УДК 378.162.33

ББК 39.5

В. М. Попов

Иркутск, Россия

С. В. Здрачук

Иркутск, Россия

УЧЕБНЫЙ ТРЕНАЖЕР КАБИНЫ ВЕРТОЛЕТА МИ-8Т НА БАЗЕ АВИАЦИОННОГО СИМУЛЯТОРА

В статье рассматривается применение авиационного симулятора X-plane для имитации динамики полета и аппаратной среды Arduino для создания имитаторов системы управления, приборного и пилотажно-навигационного оборудования при разработке тренажера кабины вертолета Ми-8Т.

Ключевые слова: авиационный симулятор, учебный авиационный тренажер, аппаратная среда Arduino

V. M. Popov

Irkutsk, Russia

S. V. Zdrachuk

Irkutsk, Russia

THE HELICOPTER COCKPIT TRAINING SIMULATOR OF MI-8T BASED ON THE AIRCRAFT SIMULATOR

The application of aircraft simulator X-plane for simulating flight dynamics and Arduino hardware environment for creating control system simulators, instrument and flight navigation equipment during the development of the cockpit of Mi-8T simulator is presented in the article.

Keywords: aircraft simulator, training aircraft simulator, Arduino hardware environment

ВВЕДЕНИЕ

Авиационный (пилотажный) тренажер – это имитатор полета, предназначенный для наземной подготовки пилотов. В авиационном тренажере (АТ) имитируется динамика полета и работа самолетных систем и двигателей с помощью специальных моделей, реализованных в программном обеспечении вычислительного комплекса тренажера.

АТ получил большое распространение в гражданской авиации по двум причинам. Во-первых, АТ позволяет сделать подготовку пилотов абсолютно безопасной, т. к. подготовка пилотов в реальном полете всегда связана с повышенным риском. Кроме этого, тренажер позволяет сделать безопасной отработку нештатных ситуаций, некоторые из которых либо крайне опасны для отработки в реальном полете, либо вообще их отработка в реальном полете запрещена законодательно. Во-вторых, АТ позволяет сэкономить значительные финансовые средства на летной подготовке экипажей ввиду того, что стоимость эксплуатации реального самолета кратно превосходит стоимость эксплуатации тренажера (несмотря на высокую стоимость современных тренажеров, порой превышающую 10 миллионов долларов).

АТ используется для теоретической подготовки как летного, так и инженерно-технического состава.

а) Для летного состава:

- изучение теоретического материала по конструкции вертолета;
- закрепление теоретических знаний по эксплуатации бортовых систем на программном тренажере;
- отработка первоначальных навыков и умений по выполнению полета на боевое применение, в том числе в особых случаях полета;
- процедурный контроль знаний.

б) Для инженерно-технических специалистов:

− изучение назначения, состава, размещения и принципа работы вертолетного оборудования;

− проверка и настройка бортового оборудования;

− техническое обслуживание вертолета.

Одним из основных требований при работе современных АТ является обеспечение полной идентичности работы имитаторов приборов и кабинного оборудования реальным самолетным системам. Точная и качественная симуляция работы приборов и кабинного авиационного оборудования является неотъемлемым условием получения правильных навыков обучения на авиационном тренажере.

Российское тренажеростроение в настоящее время представлено, в основном, центром научно-технических услуг «Динамика», ЗАО «Транзас», компанией ЗАО «Р.Е.Т. Кронштадт», которые разрабатывают, модернизируют и производят комплексные тренажеры различных уровней практически всех модификаций вертолетов марки Ми и Ка. Разработчиком авиасимуляторов выступает компания «Модернизация авиационных комплексов» (М.А.К.). Сегодня в России ведущими предприятиями по разработке и производству авиационных тренажеров являются ЦНТУ «Динамика» и ЗАО «Транзас».За время работы на рынке авиационных тренажеров «Транзас» создал тренажеры нового поколения с использованием современных электронных технологий, разработанных и запатентованных компанией. По заказу отечественных и зарубежных партнеров было освоено серийное производство тренажеров вертолетов всего семейства Ми-8/Ми-17. Недавно ЗАО «Транзас» сдал в эксплуатацию для компании UTair еще два комплексных тренажера вертолетов Ми-8Т и Ми-8МТВ [Попов, 2017].

В настоящее время на кафедре авиационных электросистем и пилотажно-навигационных комплексов Иркутского филиала МГТУ ГА реализуется проект по созданию учебного тренажера на базе кабины вертолета Ми-8Т. Этот тренажер будет имитировать приборные доски, пульты и органы управления

вертолетом и предназначен для изучения вертолетных систем и отработки процедур управления вертолетом. Учебный тренажер позволит проводить обучение инженерно-технического персонала по эксплуатации, контролю систем вертолета и запуску двигателей.

В основе предлагается использовать авиационный симулятор на базе X-plane для имитации динамики полета вертолета Ми-8Т и аппаратную средуArduino для управления системами и симуляторами аналогового приборного и пилотажно-навигационного оборудования вертолета.

Внешний вид кабины вертолета представлен на рисунке 1.

Железная часть тренажёра представляет из себя макет рабочего места КВС со всеми реальными органами управления вертолетом и приборной панелью, персонального компьютера, системы шумовой имитации и мультимедийного проектора для имитации визуального закабинного пространства.

Рассмотрим процесс разработки тренажера кабины вертолета Ми-8Т с применением авиационного симулятора X-plane для имитации динамики полета и аппаратной среды Arduino для управления системами самолета и вывод полетной информации на имитаторы приборного оборудования.

