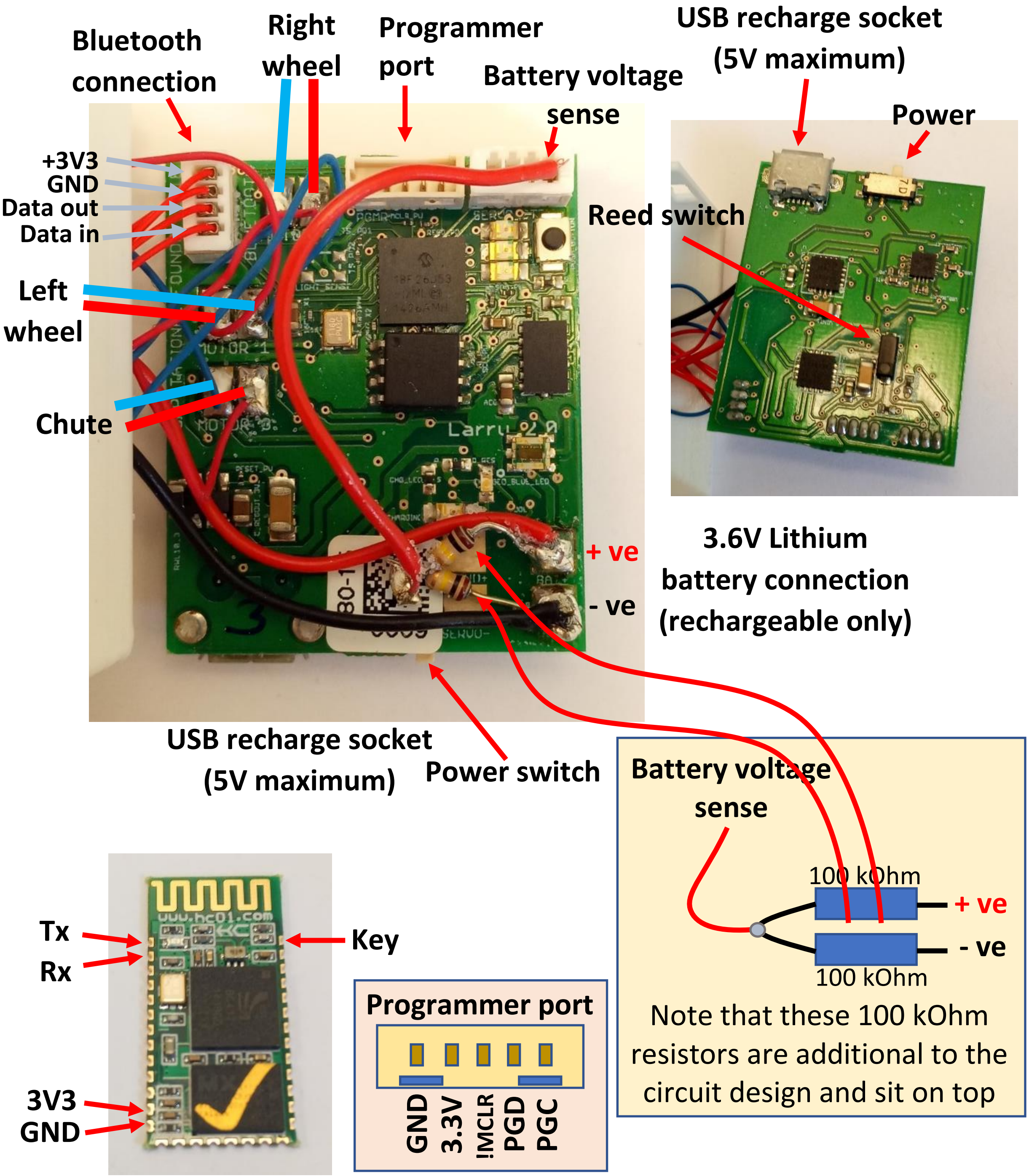
