Project

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

A microcontroller board is connected to two motors, forming a differential drive. Let l = 0.5 m be the distance between the wheels, and r = 0.2 m the radius of the two wheels.

The microcontroller receives desired reference values for the speed and the angular velocity of the robot. These reference signals are sent through a serial interface. The microcontroller sends feedback messages back to the control PC to report a few status information.

Hardware specifications

- Each motor can run from -50 to +50 RPMs
- The motors are controlled through a PWM signal.
    - The frequency must be 1 kHz.
    - 50% duty cycle corresponds to 0 RPM, 0% corresponds to -60 RPM and 100% corresponds to 60 RPMs.
    - A dead time of at least 3 microseconds should be used to prevent problems with the H-bridge controlling the motors.
    - Running the motors above +-50 RPMs can damage them and should be avoided.

Firmware requirements

- The control systems should compute the desired RPMs of the wheels using a differential drive kinematic model.
- The control system must never generate PWM signals outside of the specifications of the motors.
    - If the requested speed and angular velocities of the robot are such that the necessary motor RPMs are outside the maximum specifications, the system should saturate them to the minimum/maximum allowed value.
    - An alert message (MCALE) should be sent at 1 Hz until the desired references are changed again and fall below the threshold.
- If no references (i.e., the MCREF command) are received from the PC for more than 5 seconds, the firmware should enter a timeout mode:
    - Both motors velocity should be set to zero.
    - Led D4 should blink at 5 Hz to signal timeout.
    - When a new reference is read, then the led D4 should stop blinking and commands should be given again to the motors.
- The firmware must support receiving references at least at 10 Hz frequency (through a proper choice of baud rate).
    - The firmware does not reply with an ack to reference messages.
    - The frequency should be guaranteed even in case of ENA messages (assume at least 1 per second)
- The firmware must refresh the PWM value at least at 10 Hz frequency.
- The firmware must acquire the temperature sensor at 10 Hz frequency and average the last 10 readings. The averaged value is sent to the PC at 1 Hz frequency with the MCTEM message.
- The firmware must send the feedback message MCFBK at 5 Hz frequency.
- The control system should blink led D3 at 1 Hz to always signal a correct functioning of the main loop, regardless of any state.
- Given the chosen UART baudrate, the firmware should never lose a message due to its implementation (i.e., proper dimensioning of buffers), even with full use of the bandwidth
- If button S5 is pressed, the firmware should enter a safe mode:
    - Motors are stopped *immediately,* and reference signals are *ignored* until the microcontroller receives an enable message (HLENA).
    - After exiting safe mode, the motors should be set to zero. Motors should move only after receiving a *new* reference. Once an enable command is received, the firmware should send a positive ack to the PC.
- The firmware should write on the LCD
    - First row: “STATUS: x”, where x = H/T/C (halt/timeout/controlled)
    - Second row: “R: n1; n2”, where n1 and n2 are the applied RPM (e.g. “R: -20; 33”)
    - If the button S6 is pressed, the data displayed on the LCD changes as follows:
        - First row: “STATUS: x”, where x = H/T/C (halt/timeout/controlled)
        - Second row: “R: omega; v”, where omega is the reference angular velocity, and v is the desired linear speed for the ground vehicle
    - If the button S6 is pressed again, the data displayed toggles again to the first one.

Messages from the PC

$HLREF,omega,speed* where omega (rad/s) is the desired angular velocity and speed (m/s) is the desired linear velocity for the robot.

$HLENA* enables the firmware to send references to the motors (to exit safe mode)

Messages to the PC

$MCFBK,n1,n2,state* where *n1* and *n2* are the *applied* RPM values and *state* is 2 if the microcontroller is in safe mode, 1 if it is in timeout mode, 0 otherwise.

$MCALE,n1,n2*, where n1 and n2 are the compute wheel RPM values that are outside the maximum allowed values.

$MCTEM,temp* where *temp* is the temperature

$MCACK,msg_type,value* where *msg_type* is the command (e.g. ENA, SAT), and *value* is 1 if the message was applied and 0 otherwise (example: $MCACK,SAT,0* for a negative ack to the saturation command)