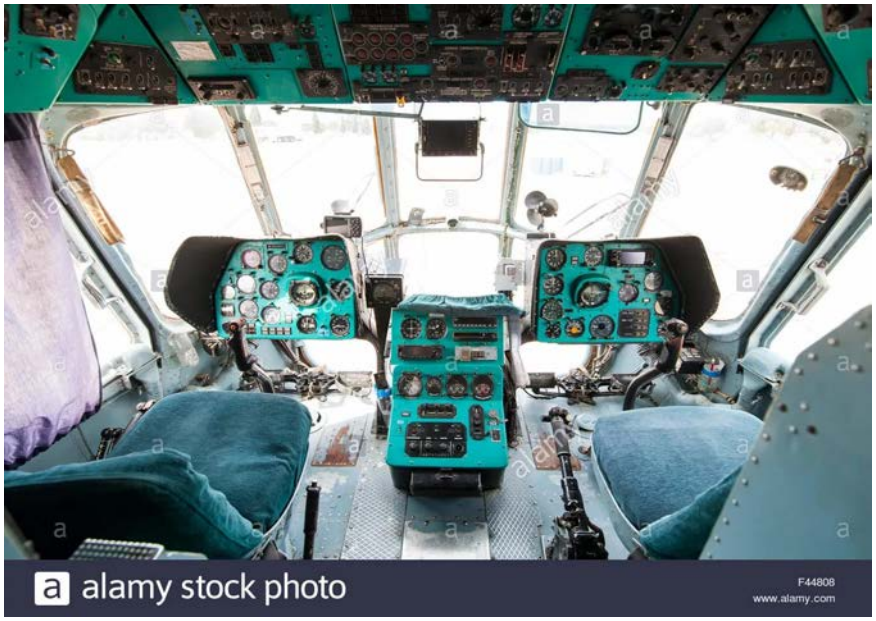

Рисунок 1 – Кабина вертолета Ми-8Т

Структура учебного тренажера кабины вертолета Ми-8Т представлена на рисунке 2.

Для того, чтобы управлять вертолетом, необходимо было подключить следующие органы управления: ручку управления, ручку «Шаг-Газ», педали, тумблеры и кнопки, расположенные на приборных панелях. Для этой цели были выбраны платы Arduino, которые представляют собой печатные платы с микроконтроллером фирмы Atmel, как сердце платы. Для написания программ и отправки их в микроконтроллер используется программная среда Arduino IDE, разработанная создателями плат.

Она представляет стандартную среду для написания программ с учетом того, что в ней уже находятся библиотеки, которые поддерживают множество сторонних устройств (серводвигатели, шаговые двигатели, различные датчики и т. д.), что значительно упрощает программирование микроконтроллера.

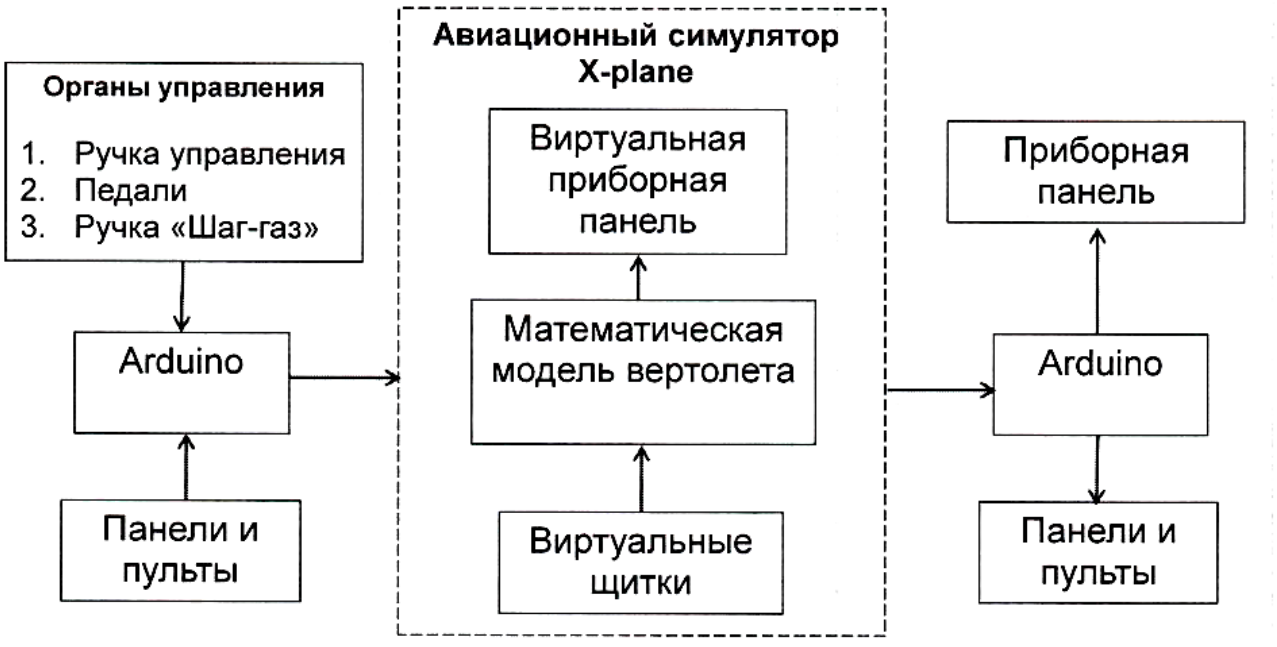

Рисунок 2 – Структура учебного тренажера кабины вертолета Ми-8Т

На рисунке 3 показана плата Arduino MEGA 2560, которая используется в работе.