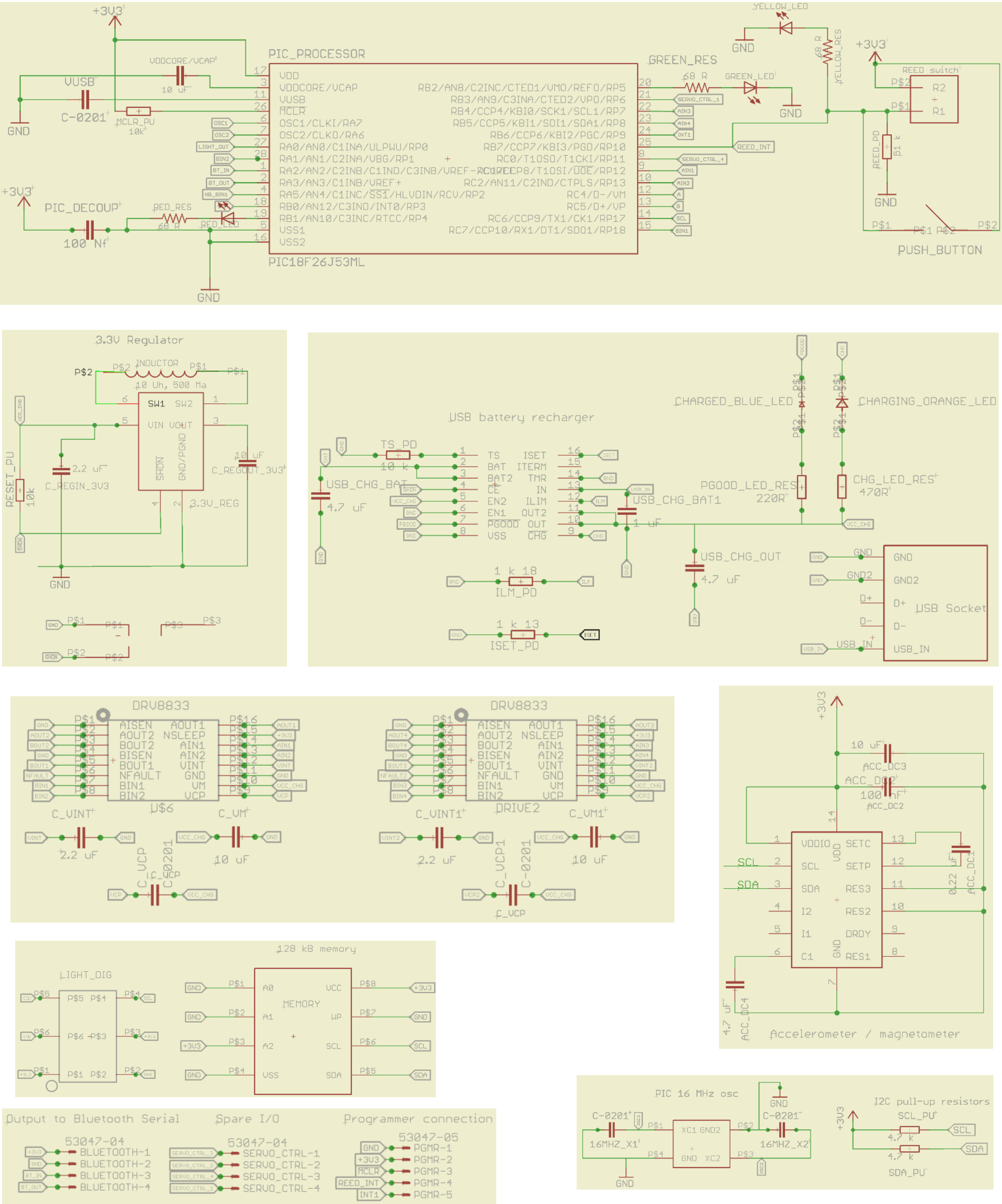
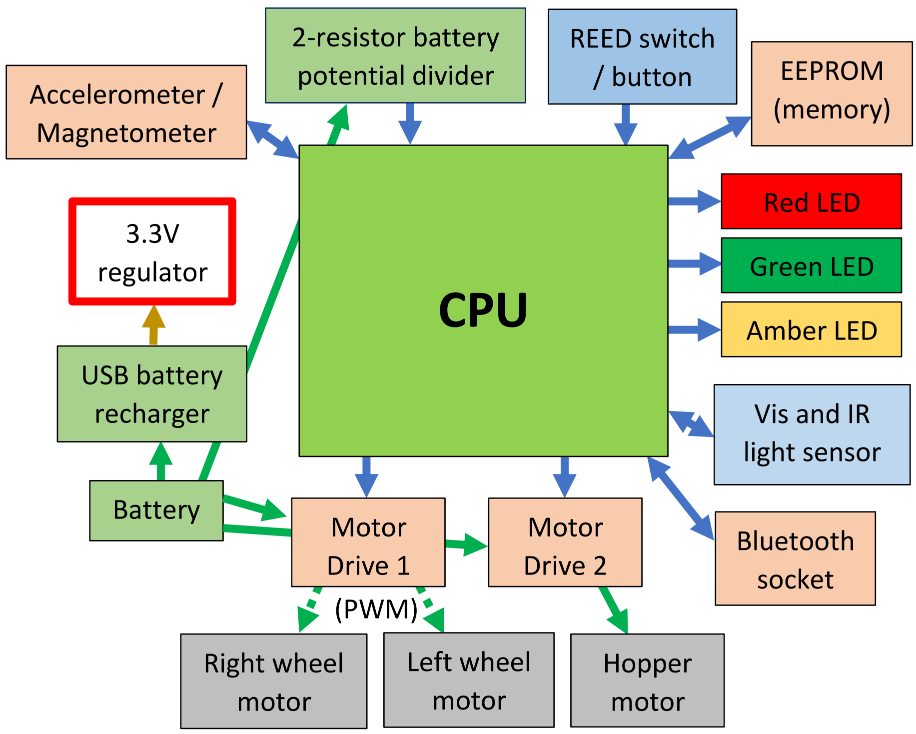
