Fujitsu Semiconductor Europe
User Guide

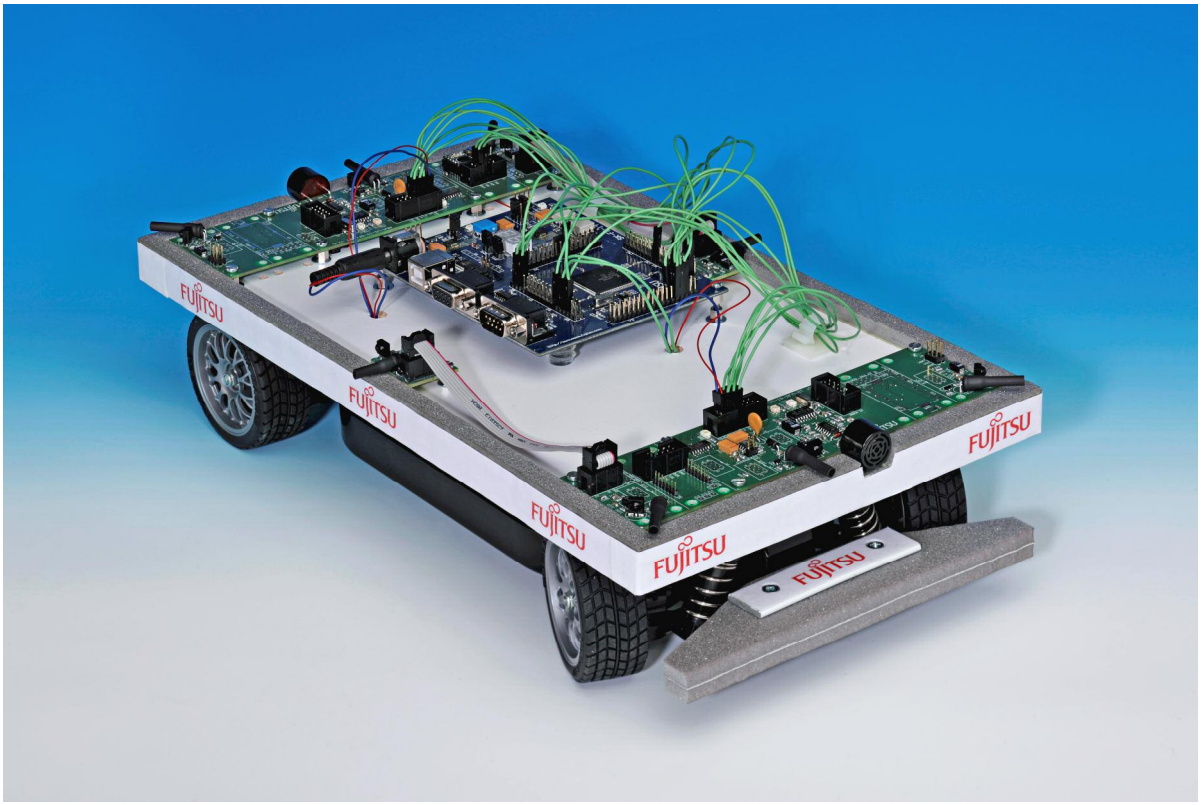
FSEUMCU-ug-000002-16

16FX/FR

EVALUATION SYSTEM

STUDENT-MODEL-CAR

USER GUIDE



FUJITSU

Revision History

Date	Issue
2010-01-19	CII – Initial Version 1.0
2010-02-10	CII – V1.1
2010-02-25	CII – V1.2: Added 3.3.1 - Speed controller calibration
2010-08-12	DWi – V1.3: IR line sensor added Updated table 1 + 5 Added 3.8 IR line sensor Added 6.11 IR line sensor module
2010-09-27	DWi – V1.3.1: Updated headlines Updated 3.8 – ADA-IR-LINE-SENSOR
2010-09-29	DWi – V1.4: Updated Warranty and Disclaimer
2010-09-30	DWi – V1.5: Updated 6.12 IR line sensor module Added 6.12.1 Software Added 6.12.2 Test software
2011-06-10	MWi – V1.6: Line-Sensor connector numbering corrected in Table 1: Modules to MCU connections

This document contains 55 pages.

Warranty and Disclaimer

The use of the deliverables (e.g. software, application examples, target boards, evaluation boards, starter kits, schematics, engineering samples of IC's etc.) is subject to the conditions of Fujitsu Semiconductor Europe GmbH ("FSEU") as set out in (i) the terms of the License Agreement and/or the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials.

Please note that the deliverables are intended for and must only be used for reference in an evaluation laboratory environment.

The software deliverables are provided on an as-is basis without charge and are subject to alterations. It is the user's obligation to fully test the software in its environment and to ensure proper functionality, qualification and compliance with component specifications.

Regarding hardware deliverables, FSEU warrants that they will be free from defects in material and workmanship under use and service as specified in the accompanying written materials for a duration of 1 year from the date of receipt by the customer.

Should a hardware deliverable turn out to be defect, FSEU's entire liability and the customer's exclusive remedy shall be, at FSEU´s sole discretion, either return of the purchase price and the license fee, or replacement of the hardware deliverable or parts thereof, if the deliverable is returned to FSEU in original packing and without further defects resulting from the customer's use or the transport. However, this warranty is excluded if the defect has resulted from an accident not attributable to FSEU, or abuse or misapplication attributable to the customer or any other third party not relating to FSEU or to unauthorised decompiling and/or reverse engineering and/or disassembling.

FSEU does not warrant that the deliverables do not infringe any third party intellectual property right (IPR). In the event that the deliverables infringe a third party IPR it is the sole responsibility of the customer to obtain necessary licenses to continue the usage of the deliverable.

In the event the software deliverables include the use of open source components, the provisions of the governing open source license agreement shall apply with respect to such software deliverables.

To the maximum extent permitted by applicable law FSEU disclaims all other warranties, whether express or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the deliverables are not designated.

To the maximum extent permitted by applicable law, FSEU's liability is restricted to intention and gross negligence. FSEU is not liable for consequential damages.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect.

The contents of this document are subject to change without a prior notice, thus contact FSEU about the latest one.

Contents

1 Overview

The Student-Model-Car is a hardware platform for students to get familiar with embedded microcontroller projects. It features a basic setup including actors, sensors and communication interfaces for a quick start but it can also be extended by additional hardware modules. Starting with simple examples which only use single features of the platform it might be used up to autonomous car control.

1.1 Features

- Starter kit SK-16FX-EUROSCOPE or SK-FR-144PMC-91467B
- Eight infrared transmitters and receivers in eight directions
- Two ultrasonic transmitters (front and back)
- RFM wireless communication module (RFM12)
- Servo for steering and motor, controlled by PWM signal
- Voltage regulation for unregulated battery supply
- Sensor modules are protected against short circuits
- Main peripheral PCB is prepared for stacked PCB providing power supply to the upper level
- Optional Bluetooth communication module (BTM-222)
- 5 infrared reflective sensors for line detection

1.2 Modules

1.2.1 Car body

The body is a TT-01 chassis taken from a Tamiya model car. It features a battery pack fixture and damped wheel suspension. The chassis also has fixtures used for the mounting plate for the electronic components.

1.2.2 Power supply

Power for the electronic components of the model car is provided by a six cell 7.2 V battery pack.

1.2.3 Drive

The chassis includes a steering servo and a speed controller which drives the motor. Both are controlled by PWM signals.

1.2.4 Shock absorbers

The front and back side of the chassis each hold a shock absorber to prevent higher damage to be dealt to the environment when the car bumps into it.

1.2.5 Mounting plate for electronic components

The electronic modules are mounted on a wooden plate which in turn is mounted on the chassis of the car. This plate is framed by a damper to reduce possible damage to the environment.

1.2.6 Starter kit

The starter kit provides the microcontroller and the peripheral resources required to use the Student-Model-Car such as power supply, oscillator, and an USB interface for flash downloading and debugging. It also provides pin headers to connect to the other resources.

1.2.7 ADA-US-IR-RFM-BT

Also referred to by main sensor modules (front and back). These modules combine three infrared modules for object detection and short distance communication, an ultrasonic distance measurement module and a RFM12 wireless communication module. Optionally a BTM-222 Bluetooth module can be mounted. Furthermore a multiplexer allows to switch between the three infrared channels on this module plus one side module, voltage regulator generates a 5 V / 3.3 V level from an unregulated battery supply and two LEDs allow for status signalling.

1.2.8 ADA-INFRARED

These modules are referred to by side infrared modules (left and right). They each contain one infrared diode and one receiver. They are powered and interfaced to the microcontroller through the front and back module respectively.

1.2.9 ADA-IR-LINE-SENSOR

The line sensor module features five reflective optical sensors directed at the ground below the car. They can be used to detect a line or edges on the ground.

Note: This module is not shipped with the first version of the Student-Model-Car.

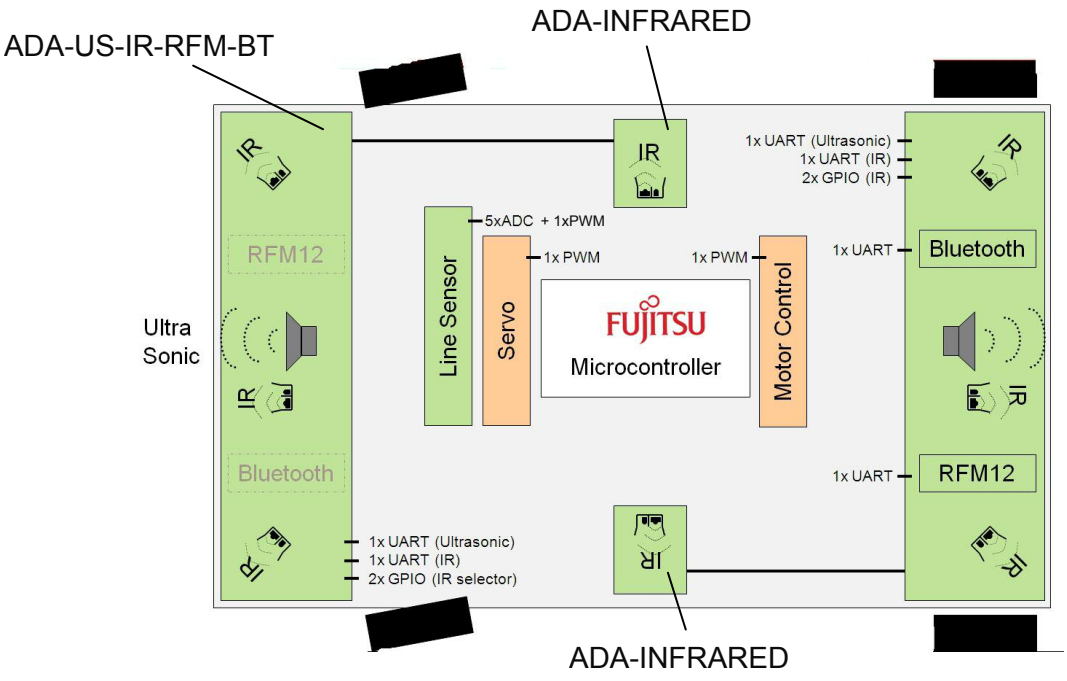


Figure 1: Hardware platform outline

2 Quick setup

This chapter will explain how to quickly connect the modules on the car. For detailed information of the signals see chapter 3 - Module description.

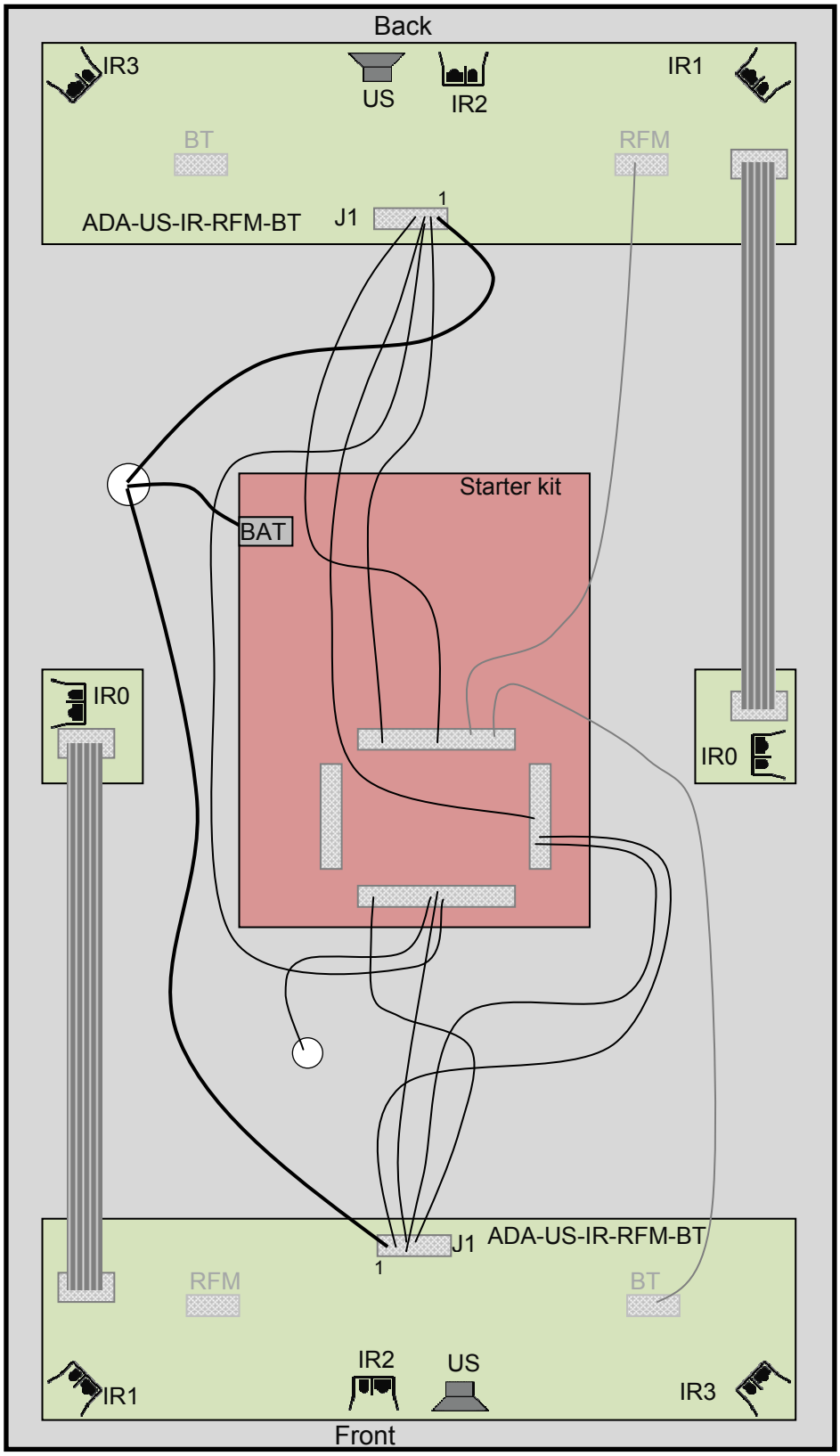


Figure 2: Schematic overview of the mounting plate

2.1 Power supply for sensor modules and drive

Plug the power supply jack from the speed controller in the battery connector.