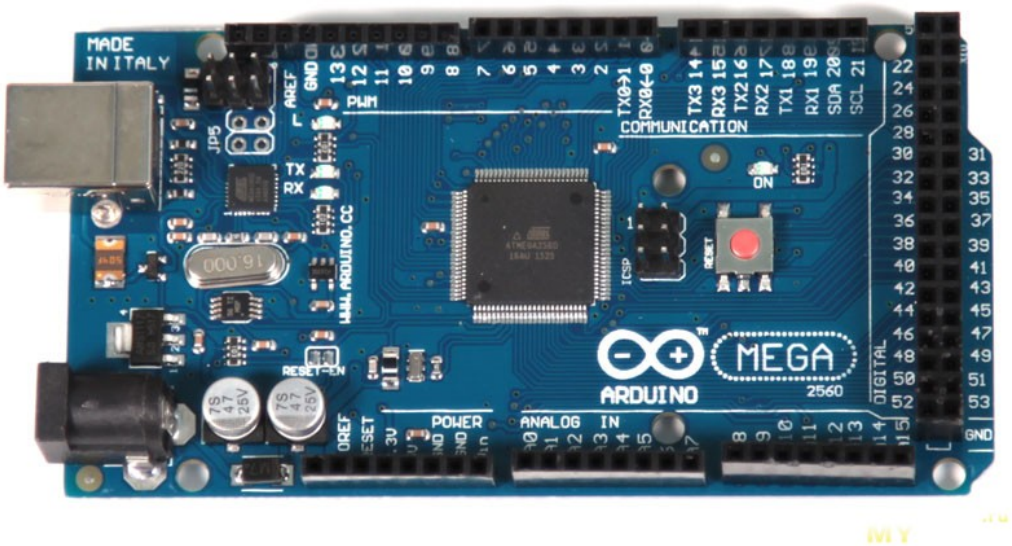

Рисунок 3 – Плата Arduino MEGA 2560

В чистом виде плата не функционирует, для ее работы нужна прошивка. Но и этого мало. Чтобы плата и симулятор корректно работали – требуется маленькая подпрограмма в симулятор. Такая подпрограмма называется – плагин.

В интернете существует несколько бесплатных проектов, которые реализуют такие задачи, но наиболее технологичным оказался проект ArdSimX [Попов, 2017]. Данный проект уже включает в себя плагин и прошивку для Arduino, а так же имеет онлайн-конфигуратор, где, выбрав плату (Uno, Nano, Mega, Micro, Mini (Pro)), можно с помощью мышки настроить нужный выход платы на нужный параметр. Платы работают через USB (используется для создания виртуального COM-порта и общения через него) и через интернет (требуется докупить интернет-плату для Arduino). Данный комплекс поддерживает до 9 плат каждого типа (при использовании самой большой – MEGA, возможно управлять около 600 параметрами самолета).

Немаловажным качеством данной среды является то, что здесь после установки плагина (требуется перенести папку с плагином в определенную папку в симуляторе) и записи прошивки на плату (записывается один раз, даже если количество параметров меняется) – больше ничего не требуется.

1. ПРОЕКТ ArdSimX

Как уже было сказано, есть конфигуратор, в котором задаются параметры для работы. На рисунке 4 показано окно этого конфигуратора.

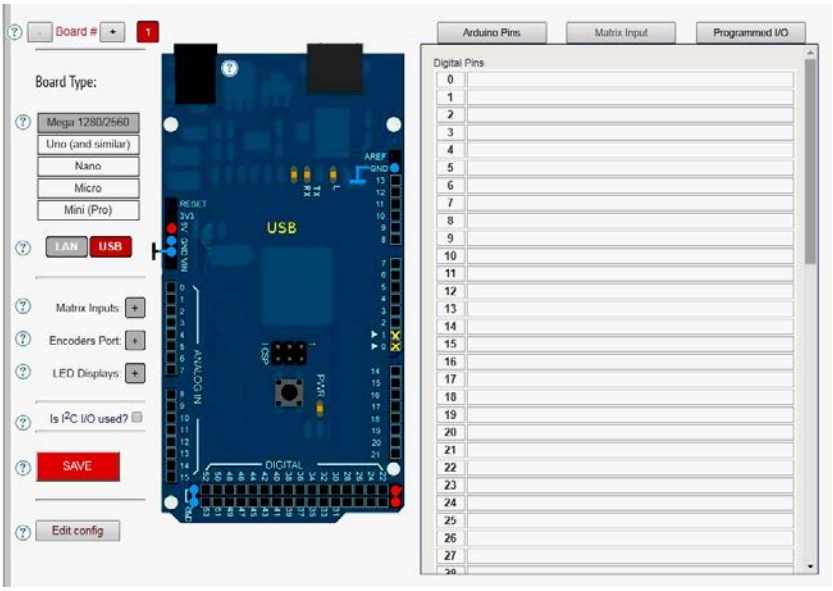

Рисунок 4 – Конфигуратор ArdSimX

После настройки конкретной ножки на конкретный параметр (на плате в реальном времени показывается, какие ножки заняты, а какие свободны) и нажатием на кнопку SAVE будет создан файл data.cfg (открывается с помощью Блокнота). Структура файла показана на рисунке 5.

```
Board #2 - Arduino (USB) ------------------------------------------------------

---- Digital pins: ----

22 - switch_invertor_115_light
23 - first_jack_on
24 - switch_emer_invertor_light
25 - invertor_works_115
26 - main_hyd_on
27 - second_jack_on
28 - sec_hyd_on
29 - fail_gen_left
30 - APD_light
31 - fail_akb
33 - fail_gen_right
35 - heat_pitot_check_light
36 - differential_fire_light
37 - heater_fire_light
39 - left_fire_light
41 - right_fire_light
43 - right_valve_light
45 - manual_fire_light
47 - auto_fire_light
49 - valve_open_light
51 - left_valve_light

Board #3 - Arduino (USB) ------------------------------------------------------

---- Digital pins: ----

22 - left_eng_on_light
23 - pzu_right_light
24 - right_eng_on_light
25 - switch_aice_on_command_light
26 - left_tank_light
27 - pzu_left_light
28 - fuel_main_light
30 - left_inlet_on_light
31 - heat_pitot_check_light
33 - ri_65_light
29 - rio_3_check_light
36 - right_inlet_on_light
38 - aice_on_light
40 - white_lamp_left
41 - fuel_heat
42 - right_tank_light
43 - heating_ok
44 - red_lamp_left
45 - lights_check_light
47 - ignition_ko50
53 - red_lamp_right
```

