РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ АВТОМАТИЗИРОВАННЫМ СТОЛОМ ДЛЯ НАСТОЛЬНОГО ФУТБОЛА

Гриценко К. В.[1], *Поберезкин Н. И.*[2]
[1] *ТПУ, ИШИТР, гр. 8Е11, e-mail: kvg15@tpu.ru*
[2] *ТПУ, ИШИТР, ассистент ОАР, e-mail: nip6@tpu.ru*

Аннотация

В рамках работы был спроектирован протокол передачи данных между одноплатным компьютером и микроконтроллером. Разработана система управления и рассчитана кинематика и динамика системы.

Ключевые слова: Настольный футбол, кикер, управление системой, кинематика, голономный.

Введение

В настоящее время на отделении автоматизации и робототехники ведется разработка автоматизированного тренажера для настольного футбола, представленного на рисунке 1. На прошлом этапе был разработан механизм, представленный на рисунке 2, осуществляющий вращательное и поступательное движения штанги, система управления моторами и PI-регулятор по контурам положения и скорости.

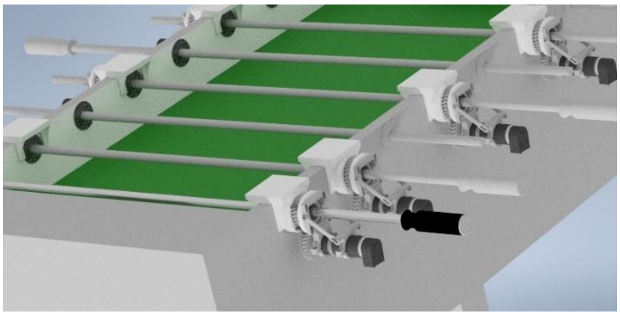

Рис. 1. Автоматизированный тренажер для настольного футбола

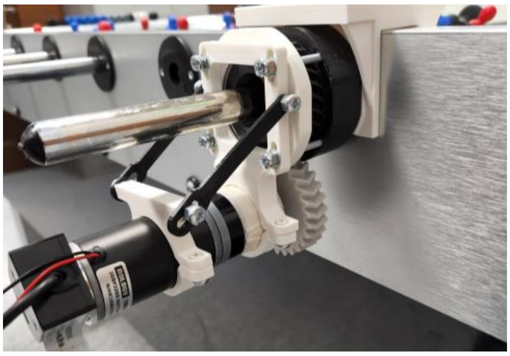

Рис. 2. Исполняющий механизм

В состав исполняющего механизма (рис. 2) входит безрезьбовой роликовый винт, ролики которого повернуты относительно оси вращения на 45°, представлен на рисунке 3. Пара таких роликов позволяет обеспечить как вращательное, так и поступательное движение.

Структурная схема системы представлена на рисунке 4, где n_i – сигналы, передаваемые между контроллером среднего уровня и контроллером нижнего уровня, а ω_s – угловая скорость входного вала мотора.