Connect the power supply board through header J1 to the battery.

Connect the servo power supply cable with the supply board's regulated output J2.

Note: Each module has its own power LED which should light up when the module is connected to the power supply. If one module does not light up its power LED recheck the connections. Make sure the jumper setting of JP1 on the ADA-US-IR-RFM-BT boards is correct, that is open for 5 V or closed for 3.3 V operation.

Connect the power pins from the front and back side main sensor modules (pins 1 and 2) with the appropriate pins on the power supply board on J3. Use the power connector cable to connect the starter kit with the power supply board on J3. Fit the cables through the two cable holes.

If a Bluetooth module is mounted on one of the main sensor modules set the power select jumper according to the operating voltage of the sensor module:

Jumper	Setting	Description
JP2	1-2	Board operates at 5 V
	2-3	Board operates at 3.3 V

2.2 Mounting plate

Connect signal cables (green) to the input of the speed controller and the steering servo and fit both cables through a cable hole of the plate.

Put the plate on the car so that the four fixtures of the chassis go through the four holes of the plate. Put the holding clamps through the fixtures.

2.3 Side infrared modules

Connect the infrared modules on both sides of the car to the corresponding main sensor modules with a ribbon cable.

2.4 Microcontroller

Connect the signals from the different modules to the microcontroller with green connector cables.

Module			SK-16FX-EUROSCOPE MB96F348HS		SK-FR-144PMC-91467B MB91F467B	
Module	Pin	Signal	Pin	Signal	Pin	Signal
Main sensor module front	3	IR_A	55	P08_0	110	P29_0
	4	IR_B	56	P08_1	111	P29_1
	5	IR_RX	54	SIN9R (P07_7)	98	SIN6 (P18_0)
	6	IR_TX	36	PPG2 (P06_2)	66	PPG10 (P16_2)
	7	US_SDI	23	SOT2 (P05_1)	39	SOT2 (P20_1)
	8	US_SDO	22	SIN2 (P05_0)	38	SIN2 (P20_0)
	Not connected		53	SOT9R/INT6 (P07_6) [1]	128	INT2 (P24_2) / SOT6 (P18_1) (cable) [2]
	Not connected				99	
Main sensor	3	IR_A	95	P02_0	118	P28_0

Module	Module Pin	Module Signal	SK-16FX-EUROSCOPE MB96F348HS Pin	SK-16FX-EUROSCOPE MB96F348HS Signal	SK-FR-144PMC-91467B MB91F467B Pin	SK-FR-144PMC-91467B MB91F467B Signal
module back	4	IR_B	96	P02_1	119	P28_1
	5	IR_RX	60	SIN1 (P08_5)	16	SIN0 (P21_0)
	6	IR_TX	37	PPG3 (P06_3)	67	PPG11 (P16_3)
	7	US_SDI	86	SOT3 (P01_3)	42	SOT3 (P20_5)
	8	US_SDO	85	SIN3 (P01_2)	41	SIN3 (P20_4)
	Not connected		59	INT15R (P08_4) / SOT1 (P08_6) (jumper) [1]	15	INT7 (P24_7) / SOT0 (P21_1) (jumper) [2]
	Not connected		61		17	
RFM module	3	SDI	79	SOT8R (P00_4)	102	SOT7 (P18_5)
	4	SDO	80	SIN8R (P00_5)	101	SIN7 (P18_4)
	5	SCK	78	SCK8R (P00_3)	103	SCK7 (P18_6)
	6	xSEL	81	P00_6	61	P22_5
BTM module	3	SDI	76	SOT7R (P00_1)	93	SOT4 (P19_1)
	4	SDO	77	SIN7R (P00_2)	92	SIN4 (P19_0)
Servo		PWM	34	PPG0 (P06_0)	64	PPG8 (P16_0)
Motor		PWM	35	PPG1 (P06_1)	65	PPG9 (P16_1)
IR line sensor	8	GND	GND	GND	GND	GND
	7	Sensor 4	29	AN15 (P05_7)	116	AN6 (P29_6)
	6	Sensor 3	28	AN14 (P05_6)	115	AN5 (P29_5)
	5	Sensor 2	27	AN13 (P05_5)	114	AN4 (P29_4)
	4	Sensor 1	26	AN12 (P05_4)	113	AN3 (P29_3)
	3	Sensor 0	25	AN11 (P05_3)	112	AN2 (P29_2)
	2	PPG/PWM	38	PPG4 (P06_4)	70	PPG14 (P16_6)
	1	VCC	VCC	VCC	VCC	VCC

Table 1: Modules to MCU connections

Note:

[1]: For the MB96F348HS the pins 59 and 61 have to be connected together and pin 53 has to be left unconnected! On JP4 no connection may be made to position '1'.

[2]: For the MB91F467B pins 128 and 99 have to be connected together and pins 15 and 17 also! On JP4 no connection may be made to position '0' when a Bluetooth module is used.

See chapter 3.5.3 - SOTx -> INTx connections for further details.

3 Module description

In this chapter the different modules of the Student-Model-Car will be described.

3.1 Mechanic components

3.1.1 Chassis

The chassis is a Tamiya TT-01 equipped with a Tamiya TSU-01 servo and a Tamiya TEU-101BK speed controller.

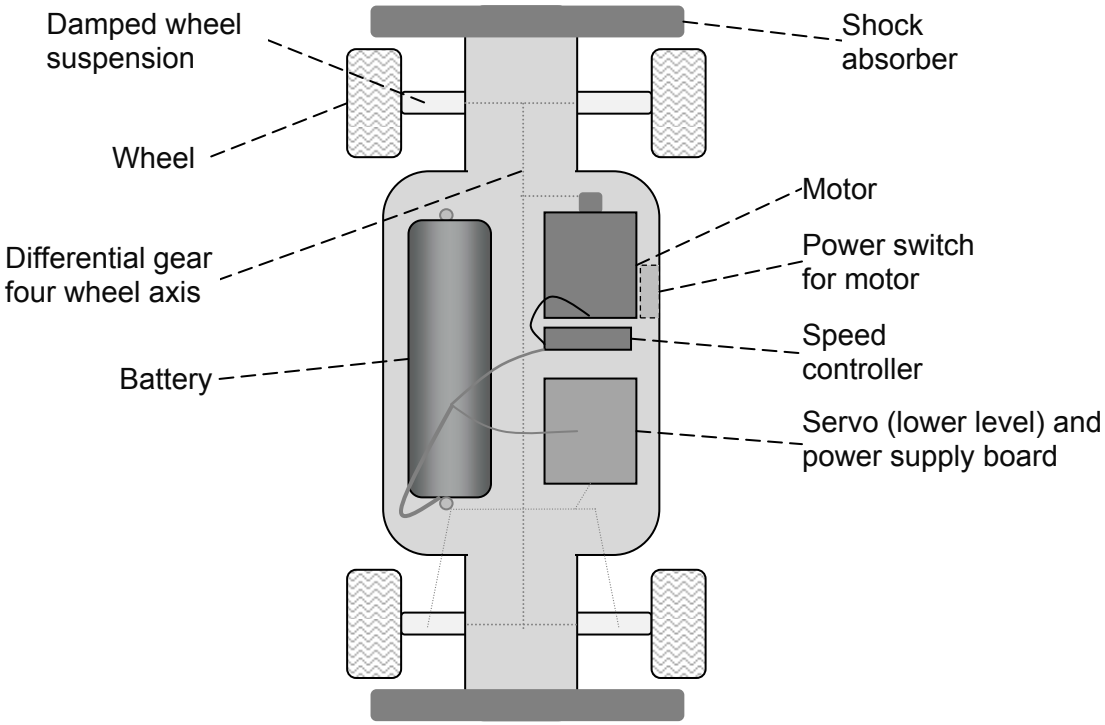


Figure 3: Schematic overview of chassis

3.1.2 Mounting plate

The electronic components such as sensor modules and the starter kit are mounted on a wooden plate. For that purpose the panel is supplied with holes which contain T-nuts to screw the boards. Furthermore the panel has a foam border to reduce possible risks of damaged objects by collisions.

3.2 Power supply

Power is supplied by a 6 cell / 7.2 V battery pack with Tamiya connector. It is then directly fed into the speed controller and supplied to the electronic boards through a power supply board. This board also generates a regulated 5 V output for the steering servo.

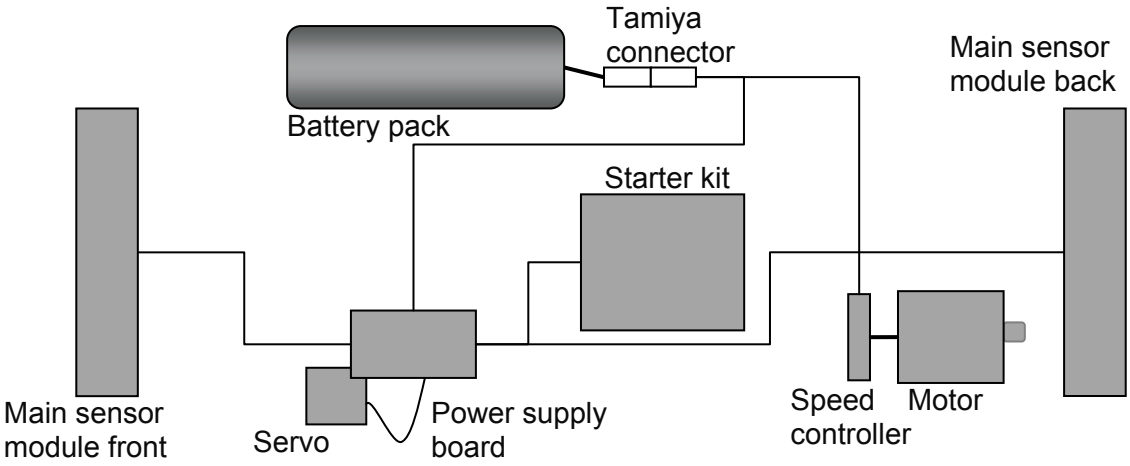


Figure 4: Power distribution

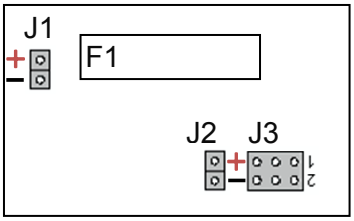


Figure 5: Power supply board pinout

The power supply board includes a 0.5 A fuse which protects the electronic boards and the servo. J1 is the power input from the battery, J2 the regulated 5 V output to the servo and J3 provides the battery voltage 3 times for the electronic boards.

Note: Minimum input voltage for the whole system to function properly is about 6.8 V.

3.3 Speed controller

The speed controller has five connections:

Cable	Description
Red / black	Power supply input from battery
Red / black	Power switch connection
Red / black	Provides the power for the receiver module (not available in this setup)
Red / black / green	Control input Red: unconnected Black: ground Green: PWM signal
Orange / blue	Motor power output

Table 2: Speed controller connections

The PWM input signal has a period of 17.43 ms. Duty cycle for null value is about 1.53 ms, lower values down to 1.14 ms are for forward drive whereas higher values 1.86 ms are for backward drive.

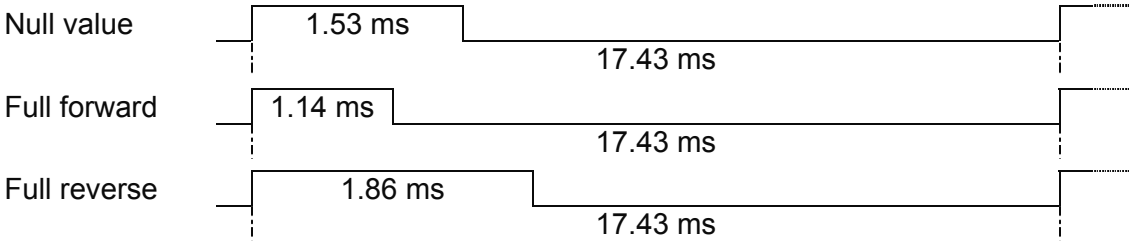


Table 3: Timings for speed controller

If a PWM signal for backward is send to the speed controller while driving in forward direction the speed controller makes the motor actively brake. Braking is not provided for backwards direction.

3.3.1 Calibration

The speed controllers can have different settings for their null value and minimum / maximum. In order to get them behave the same with one software running on the microcontroller it is necessary to calibrate them to the same settings.

Note: Calibration is done on shipped cars. Only if a new speed controller is used calibration has to be done again.

Calibration of the steering servo can be achieved by the following procedure:

1. Power on the system
2. While the null value signal is applied to the steering servo press the SET button on the speed controller for at least 0.5 seconds. The controller will beep once and the LED should flash in a short interval.
3. Output the full forward signal and press the SET button. The speed controller will beep and the LED should flash twice in short intervals.
4. Output the full reverse signal and press the SET button again. The speed controller will beep and the LED will stop blinking.

3.4 Steering servo

The servo is interfaced with a three wire cable. The red wire is for a 5 V to 6 V supply, black should be connected to ground and white is the PWM input signal. PWM period is 17.43 ms, duty cycle for neutral position is 1.53 ms. Maximum position to the right has a duty cycle of 1.81 ms, maximum to the left 1.24 ms.

