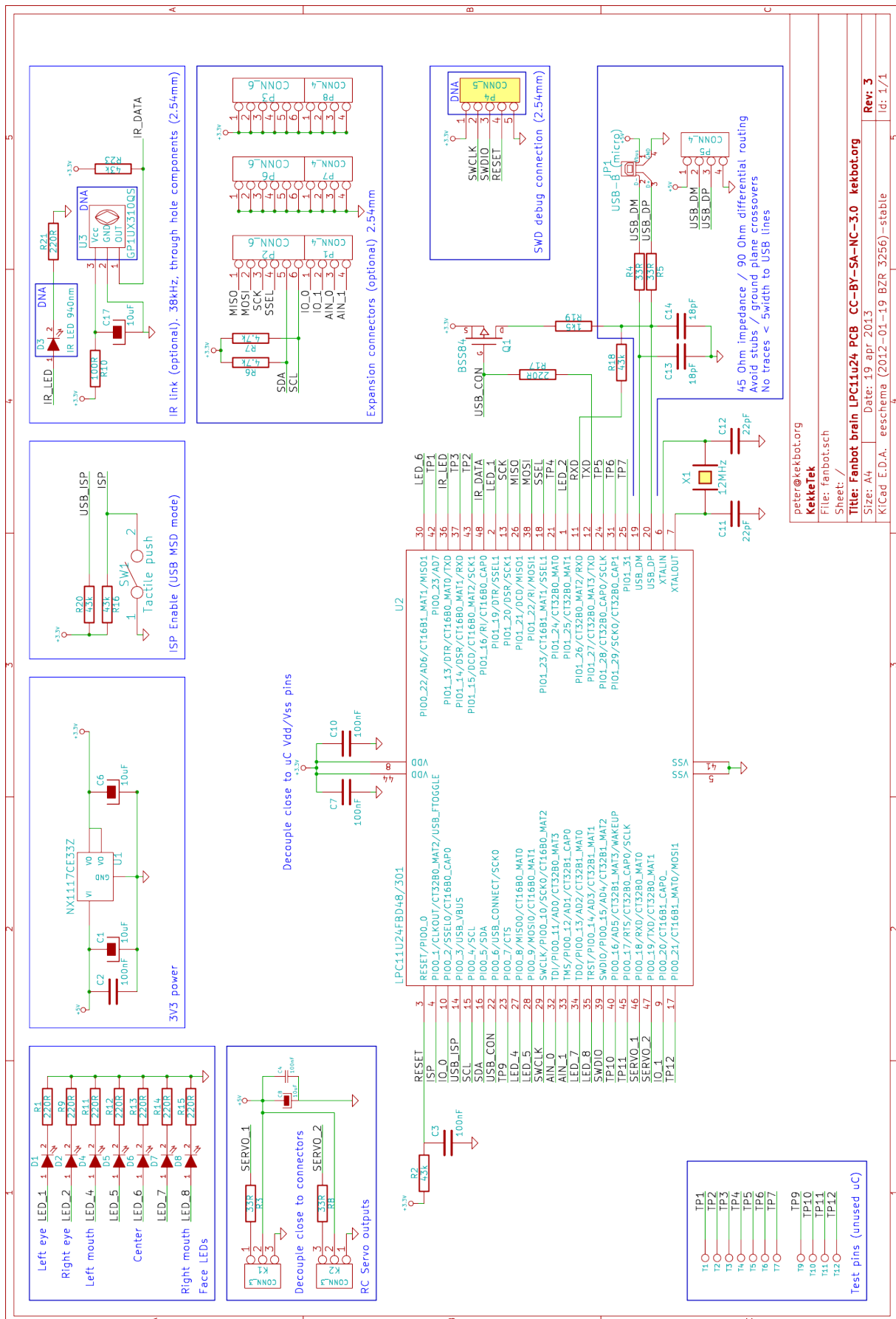
5 Stand



Stand matrix (44x24 Fanbots, 24 per Hub), 10,5m x2,9m x 2,4m (lxwxd)