Рисунок 5 − Структура файла data.cfg

С помощью плагина Data RefEditor можно считать, так называемые, dataref строчки. Если переключатель или индикатор в вертолете задействованы, то у них прописаны: их имя в симуляторе и принимаемые значения. Вся эта информация содержится в dataref строке. Она индивидуальна для каждого

параметра. Это лишь вспомогательный плагин, и он используется только для нахождения имени dataref строчек и их значений, а общение осуществляется через плагин проекта (ArdSimX).

Так как дополнительно программировать ничего не требуется, плата сама реализует логику работы системы. Т.е., если переключатель, соединенный между контактом и землей, не замыкает цепь, это означает уменьшение текущего параметра. Если же замыкает, это означает увеличение.

Для галетного переключателя используется подобная логика. Здесь либо переключатель представляется как набор из двухпозиционных переключателей, либо между ножками впаиваются резисторы, и подключается переключатель к одному из аналоговых входов, которые представляют из себя каналы АЦП. В таком случае, переключение будет эквивалентно первому методу, однако происходит существенная экономия доступных контактов. Вся информация по подключению переключателей и индикаторов указана на сайте разработчика [ArdSimX, ArdSimXConfigurator]. На рисунке 6 показаны оба метода.

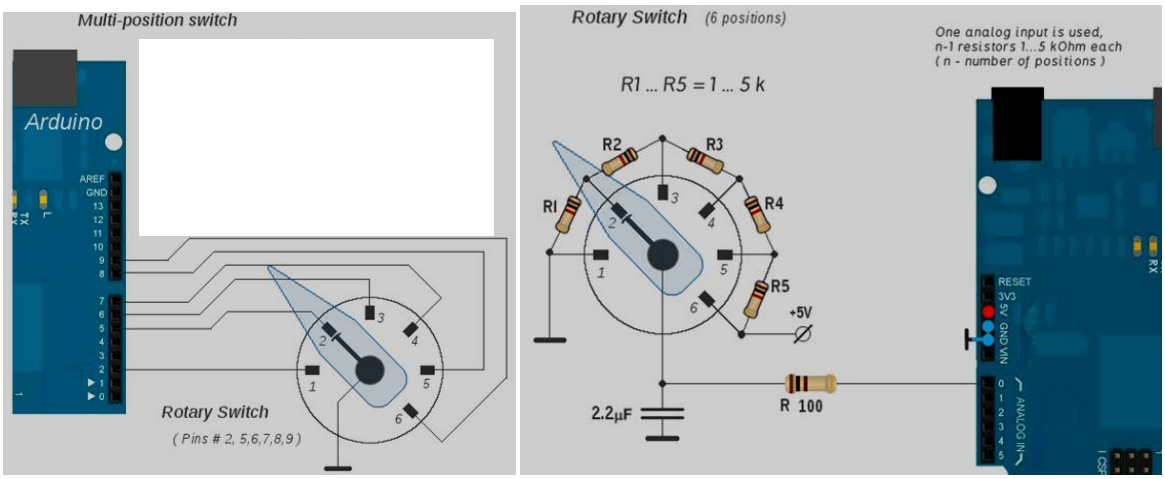

Рисунок 6 – Способы подключения галетного переключателя

После внесения данных в конфигурационный файл при каждом запуске плагин читает его, устанавливает связь между платами и понимает, какая плата за какой функционал отвечает, а плата, в свою очередь, управляет периферией.

Теперь же расшифруем этот файл. В данном примере используется центральный пульт с сигнализацией работы гидросистемы.

Board #2 – Arduino (USB). Как уже было сказано, комплекс поддерживает 9 плат. Данный пункт показывает, что данные параметры используются для 2 платы, и она подключена по USB интерфейсу.

Board #3 – все аналогично.

26 – main_hyd_onозначает, что 26 контакт разъема платы подключается к схеме коммутации сигнальной лампы, отвечающей за индикацию работы основной гидросистемы. Это лишь текстовые пояснения, они не несут функциональной нагрузки.

@====CONFIG===== − собственно сама настройка платы. В данном разделе идет привязка параметров к портам на плате.

26L 0 sim/custom/xap/Mi8/hydr/main_hyd_on 1

– L – Light – указатель настройки порта на вывод сигнальной информации;

– 26 – номер этого контакта;

– 0 – выключено;

– 1 – включено;

– sim/custom/xap/Mi8/hydr/sec_hyd_on – сама переменная. Имя для нее вы можете выбрать сами, однако для меньшей путаницы можно использовать принцип операционной системы, где каждый последующий раздел является подразделом родительского каталога. Можно было указать лишь sim/ main_hyd_on и симулятор бы понял это. Надпись custom означает, что данный параметр является созданным самостоятельно, а не выбранным из перечня заранее созданных параметров симулятора. В данном случае, переменная отвечает за сигнализацию включения гидросистемы, когда она включена и работает, лампа горит.