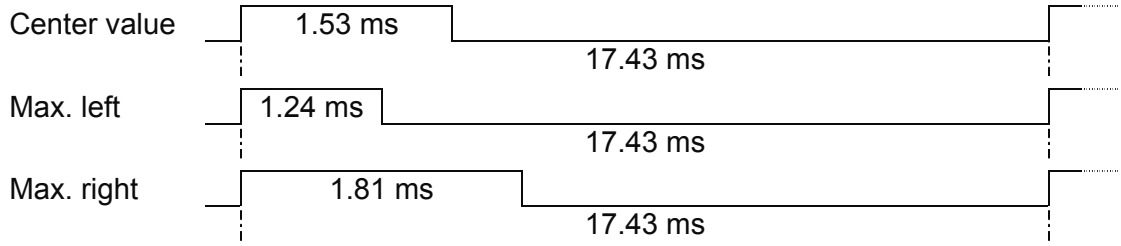


Table 4: Timings for steering servo

3.5 Microcontroller board

Different microcontrollers can be used to control the car. Currently software examples for the MB96F348HS and the MB91F467B are prepared. For those the starter kits SK-16FX-EUROSCOPE and SK-FR-144PMC-91467B respectively can be used. For further details on the starter kits or the microcontrollers used see the websites given in chapter 8.

3.5.1 SK-16FX-EUROSCOPE

Pin	Name	Used by Modules	Used by SK-16FX-EUROSCOPE
1	P02_6/A22/IN2/TTG2/TTG10		
2	P02_7/A23/IN3/TTG3/TTG11		
3	P03_0/ALE/IN4/TTG4/TTG12		
4	P03_1/RDX/IN5/TTG5/TTG13		
5	P03_2/WRLX/WRX/INT10R		
6	P03_3/WRHX		
7	P03_4/HRQ/OUT4		
8	P03_5/HAKX/OUT5		
9	P03_6/RDY/OUT6		
10	P03_7/CLK/OUT7		
11	P04_0		
12	P04_1		
13	Vcc		Vcc
14	Vss		Ground
15	C		C capacitors
16	P04_2/IN6/RX1/TTG6/TTG14		
17	P04_3/IN7/TX1/TTG7/TTG15		
18	P04_4/SDA0/FRCK0		
19	P05_5/SCL0/FRCK1		
20	P04_6/SDA1		
21	P04_7/SCL1		
22	P05_0/AN8/ALARM0/SIN2	US front (SDO)	
23	P05_1/AN9/ALARM1/SOT2	US front (SDI)	
24	P05_2/AN10/SCK2		
25	P05_3/AN11/TIN3	Line sensor 0	
26	P05_4/AN12/TOT3/TIN2R	Line sensor 1	
27	P05_5/AN13/INT0R/NMIR	Line sensor 2	
28	P05_6/AN14/INT4R	Line sensor 3	
29	P05_7/AN15/INT5R	Line sensor 4	
30	AVcc		Vcc
31	AVRH		Vcc
32	AVRL		Ground
33	AVss		Ground
34	P06_0/AN0/PPG0	Servo	
35	P06_1/AN1/PPG1	Motor	
36	P06_2/AN2/PPG2	IR front (TX)	
37	P06_3/AN3/PPG3	IR back (TX)	
38	P06_4/AN4/PPG4	Line sensor PPG	
39	P06_5/AN5/PPG5		
40	P06_6/AN6/PPG6		
41	P06_7/AN7/PPG7		

Pin	Name	Used by Modules	Used by SK-16FX-EUROSCOPE
42	Vss		Ground
43	P07_0/AN16/INT0/NMI		Button “INT0/NMI”
44	P07_1/AN17/INT1		Button “INT1”
45	P07_2/AN18/INT2		
46	P07_3/AN19/INT3		
47	P07_4/AN20/INT4		
48	P07_5/AN21/INT5		
49	MD2		Ground
50	MD1		Vcc
51	MD0		Mode switch S1
52	RSTX		Key button “Reset”
53	P07_6/AN22/INT6/SOT9R	Do not connect!*	
54	P07_7/AN23/INT7/SIN9R	IR front (RX)	
55	P08_0/TIN0/CKOTX0/ADTG/INT12R	IR front (Sel. A)	
56	P08_1/TOT0/CKOT0/INT13R	IR front (Sel. B)	
57	P08_2/SIN0/TIN2/INT14R		UART0 (RXD)
58	P08_3/SOT0/TOT2		UART0 (TXD)
59	P08_4/SCK0/INT15R	Jumper to pin 61*	
60	P08_5/SIN1/INT1R	IR back (RX)	UART1 (RXD)
61	P08_6/SOT1	Jumper to pin 59*	UART1 (TXD)
62	P08_7/SCK1		
63	Vcc		Vcc
64	Vss		Ground
65	P09_0/PPG8/UBX		SEG1-A
66	P09_1/PPG9/LBX		SEG1-B
67	P09_2/PPG10/CS5		SEG1-C
68	P09_3/PPG11/CS4		SEG1-D
69	P09_4/OUT0/CS3		SEG1-E
70	P09_5/OUT1/CS2		SEG1-F
71	P09_6/OUT2/CS1		SEG1-G
72	P09_7/OUT3/CS0		SEG1-DP
73	P10_0/RX0/INT8R		CAN0 (RX)
74	P10_1/TX0		CAN0 (TX)
75	P00_0/AD00/INT8/SCK7R	BTM-222 (Reset)	SEG2-A
76	P00_1/AD01/INT9/SOT7R	BTM-222 (SDI)	SEG2-B
77	P00_2/AD02/INT10/SIN7R	BTM-222 (SDO)	SEG2-C
78	P00_3/AD03/INT11/SCK8R	RFM12 (SCK)	SEG2-D
79	P00_4/AD04/INT12/SOT8R	RFM12 (SDI)	SEG2-E
80	P00_5/AD05/INT13/SIN8R	RFM12 (SDO)	SEG2-F
81	P00_6/AD06/INT14	RFM12 (SEL)	SEG2-G
82	P00_7/AD07/INT15		SEG2-DP
83	P01_0/AD08/CKOT1/TIN1		
84	P01_1/AD09/CKOTX1/TOT1		
85	P01_2/AD10/INT11R/SIN3	US back (SDO)	
86	P01_3/AD11/SOT3	US back (SDI)	
87	P01_4/AD12/SCK3		
88	Vcc		Vcc
89	Vss		Ground
90	X1		4 MHz crystal
91	X0		4 MHz crystal

Pin	Name	Used by Modules	Used by SK-16FX-EUROSCOPE
92	P01_5/AD13/INT7R/SIN2R		
93	P01_6/AD14/SOT2R		
94	P01_7/AD15/SCK2R		
95	P02_0/A16/PPG12	IR back (Sel. A)	
96	P02_1/A17/PPG13	IR back (Sel. B)	
97	P02_2/A18/PPG14		
98	P02_3/A19/PPG15		
99	P02_4/A20/TTG8/IN0		
100	P02_5/A21/TTG9/TTG1/IN1/ADTGR		

Table 5: Pinout of MB96F348HS

*: See chapter 3.5.3 - SOTx -> INTx connections

For more information on the functions of the pins refer to the datasheet and hardware manual of the microcontroller.

3.5.2 SK-FR-144PMC-91467B

Pin	Name	Used by Modules	Used by SK-FR-144PMC-91467B
1	VSS5		Ground
2	P27_6/AN22		
3	P27_7/AN23		
4	P26_0/AN24		SEG1-A
5	P26_1/AN25		SEG1-B
6	P26_2/AN26		SEG1-C
7	P26_3/AN27		SEG1-D
8	P26_4/AN28		SEG1-E
9	P26_5/AN29		SEG1-F
10	P26_6/AN30		SEG1-G
11	P26_7/AN31		SEG1-DP
12	P24_4/INT4		
13	P24_5/INT5		
14	P24_6/INT6		
15	P24_7/INT7	Jumper to pin 17*	
16	P21_0/SIN0	IR back (RX)	
17	P21_1/SOT0	Jumper to pin 15*	
18	VDD35		VDD35
19	VSS5		Ground
20	P14_4/ICU4/TIN4/TTG12/4		
21	P14_5/ICU5/TIN5/TTG13/5		
22	P14_6/ICU6/TIN6/TTG14/6		
23	P14_7/ICU7/TIN7/TTG15/7		
24	P15_4/OCU4/TOT4		
25	P15_5/OCU5/TOT5		
26	P15_6/OCU6/TOT6		
27	P15_7/OCU7/TOT7		
28	P17_0/PPG0		SEG2-A
29	P17_1/PPG1		SEG2-B
30	P17_2/PPG2		SEG2-C

Pin	Name	Used by	
		Modules	SK-FR-144PMC-91467B
31	P17_3/PPG3		SEG2-D
32	P17_4/PPG4		SEG2-E
33	P17_5/PPG5		SEG2-F
34	P17_6/PPG6		SEG2-G
35	P17_7/PPG7		SEG2-DP
36	VDD35		VDD35
37	VSS5		Ground
38	P20_0/SIN2/AIN0	US front (SDO)	
39	P20_1/SOT2/BIN0	US front (SDI)	
40	P20_2/SCK2/ZIN0/CK2		
41	P20_4/SIN3/AIN1	US back (SDO)	
42	P20_5/SOT3/BIN1	US back (SDI)	
43	P20_6/SCK3/ZIN1/CK3		
44	P24_0/INT0		
45	P24_1/INT1		
46	P23_0/RX0/INT8		CAN0 (RX)
47	P23_1/TX0		CAN0 (RX)
48	P23_2/RX1/INT9		
49	P23_3/TX1		
50	P23_4/RX2/INT10		
51	P23_5/TX2		
52	P23_6/RX3/INT11		
53	P23_7/TX3		
54	VDD5		VDD5
55	VSS5		Ground
56	P22_0/RX4/INT12		Key button ‘SW2’
57	P22_1/TX4		
58	P22_2/RX5/INT13		Key button ‘SW3’
59	P22_3/TX5		
60	P22_4/SDA0/INT14		
61	P22_5/SCL0	RFM12 (SEL)	
62	P22_6/SDA1/INT15		
63	P22_7/SCL1		
64	P16_0/PPG8	Servo	
65	P16_1/PPG9	Motor	
66	P16_2/PPG10	IR front (TX)	
67	P16_3/PPG11	IR back (TX)	
68	P16_4/PPG12/SGA		
69	P16_5/PPG13/SGO		
70	P16_6/PPG14	Line sensor PPG	
71	P16_7/PPG15/ATGX		
72	VDD5		VDD5
73	VSS5		Ground
74	MD_0		Ground
75	MD_1		Ground
76	MD_2		Ground
77	MONCLK		
78	MD_3		Ground
79	X1		4 MHz crystal
80	X0		4 MHz crystal
81	VSS5		Ground

Pin	Name	Used by	
		Modules	SK-FR-144PMC-91467B
82	X0A		Ground
83	X1A		
84	INITX		Key button ‘Reset’
85	NMIX		VDD5
86	VSS5		Ground
87	VCC18C		C23||C24
88	VDD5R		VDD5R
89	VDD5R		VDD5R
90	VDD5		VDD5
91	VSS5		Ground
92	P19_0/SIN4	BTM-222 (SDO)	UART4 (RXD)
93	P19_1/SOT4	BTM-222 (SDI)	UART4 (TXD)
94	P19_2/SCK4/CK4	BTM-222 (Reset)	
95	P19_4/SIN5		UART5 (RXD)
96	P19_5/SOT5		UART5 (TXD)
97	P19_6/SCK5/CK5		
98	P18_0/SIN6	IR front (RX)	
99	P18_1/SOT6	Cable to pin 128*	
100	P18_2/SCK6/CK6		
101	P18_4/SIN7	RFM12 (SDO)	
102	P18_5/SOT7	RFM12 (SDI)	
103	P18_6/SCK7/CK7	RFM12 (SCK)	
104	ALARM_0		
105	AVSS		Ground
106	AVRH5		VDD5
107	AVCC5		VDD5
108	VDD5		VDD5
109	VSS5		Ground
110	P29_0/AN0	IR front (Sel. A)	
111	P29_1/AN1	IR front (Sel. B)	
112	P29_2/AN2	Line sensor 0	
113	P29_3/AN3	Line sensor 1	
114	P29_4/AN4	Line sensor 2	
115	P29_5/AN5	Line sensor 3	
116	P29_6/AN6	Line sensor 4	
117	P29_7/AN7		
118	P28_0/AN8	IR back (Sel. A)	
119	P28_1/AN9	IR back (Sel. B)	
120	P28_2/AN10		
121	P28_3/AN11		
122	P28_4/AN12		
123	P28_5/AN13		
124	P28_6/AN14		
125	P28_7/AN15		
126	VDD5		VDD5
127	VSS5		Ground
128	P24_2/INT2	Cable to pin 99*	
129	P24_3/INT3		
130	P14_0/ICU0/TIN0/TTG8/0		
131	P14_1/ICU1/TIN1/TTG9/1		
132	P14_2/ICU2/TIN2/TTG10/2		

Pin	Name	Used by	
		Modules	SK-FR-144PMC-91467B
133	P14_3/ICU3/TIN3/TTG11/3		
134	P15_0/OCU0/TOT0		
135	P15_1/OCU1/TOT1		
136	P15_2/OCU2/TOT2		
137	P15_3/OCU3/TOT3		
138	P27_0/AN16		
139	P27_1/AN17		
140	P27_2/AN18		
141	P27_3/AN19		
142	P27_4/AN20		
143	P27_5/AN21		
144	VDD35		VDD35

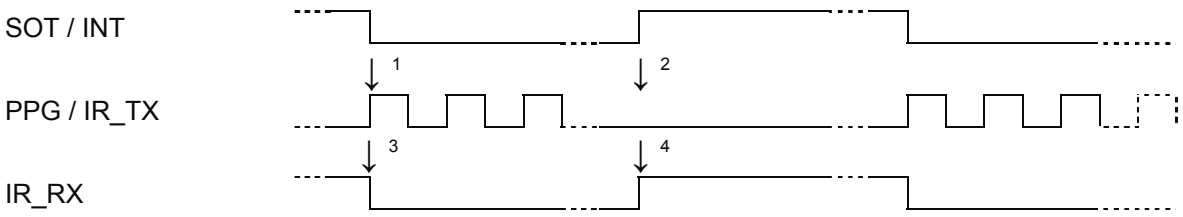
Table 6: Pinout of MB91F467B

*: See chapter 3.5.3 - SOTx -> INTx connections

For more information on the functions of the pins refer to the datasheet and hardware manual of the microcontroller.

