PROJECT AT KUMC PRECISION NEURAL DYNAMICS LAB:

OBJECT TRACK MATRIX

Presenter: Thresa Kelly

Authors: Thresa Kelly, Ryan Coppens, Dr. Alice Bean, & Dr. Adam Rouse

INTRODUCTION

Project Goal

- Design an activity board with software for monkey reach-and-grasp experiments
- 4x3 grid of motion sensors (12 total)
- LED lights up object → indicate to grab object
- Software to customize experiment and record real-time data

Background

- People with neurological diseases, movement disorders, or amputations may have difficulty controlling their body
  - New technologies and robotics can help!
- University of Kansas (KU) **Department of Neurosurgery Precision Neural Dynamics Lab** → research on how the brain controls the body
  - <u>Study brain/spinal cord activity</u> in animals → better understand how groups of neurons coordinate for…
    - Arm reaching
    - Hand grasping
  - Develop computational methods for <u>analyzing large datasets</u> from experiments

Project Motivation

- Object track matrix → reach & grasp experiments with rhesus monkeys
- **Research:**
  - Study neural encoding of precision movements → target-independent and target-dependent neural signal limitations of speed and precision
    - <u>Independent</u> = neurons that fire for any movement in a given motor area
    - <u>Dependent</u> = level of activity of specific neurons for movement direction
  - Neural mapping of hand function → brain maps specific hand grasp, then sends signal to hand via spinal cord

