Задание: необходимо построить ПИД-регулятор к объекту управления, который описывается передаточной функцией

$$W(s) = \frac{b_0}{s^2 + sa_1 + a_0} \tag{1}$$

В задании необходимо выполнить ряд ключевых действий, присущих процессу разработки программы для регуляторов. Реальный объект управления, как правило, представляет собой физическое устройство, оснащенное датчиками. Взаимодействие с датчиками преимущественно осуществляется через цифровой протокол обмена данными. Таким образом, чаще всего в реальной жизни от объекта управления принимается массив байт, которые необходимо интерпретировать в соответствии с форматом протокола.

1. Получить данные с объекта управления (в программе – объект типа ControllerBackend). Для этого необходимо подключаться к соответствующему сигналу). Синтаксис подключения следующий:

   ```
   QObject::connect(emitter_address, signal_address, catcher_address, slot_address);
   ```

   Для того чтобы концепция сигнал-слот работала, необходимо, чтобы оба объекта были унаследованы от *QObject* (как в предыдущем семестре было сделано в задании с фильтрами). После подключения сигнала к слоту каждый раз, когда сигнал издается, запускается функция в слоте. Таким образом, необходимо создать свой класс, в котором будет производиться вся дальнейшая обработка, унаследоваться от *QObject*, создать слот и подключиться к сигналу. Каждый раз, когда объект управления испускает сигнал об изменении состояния, подключенный слот будет запускаться. В этом слоте необходимо рассмотреть данные, которые приходят от объекта управления. Для этого можно использовать синтаксис вывода в консоль, предварительно подключив соответствующую библиотеку (*# include <QDebug>*):

   ```
   qDebug() << someVariable;
   ```

2. Разобрать данные, которые приходят с объекта управления в виде массива байтов *QByteArray*. Каждое сообщение содержит стартовый байт, тип команды, непосредственно сами данные и проверочную сумму. Требуется определить, каким образом рассчитывается проверочная сумма, и реализовать проверку корректности сообщений. Некорректные сообщения необходимо отбрасывать.

3. Для последующей обработки данных в программной среде Matlab необходимо настроить сохранение логов. Для этого используется *QFile* и *QTextStream*, позволяющие осуществлять запись данных в текстовом виде в файл.

4. Написать скрипт в среде Matlab для чтения сохраненного лога, например, с использованием функций *fileread*, *strsplit*, *str2double*. Для проверки правильности отобразить данные и сравнить с тем, что было отображено в программе при записи лога.

5. Идентифицировать параметры объекта управления (1) с использованием метода наименьших квадратов. Для этого провести численное дифференцирование сигналов для получения значений скорости и ускорений. Предварительно данные следует отфильтровать с использованием функции *lowpass* и рассчитать значение частоты среза для фильтра низких частот. Этот фильтр впоследствии необходимо реализовать в коде для фильтрации сигнала выхода с объекта управления перед подачей на регулятор. Для получения корректных параметров необходимо задать вход объекта управления вручную (с использованием курсора) или с использованием подготовленной функции (желательно содержащей как можно больше гармоник). Перед реализацией регулятора необходимо согласовать с преподавателем полученные параметры во избежание неправильной настройки регулятора.

6. Настроить в среде Simulink ПИД-регулятор, например, с использованием специальных инструментов (блок PID Controller). Перевести полученную модель регулятора в вид Вход-Состояние-Выход (непрерывный или дискретный) для последующей реализации в коде. Основное требование: время переходного процесса должно составлять не более трех секунд.

7. Реализовать рассчитанный на предыдущем пункте регулятор в коде. На вход регулятора подавать ошибку между задающим сигналом и отфильтрованным выходом. Проверить качество регулятора на основе следующего задающего сигнала (на интервале до 60 секунд):

$$70 - 10e^{-0.1t}\Big(\frac{-(t-2)^2 + 60t}{100}\Big)\sin\big(\frac{t}{2}\big) \tag{2}$$

Для реализации экспоненты, синуса и степени в C++ использовать функции *std::exp()*, *std::sin()*, *std::pow()* соответственно. Результирующий лог необходимо загрузить в Matlab и рассчитать качество управления:

$$E = \sum_{i=0}^{n}(r[i] - y[i])^2, \tag{3}$$

где r – задающее воздействие, y – выход объекта управления. Для получения усредненного значения применить формулу

$$\overline{E} = \sqrt{\frac{E}{n}}. \tag{4}$$