3.5.3 SOTx -> INTx connections

In the default setup the signal for infrared transmission is generated by the USART. Since the infrared detector requires a carrier frequency of 36 kHz with a data signal modulated at an at least 10 times lower frequency the USART only generates the data output. This data is fed into an interrupt pin which enables or disables a 36 kHz PPG output depending on the state of the USART output value. Therefore the SOTx pins used for infrared have to be connected to external interrupt pins.



[1]: Interrupt caused by SOT falling edge starts PPG output
[2]: Interrupt caused by SOT rising edge disables PPG output
[3]: IR receiver detects 36 kHz carrier and sets its output low
[4]: No carrier detected by the IR receiver makes its output go high
Note: The signal output by the IR receiver has a time offset related to the PPG output because the receiver has to detect a few 36 kHz pulses before it recognizes the signal.

Table 7: IR communication outline

3.6 ADA-US-IR-RFM-BT

This is the main peripheral module on the Student-Model-Car. It is built of five independent sub-boards which can be split up into single PCBs:

- ADA-INFRARED (ADA-IR1 / ADA-IR3)
- ADA-RFM12
- SK-95F284-US-IR
- ADA-BT

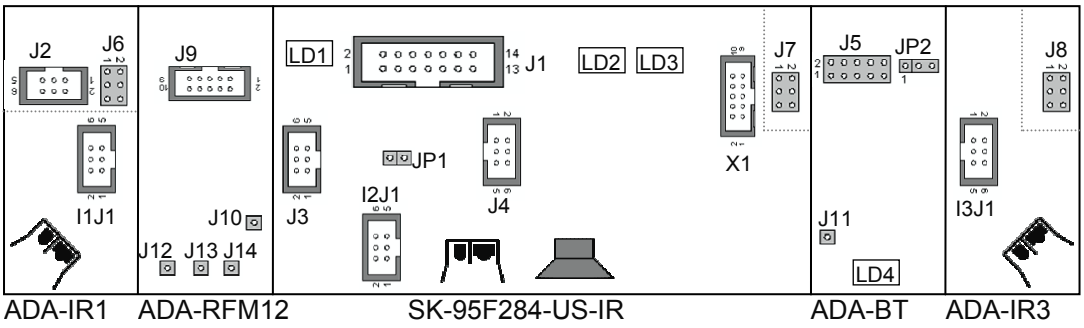


Figure 6: Outline of ADA-US-IR-RFM-BT

It features an ultrasonic distance measurement unit based on a MaxBotix MaxSonar-UT ultrasonic transducer and an analogue amplification circuitry. Conversion and evaluation of the data is handled by an 8 bit microcontroller MB95F284KPF which also allows access to four general purpose inputs / outputs. Two of those GPIOs have LEDs connected.

Furthermore this board has three ADA-INFRARED units which are connected to a multiplexer. The fourth ADA-INFRARED unit from the side of the car can be connected to the multiplexer by a four wire ribbon cable.

Power for this board is regulated by an adjustable linear voltage regulator LT1763 which has a very low dropout voltage of about 300 mV, overcurrent and overtemperature protection. The supply of the board is protected against short circuits by a self-resetting fuse.

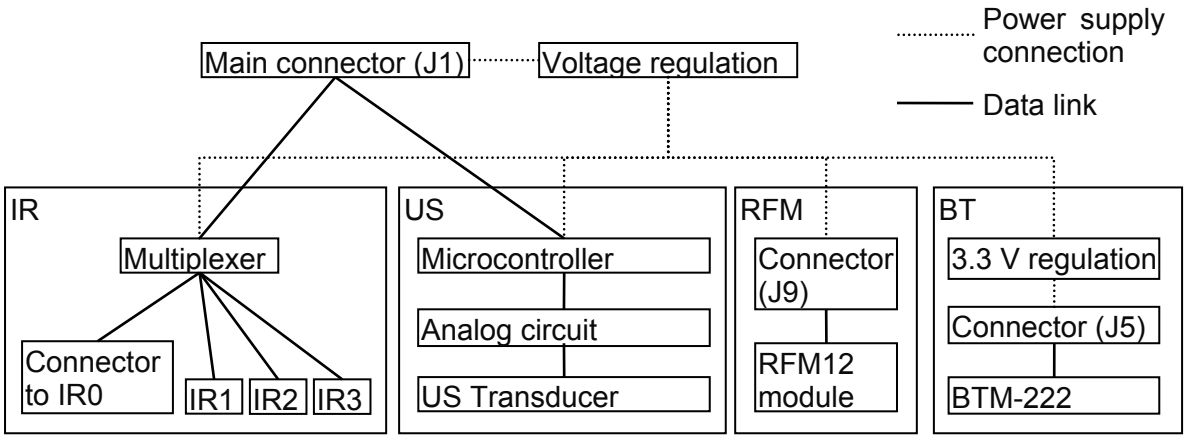


Figure 7: Relationship of modules on ADA-US-IR-RFM-BT

3.6.1 Primary interface connector (J1)

The primary interface to the microcontroller is J1 which connects to the multiplexer for the infrared units and the ultrasonic module.

Pin	Name	Description
1	VIN	Battery power supply

Pin	Name	Description
2	GND	Ground
3	IR_A	Infrared multiplexer channel select A
4	IR_B	Infrared multiplexer channel select B
5	IR_RX	Infrared receiver output of multiplexer
6	IR_TX	Infrared transmitter input to multiplexer
7	US_SIN	Ultrasonic MCU serial data input pin
8	US_SOT	Ultrasonic MCU serial data output pin
9	US_PG1	Ultrasonic MCU GPIO pin PG1
10	US_PG2	Ultrasonic MCU GPIO pin PG2
11	US_P01	Ultrasonic MCU GPIO pin P01
12	US_P02	Ultrasonic MCU GPIO pin P02
13		Not connected
14		Not connected

Table 8: J1 – Main interface connector of ADA-US-IR-RFM-BT

The voltage on the VIN pin should not exceed 10 V as the power dissipation could damage the regulator. Minimum operating voltage depends on the target board voltage level selected by JP1.

3.6.2 Power level select (JP1)

Jumper	Setting	Description	Starter kit
JP1	Open	Board operates at 5 V	SK-16FX-EUROSCOPE: JP10: 1-2 SK-FR-144PMC-91467B: JP10: 1-2
	Closed	Board operates at 3.3 V	SK-16FX-EUROSCOPE: JP10: 2-3 SK-FR-144PMC-91467B: JP10: 2-3

Table 9: JP1 board voltage selection

Note: The selected voltage has to be the same as for the microcontroller! See 3.2 - Power supply.

3.6.3 Power supply state LED (LD1)

LD1 (yellow) is lit when the board is powered up.

3.6.4 Side infrared module connector (J2)

Connector J2 is used to connect the standalone ADA-INFRARED module of the side of the car. Its pinout is the same as J1 of ADA-INFRARED.

3.6.5 Stacked PCB connectors (J6, J7, J8)

Connectors J6, J7 and J8 can be used to supply power to a stacked PCB on top of this PCB.

Pin	Name	Description
1, 2	VBAT	Protected (self resetting fuse, diode) battery power supply
3, 4	VCC	5 V / 3.3 V regulated power
5, 6	GND	Ground

Table 10: J6, J7, J8 connectors for stacked PCB

Note: Maximum current from one ADA-US-IR-RFM-BT modules' VCC line may not exceed 200 mA. Current taken from VBAT pins may not exceed 400 mA.

3.6.6 SK-95F284-US-IR

The infrared module on this board (IR2) is connected to the *Infrared channel multiplexer*. Also refer to chapter 3.7 - ADA-INFRARED for details on the signals.

The ultrasonic module is interfaced through pins 7 to 12 on J1. Control of the microcontroller operation is sent over an USART channel. The default firmware provides a command based control interface which is described in 5.1 - Interface. The four GPIO pins can be used to exchange status information from the ultrasonic controller to the main microcontroller. Furthermore the LEDs LD2 (green) and LD3 (red) are connected to the ultrasonic microcontroller pins PG1 and PG2 and can be controlled through the USART command interface.

Connector X1 provides the interface for the external programmer/debugger unit MB2146-08-E. To program the flash or debug the application first connect the debugger to X1, connect the debugger to the USB port of the PC and finally power up the microcontroller on this module.

3.6.7 Infrared channel multiplexer

In order to save resources on the microcontroller all three infrared units of one main sensor module and one side module are connected to one USART channel on the microcontroller through the multiplexer. The channel select pins and the infrared data interface pins can be found on J1 pins 3 to 6. Pins A and B select the channel where BA can be handled as a 2 bit number indicating the selected IR module.

B (JP1_4)	A (JP1_3)	IR module
0	0	0 – Side module
0	1	1 – Right module
1	0	2 – Middle module
1	1	3 – Left module

Table 11: IR multiplexer channel selection

3.6.8 ADA-IR1 & ADA-IR3

These modules are functionally the same as the standalone ADA-INFRARED module. The optionally mounted I1J1 / I3J1 can be used to connect these modules to the multiplexer if the board is split into its individual sub-boards. In that case I1J1 has to be connected to J3 and I3J1 to J4. Those connectors have the same pinout as J1 of ADA-INFRARED.

Refer to 3.6.7 - Infrared channel multiplexer for details on how to interface these modules and chapter 3.7 - ADA-INFRARED for further details on the signals.

3.6.9 ADA-RFM12

If a RFM12 wireless module is mounted it can be interfaced with SPI on J9:

Pin	Name	Description
1	VCC	Power supply
2	GND	Ground
3	SDI	Serial data input to RFM12
4	SDO	Serial data output from RFM12
5	SCK	Serial clock input to RFM12
6	xSEL	Chip select input (low active)
7	xIRQ	Interrupt output (low active)
8	xRES	Reset input (low active)

Pin	Name	Description
9		Not connected
10		Not connected

Table 12: J9 – RFM12 connector

Additional signals of the RFM12 module are provided on test points. For more details on those signals see the RFM12 manual.

Pin	Name	Description
J12	DCLK/CFIL/FFIT	Clock output / external filter capacitor for analogue mode / FIFO interrupts
J13	CLK	Clock output
J14	xINT/VDI	Interrupt input / Valid data indicator
J10	ANT	Antenna output. A 8.6 cm (λ/4) antenna should be connected

Table 13: RFM12 test points

3.6.10 ADA-BT

The Bluetooth module operates at 3.3 V. Because of this a 3.3 V regulator for the Bluetooth module is required if the board operates at 5 V. JP2 selects whether the board voltage is directly used or the 3.3 V regulator for the Bluetooth module is used.

Jumper	Setting	Description
JP2	1-2	Board operates at 5 V
	2-3	Board operates at 3.3 V

Table 14: JP2 – Source voltage selection for Bluetooth

The interface of the Bluetooth module is a normal UART protocol, control of the reset signal is optional.

Pin	Name	Description
1	VCC	Power supply
2	GND	Ground
3	SDI	Serial data input to BTM-222
4	SDO	Serial data output from BTM-222
5		Not connected
6		Not connected
7		Not connected
8	xRES	Reset input (low active)
9		Not connected
10		Not connected

Table 15: J5 – Bluetooth connector

LD4 signals incoming data over an active Bluetooth connection.

Antenna connection is provided through a test point:

Pin	Name	Description
J11	ANT	Antenna output. A 3.1 cm (λ/4) antenna should be connected

Table 16: BTM-222 Test points

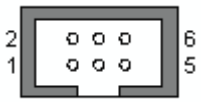
3.7 ADA-INFRARED

This board allows infrared object detection and short range infrared communications. It can be interfaced directly although in the normal setup it is connected to multiplexer of the main sensor module to save microcontroller resources. The data is modulated on a 36 kHz carrier

frequency and output to an infrared LED. Signal reception is done with a SFH5110 with integrated demodulator. The output signal of the receiver is fed into a low-pass filter with a cut-off frequency of approximately 10.5 kHz.

Note: The infrared receiver on these modules requires an operating voltage of 5 V.

The interface of this module is provided through the header J1.



Pin	Name	Description
1	VCC	Power supply (5 V)
2	GND	Ground
3	RX	Receiver output
4	TX	Transmitter input
5		Not connected
6		Not connected

Table 17: J1 – ADA-INFRARED connector

The receiver is designed for 36 kHz infrared signals; length of one burst should be at least 10 cycles for the receiver to detect the transmitted signal. The infrared LED is driven active high whereas the receiver outputs a low signal when it detects a 36 kHz infrared signal.

See chapter 3.5.3 - SOTx -> INTx connections for an overview of the signal relations.