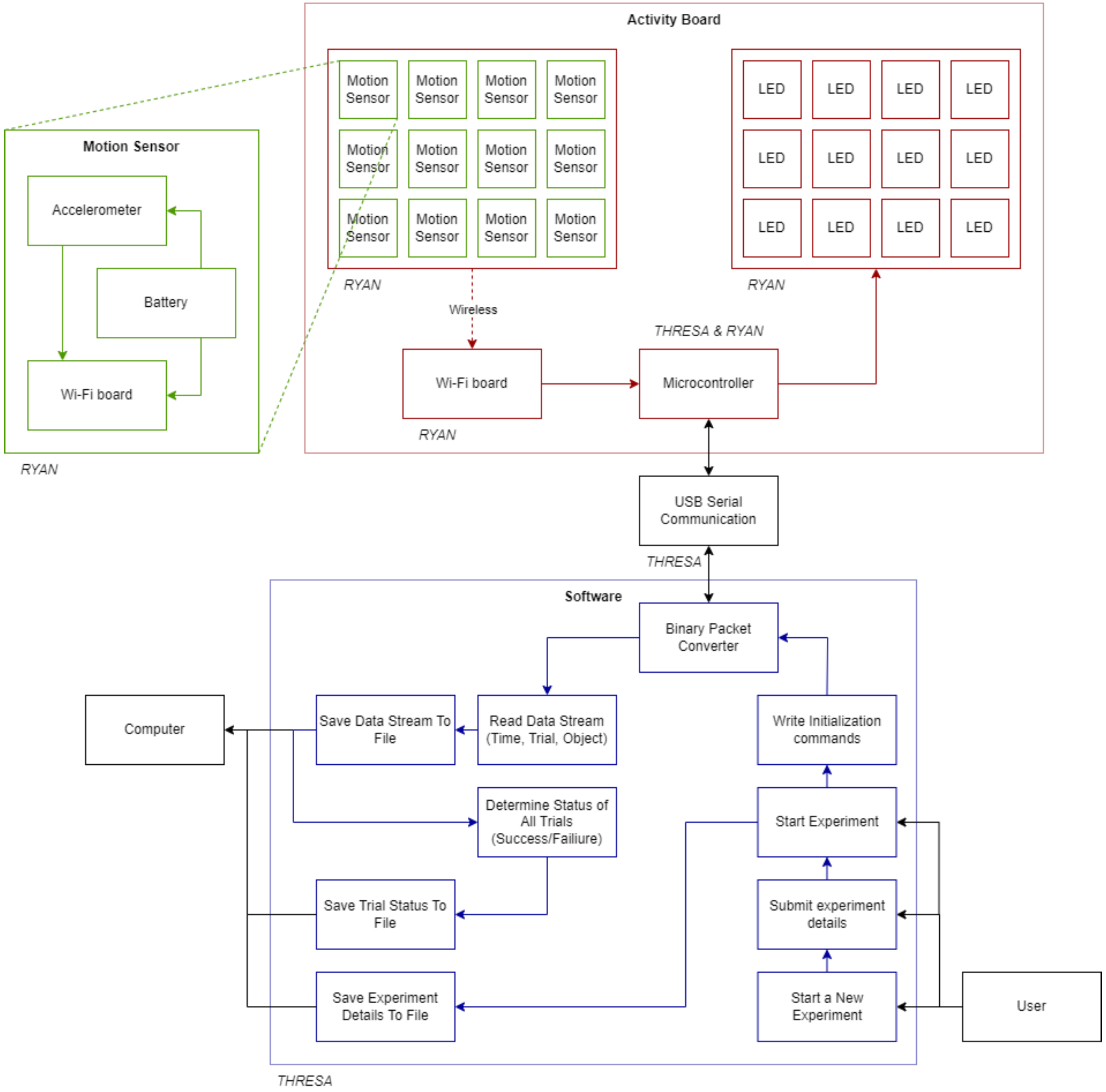
OVERALL DESIGN

Requirements (1/2)

- **Physical board:**
  - Objects must be able to be disconnected from the activity board.
  - Rods must be able to support objects of different size, shape, and weight.
  - The activity board must support 12 objects in a 4 x 3 grid.
- **Motion sensing:**
  - Accelerometer must be able to detect motion in one dimension.
  - Noise from the accelerometer must not be interpreted as an interaction from the monkey.
  - The motion sensor package must be wirelessly attached to the end of the rod with the object.

Requirements (2/2)

- **Lighting:**
  - Only one LED should turn on at a time.
  - The LED should turn on for a duration set by the user.
- **Software:**
  - The user must be able to plan an experiment by determining the LED sequence and timing durations.
  - The software must read from the hardware in real time.
  - The timing resolution of collected data must be accurate to 10 ms.
  - The software must determine if a trial was successful (the correct object was manipulated) or a failure (no object or the incorrect object was manipulated).

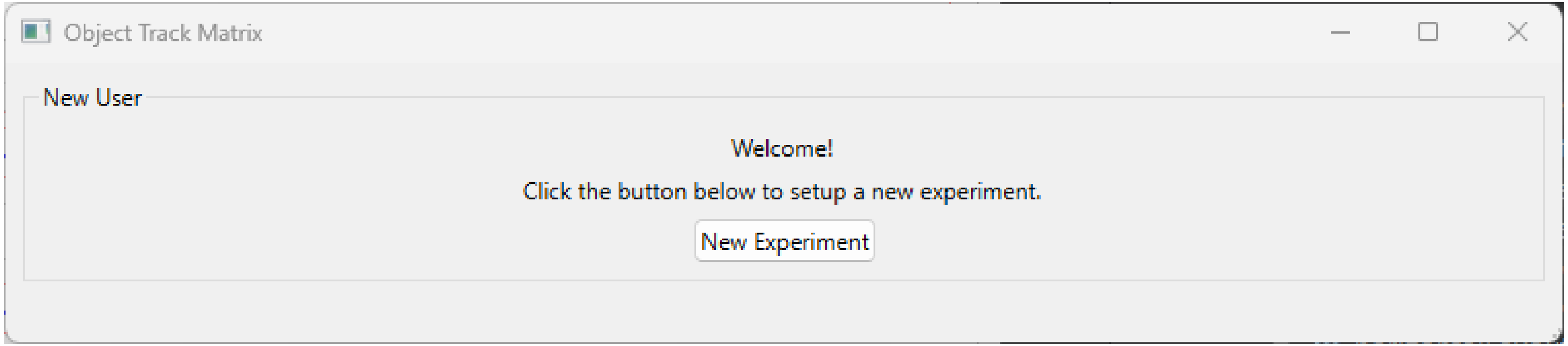


Control Flow

- Main components:
  - Software
  - Activity board
  - Motion Sensors
- Motion sensors communicate with the activity board via Wi-Fi
- The activity board communicates with the software via serial communication via USB port