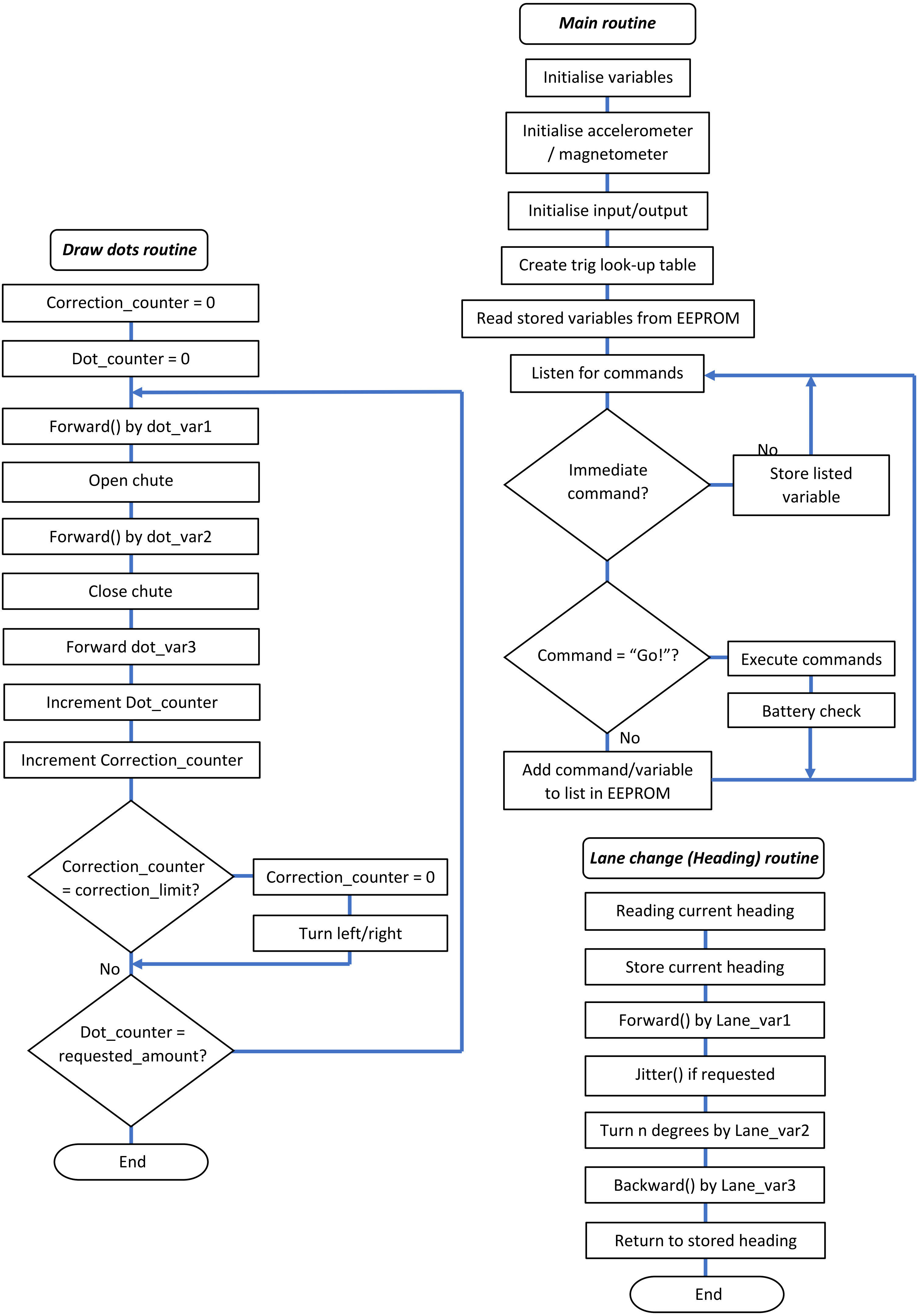
