Exported for Brian O'Connell on Tue, 27 May 2025 19:32:02 GMT

Programming 7 Homework - Serial Communication

This homework is going to be very open-ended. You will need to create bi-directional communication between your Arduino and MATLAB. This means you need them to be both listening and receiving from one another.

The requirements in Part 1 will give some guidance, BUT if you have another idea that maybe say happens to correspond with an aspect of your project, maybe something being developed for your proof of concept, you may ask if you can use that. It just needs to meet the requirements listed.

Collaborating with one another is encouraged although you need to submit your own work.

READ THE FOLLOWING VERY CAREFULLY

- Pseudocode or flowchart.
  - Handwritten flowcharts are not acceptable
    - Use a word or drawing program to assist
  - Pseudocode must be typed
  - Must adhere to the course syntax
    - Pseudocode is not just your code in a text file
  - Must utilize plain language so anyone can understand your overall logic
- This is not a group assignment. It is an INDIVIDUAL one.
  - But I do understand if you work together.
  - You must turn in your own code
  - You must acknowledge who helped you in the Title block.
    - If it was another student, name them.
    - If it was a Red Shirt, just say Red Shirts and their first name if you can recall it. The same goes for if you get help from the TA.
- INCLUDE comments in your code

- Code is not complete if it is not annotated to help us understand what you are trying to do in your code.
- Basic rule of thumb is you need enough plain language comments so that:
  - Someone with a similar ability level could just look at the comments and fully understand what you're trying to do with your code without having to interpret your actions through the code itself.
  - Someone with no programming experience should also be able to read and have a basic understanding of what you're doing, but maybe not how. They'll likely not understand every comment.

Part 1 - The Serial Games

Create a system between your Arduino and MATLAB that produces a game that utilizes both Arduino and MATLAB features to take inputs from the user(s) and provide feedback

Requirements:

It must achieve the following:

- Arduino must have at least 1 input and 1 output
- Arduino must transmit sensor data from at least 1 input to MATLAB
- Arduino must react to data received from MATLAB
- MATLAB must utilize user Interface objects to provide information to the user
  - Specifically must include some images
- MATLAB must utilize user interface objects to take inputs from the user
- MATLAB must react to sensor data take from the Arduino
- MATLAB must transmit data to the Arduino
  - Must be important to the concept of your game but not necessarily a direct user input. Could be calculated values, randomized elements, or other such programmatically generated game details.

You must provide:

- A full representation of the logic for your completed game through a pseudocode or flowchart
- A complete and well-commented Arduino script
- A complete and well-commented MATLAB script

- A video of a user, other than yourself, playing an early version of your game and providing feedback
- A video that goes over the final version of your game with a detailed walkthrough noting the the major logic points
- A SHORT write-up covering the background/inspiration for your game and how the user feedback influenced the final design
- Images of your user interface and circuit

Game Recommendations:

Here are some recommendations for useful tips, commands and even some loose game ideas (you have to make sure your version meets all requirements) to help get you started.

- Look through the available MATLAB uifigure elements - https://www.mathworks.com/help/matlab/develop-apps-using-the-uifigure-function.html
- Go into FYELIC or ask your professor to see about different sensors and actuators that are available to expand your options for Arduino Inputs and Outputs
  - This might help inspire your game design as well
- Make use of random commands - https://www.mathworks.com/help/matlab/random-number-generation.html
  - `randi()` is the most commonly used as it generates a random integer, which is typically useful in most games
- A guessing game where MATLAB creates the random values and guesses get inputted via the Arduino. Using images in MATLAB or lights on the Arduino to indicate if you're higher or lower.
- A game that involves matching timing between MATLAB and Arduino, getting a gauge to land at a specific value and a servo to land at that same value, through inputs from either system component.
- A version of wordle with an external button to lock in guesses and LEDs indicating correct guesses
- Borrow a joystick and create a maze game with traps indicated by Arduino outputs.

I will add to this list as questions are asked and further recommendations are given out to individuals so everyone can benefit.

Submission Requirements:

- .ino file
  - The raw Arduino file
- .m file
  - The raw MATLAB file
- .pdf file
  - Flowchart or pseudocode
  - Write up of background/inspiration and influence of user feedback
  - Images of user interface
    - Include multiple to represent each variation if it changes
    - Include descriptive text as necessary (Most likely necessary)
  - Images of the Circuit
    - EC to use fritzing, tinkercad, or other means to create a clean digital version of your wiring diagram. Wire must have 90 degree bends and minimize overlap to make it easier to follow.
- Video files
  - Can be split into 2 or combined
  - If split, append '_Walkthrough' and '_usertest' to the filename to indicate which is which.

The following survey is for my reference to help improve future assignments. The results are not checked until after the semester is completed and participation is not required.

Rubric

P1: Serial Game - 100% Total

- Meets Requirements - 30%
  - Arduino component minimums
  - Arudino works as transmitter and receiver
  - MATLAB user interface requirements
  - MATLAB as transmitter and receiver
- Pseudocode/Flowchart - 30%
  - Logical Approach to problem
  - Well communicated/written
- Programming - 20%
  - Runs without errors
  - Appropriate Commands
  - Clean and clear UI
  - Commented throughout and cleanly