УНИВЕРСИТЕТ ИТМО

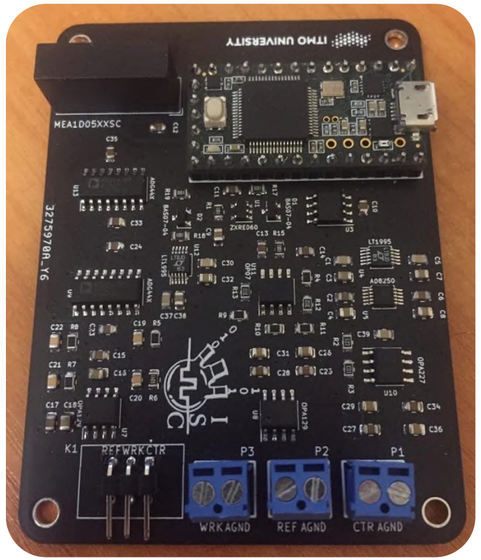
Промежуточные итоги изучения прошивки потенциостата

Ганькин Владимир
Котлярова Софья
Панаёт Виктор
Панаёт Роман

Содержание презентации

1. Что такое потенциостат?
2. Схема потенциостата;
3. Функционал прошивки;
4. Изученные модули:

Что такое потенциостат?

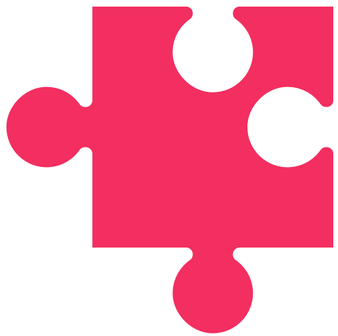


Потенциостат – электрическое устройство, служащее для поддержки потенциала или тока в отдельных точках независимо от их полярности.

Электроды потенциостата

- Токовый электрод - вывод или провод, по которому прибор передает в исследуемый объект рабочий ток (в любом режиме - потенциостатическом или гальваностатическом).
- Потенциальный электрод - вывод или провод, с помощью которого прибор измеряет напряжение. Считается, что по этому проводу течет пренебрежимо малый ток.
- Рабочий электрод - тот электрод, электрохимические процессы и явления на котором (или на границе этого электрода с электролитом) исследуются.

Электроды потенциостата



- Вспомогательный электрод - второй токовый электрод (Counter, он же поляризующий) в электрохимической ячейке, нужен для поляризации рабочего электрода, производство и поставка приборов для электрохимических исследований проще говоря, нужно как минимум два провода, чтобы пропустить через исследуемый объект электрический ток.
- Электрод сравнения, он же референсный электрод - нужен как точка отсчета абсолютного значения потенциала в трехэлектродной схеме, и как точка ввода в усилитель потенциостата сигнала обратной связи от электрохимической ячейки.

Краткая характеристика

1. teensy 3.2 - компактная платформа для разработки на базе микроконтроллера NXP MK20DX256VLH7 с вычислительным ядром ARM Cortex® M4.
2. 12-битный выход напряжения с четырьмя доступными диапазонами: ±1, 2, 5, 10В
3. 16-битное измерение тока с четырьмя доступными диапазонами: ±1, 10, 100, 1000uA
4. Управление осуществляется через USB с помощью простых сообщений JSON
5. Может быть запрограммирован через USB с помощью Arduino IDE
6. Прошивка поддерживает многие стандартные вольтамперометрические методы, включая:
   - постоянное напряжение;
   - циклическая вольтамперометрия;
   - синусоидальная вольтамперометрия;
   - вольтамперометрия линейной развертки;
   - хроноамперометрия;
   - многоступенчатый.

Изученные модули

- **ps_base_test**
- **ps_constant_test**
- **ps_multistep_test**
- **ps_periodic_test**
- **ps_cyclic_test**

Представленные здесь модули не единственные, что мы сделали, но чтобы не увеличивать объём и так большой презентации было решено вставить только эти модули. В будущих презентациях будет рассказано об остальных модулях.

ps_base_test

ps_base_test – это модуль для создания основы теста. Имеет четыре важных параметра - время молчания, значение вольт на время молчания, редактируемость и время между тестами.