После того как плагин прочитал данную строку, он знает, какой контакт на плате связан с параметром, и его значение. Создание данного файла

возможно и вручную, однако конфигуратор максимально упрощает и автоматизирует процесс настройки. На рисунке 7 представлена работа сигнальных ламп на центральном электропульте.

Рисунок 7 – Сигнальные лампы на центральном электропульте

В онлайн-конфигураторе доступны следующие электротехнические элементы. На рисунке 8 показан набор поддерживаемых изделий.

Рисунок 8 – Список поддерживаемых изделий

Слева направо:

– T-SWITCH: переключатель;
– BUTTON: кнопка;

– ENCODER: это так называемый датчик угла поворота, то есть устройство, которое предназначено, чтобы преобразовать угол поворота вала (измеряемого объекта) в электрические импульсы, по которым можно определить: угол поворота, скорость вращения, направление вращения и текущее положение относительно начальной точки [Здрачук, 2017]. Можно использовать как кремальеру:

– AXIS: потенциометр. Можно использовать в качестве органа управления (штурвал, РУД, педали) или другое подобное применение;

– RT-SWITCH: галетный переключатель;

– ModeSwitch: согласно документации, специальный тип кнопки для переключения типа одного или нескольких энкодеров.

Это были элементы, с помощью которых можно было управлять параметрами симулятора. Теперь рассмотрим элементы, с помощью которых можно отображать информацию из симулятора.

Слева направо:

– DIGITALOUTPUT: лампочка, которая при поступлении информации из симулятора загорается – простой сигнализатор;

– SHIFT OUTPUT: индикационное табло;

– 7-Segment: семисегментный индикатор;

– STEPPER: шаговый двигатель;

– AMMETER: миллиамперметр. Можно использовать для различных указателей, например, топлива;

– Servo: серводвигатель.

Для каждого из вышеописанных элементов есть подробная информация по настройке [ArdSimX, ArdSimXConfigurator].

2. ПРОШИВКА ПЛАТЫ ARDUINO

Теперь, когда все элементы разобраны, строчки к контактам привязаны –

перейдем к прошивке платы. Ниже представлен листинг программы:

```
#include<ArdSimX.h>
voidsetup()
{
}
voidloop()
{
  ArdSimX (1);
}
void ProgOut(byte id, float val)
{
}
```

Функция ProgOut – функция программируемого вывода информации из плагина в Arduino. Если есть необходимость использования какого-либо параметра из симулятора в своем коде (например – вычисления разного рода), то понадобится эта функция. Она будет обращаться к плагину за значением конкретной dataref строки. Подробная информация есть на сайте разработчика и в базовом примере [ArdSimX, ArdSimXConfigurator].

Помимо этой функции есть и другая, выполняющая обратную задачу SimInput. Она выполняет передачу информации из Arduino в плагин (как пример − значение переменной на основе расчетов) [ArdSimX, ArdSimXConfigurator].

Вся работа завязана на функции ArdSimScan. Она общается с плагином, считывает все вводы/выводы и по алгоритму выполняет действия. В круглых скобках указывается номер платы, который используется плагином для передачи управления именно той плате, которая на это настроена. Максимум плат 9.

Так как на панелях управления вертолета расположено более 160 тумблеров и кнопок, из которых задействовано для использования 158, то все

тумблеры и кнопки панелей управления объединены в 14 групп по 12 штук в каждой, которые имеют свою общую «землю». Это позволяет создать матрицу переключателей из 168 элементов, которая создается в конфигурационном файле плагина.

На рисунке 9 для примера приведена электрическая схема подключения панели левого электрощитка вертолета Ми-8Т. Каждый элемент подключается к конкретному разъему на плате.

Вывод 7-2 означает: 7 – номер контакта на разъеме платы Arduino, 2 – «земля» группы переключателей.

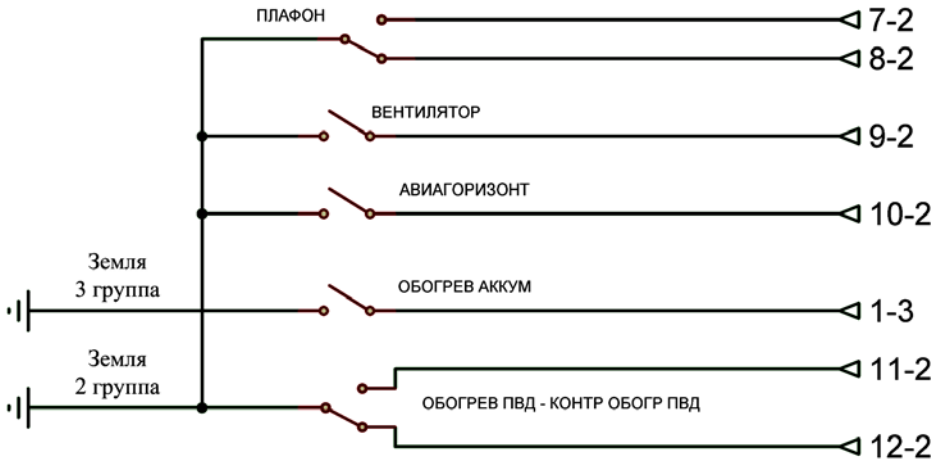

Рисунок 9 − Схема подключения тумблеров левого электрощитка

3. ОРГАНЫ УПРАВЛЕНИЯ

Помимо переключателей так же можно реализовать органы управления, у которых множество промежуточных положений. Такие органы – ручка управления, ручка «Шаг-Газ» и педали. Рассмотрим пример только для ручки «Шаг-Газ», т. к. ручка управления и педали настраиваются аналогичным способом.