3.8 ADA-IR-LINE-SENSOR

3.8.1 Installation instruction

The IR line sensor has to be mounted under the front bumper. Therefore the mounting plate needs to be removed:

1. Remove the screws
2. Remove the ADA-US-IR-RFM-BT modules on both sides
3. Remove the brackets under the ADA-US-IR-RFM-BT modules
4. Remove the mounting plate carefully because connections cables are still attached (note: cables don’t have to be removed)
5. Remove the screws of the front bumper and the front bumper
6. Cut out a rectangle (24mm x 8mm) of the bumper for the connectors (see figure 8)
7. Install the IR line sensor (figure 8)
8. Reassemble the car

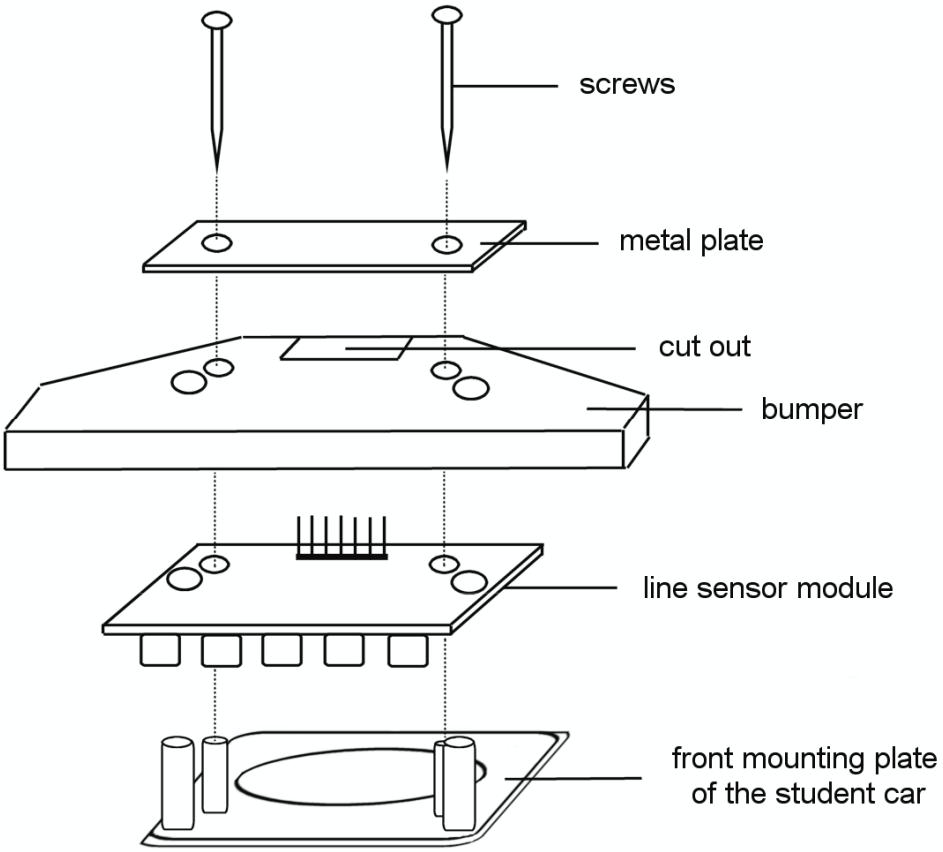


Figure 8: Installation instruction of IR line sensor

The front bumper of the student car should look like this:

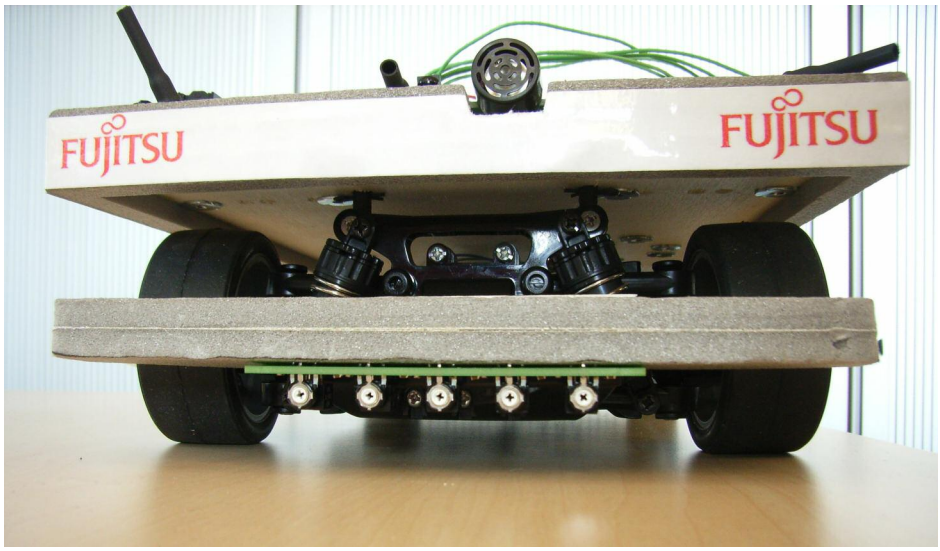


Figure 9: IR line sensor attachment

3.8.2 Features and usage

The IR line sensor is able to detect the location of a line below the sensors. The line should have a width of about 1.0 – 1.5 cm. The line also should be very bright if one uses a dark underground or vice versa.

Example:

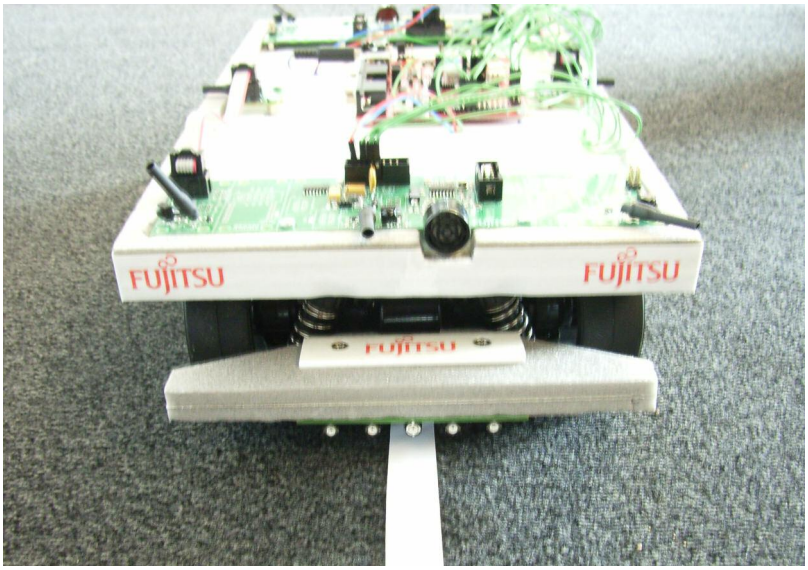


Figure 10: Example: line detection

All sensor ADC values during the test conditions (wooden desk) added up to about 700 of 1024 maximum. By planting a white line below one of the sensors the sensor value drop to about 20.

If the sensor values have big differences in their range caused of part variances the potentiometers in front of the sensors can be used in order to adjust the output values so that all five sensors nearly have the same level.

The PPG / PWM functionality can be used in order to realize a power management by adjusting the duty cycle of the PPG signal.

3.8.3 Pin description

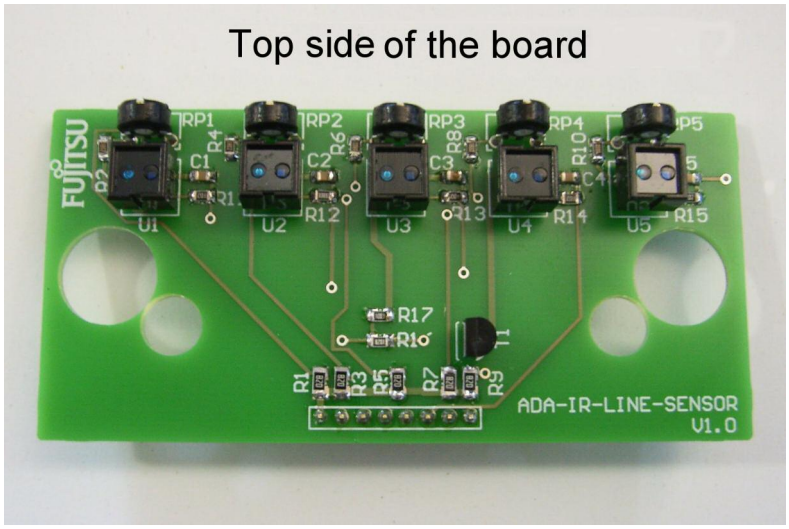
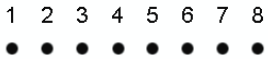


Figure 11: ADA-IR-LINE-SENSOR



Pin	Name	Description
1	VCC	Power supply (VCC)
2	PPG/PWM	Programmable Pulse Generator / Pulse Width Modulation (Power management)
3	Sensor 0	Sensor output
4	Sensor 1	Sensor output
5	Sensor 2	Sensor output
6	Sensor 3	Sensor output
7	Sensor 4	Sensor output
8	GND	Ground (GND)

Table 18: ADA-IR-LINE-SENSOR connector

Use ADC channels to readout sensor values.

Use a PPG channel for a PPG / PWM signal (power management) or VCC for permanent current supply.

4 Getting started

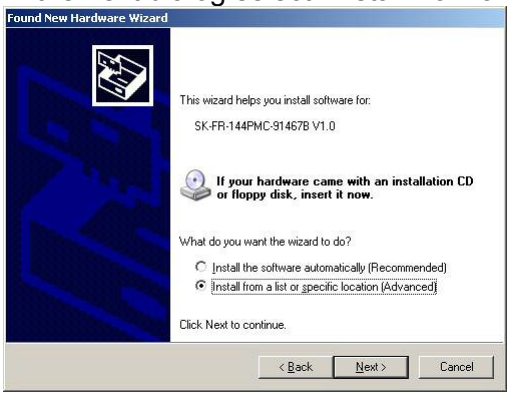
This chapter will describe how to download and use the test software for the Student-Model-Car.

4.1 Installing the USB driver

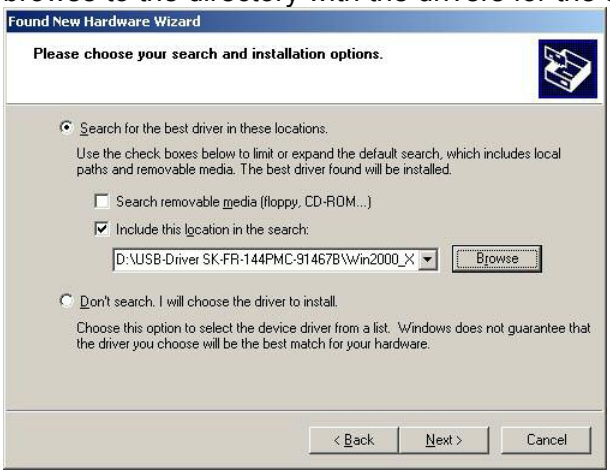
The first time the starter kit on the Student-Model-Car is connected to a Windows PC through USB the driver for the USB to serial converter has to be installed.

1. Connect the starter kit with an USB A to USB B cable
2. Windows will pop up a new hardware wizard. Do not connect to Windows Update.

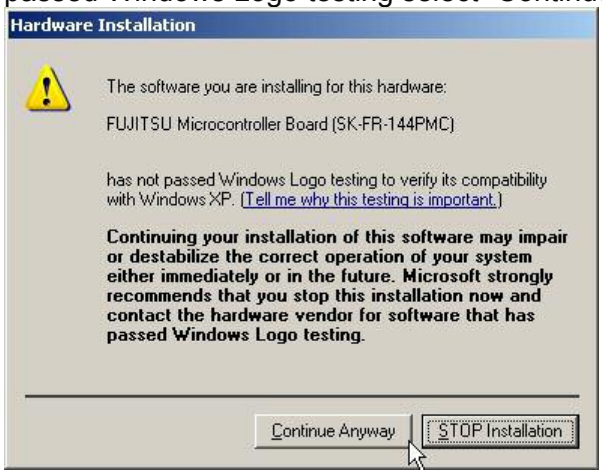
3. In the next dialog select “Install from a list or specific location (Advanced)”.



4. Select “Search for the best driver in these locations”, uncheck “Search removable media (floppy, CD-ROM...)” and select “Include this location in the search”. Then browse to the directory with the drivers for the starter kit.



5. When asked if the installation should be continued even though the drivers have not passed Windows Logo testing select “Continue Anyway”.



6. Press finish when the installation of the driver is complete.

Windows will now show a second driver installation dialog for an “USB Serial Port”. Repeat the instructions for this device.

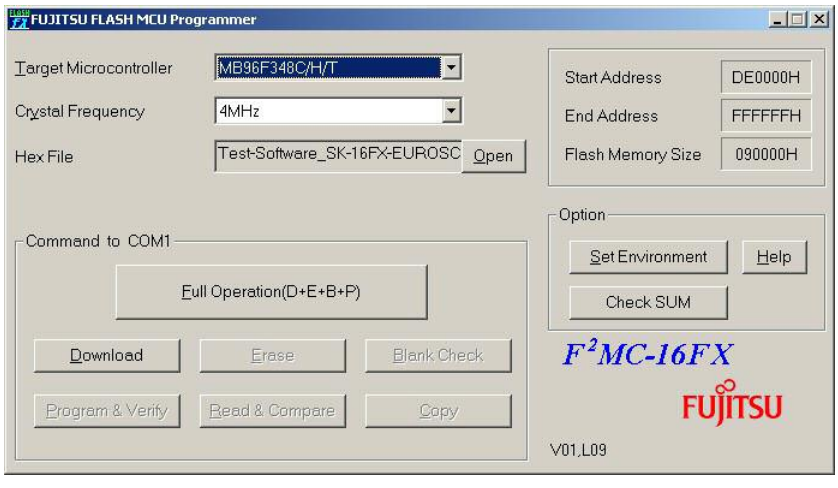
Note: Always use the same USB port with the same starter kit. Otherwise Windows will ask for another driver installation and will assign a different COM port number.

4.2 Downloading the firmware to flash

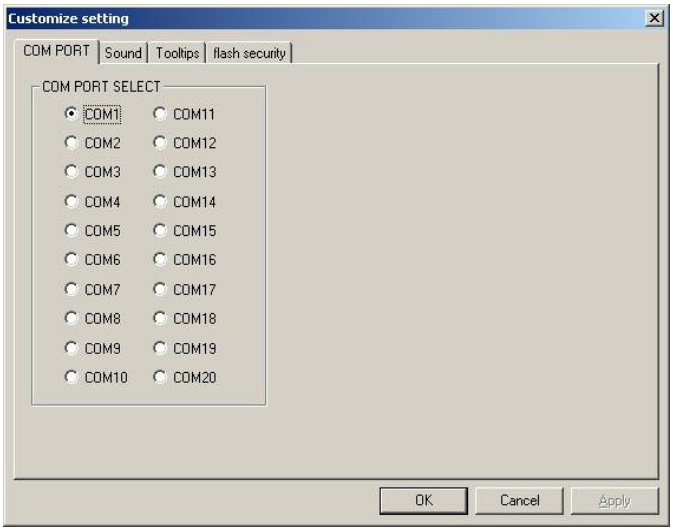
Depending on the microcontroller series respectively the starter kit used a different programming software is provided by Fujitsu.

4.2.1 MB96340 series / SK-16FX-EUROSCOPE

1. Start the Fujitsu Flash MCU Programmer for 16FX.
2. Select the MB96F348C/H/T target microcontroller and 4 MHz crystal frequency. Also select the firmware hex file.