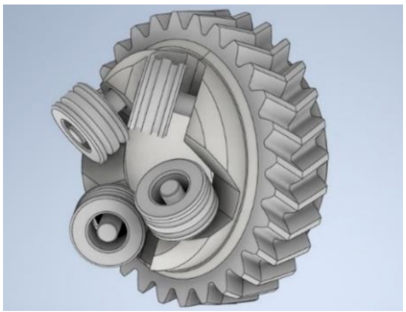

Рис. 3. Безрезьбовой роликовый винт

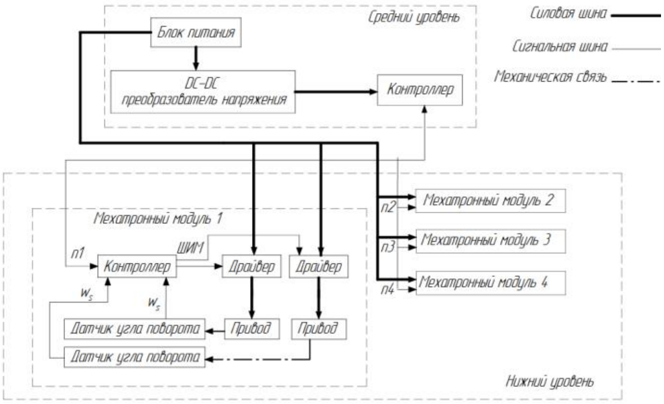

Рис. 4. Структурная схема системы

Поскольку уже есть реализованные регуляторы и механическая система, то следующим этапом проекта становится организация управления системой. Для этого нужен средний уровень, с которым нужно наладить связь и обеспечить точную и бесперебойную передачу данных. Для решения этой задачи было решено посчитать механику и кинематику используемой в механизме передачи, изучить и реализовать протоколы связи между микроконтроллерами и методы верификации данных при передаче.

Расчеты механики и кинематики системы

Расчет механики требуется в целях определения основных характеристик моторов, которые будут использоваться в системе, чтобы обеспечить предъявляемые к ней требования. Для этого нужно отдельно провести расчеты вращательного и поступательного движений, а также определить КПД передачи.

В расчетах вращательного движения штанги использовались формулы (1), (2), (3) и (4).

$$\varepsilon = 2 \cdot \frac{\varphi}{t_{уск}^2}, \tag{1}$$

$$\omega = \varepsilon \cdot \frac{t_{уск}}{360} \cdot 60, \tag{2}$$

$$J = m \cdot R^2, \tag{3}$$

$$M = J \cdot \varepsilon, \tag{4}$$

где φ – угол, который проходит игрок для удара, $t_{уск}$ – время ускорения.

Для расчета линейного движения штанги использовались формулы (5), (6), (7) и (8).

$$a = \frac{v}{t}, \tag{5}$$

$$F = m \cdot a, \tag{6}$$

$$A = F \cdot S, \tag{7}$$

$$P = \frac{A}{t}. \tag{8}$$

После того, как были посчитаны основные параметры для каждого из видов движений, был выполнен расчет КПД передачи, которая представляет собой безрезьбовой роликовый винт, ролики которого повернуты на 45^{o} относительно оси вращения и придают валу необходимое движение.

На рисунке 5 изображен упрощенный вид конструкции, где каждый из роликов имеет с валом определенный натяг F_{ni}. Где максимальная выходная сила $F_{t\max}$ равна сумме всех четырех нормальных сил, помноженных на статический коэффициент трения μ_s.

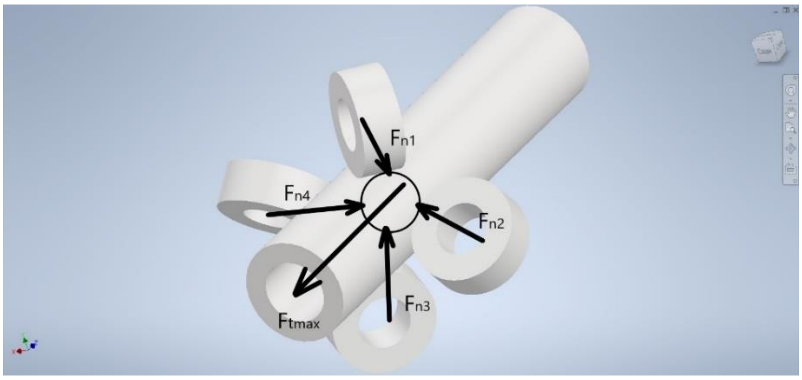

Рис. 5. Упрощенный вид конструкции

Для нахождения КПД передачи была использована формула (7), а также формулы КПД (9) и потерь крутящего момента из-за трения (10) КПД передачи можно найти из выражений 7 и 9, ещё стоит учесть потери крутящего момента по причине возникновения трения качения между роликами и штангой, они представлены в выражении 10, которые представлены ниже:

$$\eta = \frac{\tau_{входной} - \tau_{потерь}}{\tau_{входной}}, \tag{9}$$

$$\tau_{F_{TP}} = r \cdot \mu_r \cdot F_N, \tag{10}$$

где, μ_r – коэффициент трения качения, τ – крутящий момент.