6 Schematics & Bills of materials

6.1 Fanbot Electrical schematics (obsolete by last production designs)



6.2 Fanbot Bill of materials (obsolete by last production designs)

Fanbot Components	spec	Nr / robot	Tot. nr	ref	Remarks
Capacitor	100nF	6	9000	C2, C3, C4, C7, C10, C16	
Resistor	100R	1	1500	R10	Optional (IR)
Resistor	10K	1	1500	R23	Optional (IR)
Capacitor	10uF	4	6000	C1, C6, C8, C15	
Capacitor	10uF	1	7500	C17	Optional (IR)
Crystal	12MHz	1	1500	X1	
Capacitor	18pF	2	3000	C13, C14	
Resistor	1k5	1	1500	R19	
Resistor	220R	7	10500	R1, R9, R11, R12, R13, R14, R15,	
Resistor	220R	1	1500	R21	Optional (IR)
Capacitor	22pF	2	3000	C11, C12	
Resistor	2k	2	3000	R6, R7	
Resistor	33R	2	3000	R4, R5	
Resistor	43k	3	4500	R2, R16, R?	
Resistor	5K	2	3000	R17, R18	
Resistor	68R	2	3000	R3, R8	
Connector	CONN_10	3	4500	P1, P2, P3	
Connector	CONN_3	2	3000	K1, K2	
LED	IR_LED	1	1500	D3	Optional (IR)
Microcontroller	LPC11U24FBD 48/301	1	1500	U2	NXP microcontroller, to be confirmed.
	NX1117C	1	1500	U1	
LED	RED	2	3000	D1, D2,	Different colours are nice, there are 2 LEDs for the eyes and 5 for the mouth
LED	GREEN	3	4500	D4, D5, D7, D8	
LED	YELLOW	2	3000	D6	
IR reciever	SFH-5110	1	1500	U3	Optional (IR)
	SPST	1	1500	SW1	
	SWD	1	1500	P4	
Connector	USB-B	1	1500	JP1	
others					
PCB Fanbot	2 layer, 5x5 cm	1	1500	Eurocircuits	
SERVO		1	1500	Ordered	
USB cable	Micro USB, 1m	1	1500	Ordered in 9 colours	
Carboard body		1	1500	Laser cut material	
Split pins		4	6000	Officecenter?	

6.3 Fanbot PCB layout

6.3.1 Mechanical and layout considerations

Important mechanical and layout considerations:

- All through hole connectors on 2.45 mm pitch, at the edge of the board as much as possible
- Signal pins of option connectors (P1/P2) can be swapped if required for proper layout
- Test pin pads 1.5 mm, any location allowed
- LED locations and mounting hole exact position defined in drawing

Component	Type	Location
SW1	Tactile switch	Center of board (front side)
JP1	USB micro B	Bottom edge, left aligned, cable facing down (front side of PCB)
K1	2.54mm pins, right angle	Bottom edge, right aligned, cable facing down (front or back side of PCB)
K2	2.54mm pins, right angle	Right edge, bottom aligned, cable facing right (front or back side of PCB)
D3 / U3	IR LED, IR receiver, 2.54mm pitch (not mounted)	Top edge, right aligned
P4	2.54 mm header (not mounted)	Right edge
D1 .. D8	SMD Leds	Exact location defined in drawing "fanbot-brain.dxf"
H1	Hole 3.5 mm	Exact location defined in drawing "fanbot-brain.dxf"






6.3.2 Silkscreen Front

The front silkscreen will include the following identifications:

Identification	Text
PCB name	Fanbot brain
Licence	CC-BY-SA-NC-3.0
Website	KekBot.org
SW1	Program
JP1	USB
K1	SERVO1, +5V, SIG1, GND
K2	SERVO2, +5V, SIG2, GND
D3/U3	IR
P4	SWD, -, R, D, C, +
P1,P2,P3,P6,P7,P8	1, 2, 3, 4, SDA, SCL, 7, 8, 9, 10, 3V3, GND, SIG

6.3.3 Silkscreen Back

The back silkscreen will include the logos of the various sponsors

Logo	Bitmap
AME	
Arrow	
Eurocircuits	
KekkeTek	
RoboCup	

6.4 Fanbot HUB Electrical schematics

t.b.d.