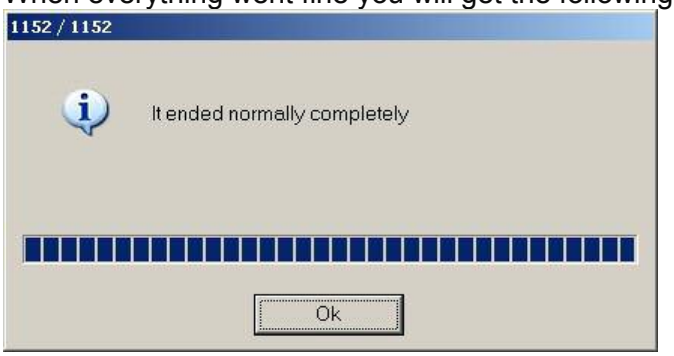


3. Open the settings by clicking on “Set Environment” and select the COM port the starter kit is connected to.



4. On the starter kit select the programming mode by setting the switch S1 to “PROG”.
5. Start the flash download operation by clicking on “Full operation(D+E+B+P)”. When asked to press the reset button on the starter kit and click OK in the flash programmer.

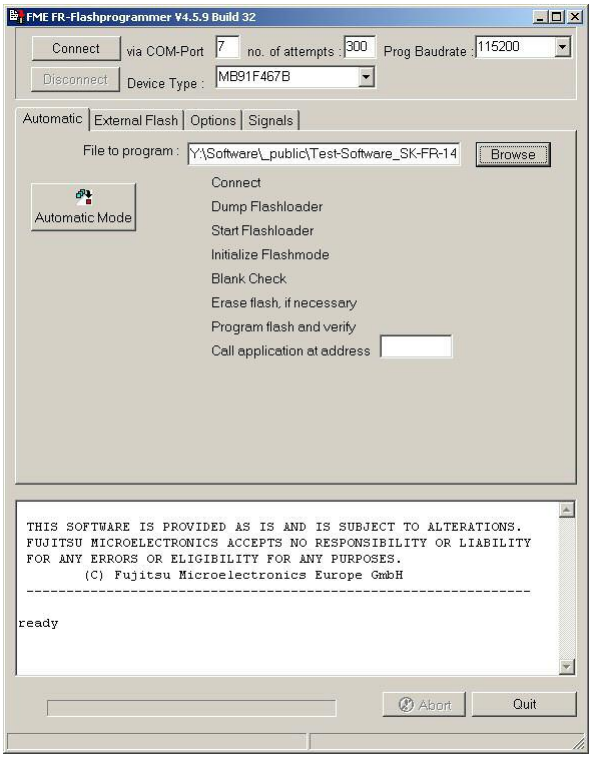
6. When everything went fine you will get the following message:



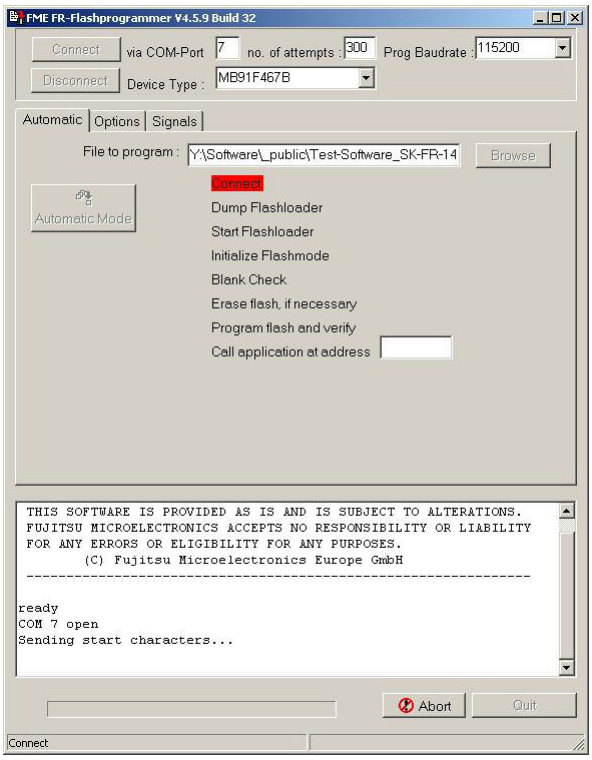
7. Select the run mode on the starter kit by setting S1 to “RUN” and reset the microcontroller to start the application.

4.2.2 MB91460 series / SK-FR-144PMC-91467B

1. Start the Fujitsu FME FR-Flash Programmer.
2. Set the COM port number to the virtual COM port of the starter kit, the programming baud rate to 115200 and select the MB91F467B device. Also select the firmware hex file in the automatic tab.

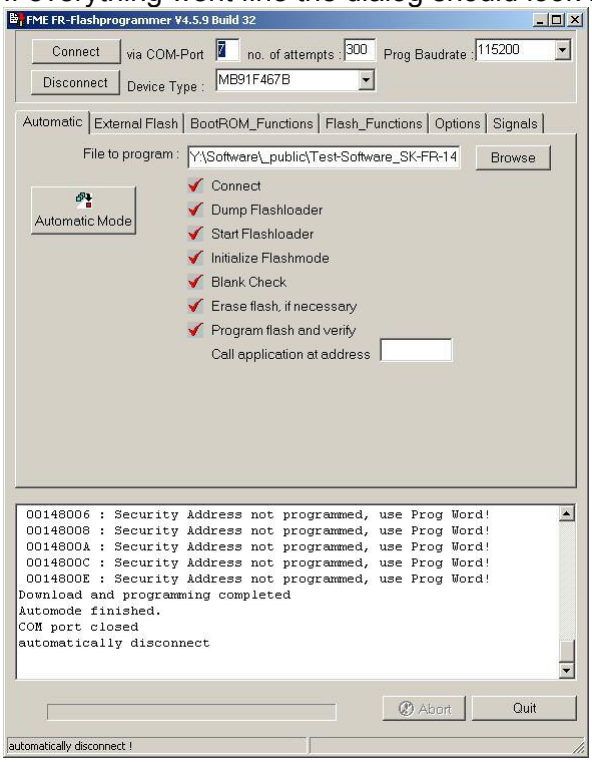


3. Start the automatic mode.



4. Reset the microcontroller by pressing the blue button on the starter kit. Flash downloading will start.

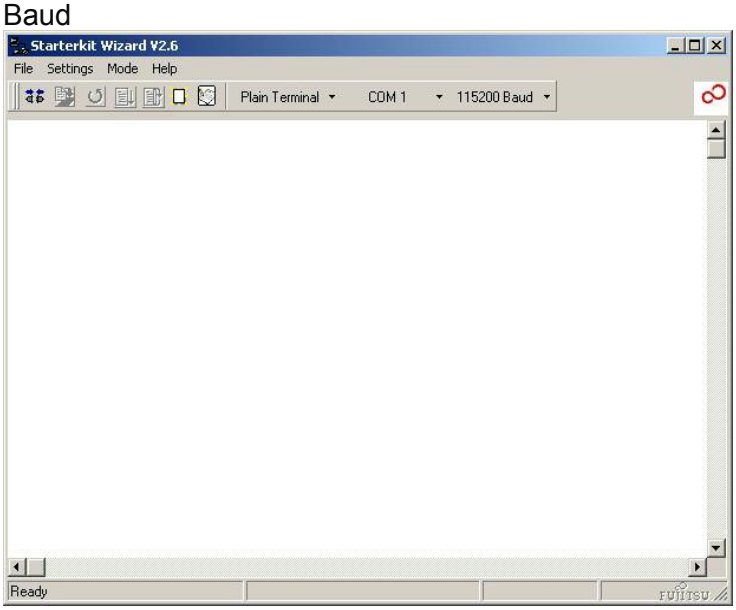
5. If everything went fine the dialog should look like this:



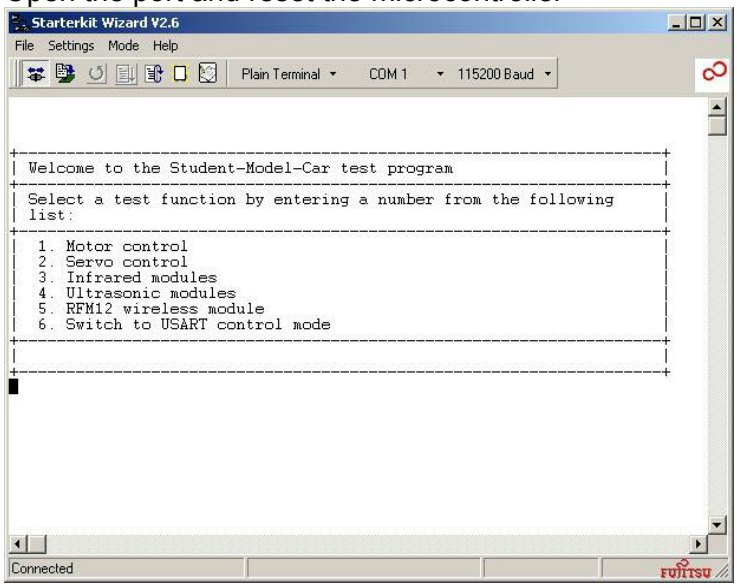
6. Press reset on the starter kit to start the application.

4.3 Usage of the test program

1. Connect the interface port of the starter kit to the PC
   Depending on the starter kit the terminal interface of the test program is on different ports:
   a. SK-16FX-EUROSCOPE: USB (X5)
   b. SK-FR-144PMC-91467B: RS232 (X4)
2. Open SK Wizard
3. Select the COM port the starter kit is connected to and set the baud rate to 115200 Baud



4. Open the port and reset the microcontroller



5. Select a test from the list by pressing its number. Further instructions will be given in the test routines.

The provided tests cover:

1. Motor control:
   Tests the operation of the motor module by sending appropriate PPG signals to the speed controller for forward driving, braking and backward driving.
2. Servo control:
   Sends PPG signals to the servo to move from maximum left position to maximum right position.
3. Infrared modules:
   Displays the state of infrared object detection for each sensor module.
4. Ultrasonic modules:
   Displays distances to detected objects, send calibration command and make the LEDs of the ultrasonic modules blink.
5. RFM12 wireless module:
   Send and receive messages over wireless communication.
6. USART control mode:
   In this mode the motor and servo operation can be controlled manually by sending different characters.

5 Ultrasonic module software

The preprogrammed firmware on the ultrasonic module samples the echo of the ultrasonic transmission and provides an evaluated output of that data which gives the distance to a detected object. Communication is done via a USART interface.

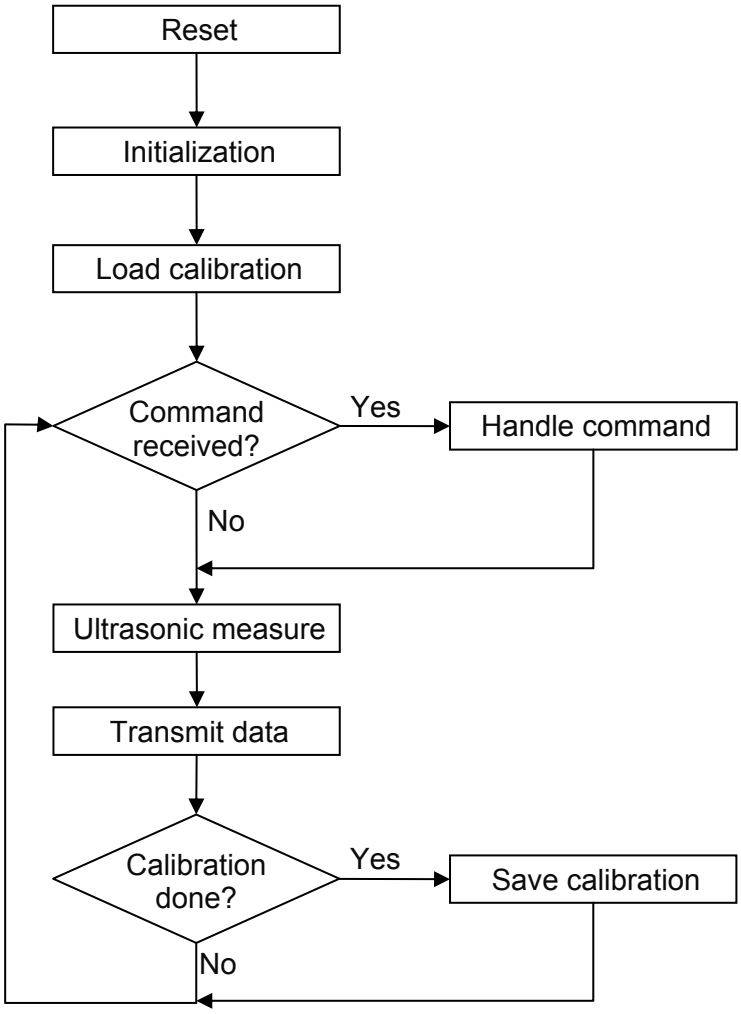


Figure 12: Flowchart of ultrasonic module firmware

The measurement routine first sends 13 ultrasonic bursts. Then it converts 8 samples of the returned signal which are averaged in each of 250 timeslots. From the averaged value the stored calibration value is subtracted. The timeslots are 128 µs each which results in about 2 cm of additional distance per timeslot until a peak is detected.