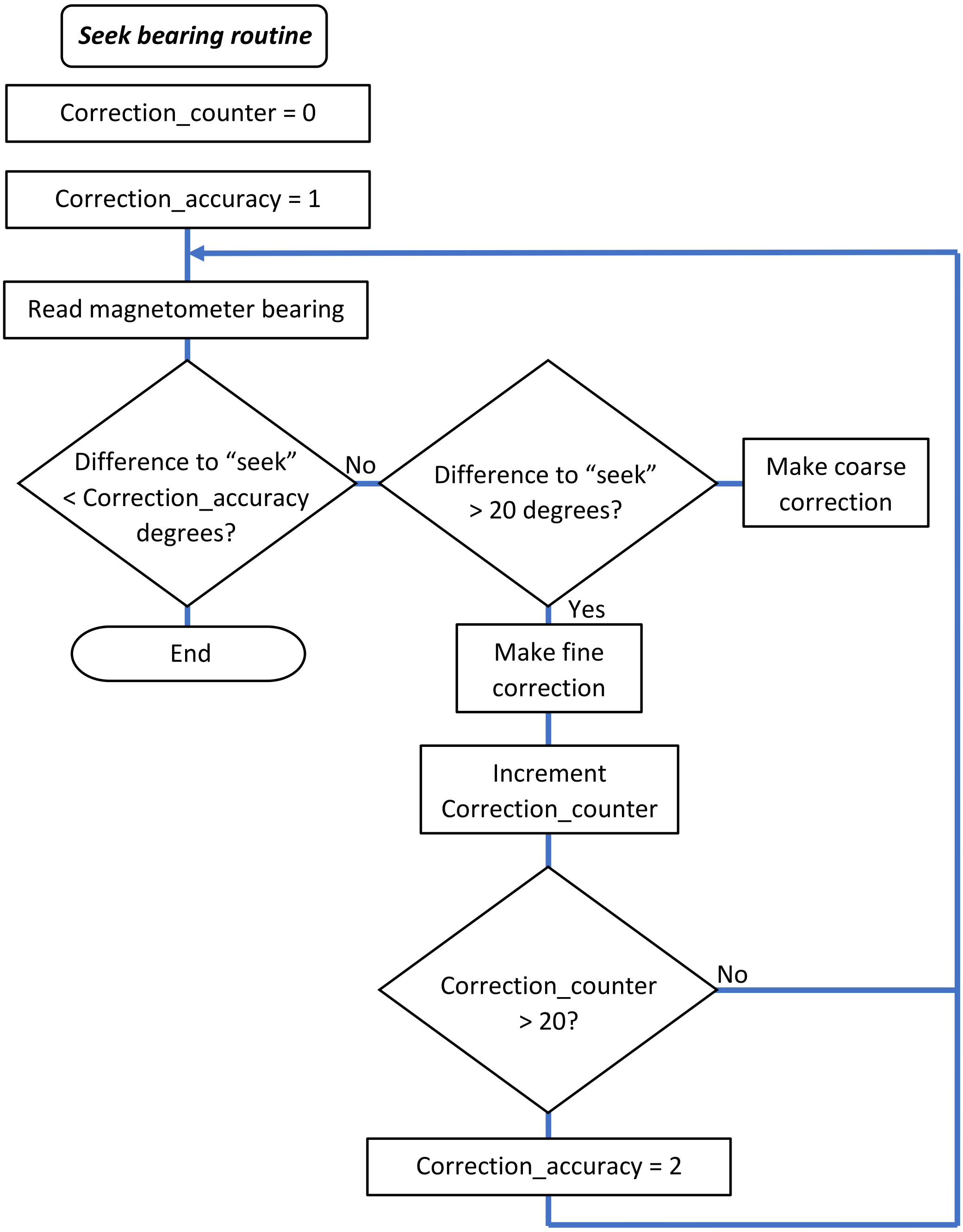