MY CONTRIBUTIONS

User Interface: Welcome

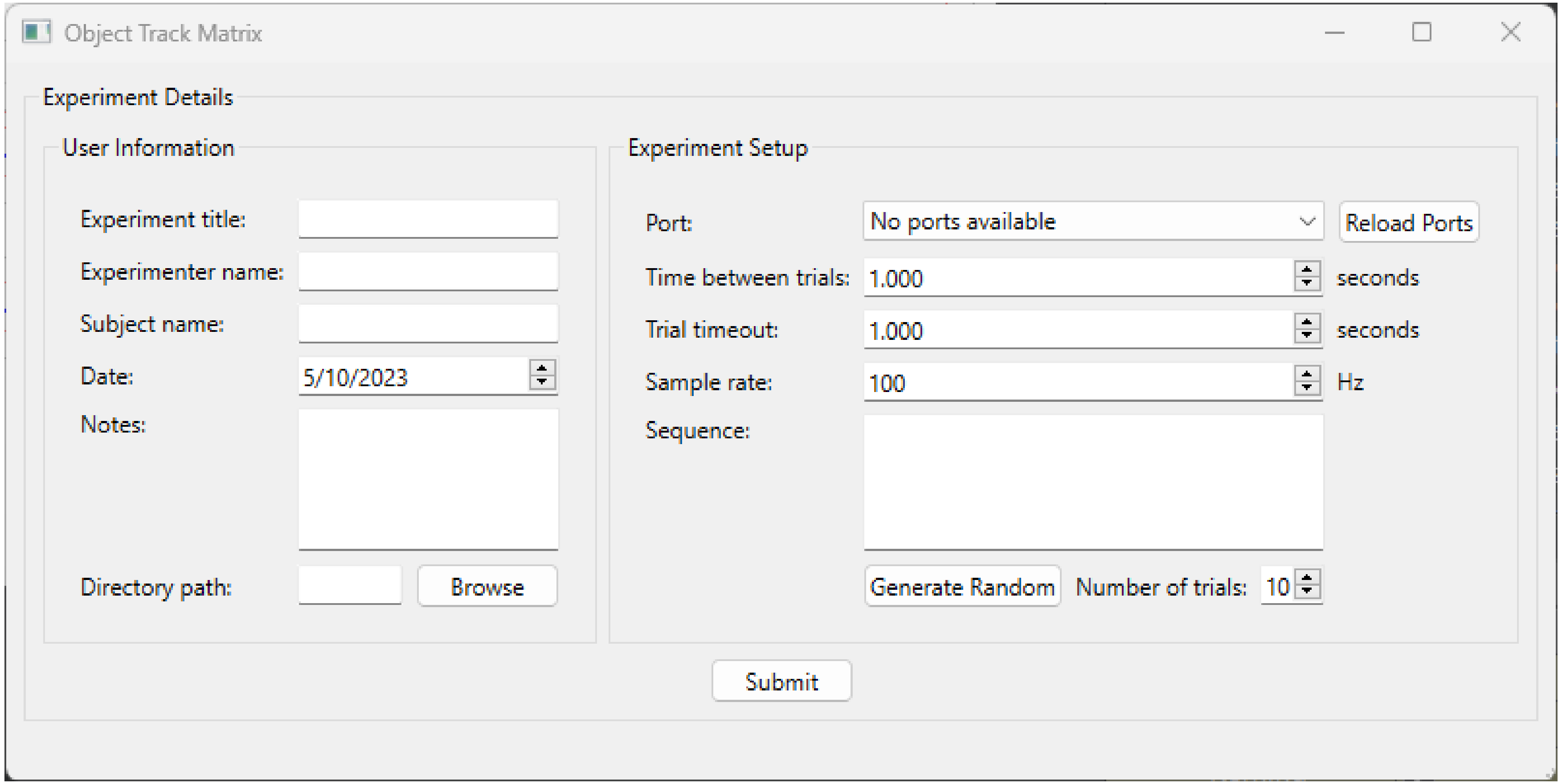


New User

Welcome!

Click the button below to setup a new experiment.

