Project

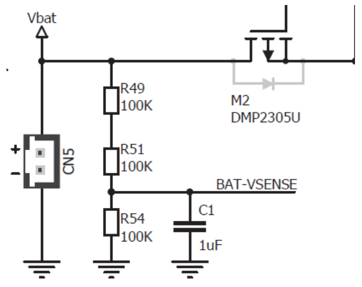
Goal: Implementing a basic control system for an autonomous ground vehicle.

Firmware requirements

- Main loop requirements
    - The control loop should be implemented at 1 kHz frequency.
    - The motor PWM should be updated at 1 kHz and the IR sensor should be read at 1 kHz frequency.
    - Initially, the robot should be in “Wait for start” state.
        - In this state, the PWM DC of all the motors should be 0.
        - The LED A0 and should blink at 1 Hz frequency.
        - The left and right indicators should blink at 1 Hz frequency.
    - Once the button RE8 is pressed, the robot should go in the “Execute” state.
        - In this state, the robot processes the commands received (in order).
            - The commands are sent using the message $PCCMD,x,t*, where t is an integer number and represent the duration of the action in milliseconds, and x can be
                - 1 = forward motion
                - 2 = counterclockwise rotation
                - 3 = clockwise rotation
                - 4 = backward motion
            - Multiple PCCMD can be sent by PC at any time. The micro should thus keep a FIFO of commands received and execute them in order of arrival. The queue can be of maximum 10 commands. Each time a command that can be put in the FIFO is received, the micro acknowledges the command with the message $MACK,1*. If the FIFO is full, the micro sends to the PC a message $MACK,0* to signal the error and discards the message.
            - The commands can also be received in the “wait for start” state and should be kept in the FIFO to be executed as soon as the system goes in the “execute” state.
            - If the FIFO is empty, the robot should simply stop, waiting for new commands, maintaining the state “executing”.
        - The PWM is generated to execute the current command.
        - The LED A0 should blink at 1 Hz frequency.
        - If an obstacle is sensed, the robot should not get closer than 20 cm, and should wait until the current action completes its duration.
    - If the button RE8 is pressed for a second time, the robot should go back in the “Wait for start” state. If any action was being executed, it should be stopped and cleared from the FIFO.
- Motor control
    - Four PWM signals must be generated to control the buggy, on pins RD1 to RD4 (PR65 to RP68) using four Output Compare peripherals.
    - The frequency of the PWM signals must be 10 kHz.
    - The actuation of the wheels follows the specification reported in the table below:

Command	**PWM with DC > 0**	**PWM with DC = 0**
Left wheels forward (left_pwm > 0)	RD2 = left_pwm	RD1 = 0
Left wheels backward (left_pwm < 0)	RD1 = -left_pwm	RD2 = 0
Right wheels forward (right_pwm > 0)	RD4 = right_pwm	RD3 = 0
Right wheels backward (right_pwm < 0)	RD3 = -right_pwm	RD4 = 0

- Battery sensing
    - The voltage of the battery (BAT-VSENSE in the figure below) is available on pin AN11. It is sensed after a partitioning circuit, i.e., in between a 200 kohm resistor and a 100 kohm resistor.



- IR sensor
    - The infrared sensor should be mounted on the *Buggy Mikrobus* 1 or 2 (i.e., in front of the buggy). The signal can be read on AN14/AN15, while the enable to the IR sensor must be given on the digital I/O on RB9/RA3.
- Data logging / command interface through UART
    - The UART to RS232 module should be installed on the *Clicker Mikrobus* 2. The TX signal should be remapped to RD0/RP64, while the RX signal should be remapped to RD11/RPI75.
    - The microcontroller should send, to the PC, the following messages (in all the states)
        - $MBATT,v_batt* where v_batt is the sensed battery in Volt, at 1 Hz frequency. Use two digits, i.e., X.YZ
        - $MDIST,distance* where distance is the sensed distance in cm, at 10 Hz frequency. Use an integer.
    - Given the chosen UART baud rate, the firmware should never lose a message due to its implementation (i.e., proper dimensioning of buffers), even with full use of the bandwidth.

Evaluation criteria

Among other things, these criteria will be used:

- Adherence to the provided specifications
- Correctness of the interrupts service routines
- Correct handling of shared data
- Management of the UART FIFO and circular buffers on both sending and receiving
- General code cleanliness

Pin Mapping

- RB8 Left side lights
- RF1 Right-side lights
- RF0 brakes
- RG1 low intensity lights
- RA7 beam headlights
- AN11 battery sensing
- RD1/RP65 left PWM backward motion
- RD2/RP66 left PWM forward motion
- RD3/RP67 right PWM backward motion
- RD4/RP68 right PWM forward motion
- AN14 or AN15 IR sensor voltage
- RB9 or RA3 IR sensor enable
- RD0/RP64 UART TX
- RD11/RPI75 UART RX