Bluetooth connection
Right wheel
Programmer port
Battery voltage sense
USB recharge socket (5V maximum)
Power
+3V3
GND
Data out
Data in
Left wheel
Chute
Reed switch
+ ve
- ve
3.6V Lithium battery connection (rechargeable only)
USB recharge socket (5V maximum)
Power switch
Battery voltage sense
100 kOhm
+ ve
- ve
100 kOhm
Note that these 100 kOhm resistors are additional to the circuit design and sit on top
Tx
Rx
Key
3V3
GND
Programmer port
GND
3.3V
!MCLR
PGD
PGC

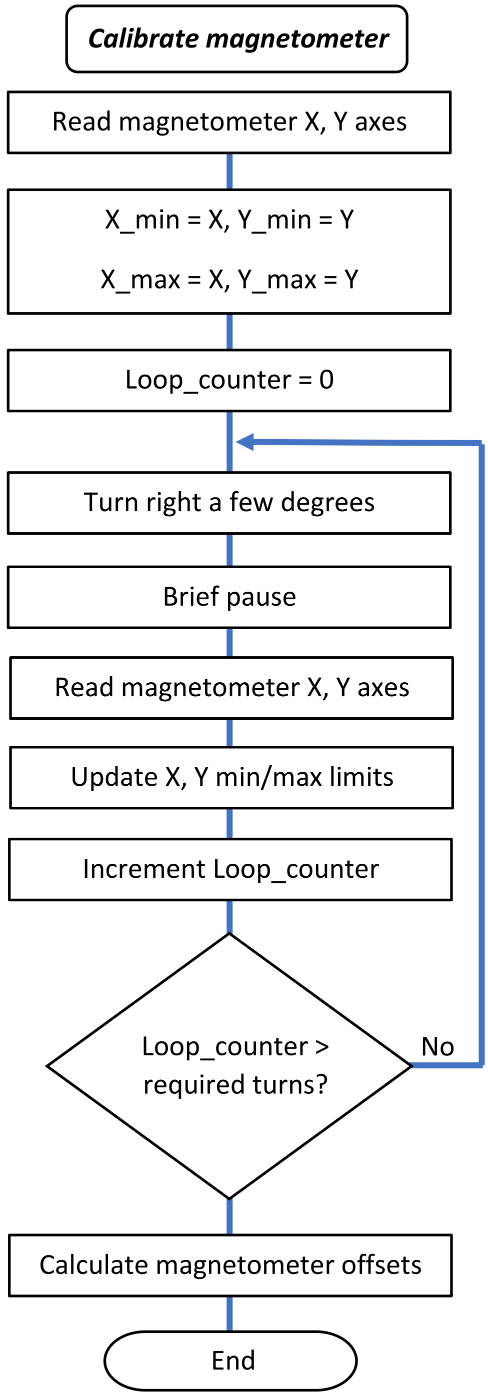


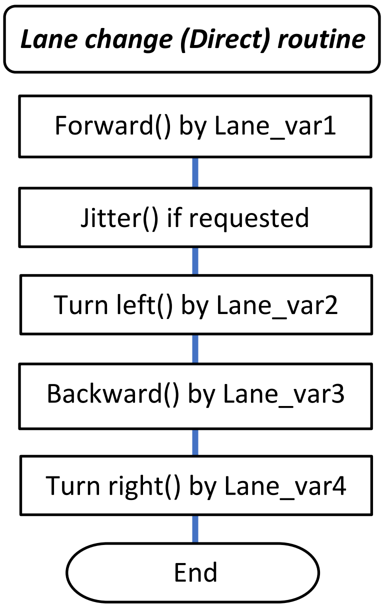
PIC_PROCESSOR
PIC18F26J53ML
VDD
VDDCORE/VCAP
VUSB
MCLR
OSC1/CLKI/RA7
OSC2/CLKO/RA6
RA0/AN0/C1INA/ULPWU/RP0
RA1/AN1/C2INA/VBG/RP1
RA2/AN2/C2INB/C1IND/C3INB/VREF-/CVREF
RA3/AN3/C1INB/VREF+
RA5/AN4/C1INC/SS1/HLVDIN/RCV/RP2
RB0/AN12/C3IND/INT0/RP3
RB1/AN10/C3INC/RTCC/RP4
VSS1
VSS2
RB2/AN8/C2INC/CTED1/VMO/REFO/RP5
RB3/AN9/C3INA/CTED2/VPO/RP6
RB4/CCP4/KBI0/SCK1/SCL1/RP7
RB5/CCP5/KBI1/SDI1/SDA1/RP8
RB6/CCP6/KBI2/PGC/RP9
RB7/CCP7/KBI3/PGD/RP10
RC0/T1OSO/T1CKI/RP11
RC1/CCP8/T1OSI/UOE/RP12
RC2/AN11/C2IND/CTPLS/RP13
RC4/D-/VM
RC5/D+/VP
RC6/CCP9/TX1/CK1/RP17
RC7/CCP10/RX1/DT1/SDO1/RP18
+3V3
GND
VUSB
C-0201
VDDCORE/VCAP
10 uF
MCLR_PU
10k
PIC_DECOUP
100 Nf
RED_RES
68 R
RED_LED
GREEN_RES