User Interface: Experiment Details



Object Track Matrix

Experiment Details

User Information

Experiment title:

Experimenter name:

Subject name:

Date: 5/10/2023

Notes:

Directory path:

Experiment Setup

Port: No ports available

Time between trials: 1.000 seconds

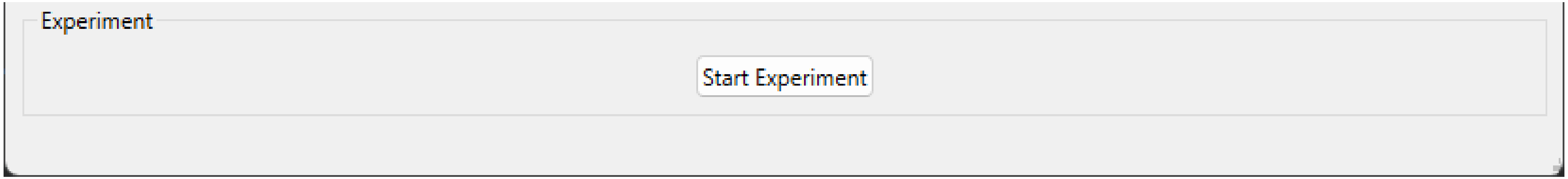
Trial timeout: 1.000 seconds

Sample rate: 100 Hz

Sequence:

Number of trials: 10

User Interface: Experiment

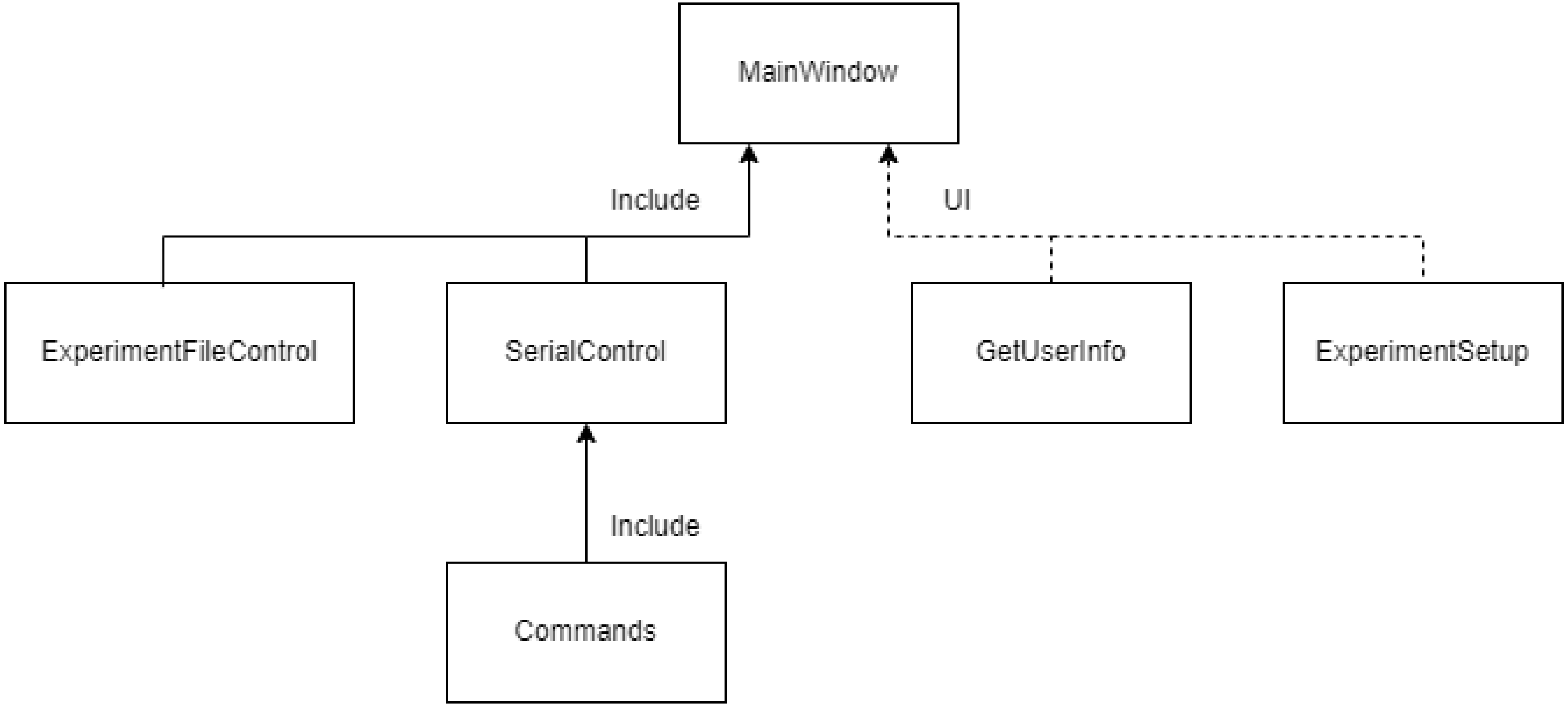


- Experiment data is saved to three files:
- <Directory Path>/<Experiment Name>/
  - Experiment_Information.csv
  - Experiment_Data_Stream.csv
  - Experiment_Data_Stream.csv