Шаг-Газ — это орган управления тягой двигателя и шагом винта. Лётчик управляет режимом работы двигателя с помощью вращательного движения

ручки, а перемещением вверх-вниз — увеличивает или уменьшает шаг винта. Внешний вид ручки Шаг-Газ показан на рисунке 10.

Рисунок 10 – Внешний вид ручки Шаг-Газ

Управление двигателем представляет собой процесс, при котором каждому изменению положения газа соответствует вполне определённое изменение положения дроссельного крана, а на большинстве турбореактивных двигателей – определённые заданные обороты ротора высокого давления или вентилятора, которые поддерживаются автоматически за счёт изменения подачи топлива. Поворот ручки изменяет количество топлива, поступающего в камеру сгорания, и режим работы двигателя. Рычаги управления двигателями соединены с рычагами топливных насосов-регуляторов двигателей системой тросов, тяг и поводков.

Для работы ручки, по аналогии с панелью запуска двигателей, составим конфигурационный файл (data.cfg). Он представлен на рисунке 11.

```
[A]
[I]
A0 sim/cockpit2/controls/yoke_heading_ratio -6.5,6.5 1 50 20 80
A1 sim/cockpit2/controls/yoke_pitch_ratio 6.5,-7 1 50 20 80
A2 sim/cockpit2/controls/yoke_roll_ratio -6.5,6.5 1 50 20 80
A3 sim/cockpit2/engine/actuators/throttle_ratio 1,0 1 200 0 70
A4 sim/cockpit2/engine/actuators/throttle_ratio 0,1 2 200 50 80
A5 sim/cockpit2/engine/actuators/prop_angle_degrees 17,-3 1 1023 78 100
D53- sim/cockpit2/switches/rotor_brake 1 0
D53+ sim/cockpit2/switches/rotor_brake 0 1
D36- sim/custom/xap/Mi8/engine/right_sv 1 0
D36+ sim/custom/xap/Mi8/engine/right_sv 0 1
D38- sim/custom/xap/Mi8/engine/left_sv 1 0
D38+ sim/custom/xap/Mi8/engine/left_sv 0 1
```

Рисунок 11– Структура файла data.cfg

На рисунке 11символами A3, A4 обозначены аналоговые входы на плате:

– 0,1 – диапазон значений переменной от нуля до одного включительно;

– 1, 2 соответственно, управление левого и правого двигателей;

– 200 – количество промежуточных положений потенциометра на полный оборот (чувствительность);

– 0 70 – процентное соотношение использования потенциометра (если значение, например, 50 80, то Arduino будет считывать изменения значений напряжения при нахождении ротора потенциометра именно в диапазоне от 50% до 80%).

Электрическая схема выходных сигналов для каждого рычага представлена на рисунке 12. Проблемой было преобразование поступательного движения тяги во вращательное движение потенциометра. Проблема была решена путем сборки тяги типа «колено», которое с одной стороны крепилось к тяге, с другой стороны к потенциометру. Вал потенциометра прочно крепился к корпусу кабины, а корпус потенциометра свободно вращался тягой. Достоинством такой конструкции является возможность быстрого и точного регулирования положения потенциометра при замене или настройке.

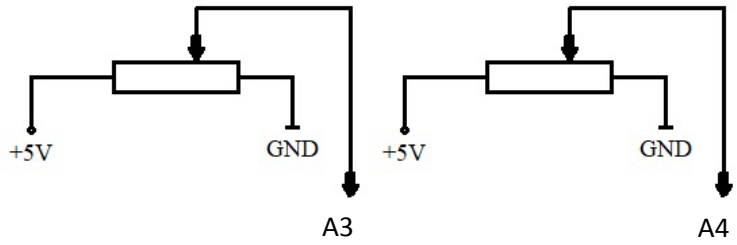

Рисунок 12 – Электрическая схема выходных сигналов

Достаточно ослабить фиксирующую гайку потенциометра, провернуть в нужную сторону и закрутить обратно. Техническая реализация представлена на рисунке 13.

Рисунок 13 − Техническая реализация подключения потенциометров к тягам

Работа основана на измерении сопротивления потенциометра платой Arduino. Каждый потенциометр присоединен к своей ручке. Каждая из ручек подсоединена к своему входу.

Перемещение ручки ведет к эквивалентному вращению вала потенциометра. Далее прошивка платы обрабатывает эти сигналы и отправляет

их значения в плагин, после чего эти значения присваиваются своим переменным. В виртуальной кабине происходит эквивалентное перемещение.

4. ПРИБОРНЫЕ СИМУЛЯТОРЫ

Большинство приборов, расположенных на приборных досках кабины вертолета – аналоговые, которые имеют подвижные стрелочные указатели (рис. 14).

Рисунок 14 – Внешний вид левой приборной доски

Типичным представителем данного класса приборов является двухстрелочный магнитоиндукционный тахометр ИТЭ-2 (рис.15), на примере которого ниже будет раскрыт подход в разработке имитаторов пилотажных приборов кабины вертолета.

Основным элементом в приборном имитаторе является сервопривод, который преобразует сигнал управления в угловое перемещение исполнительного механизма, согласно требуемого закона управления.

Рисунок 15 – Внешний вид указателя двухстрелочного тахометра ИТЭ-2

При разработке имитаторов приборов были использованы серийные малогабаритные электроприводы постоянного тока (рис.16) [Здрачук, 2017].

Рисунок 16 – Внешний вид типового серийного электропривода

В качестве сигналов управления исполнительными сервоприводами приборов используются соответствующие сигналы авиационного симулятора.