В результате математических преобразований было получено КПД, равное 69,4 %. Следовательно, из полученных крутящего момента, частоты оборотов и мощности, с учетом КПД передачи, появилась возможность подобрать для проекта те моторы, которые смогут обеспечить требуемые показатели системы.

Вследствие того, что вал при линейном движении описывает винтовую траекторию, то величина шага вала, с учетом того, что радиус ролика равен радиусу вала, определяется как:

$$L = 2 \cdot \pi \cdot r \cdot \tan(\theta) \tag{11}$$

Из уравнения (11) было определено, что шаг вала за один оборот винта равен 5 см.

Для проверки полученной величины шага был проведен эксперимент с прототипом, по результатам которого был получен шаг, равный 4,62 см. Таким образом, погрешность составила 7,6 %, что может быть связано с наличием проскальзывания роликов, а также неровности вала.

После проведения расчётов необходимо было обеспечить взаимодействие среднего и нижнего уровня.

Анализ методов передачи данных

Для осуществления управления столом было решено использовать, в качестве среднего уровня, одноплатный компьютер Raspberry PI 4, который должен подключаться ко всем микроконтроллерам нижнего уровня, принимать с них данные о состоянии системы и отдавать им команды на выполнение различных действий.

Основными требованиями к передаче данных в разрабатываемой системе являются минимальная зависимость от длины линии связи, высокая скорость передачи и помехозащищенность передаваемого сигнала, поскольку управление столом осуществляется за счёт приводов, которые синтезируют электромагнитные помехи. Исходя из этого, в рамках работы был проведен анализ следующих интерфейсов:

– SPI;
– I2C;
– CAN;
– USB 2.0.

Информация о каждом из интерфейсов представлена в источниках [1–6].

По итогу проведенного анализа был выбран протокол USB 2.0. Такой выбор обоснован высокой скоростью передачи данных, возможностью передавать информацию нескольким ведомым устройствам и устойчивостью к помехам.

SPI протокол менее предпочтителен из-за условий передачи данных, а именно длина линии передачи данных и слабой защищенностью от электромагнитных помех со стороны используемых моторов.

I2C протокол также как SPI имеет плохую помехозащищенность, что скажется на качестве передачи данных.

CAN протокол наравне с USB имеет отличную помехозащищенность, однако из-за сложности реализации и необходимости в дополнительной конфигурации, в виде конвертера CAN для Raspberry PI, был выбран USB 2.0, как наиболее простой, надежный и быстрый протокол.

Анализ методов подтверждения и верификации при передаче данных

При использовании различных методов контроля информации, можно встретить определённый вид баланса, когда высокая эффективность нахождения ошибок влечет за собой дополнительное процессорное время, а менее эффективное – высокую скорость работы и меньшую затратность на ресурсы процессора. Поэтому необходимо провести анализ существующих алгоритмов решения данной задачи для обеспечения максимальной точности передачи данных между средним и нижним уровнями.

В рамках данной работы были рассмотрены следующие методы подтверждения и верификации при передаче данных:

– Проверка четности (Parity checking);
– Код Хэмминга;
– Контрольная сумма;
– CRC (Циклический избыточный код).

Из самых часто встречаемых ошибок при передаче данных являются:

– Ошибка отдельного бита - когда один из битов инвертирован;
– Ошибка неупорядоченности – когда отправляются те же самые символы, но в неверном порядке;
– Пакетная ошибка – повреждение сразу нескольких битов.

Основная информация по “Коду Хэмминга” и CRC представлена в источниках [7-9].

После того, как методы подтверждения и верификации были рассмотрены, была составлена сравнительная таблица 1.

Проведя анализ методов и сравнив их, был сделан вывод, что наиболее подходящим методом будет являться аппаратный CRC (Циклический избыточный код). Такой выбор обусловлен тем, что его эффективность выявления однобитных и двухбитных ошибок составляет 100 %, а для многобитных – 99 %. Также, известно, что в используемом протоколе USB 2.0 есть встроенный CRC, обеспечивающий постоянное определение ошибок и отправку запросов на повторную отправку неверно пришедших данных. Также, аппаратная реализация позволит поддержать высокую скорость обработки данных при минимальных ресурсных затратах.