Software Internal Design

Class Name	Description
MainWindow	Top-level class and UI to request and submit experiment information, perform an experiment, and save data to files.
GetUserInfo	UI with user input fields for the experiment title, experimenter name, subject name, date, notes, and directory path. This information is accessed using getter functions.
ExperimentSetup	UI with user input fields for the serial port, time between trials, trial timeout, sample rate, and trial sequence. This information is accessed using getter functions.
Commands	Handles building and interpreting serial packets. Also has functions to convert integers to hexadecimal strings, check the command parameter requirements, and get the number of bytes of the packet components.
SerialControl	Handles reading and writing command packets through a serial port.
ExperimentFileControl	Writes files to the experiment directory to store the experiment information, streaming data, and trial status as *.csv files.

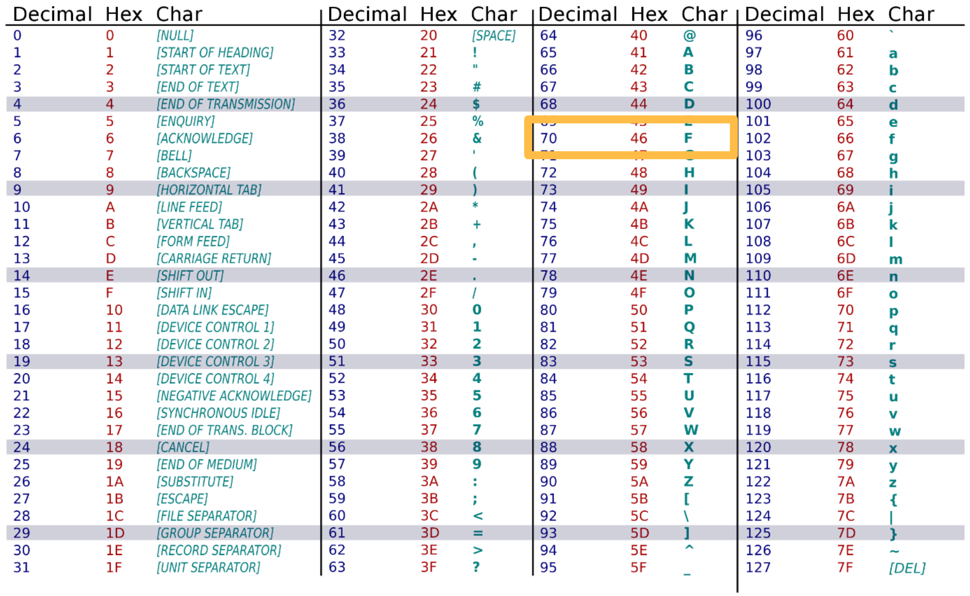
Class Structure



Serial Communication Protocol

Bytes	0	1	2	3	4	5	6	7
Component	STX	Command	ID	Data				ETX

- 1 byte = 8 bits = 1 ASCII character
- Building a packet:
  - Decimal integer → hexadecimal string (15 → “F”)
  - Hex character → ASCII code (“F” → 70)