Формирование закона управления было возложено на серийный программно-аппаратный комплекс Arduino, представляющий собой средство для построения простых систем автоматики и робототехники, ориентированное на непрофессиональных пользователей. Его программная часть состоит из бесплатной программной оболочки (IDE) для написания программ, их компиляции и программирования аппаратуры. Аппаратная часть представляет собой набор смонтированных печатных плат, продающихся как официальным производителем, так и сторонними производителями (рис.17). Полностью

открытая архитектура системы позволяет свободно копировать или дополнять линейку продукции.

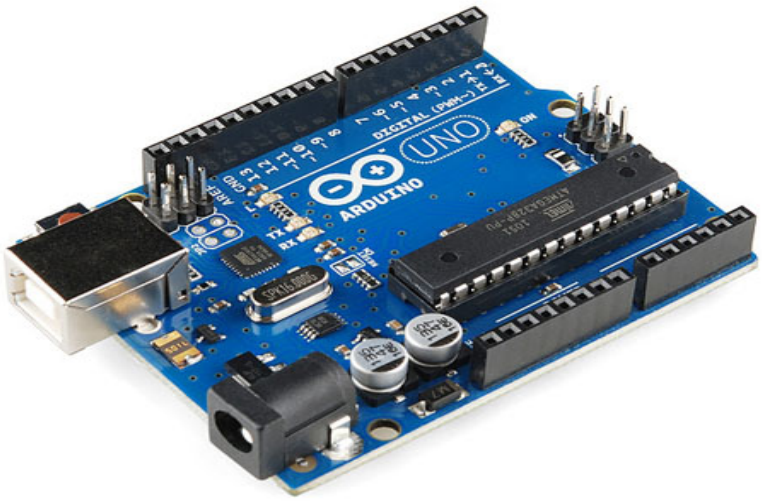

Рисунок 17 – Внешний вид серийной платы Arduino

Реализация проекта по разработке имитаторов пилотажных приборов учебного тренажера кабины вертолета Ми-8 включает в себя два основных этапа:

− изготовление имитаторов (в штатный прибор с учетом его кинематических особенностей интегрируется исполнительный электропривод);

− согласование пилотажных и других текущих параметров, получаемых от авиационного симулятора с параметрами конкретных приборов и их симуляторов (в том числе разработка соответствующих «скетчей»).

Ниже представлены основные этапы разработки имитатора на примере двухстрелочного магнитоиндукционного тахометра ИТЭ-2.

Кинематика подключения выходных валов сервоприводов к осям стрелок указателя тахометра ИТЭ-2 представлена на рисунке 18. Здесь хорошо видно, что при поступлении на сервоприводы 5 сигналов управления, они начинают вращать оси стрелок 2 через зубчатые передачи 4, тем самым задавая стрелкам 3 угол, соответствующий определенному значению числа оборотов вала двигателя в процентах на циферблате 31.

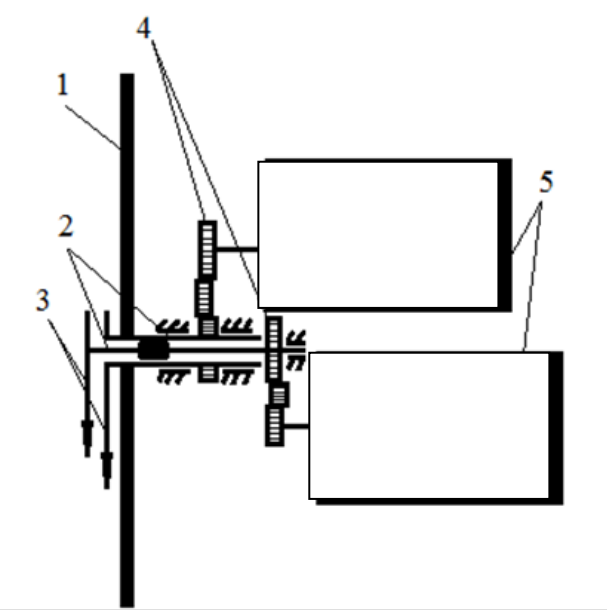

Рисунок 18 − Кинематическая схема подключения сервоприводов к осям стрелок указателя тахометра ИТЭ-2

При построении электропривода устройства автоматики, которое бы гарантированно обеспечивало поворот исполнительного механизма на требуемый законом управления угол, были использованы следующие элементы:

− плата Arduino(в работе использована MEGA 2560);

− USB драйвер CH340G для операционной системы;

− комплект соединительных проводов;

− провод USB 2.0 (A) – USB 2.0 (B);

− серводвигатель (в работе использован TowerProSG90).

Ниже представлен листинг программы для решения задачи установки ротора сервопривода постоянного тока на угол, эквивалентный значению сигнала управления, а на рисунке 19 – функциональная схема подключения сервопривода к плате.

```
#include<Servo.h>
Servoservo1;
voidsetup()
```

```
{
  servo1.attach(5);
}
void loop()
{
  int pot = analogRead(5);
  pot = map(pot, 0, 1000, 0, 180);
  servo1.write(pot);
}
```