Таблица 1

Сравнительна таблица методов подтверждения и верификации при передаче данных

Метод	Сложность реализации (1 – сложная; 5 – легкая)	Используемые ресурсы (1 – много; 5 – мало)	Работа с разными объемами данных (1 – только с малыми объемами данных; 5 – от малых до больших)	Эффективность нахождения ошибок (1 – низкая; 5 – высокая)	Способ исправления ошибок
Parity checking (проверка четности)	3	4	2	3	Исправление битовых ошибок
Расстояние Хэмминга	2	3	3	4	Собственный метод исправления битовых ошибок
Контрольная сумма	3	3	3	3	Запрос на повторную отправку
CRC	2	2	5	5	Запрос на повторную отправку

Реализация системы управления

Следующим этапом работы является реализация системы управления на существующем прототипе.

Первоначально необходимо обеспечить передачу величины уставки регулятора с одноплатного компьютера на микроконтроллер и декодировку этих данных. На рисунке 6 происходит отправка данных с одноплатного компьютера, на рисунке 7 прием данных на аппаратной платформе STM, в ПО STMStudio.

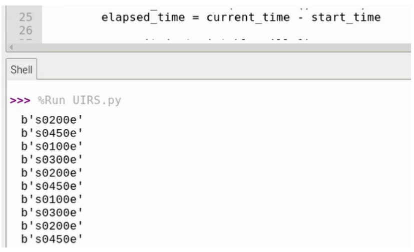

Рис. 6. Отправка данных с Raspberry PI 4

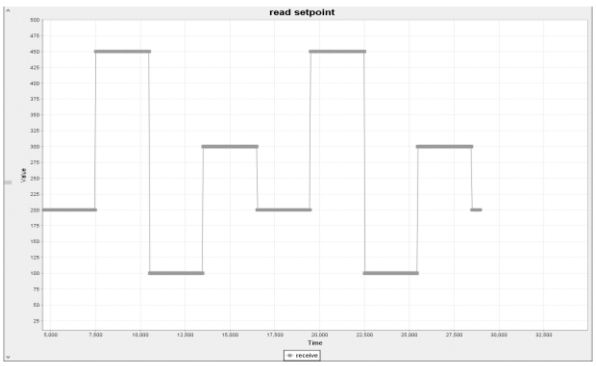

Рис. 7. Прием данных на STM32

Вместе с этим, в рамках выполняемой работы была реализована передача значения текущего угла поворота штанги, а также алгоритм отработки аварийной ситуации – разрыв соединения между одноплатным компьютером и микроконтроллером. Реакцией системы на такую ситуацию является ожидание переподключения устройства к порту USB одноплатного компьютера, в случае восстановления соединения между компьютером и микроконтроллером, порт снова открывается, и передача данных возобновляется. Отработка данной ситуации показана на рисунках 8, 9, 10. На рисунках изображен стенд, состоящий из Raspberry PI 4 Model B, STM32F103C8T6, блока питания и мотора постоянного тока с драйвером. Микроконтроллер в такой ситуации заканчивает последнее данное ему задание и ожидает повторного подключения и получения новых данных со среднего уровня.