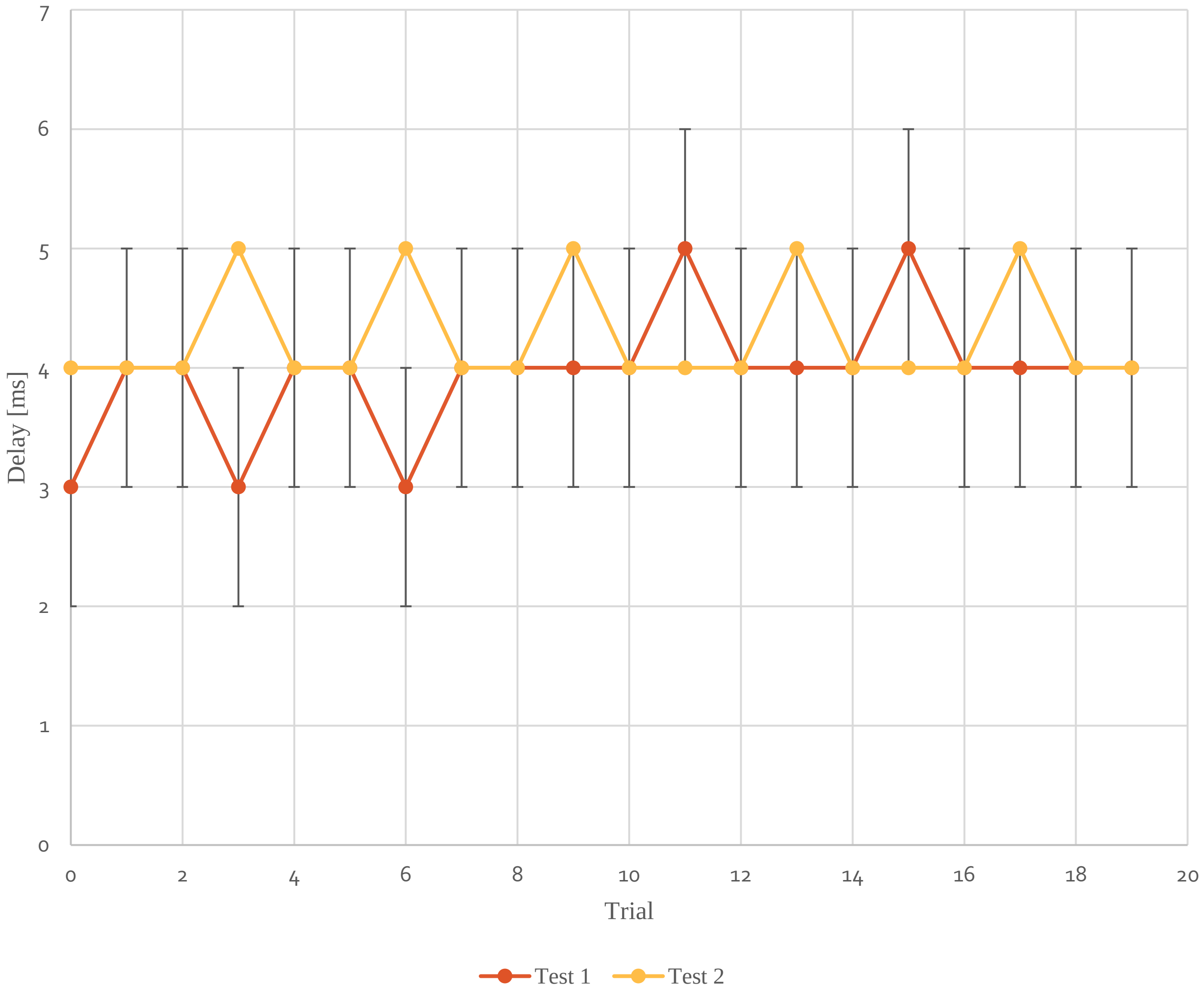


Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	*[NULL]*	32	20	*[SPACE]*	64	40	@	96	60	`
1	1	*[START OF HEADING]*	33	21	!	65	41	A	97	61	a
2	2	*[START OF TEXT]*	34	22	"	66	42	B	98	62	b
3	3	*[END OF TEXT]*	35	23	#	67	43	C	99	63	c
4	4	*[END OF TRANSMISSION]*	36	24	$	68	44	D	100	64	d
5	5	*[ENQUIRY]*	37	25	%	69	45	E	101	65	e
6	6	*[ACKNOWLEDGE]*	38	26	&	70	46	F	102	66	f
7	7	*[BELL]*	39	27	'	71	47	G	103	67	g
8	8	*[BACKSPACE]*	40	28	(	72	48	H	104	68	h
9	9	*[HORIZONTAL TAB]*	41	29	)	73	49	I	105	69	i
10	A	*[LINE FEED]*	42	2A	*	74	4A	J	106	6A	j
11	B	*[VERTICAL TAB]*	43	2B	+	75	4B	K	107	6B	k
12	C	*[FORM FEED]*	44	2C	,	76	4C	L	108	6C	l
13	D	*[CARRIAGE RETURN]*	45	2D	-	77	4D	M	109	6D	m
14	E	*[SHIFT OUT]*	46	2E	.	78	4E	N	110	6E	n
15	F	*[SHIFT IN]*	47	2F	/	79	4F	O	111	6F	o
16	10	*[DATA LINK ESCAPE]*	48	30	0	80	50	P	112	70	p
17	11	*[DEVICE CONTROL 1]*	49	31	1	81	51	Q	113	71	q
18	12	*[DEVICE CONTROL 2]*	50	32	2	82	52	R	114	72	r
19	13	*[DEVICE CONTROL 3]*	51	33	3	83	53	S	115	73	s
20	14	*[DEVICE CONTROL 4]*	52	34	4	84	54	T	116	74	t
21	15	*[NEGATIVE ACKNOWLEDGE]*	53	35	5	85	55	U	117	75	u
22	16	*[SYNCHRONOUS IDLE]*	54	36	6	86	56	V	118	76	v
23	17	*[END OF TRANS. BLOCK]*	55	37	7	87	57	W	119	77	w
24	18	*[CANCEL]*	56	38	8	88	58	X	120	78	x
25	19	*[END OF MEDIUM]*	57	39	9	89	59	Y	121	79	y
26	1A	*[SUBSTITUTE]*	58	3A	:	90	5A	Z	122	7A	z
27	1B	*[ESCAPE]*	59	3B	;	91	5B	[	123	7B	{
28	1C	*[FILE SEPARATOR]*	60	3C	<	92	5C	\	124	7C	|
29	1D	*[GROUP SEPARATOR]*	61	3D	=	93	5D	]	125	7D	}
30	1E	*[RECORD SEPARATOR]*	62	3E	>	94	5E	^	126	7E	~
31	1F	*[UNIT SEPARATOR]*	63	3F	?	95	5F	_	127	7F	*[DEL]*

Commands

Name	Number	Software		Hardware		Description
Testing		ID	Data			
PING	0					call and response to test communication
TEST LED	1	Object ID	Time (ms)			Command that turns on all LEDs to test functionality
BATTERY	2	Object ID		Object ID	Battery percent	Get the percent of the battery life of one motion sensor
Setup						