68 R
GREEN_LED
YELLOW_LED
68 R
YELLOW_RES
REED suitch
REED_PD
51 k
REED_INT
PUSH_BUTTON
3.3V Regulator
INDUCTOR
10 Uh, 500 Ma
SW1
SW2
VIN
VOUT
SHDN
GND/PGND
2.2 uF
C_REGIN_3V3
10 uF
C_REGOUT_3V3
3.3V_REG
RESET_PU
10k
USB battery recharger
TS_PD
10 k
USB_CHG_BAT
4.7 uF
TS
BAT
BAT2
CE
EN2
EN1
PGOOD
VSS
ISET
ITERM
TMR
IN
ILIM
OUT2
OUT
CHG
USB_CHG_BAT1
1 uF
USB_CHG_OUT
4.7 uF
1 k 18
ILM_PD
1 k 13
ISET_PD
CHARGED_BLUE_LED
CHARGING_ORANGE_LED
PGOOD_LED_RES
220R
CHG_LED_RES
470R
USB Socket
GND
GND2
D+
D-
USB_IN
DRV8833
AISEN
AOUT2
BOUT2
BISEN
BOUT1
NFAULT
BIN1
BIN2
AOUT1
NSLEEP
AIN1
AIN2
VINT
GND
VM
VCP
U$6
DRIVE2
C_VINT
2.2 uF
C_VM
10 uF
C_VCP
C-0201
C_VINT1
C_VM1
C_VCP1
+3V3
10 uF
ACC_DC3
ACC_DC2
100 nF
VDDIO
SCL
SDA
I2
I1
C1
SETC
SETP
RES3
RES2
DRDY
RES1
VDD
GND
0.22 uF
ACC_DC1
4.7 uF
ACC_DC4
Accelerometer / magnetometer
128 kB memory
LIGHT_DIG
MEMORY
A0
A1
A2
VSS
VCC
WP
SCL
SDA
Output to Bluetooth Serial
53047-04
BLUETOOTH-1
BLUETOOTH-2
BLUETOOTH-3
BLUETOOTH-4
Spare I/O
SERVO_CTRL-1
SERVO_CTRL-2
SERVO_CTRL-3
SERVO_CTRL-4
Programmer connection
53047-05
PGMR-1
PGMR-2
PGMR-3
PGMR-4
PGMR-5
GND
+3V3
MCLR
REED_INT
INT1
PIC 16 MHz osc
C-0201
16MHZ_X1
16MHZ_X2
XC1 GND2
GND XC2
I2C pull-up resistors
SCL_PU
SDA_PU
4.7 k
SCL
SDA