6.5 Fanbot HUB Bill of materials

!!! TO BE UPDATED WITH FINAL BOM !!!

HUB Components	spec	Nr / robot	Tot. nr	ref	Remarks	
LITTELFUSE-1812L125/16DR	1.25A	24	960	F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, F13, F14, F15, F16, F17, F18, F19, F20, F21, F22, F23, F24	1836775	0.34
Capacitor	10000uF	4	160	C2, C16, C19, C20		
Capacitor	100nF	10	400	C1, C3, C4, C6, C7, C11, C12, C15, C17, C18		
Resistor	100R	24	960	R7, R8, R11, R12, R20, R21, R22, R23, R24, R25, R26, R27, R28, R29, R30, R31, R32, R33, R34, R35, R36, R37, R38, R39		
Resistor	10k	3	120	R4, R5, R19		
Capacitor	10uF	2	80	C9, C10		
Resistor	120R	2	80	R2, R42		
Resistor	12MHz	1	40	X1		
Capacitor	18pF	4	160	C13, C14, C21, C22		
Resistor	1k	3	120	R15, R44, R45		
Resistor	1k5	1	40	R16		
Resistor	22k	1	40	R13		
Resistor	330R	5	200	R1, R3, R6, R43, R46		
Resistor	33R	2	80	R17, R18		
Capacitor	4u7	2	80	C5, C8		
	5.6V	1	1206	VR1	1189317	0.6 AVX - VC120605D150 DP
Resistor	560R	4	160	R9, R10, R40, R41		
	ADM2484	1	16SOIC	U2	2067811	ANALOG DEVICES
MOSFET, P, SOT-23, 0.1	BSS84	1	SOT-23	Q1	1510765	0.1
DIODE (FAIRCHILD SEMICONDUCTOR - RURG5060)	DIODE	2	80	D2, D8	1651155	2 FAIRCHILD SEMICONDUCTOR - RURG5060
	DIPS_04	1	40	SW3		
LED	GREEN	2	80	D1, D6		
JUMPER	JUMPER	1	40	JP1		
	LPC11U35FBD 64/401	1	LQFP64	U4	2115655	3.00
	NX1117CE33Z	1	SOT-223	U1	2057285	0.25
	NX1117CE50Z	1	SOT-223	U3	2057286	0.27 NXP -

HUB Components	spec	Nr / robot	Tot. nr	ref	Remarks	
						NX1117CE50Z
LED	ORANGE	1	40	D3		
	POWER	2	80	P1, P2		
	R	1		R14		
LED	RED	2		D4, D7		
	RJ45IN	1		J13		
	RJ45OUT	1		J14		
DIODE, SWITCH, 100V, 0.7A.DO-219AB	S07B-GS08(>1A)	1	DO-219AB	D5	1864884	0.06 VISHAY SEMICONDUCTOR - S07B-GS08 -
	SPST	2		SW1, SW2		
	USB	1		J15		
Connector	USBA-D	12		J1, J2, J3, J4, J5, J6, J7, J8, J9, J10, J11, J12	0.28	USB-AW-2
others						
PCB HUB	2 layer, 10x15 cm	1	40	Eurocircuits, to be confirmed		

6.6 Tinkering materials

During the workshop for pimping the fanbots:

- Tinkering materials: tape, glue, felt-tips, coloured paper, wire,..
- Scrap: wiring harness, small plastic caps, connectors, nice looking parts, ...