Рис. 8. Включенное состояние системы

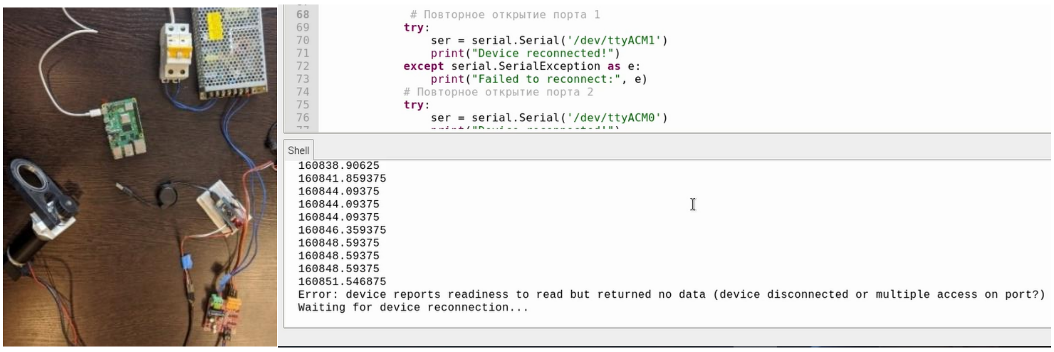

Рис. 9. Разрыв линии связи

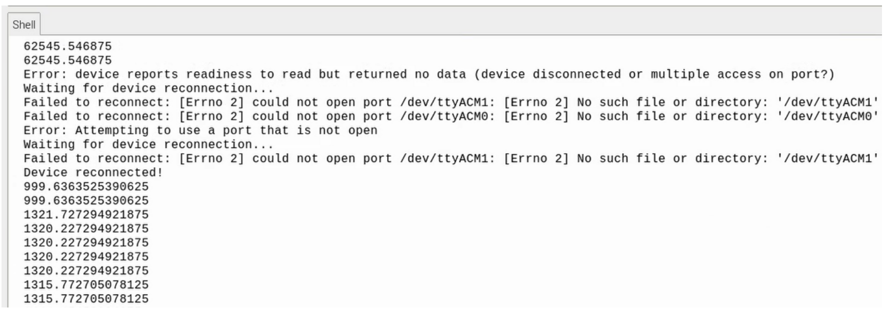

Рис. 10. Восстановление подключения между одноплатным компьютером и микроконтроллером

Заключение

В рамках данной работы был проведен анализ методов передачи данных между одноплатным компьютером Raspberry PI 4 и микроконтроллерами семейства STM32, а также методов верификации и подтверждения данных при передаче. По результатам анализов было решено применить USB интерфейс и аппаратно встроенный в него CRC, так как при данной реализации передача данных будет осуществляться на высокой скорости, точно и непрерывно, и с хорошей помехозащищенностью.

Также, в рамках работы были проведены расчеты механики и кинематики механизма. Посчитанные крутящий момент, частота оборотов, мощность и КПД передачи позволило сделать выводы о том, какие моторы необходимо использовать для обеспечения требуемых показателей системы. Расчеты кинематики заключались в определении шага вала при одном обороте безрезьбового роликового винта. Определив шаг, 5 см, был проведен эксперимент с прототипом, где реальный шаг составил 4,62 см, такая погрешность обусловлена проскальзыванием роликов о штангу, и неровностью вала. В дополнении к этому, в будущем планируется реализовать контур тока с последующей интеграцией в систему и согласованное управление двумя штангами настольного футбола.

Список использованных источников

1. BASICS OF THE SPI COMMUNICATION PROTOCOL. [Электронный ресурс]. – URL: https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
2. BASICS OF THE I2C COMMUNICATION PROTOCOL. [Электронный ресурс]. – URL: https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/
3. Интерфейс передачи данных - I2C. [Электронный ресурс]. – URL: https://3d-diy.ru/wiki/arduino-moduli/interfeys-peredachi-dannykh-i2c/
4. Ultimate CAN Bus Guide 2023: A Detailed Look at the Protocol. [Электронный ресурс]. – URL: https://www.autopi.io/blog/can-bus-explained/#:~:text=One%20ECU%20can%20formulate%20and,(CANL%20and%20CAN%20H).
5. Интерфейс USB. Полный обзор и структура пакетов данных. [Электронный ресурс]. – URL: https://microtechnics.ru/osnovy-interfejsa-usb/
6. About the USB Protocol, Common USB Bus Errors, and How to Troubleshoot Them. [Электронный ресурс]. – URL: https://www.totalphase.com/blog/2020/07/about-the-usb-protocol-common-usb-bus-errors-and-how-to-troubleshoot-them/
7. Код Хэмминга. [Электронный ресурс]. – URL: https://habr.com/ru/articles/140611/
8. CRC. [Электронный ресурс]. – URL: https://www.totalphase.com/blog/2020/07/about-the-usb-protocol-common-usb-bus-errors-and-how-to-troubleshoot-them/
9. CRC. [Электронный ресурс]. – URL: https://radiohlam.ru/crc_calc/