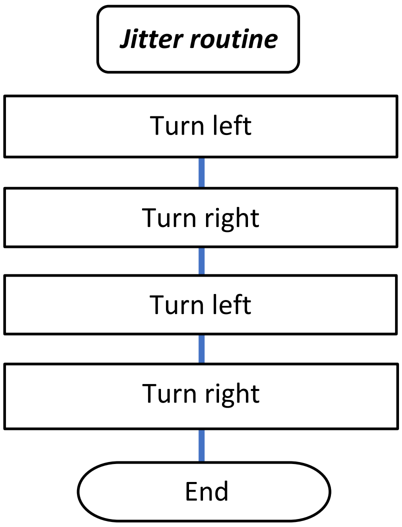

Accelerometer / Magnetometer
2-resistor battery potential divider
REED switch / button
EEPROM (memory)
Red LED
Green LED
Amber LED
3.3V regulator
CPU
USB battery recharger
Vis and IR light sensor
Battery
Motor Drive 1
Motor Drive 2
Bluetooth socket
(PWM)
Right wheel motor
Left wheel motor
Hopper motor

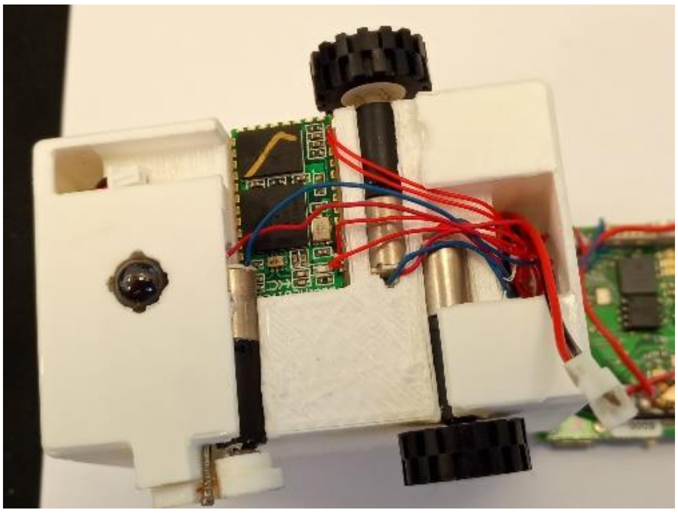
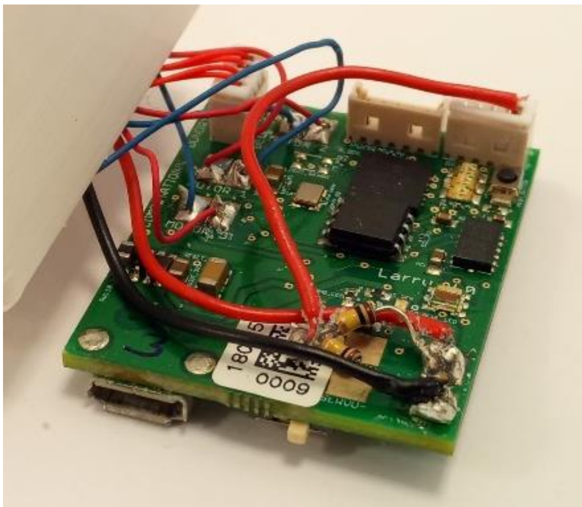

Main routine
Initialise variables
Initialise accelerometer / magnetometer
Initialise input/output
Create trig look-up table
Read stored variables from EEPROM
Listen for commands
Immediate command?
No
Store listed variable
Command = "Go!"?
Execute commands
Battery check
No
Add command/variable to list in EEPROM
Draw dots routine
Correction_counter = 0
Dot_counter = 0
Forward() by dot_var1
Open chute
Forward() by dot_var2
Close chute
Forward dot_var3
Increment Dot_counter
Increment Correction_counter
Correction_counter = correction_limit?
Correction_counter = 0
Turn left/right
No
Dot_counter = requested_amount?
End
Lane change (Heading) routine
Reading current heading
Store current heading
Forward() by Lane_var1
Jitter() if requested
Turn n degrees by Lane_var2
Backward() by Lane_var3
Return to stored heading
End