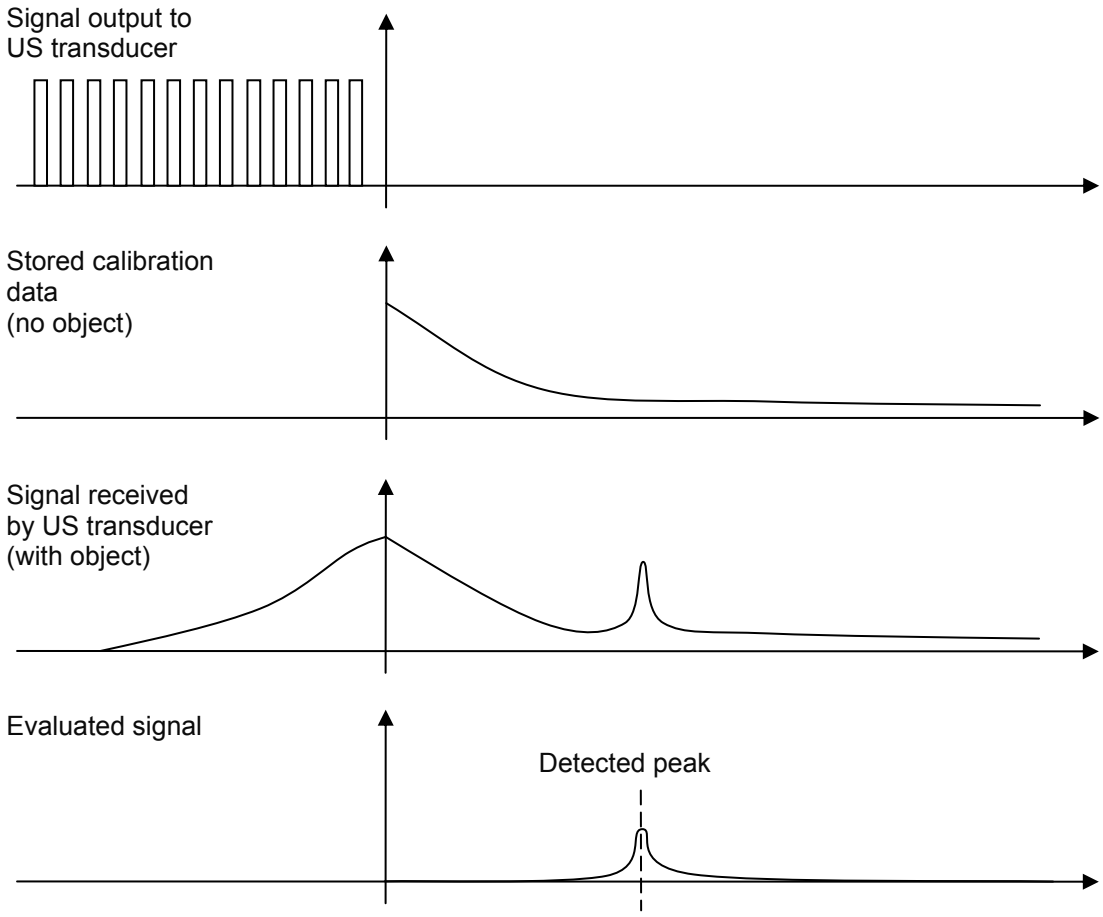


Figure 13: Ultrasonic measurement process

5.1 Interface protocol

The firmware features a small command protocol to control the behavior of the ultrasonic module. Data sent from the module is fit into packages. Each package is prefixed with a 0 value byte whereas no other 0 value will be in the data bytes of the package. The second byte in a package determines the type of the package.

Byte 0	Byte 1	Byte 2..n
0x00	Type of package	Data; length depending on type of package

Table 19: Ultrasonic module package structure

The following types of packages are defined and will be described in the following chapters:

Type value	Data length n	Description
0x01	250	Raw sample data
0x02	3	Evaluated measurement values
0x03	1	Reply to command
0x04	1	Calibration saved or unknown command

Table 20: Ultrasonic module package types

5.1.1 Raw sample data

This package is sent during each measurement run if the raw data mode is enabled (see 5.2.1 - Raw data mode). It represents the averaged sample values of the analog to digital converter per timeslot. Each of the bytes is one average value.

Byte 0	Byte 1	Byte 2..251
0x00	0x01	Averaged AD sample values

The first value is for timeslot 0 which starts at about 581 us. The value for timeslot 0 is always 1.

5.1.2 Evaluated measurement values

This package is sent after each measurement run if raw mode is disabled. It contains the timeslot index and the distance in cm where an object was detected.

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
0x00	0x02	Index of timeslot	Upper 7 bits of distance	Lower 7 bits of distance

The bytes for the distance in cm are OR'ed with 0x80 and the lower 7 bits of each byte contain the actual value. Thus the distance in cm can be taken from bytes 3 and 4 by:

Distance [cm] = ((Byte3 and 0x7f) << 7) or (Byte4 and 0x7f)

5.1.3 Reply to command

If a command was received and could be handled correctly an reply will be generated. This reply contains the original command. If the command was a read command the value fields will be filled by the current internal values of the running software.

Byte 0	Byte 1	Byte 2
0x00	0x03	Handled command

5.1.4 Calibration saved or unknown command

When a calibrate command was received and calibration is done or when an unknown command was received this package will be sent.

Byte 0	Byte 1	Byte 2
0x00	0x04	0x10 – Calibration done 0x11 – Unknown command

5.2 Commands

Commands are single bytes sent to the ultrasonic module. Since only one command is handled after each measurement run only one command may be sent until the reply is received. Alternatively a wait time may be used to make sure that the command has been processed.

Commands have the following format:

Bit 7..4	Bit 3	Bit 2..0
Parameter data	Read (0) / Write (1)	Command id

Table 21: Structure of ultrasonic commands

The commands defined are as following:

Command id	Description	Remarks
0	Raw data mode	
1	Pause between measurement runs in units of 50 ms	
2	Calibrate to current measurement values	Write only
3	Clear calibration	Write only
4	Pin data (PDR register)	
5	Pin configuration (DDR and PUL registers)	
6	Reserved	
7	Reserved	

Table 22: Ultrasonic commands

5.2.1 Raw data mode

This command allows selection of raw data mode or evaluated data mode.

Bit 7	6	5	4	3	2	1	0
1	-	-	Mode	R/W	0	0	0

Mode bit	Mode description
0	Evaluated data mode
1	Raw data mode

5.2.2 Pause

A pause between each measurement run can be set from 0 to 750 ms.

Bit 7	6	5	4	3	2	1	0
Number of 50 ms pause states				R/W	0	0	1

Default value is 0 ms delay.

5.2.3 Calibrate

Stores the current measurement values as calibration data in the Flash of the 8 bit controller. These values are subtracted from the measured values in measurements after this command has finished.

Bit 7	6	5	4	3	2	1	0
1	0	1	0	1	0	1	0

When calibration has finished a “calibration saved” package (see 5.1.4) is sent by the ultrasonic module.

Note: Calibration should be issued while the ultrasonic transducers are directed at an empty room so a good null measurement value can be taken.

5.2.4 Clear calibration

This command clears stored calibration data setting every value to 0. This way the sampled data can be received in an unaltered way in raw data mode.

Bit 7	6	5	4	3	2	1	0
0	1	1	0	1	0	1	1

Note: After this command was issued evaluated data mode will not return useful data.

5.2.5 Pin data

This command is used to write to or read from the PDR registers of the unused IO pins of the microcontroller.

Bit 7	6	5	4	3	2	1	0
-	Value	Pin1	Pin0	R/W	1	0	0

Pin1..0 bits	Pin
00	PG1
01	PG2
10	P01
11	P02

5.2.6 Pin configuration

With this command the DDR (data direction) and PUL (pull-up enable) registers can be accessed.

Bit 7	6	5	4	3	2	1	0
Register	Value	Pin1	Pin0	R/W	1	0	1

Pin1..0 are the same values as for the pin data command.

Register bit	Register
0	DDR
1	PUL

Value bit	DDR register meaning	PUL register meaning
0	Input pin	Pull-up disabled
1	Output pin	Pull-up enabled

5.2.7 Reserved IDs

These two command ids are reserved for future extensions.

Bit 7	6	5	4	3	2	1	0
-	-	-	-	-	1	1	0
-	-	-	-	-	1	1	1

6 Main microcontroller unit software

This chapter will give an overview on the provided libraries with the test software.

The following software modules are provided with the test software:

- Bluetooth module (btm.h)
- Motor and servo control (drive.h)
- Infrared (ir.h)
- RFM12 (rfm12.h)
- Real time clock (rtc.h)
- Seven segment display (seg.h)
- Systick (systick.h)
- Ultrasonic module (us.h)
- USART communication (usart.h)
- Utility routines (util.h)

The line sensor has its own test software. It is described at the end of this chapter.

6.1 Microcontroller resources used by the test software

MCU resource	Module	Pins
	Infrared front (multiplexer)	P08_0, P08_1
	Infrared back (multiplexer)	P02_0, P02_1
	RFM12	P00_6
	Seven segment display	P09
INT 15	Infrared back	P08_4
PPG 0	Servo control	P06_0
PPG 1	Motor control	P06_1
PPG 2	Infrared front	P06_2
PPG 3	Infrared back	P06_3
RLT 1	Systick	
RLT 3	Wait routines	
RTC	RTC	
UART 0	USART	P08_2, P08_3
UART 1	Infrared back	P08_5, P08_6
UART 2	Ultrasonic front	P05_0, P05_1
UART 3	Ultrasonic back	P01_2, P01_3
UART 7	Bluetooth	P00_1, P00_2
UART 8	RFM12	P00_3, P00_4, P00_5
UART 9, INT 6	Infrared front	P07_6, P07_7

Table 23: Used MCU resources

6.2 Data types

6.2.1 Integer types – [u]int(n)_t

Generic integer types are defined as [u]int(n)_t where (n) is the number of bits and the optional u makes the type unsigned. Bit widths are defined for 8, 16 and 32 bits.

6.2.2 Boolean – boolean_t

A boolean type is defined as boolean_t. Values for false and true are defined as symbols FALSE and TRUE.

6.2.3 Characters and strings – char_t

Characters are defined as char_t, strings as a pointer to a character (*char_t).

6.2.4 Generic result status – en_result_t

Enumeration which represents the status of the execution of a method. This enumeration has the following values:

Value	Description
Ok (0)	Method finished successfully
Error (1)	Unspecified error occurred
ErrorInvalidParameter (4)	One or more parameters passed were incorrect
ErrorOperationInProgress (5)	Another conflicting operation is currently running

6.2.5 Function pointers – func_ptr_t

Pointer to functions for callback functions of the type void func(void).

6.3 Bluetooth module / USART communication

Since Bluetooth communication operates via a standard USART module there is no difference between the USART and the Bluetooth module API.

6.3.1 Initialization – Btm_Init / Usart_Init

Initializes the USART. Call once after application start-up.

```
void Btm_Init(void)
void Usart_Init(void)
```

Parameters for the USARTs are:

Parameter	Value
Baudrate	115200 bit/s
Frame length	8 bit
Parity	None
Stop bits	1

6.3.2 Receive byte – Btm_GetByte / Usart_GetByte

Try to get a single byte from USART. Can be used in blocking mode (method waits until a byte is received) or non-blocking mode (method returns instantly even if no byte was received).

en_result_t Btm_GetByte(boolean_t noBlock, uint8_t *data)

en_result_t Usart_GetByte(boolean_t noBlock, uint8_t *data)

Parameters:

- noBlock: Specifies if the function should be blocking (FALSE) or non-blocking (TRUE)
- data: Pointer to an buffer where the received byte will be stored

Return value:

Ok if a byte was successfully received, Error otherwise.

6.3.3 Send byte – Btm_SendByte

Send a single byte over USART. Can be used as blocking (wait until transmission register empty) or non-blocking method.

en_result_t Btm_SendByte(uint8_t data, boolean_t noBlock)

en_result_t Usart_SendByte(uint8_t data, boolean_t noBlock)

Parameters:

- data: Data byte to send
- noBlock: Blocking (FALSE) or non-blocking (TRUE) operation

Return value:

Ok if byte was transmitted or ErrorOperationInProgress if the method was used in non-blocking mode and an transmission was still ongoing.

6.3.4 Send string – Btm_SendString

Send a null-terminated string.

en_result_t Btm_SendString(char_t* data)

en_result_t Usart_SendString(char_t* data)

Parameters:

- data: Pointer to string to send

Return value:

Ok.

6.3.5 Send decimal value as ASCII – Btm_SendDec

Send a value as an ASCII string representing the decimal value of a given parameter.

en_result_t Btm_SendDec(uint32_t data, uint8_t digits, char_t fillChar)

en_result_t Usart_SendDec(uint32_t data, uint8_t digits, char_t fillChar)

Parameters:

- data: Value to send
- digits: Number of decimal digits the value should be formatted to
- fillChar: Character used to fill leading zeros

Return value:

Ok if everything was fine or ErrorInvalidParameters if digits out of range (1 – 9).

6.3.6 Send hexadecimal value as ASCII – Btm_SendHex

Send a value as an ASCII string representing the hexadecimal value of a given parameter.

en_result_t Btm_SendHex(uint32_t data, uint8_t digits, char_t fillChar)

en_result_t Usart_SendHex(uint32_t data, uint8_t digits, char_t fillChar)

Parameters:

- data: Value to send
- digits: Number of decimal digits the value should be formatted to
- fillChar: Character used to fill leading zeros

Return value:

Ok if everything was fine or ErrorInvalidParameters if digits out of range (1 – 8).

6.4 Motor and servo control

This module is used to control the motor and servo operation.

6.4.1 Global variables

int8_t ppg_motor_power	Current motor drive level (from -3 to +3, see 6.4.3)
int8_t ppg_servo_percent	Current servo state (from -100 to +100, see 6.4.4)

6.4.2 Initialization – Drive_Init

Initializes PPGs used for motor and servo control. Should be called once after program start-up.

void Drive_Init(void)

6.4.3 Set motor power – Drive_SetMotor

Sets a new motor power target value. If timer is ignored the value is directly set to the PPG, otherwise acceleration will be limited.

```
void Drive_SetMotor(int8_t power, boolean_t ignoreTimer)
```

Parameters:

- power: Power level to set. Values from -3 to +3
- ignoreTimer: Ignore the acceleration limit timer if TRUE

6.4.4 Set servo state – Drive_SetServo

Set the servo direction.

```
void Drive_SetServo(int8_t percent)
```

Parameters:

- percent: Direction to set. -100 for full left, 0 for center, +100 for full right

6.4.5 Quick stop – Drive_QuickStop

If currently driving forward active braking is used to stop the car. If driving backwards motor is simply switched off. See chapter 3.3 - Speed controller for details.

```
void Drive_QuickStop(void)
```

6.4.6 Disable forward direction – Drive_DisableForward and disable reverse direction – Drive_DisableReverse

Forbids to drive forward or backward. Can be used by e.g. the infrared object detection to disable a direction when an object is detected.

```
void Drive_DisableForward(boolean_t disable)
```

```
void Drive_DisableReverse(boolean_t disable)
```

Parameters:

- disable: Forbids to drive in the specific direction (TRUE) or allows it (FALSE)

6.4.7 Systick routine – Drive_Systick

This routine handles acceleration and servo control and has to be called in intervals of 1 ms. This can be achieved by adding this method to the systick method list (see 6.9.2 - Register a function – Systick_AddFunction).

```
void Drive_Systick(void)
```

6.5 Infrared

This module handles object detection. In this implementation sending of identification codes and evaluation of the received data is handled internally. Only the current state of the object detection is provided by the Ir_States variable.