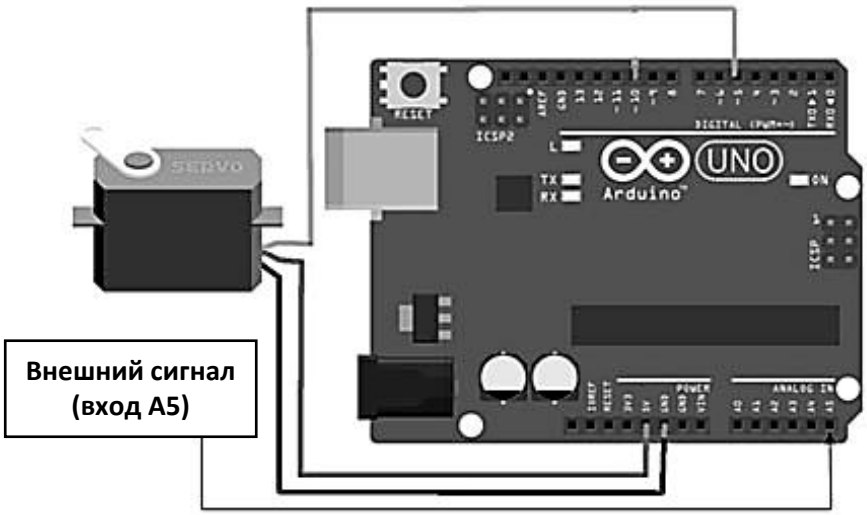

Рисунок 19 – Схема подключения серводвигателя

По USB-интерфейсу программа с компьютера записывается на микроконтроллер, который расположен на плате Arduino. Через этот же интерфейс и происходит питание всего комплекса напряжением 5В постоянного тока. К плате подключаются 2 провода: красный и черный. Это провода питания двигателя. Однако двигатель не совершит вращения без сигнала, который поступает на вход A5 площадки AnalogIN. Как только сигнал будет подан, программа обработает его, выдаст соответствующий сигнал для управления двигателем на 5 цифровой выход, к которому подключен провод управления серводвигателя, и только тогда он совершит эквивалентное вращение.

Сигналы управления сервоприводами поступают с выходов платы Arduino, соответствующие параметрам, полученным из авиасимулятора.

Параметры получаются с помощью плагина DataRefEditor. С помощью него из симулятора считываются dataref строчки, в которых прописаны: нахождение в симуляторе, параметры и принимаемые значения. Вся эта информация содержится в dataref строке, индивидуальной для каждого параметра, в рассматриваемом случае – число оборотов вала двигателя.

DataRef строки описываются в конфигурационном файле (data.cfg), представленном на рисунке 20.

```
38S 0 sim/cockpit2/engine/indicators/N1_percent 1 0,100 550 2350
39S 0 sim/cockpit2/engine/indicators/N2_percent 2 0,100 550 2350
```

Рисунок 20 – Содержимое конфигурационного файла (data.cfg)

На рисунке 20 представлена следующая информация:

− 38S, 39S – цифровые порты платы;

− 1,2 – 1й и 2й двигатели соответственно;

− 0,100 – число оборотов вала двигателя в процентах;

− 550,2350 – миллисекунды для сервопривода, соответствующие 0^{o} и 180^{o} градусам соответственно.

Принципиальная электрическая схема подключения сервоприводов к плате Arduino показана на рисунке 21.

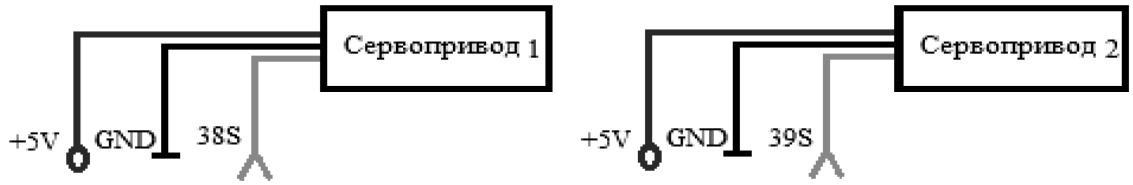

Рисунок 21 – Электрическая схема подключения сервоприводов

По аналогии с имитатором тахометра ИТЭ-2 можно реализовать любой прибор, отличие будет составлять лишь файл конфигурации (data.cfg).

Библиографический список

1. *Здрачук С.В.* Разработка авиационного тренажера на базе авиационного симулятора // X научно-практическая конференция студентов и аспирантов «Актуальные проблемы и перспективы развития авиационной техники и методов ее эксплуатации – 2017». Иркутск: Иркутский филиал МГТУ ГА, 2017. С. 264-274.

2. *Попов В.М.* Применение программных и аппаратных средств Labview при модернизации пилотажных тренажеров на базе аналогового приборного ипилотажно-навигационного оборудования // Сборник трудов Всероссийской научно-практической конференции,посвященной 50-летию Иркутского филиала МГТУ ГА«Актуальные проблемы и перспективы развитиягражданской авиации»17–19 мая 2017 г. Иркутск: Иркутский филиал МГТУ ГА, 2017. С.61-64.

3. ArdSimX, ArdSimXConfigurator // [Электронный ресурс]. – URL: http://www.simvim.com/config.html(дата обращения: 10.10.2018).

References

1. ArdSimX, ArdSimXConfigurator// [Electronic resource]. – URL: http://www.simvim.com/config.html (access date: 10.10.2018).

2. Popov V. M. (2017). The application of software and hardware Labview in the modernization of flight simulators based on analog instrumentation and flight and navigation equipment / / Collected works of the all-Russian scientific and practical conference dedicated to the 50th anniversary of the Irkutsk branch of MSTUCA "Actual problems and prospects of development of civil aviation", May 17-19, 2017 Irkutsk: Irkutsk branch of MSTUCA, 2017. p. 61-64. (In Russian).

3. *Zdrachuk S. V.* (2017). Development of aircraft simulator on the basis of aircraft simulator / / X scientific and practical conference of students and postgraduates "Actual problems and prospects of development of aviation equipment and methods of its operation – 2017". Irkutsk: Irkutsk branch of MSTUCA, 2017. p. 264-274. (In Russian).