Seek bearing routine
Correction_counter = 0
Correction_accuracy = 1
Read magnetometer bearing
Difference to “seek” < Correction_accuracy degrees?
No
End
Difference to “seek” > 20 degrees?
Make coarse correction
Yes
Make fine correction
Increment Correction_counter
Correction_counter > 20?
No
Correction_accuracy = 2


Calibrate magnetometer
Read magnetometer X, Y axes
X_min = X, Y_min = Y
X_max = X, Y_max = Y
Loop_counter = 0
Turn right a few degrees
Brief pause
Read magnetometer X, Y axes
Update X, Y min/max limits
Increment Loop_counter
Loop_counter > required turns?
No
Calculate magnetometer offsets
End


Lane change (Direct) routine
Forward() by Lane_var1
Jitter() if requested
Turn left() by Lane_var2
Backward() by Lane_var3
Turn right() by Lane_var4
End


Jitter routine
Turn left
Turn right
Turn left
Turn right
End

The Sustainabot circuit board, 29 x 36 x 10 mm, consists of two layers, with parts mounted on both top and bottom sides. These consist of a number of primary components, including a Microchip 8-bit microcontroller (PIC18F26J53) with multiple in-built timing modules, and highly flexible and programmable I/O ports. In addition to the CPU, there is an EEPROM (24AA1025 / 128kB) for storing command data, MEMS accelerometer/magnetometer (LSM303DLHC) for determining compass heading, H-bridge DC motor drivers (DRV8833), digital visible light and infrared, dual light sensor (APDS9300), 3.3V/200 mA buck-boost regulator, and a lithium recharge controller chip (BQ24074RGTR). The CPU is clocked by an external 16 MHz crystal (PLL disabled), drives 3 coloured LEDs for status indication, transfers sensor data to/from MEMS sensors and EEPROM for storage via the I2C protocol, and enables various combinations of motor drive via the two dual-port H-bridge controllers (only 3 of the 4 ports connected in-circuit).

The battery monitor, while not necessary for the operation of the system, was added as an extra feature for the user. It simply consists of two 100 kOhm resistors, connected in-series across the battery terminals, with the centre tap feeding to one of the analogue input ports on the CPU. This is a simple potential divider (/2) enabling the overall battery potential to be measured with an ADC on the CPU, by utilising a reference of 3.3V from the board's regulator. The firmware is configured to give the user a warning via the Bluetooth port when the voltage is at, or below, 3.3V, and that a recharge is required to maintain sufficient drive current to the wheel motors.

The Bluetooth module is an HC-05 board, used for transmission/reception of data. This was configured for serial communications at 38400 BAUD; any higher and the bit-error rate would have been too great due to mismatch from the CPU's 16 MHz timing crystal. This board, or any similarly 3.3V powered Bluetooth serial communications board can be used in conjunction with the controller circuit.

Some of the CPU's ports were reconfigured to make use of one of the internal timers, to output sub-microsecond-width pulses at frequencies of 100s of kHz. This, together with the ability to define different pulse-widths for different I/O ports that rapidly switch the H-bridge circuits on/off i.e. vary the 'duty cycle', enabled us to finely control the mean current delivered to each of the two motors, a technique commonly referred to as pulse width modulation (PWM). The H-bridge controllers in the circuit are configured to operate in two different modes. For the chute, this is in a full-on/off mode, while the drive wheels operate under PWM. The advantage of using PWM is that it also allows for compensation of different motor characteristics due to subtle size variations between the motors, in addition to independent drive at different speeds/directions for slow curved turning.

The drive motors we used on this device are 700:1 planetary-drive geared motors, lightweight, and with enough torque to manoeuvre a small robot around a flat surface.

The on-board magnetometer can be used to give the robot a sense of directionality, although, sensing magnetic fields can be prone to confusion due to ferrous material such as metal table legs, under-table support structures, all capable of causing a shift locally in the Earth's magnetic field. This aside, the robot can be instructed to turn in a circle of more than 360 degrees, collecting magnetic data on the two axes parallel to the ground. From this, it is able to determine any hard- or soft-iron offsets and correct for them.