6.5.1 Global variables

uint8_t carID — Stores the ID sent over infrared

uint8_t Ir_States — Contains the current object detection states for each of the eight infrared sensors.

Bit	Module
0	Left
1	Front left
2	Front
3	Front right
4	Right
5	Back right
6	Back
7	Back left

6.5.2 Initialization – Ir_Init

Initializes USARTs, PPGs and external interrupts used by the infrared module. Should be called once after program start-up.

```
void Ir_Init(void)
```

6.6 RFM12

6.6.1 Initialization – Rfm12_Init

Initializes the USART used by this module and sets up the RFM12 module. Should be called once after program start-up.

```
void Rfm12_Init(void)
```

6.6.2 Send command – Rfm12_Command

Sends a command to the RFM12 module and returns the returned value.

```
uint16_t Rfm12_Command(uint16_t cmd)
```

Parameters:

- cmd: Command to send

Return value:

Command reply data

6.6.3 Enable reception – Rfm12_RX_Enable

Enables reception of messages.

```
void Rfm12_RX_Enable(void)
```


6.6.4 Check current state – Rfm12_State

Get current receive/transmit state.

uint8_t Rfm12_State(void)

Return value:

Current state of this module. See RFM12_STATE_* defines.

6.6.5 Send a message – Rfm12_SendMessage

Put a message in the transmit buffer and initialize transmission.

en_result_t Rfm12_SendMessage(uint8_t* data, uint8_t len)

Parameters:

- data: Pointer to data buffer to transmit
- len: Length of data in buffer

Return value:

Ok if everything went fine, ErrorOperationInProgress if a transmission/reception is currently ongoing.

6.6.6 Get a received message – Rfm12_GetRxBuffer

Returns a copy of the RX buffer if a message is available.

en_result_t Rfm12_GetRxBuffer(uint8_t* buffer, uint8_t bufSize,
uint8_t* returnedBytes)

Parameters:

- buffer: Pointer to target buffer for returned message
- bufSize: Size of target buffer
- returnedBytes: Actual amount of returned bytes

Return value:

Ok if a message was received, Error if not.

6.7 Real time clock

6.7.1 Global variables

uint16_t rtcCount Increased by one every half second.

6.7.2 Initialization – Rtc_Init

Initializes the RTC module. Should be called once after application start-up.

void Rtc_Init(void)

6.7.3 Start of operation – Rtc_Start

Starts the actual RTC operation with the given time value.

```
void Rtc_Start(uint8_t h, uint8_t m, uint8_t s)
```

Parameters:

- h: Hours
- m: Minutes
- s: Seconds

6.7.4 Print time to USART – Rtc_PrintTime

Prints the current time to the USART interface defined by TESTINTERFACE in main.h.

```
void Rtc_PrintTime(void)
```

6.8 Seven segment display

6.8.1 Initialization – Seg_Init

Initializes IOs for the left seven segment display (SEG1) and disables all segments.

```
void Seg_Init(void)
```

6.8.2 Output of a decimal digit – Seg_Dec

Outputs a decimal digit to the seven segment display.

```
void Seg_Dec(uint8_t num)
```

Parameters:

- num: Digit to display (0 – 9)

6.8.3 Output of a hexadecimal digit – Seg_Hex

Outputs a hexadecimal digit to the seven segment display.

```
void Seg_Hex(uint8_t hex)
```

Parameters:

- hex: Digit to display (0 – 15)

6.9 Systick

The systick calls registered methods every 1 ms. The number of registerable methods can be defined by the symbol MAX_FUNCTIONS in systick.c which defaults to 10. Unregistering functions is not implemented.

6.9.1 Initialization – Systick_Init

Initializes the timer for the systick. Should be called once after application start-up.

```
void Systick_Init(void)
```


6.9.2 Register a function – Systick_AddFunction

Registers a function to the systick.

En_result_t Systick_AddFunction(func_ptr_t func)

Parameters:

- func: The function to be registered

Return value:

Ok if everything was fine, Error if the registered functions array is full.

6.10 Ultrasonic module

Handles communication with the ultrasonic modules. Commands which take the parameter “module” expect the following values:

Module number	Module
0	Both (not supported by all commands)
1	Front
2	Back

6.10.1 Global variables

boolean_t Us_DataValid1	Indicates whether the distances in the distance-variables for module 1 are valid
boolean_t Us_DataValid2	Indicates whether the distances in the distance-variables for module 2 are valid
uint8_t Us_Distance1Raw	Current timeslot index of detected object for module 1. If 255 no object was detected
uint16_t Us_Distance1Eval	Current distance for module 1 in cm
uint8_t Us_Distance2Raw	Current timeslot index of detected object for module 2. If 255 no object was detected
uint16_t Us_Distance2Eval	Current distance for module 2 in cm

6.10.2 Initialization – Us_Init

Initializes the USARTs for communication with the ultrasonic modules. Should be called once after application start-up.

void Us_Init(void)

6.10.3 Wait for command to be confirmed – Us_Cmd_Wait

Checks whether the last sent command was confirmed yet.

boolean_t Us_Cmd_Wait(uint8_t module)

Parameters:

- module: Module to check whether there is an outstanding command

Return value:

TRUE if command is not yet confirmed, FALSE otherwise

6.10.4 Calibrate module – Us_Calibrate

Sends a calibration command to an ultrasonic module.

en_result_t Us_Calibrate(uint8_t module)

Parameters:

- module: The module to send the command to, 0 for both modules supported

Return value:

Ok if everything was fine, ErrorInvalidParameter if module, ioNr or ddr invalid.

6.10.5 Set data direction of pin – Us_IO_DDR

Sets the data direction of an IO pin on an ultrasonic module.

en_result_t Us_IO_DDR(uint8_t module, uint8_t ioNr, uint8_t ddr)

Parameters:

- module: The module to send the command to, 0 for both modules supported
- ioNr: IO to control (0 = PG1, 1 = PG2, 2 = P01, 3 = P02)
- ddr: Data direction to set (0 = input, 1 = output)

Return value:

Ok if everything was fine, ErrorInvalidParameter if module, ioNr or ddr invalid.

6.10.6 Set data of pin – Us_IO_Value

Sets the data value of an IO pin on an ultrasonic module.

en_result_t Us_IO_Value(uint8_t module, uint8_t ioNr, uint8_t value)

Parameters:

- module: The module to send the command to, 0 for both modules supported
- ioNr: IO to control (0 = PG1, 1 = PG2, 2 = P01, 3 = P02)
- value: Data to set (0 = low, 1 = high)

Return value:

Ok if everything was fine, ErrorInvalidParameter if module, ioNr or value invalid.

6.11 Utility routines

This module contains some smaller generic utility routines.

6.11.1 ARRAY_SIZE

Macro to return the size of a given array in bytes.

ARRAY_SIZE(arr)

6.11.2 Conversion integer to string – intToStr

Converts a given integer to a decimal ASCII string.

```
en_result_t intToStr(uint32_t data, uint8_t digits, char_t fillChar, char_t* str)
```

Parameters:

- data: Value to convert
- digits: Number of digits to format to
- fillChar: Character used to fill leading zeros
- str: Target buffer for converted string

Return value:

Ok if everything went fine, ErrorInvalidParameter if number of digits out of range (1 to 10)

6.11.3 Wait number of cycles – wait

Delay loop for a number of loop cycles.

```
void wait(uint32_t del)
```

Parameters:

- del: Number of loop cycles

6.11.4 Wait number of microseconds – wait1us

Waits a specific number of microseconds.

```
void wait1us(uint16_t delay_us)
```

Parameters:

- delay_us: Number of microseconds to wait

6.11.5 Wait number of milliseconds – wait1ms

Waits a specific number of milliseconds.

```
void wait1ms(uint16_t delay_ms)
```

Parameters:

- delay_ms: Number of milliseconds to wait

6.12 IR line sensor module

6.12.1 Software

The software for the IR line sensor module is not included in the student´s car software.

Use ADC channels to readout sensor values.

Use a PPG channel for a PPG / PWM signal (power management) or VCC for permanent current supply.

6.12.2 Test software

In order to test the IR line sensor module use the test software TP_IR_LINE_SENSOR. Connect the board to the starter kit and download the test software. Press Button SW1/RESET of the evaluation board.

Check sensor functionality

After SW1/RESET is pressed SKWizard should show four columns with sensor values. But only the two left columns will already show alternating values. The two right columns will just show zeros.

Check calibration functionality

A push to the Key-button INT0 (left button) should result in calibration of the values of the second division. In Addition the two right divisions should show alternating values now as well. The values of the third division are the inverted values of the second division (values are needed for area weighting algorithm). The last division only shows the peak value of the area weighting algorithm in order to control output values.

Optional testing

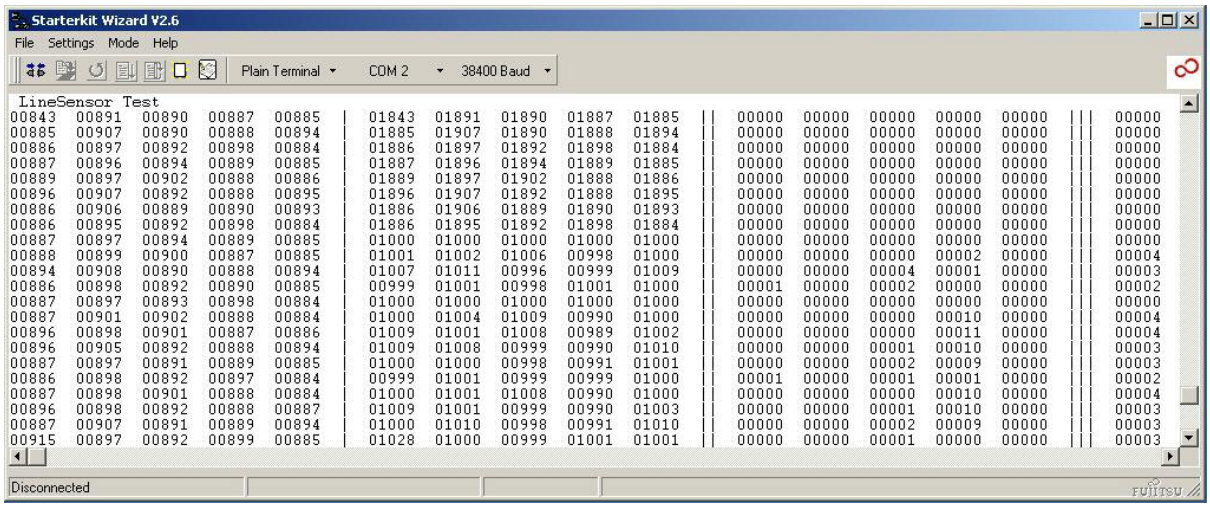
In order to check output values it is possible to apply a voltmeter to pin three to seven while affecting the sensors.

In order to check the PPG / PWM signal it is possible to apply an oscilloscope to pin 65 and ground (GND) in order to measure the frequency (example should have a value of 2 kHz).

Result within the SKWizard

After the software was started and the calibration was made the SKWizard will look like this:

COM-port (RS232) Connection:



7 Appendix

7.1 Tables

7.2 Figures

8 Information in the WWW

European website:

http://emea.fujitsu.com/

Microcontrollers (8-, 16- and 32bit)
Datasheets and hardware manuals, support tools (hard- and software)

http://mcu.emea.fujitsu.com/

8.1 16FX controller related

Microcontroller MB96F348HS:

http://mcu.emea.fujitsu.com/mcu_product/detail/MB96F348HSCPMC.htm

Starter kit SK-16FX-EUROSCOPE:

http://mcu.emea.fujitsu.com/mcu_tool/detail/SK-16FX-EUROSCOPE.htm

Flash programmer for 16FX microcontrollers:

http://mcu.emea.fujitsu.com/mcu_tool/detail/FLASH_PROGRAMMER_16FX.htm

SK Wizard:

http://mcu.emea.fujitsu.com/mcu_tool/detail/SK_WIZARD.htm

8.2 FR controller related

Microcontroller MB91F467B:

http://mcu.emea.fujitsu.com/mcu_product/detail/MB91F467BAPMC.htm

Starter kit SK-FR-144PMC-91467B:

http://mcu.emea.fujitsu.com/mcu_tool/detail/SK-91467B-144PMC.htm

Flash programmer for 32 bit FR microcontrollers:

http://mcu.emea.fujitsu.com/mcu_tool/detail/FLASH_PROGRAMMER_FR_FME.htm

SK Wizard:

http://mcu.emea.fujitsu.com/mcu_tool/detail/SK_WIZARD.htm

8.3 Hardware documentation

8.3.1 RFM12

HopeRF RFM12 product page:

http://www.hoperf.com/rf_fsk/rfm12.htm

Wiki page on microcontroller.net with fixed register information:

http://www.mikrocontroller.net/articles/RFM12

8.3.2 BTM-222

Product page:

http://www.rayson.com/btm220.html

Datasheet and supplier:

https://www.it-wns.de/themes/kategorie/detail.php?artikelid=219&source=2

9 Recycling

Gültig für EU-Länder:

Gemäß der Europäischen WEEE-Richtlinie und deren Umsetzung in landesspezifische Gesetze nehmen wir dieses Gerät wieder zurück.

Zur Entsorgung schicken Sie das Gerät bitte an die folgende Adresse:

Fujitsu Microelectronics Europe GmbH
Warehouse/Disposal
Monzastraße 4a
D-63225 Langen

Valid for European Union Countries:

According to the European WEEE-Directive and its implementation into national laws we take this device back.

For disposal please send the device to the following address:

Fujitsu Microelectronics Europe GmbH
Warehouse/Disposal
Monzastraße 4a
D-63225 Langen
GERMANY