CALIBRATE	3	Object ID				Sends the object ID to be calibrated. The respective motion sensor is moved, and its ID saved.
NTRIALS	4		Number of trials			number of trials in the experiment
TRIAL	5	Object ID	Trial number			Sets the object ID for the trial. several of these commands comprise of the experiment object sequence.
SEPARATION	6		Time (ms)			Sets the time between trials. This is the time between an LED off and on.
TIMEOUT	7		Time (ms)			Sets the maximum time an LED can be on for a trial
SAMPLE RATE	8		Frequency (Hz)			Sets the sample rate
Experiment						
STREAM	9		True / False	Object ID	Trial number	Start (true) or stop (false) the experiment. Sends the object ID (1-12) of the object that was moved and the trial number. Object ID is 0 if no object is moved

TESTING



Read/Write Delay

- Computer time for:
  - the software to build a packet and send it via USB, t
  - then for the microcontroller to interpret, execute, rebuild a packet, and write back to software,
  - Then for software to unpack and check the packet
- Goal: t<10ms
- **Average: 4+/-1 ms**

Test Alll Objects

Trial	Did the correct LED ring turn on?	Did the motion sensor activate when moved?
1	Yes	Yes
2	Yes	Yes
3	Yes	Yes
4	Yes	Yes
5	Yes	Yes
6	Yes	Yes
7	Yes	Yes
8	Yes	Yes
9	Yes	Yes
10	Yes	Yes
11	Yes	Yes
12	No	No

- Sequence: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12.
- Write sequence to the activity board, then run experiment.
- Move each object when the LEDs light up.
- Goal: all objects must light up and respond correctly when moved.
- Result: all objects by #12 work.
  - Code bug when converting 12 to a hexadecimal character “C”…

CONCLUSION

Future work

- Fix bug with #12 integer to hex character conversion.
- Implement unused commands: TEST LED, BATTERY, and CALIBRATE
- Refactor serial communication protocol to include packet number
  - Protect against data loss
- Add real-time data visualization while running experiment

QUESTIONS?

Thanks!

DEMONSTRATION