Методы

Для пользователя:

- virtual bool isDone(uint64_t *t*) const - узнать исполнен ли тест
- virtual uint64_t getDoneTime() const - узнать время исполнения теста
- virtual void reset() - сброс теста

Получение значений:

- virtual float getValue(uint64_t *t*) const;
- virtual float getMaxValue() const;
- virtual float getMinValue() const;

Методы

Установление и получение времени молчания

- virtual void setQuietTime(uint64_t *quietTime*);
- virtual uint64_t getQuietTime() const;

Установление и получение значения на время молчания

- virtual void setQuietValue(float *value*);
- virtual void setQuietValueToStart() - в начале следующего теста
- virtual float getQuietValue() const;

Установление и получение периода времени между тестами

- virtual void setSamplePeriod(uint64_t *samplePeriod*);
- virtual uint64_t getSamplePeriod() const;

Методы

Установление и получение имени теста

- virtual void setName(String *name*);
- virtual String getName();

Установление и получение редактируемости теста

- virtual void setSampleMethod(SampleMethod *sampleMethod*);
- virtual SampleMethod getSampleMethod() const;

Работа с JSON файлами

- virtual void getParam(JsonObject &*jsonDat*) - получить текущие параметры () в JSON файле
- virtual ReturnStatus setParam(JsonObject &*jsonMsg*, JsonObject &*jsonDat*) - установление параметров из JSON файла

Методы

Совместим ли тест с мультиплексором?

- virtual void setName(String *name*);
- virtual String getName();

Внутренние:

- JsonObject &getParamJsonObject(JsonObject &json, ReturnStatus &status) - получить параметр с JSON файла. Обработаны исключения когда подан не JSON файл и когда нет параметров в файле.
- void setQuietValueFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status) - установить значение для тест из JSON файла. Обработаны исключения, когда тип не float.
- void setQuietTimeFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status) - установить время на тест из JSON файла. Обработаны исключения, когда тип не unsigned long (время не может быть отрицательным).

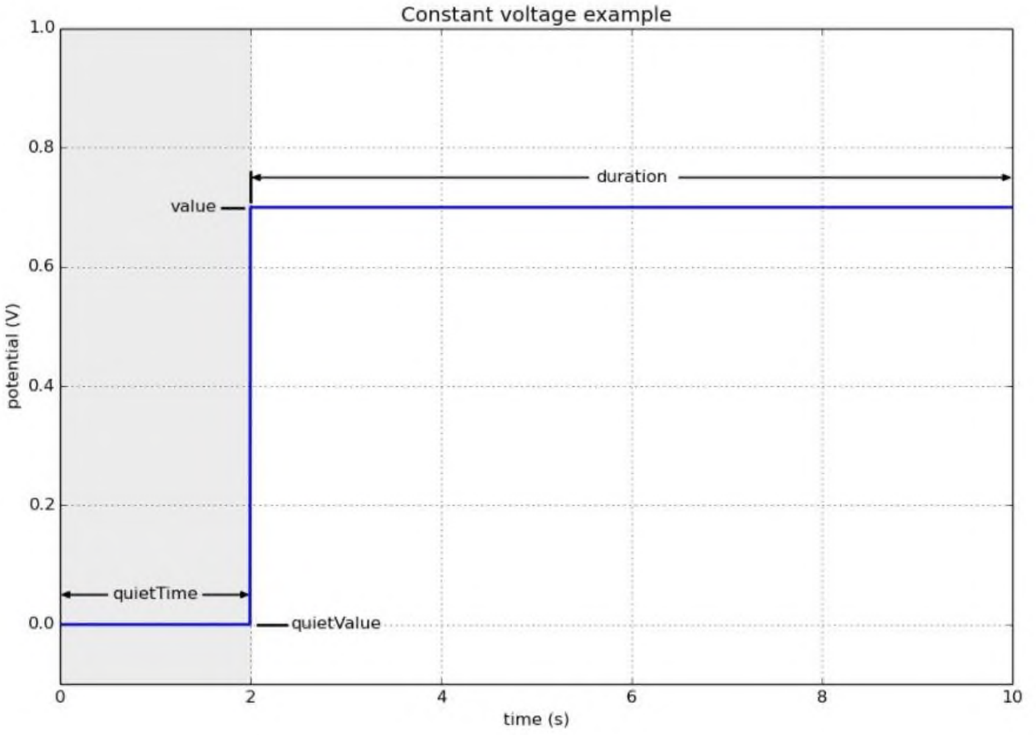
ps_constant_test

ps_constant_test – это тест, где потенциал между рабочим и эталонным электродами остается постоянным. Как и другие испытания, испытание постоянным напряжением включает в себя период молчания, в течение которого выходное напряжение удерживается, и постоянное значение (напряжение молчания) в течение фиксированной продолжительности до начала испытания.

Параметры:

Параметр	Ед. изм	Тип данных
время молчания	мс.	uint_64
напряжение во время молчания	В.	float
напряжение	В.	float
продолжительность теста	мс.	uint_64

ps_constant_test



Методы

Для пользователя:

- ConstantTest() - конструктор, в котором имя теста ставят constant и включают совместимость с мультиплексором

Установить и получить продолжительность теста

- void setDuration(uint64_t duration);
- uint64_t getDuration() const;

Установить и получить исходящее напряжение

- void setValue(float value);
- float getValue();
- virtual float getValue(uint64_t t) const override; - получить quiet_value

Методы

Готов ли тест и за сколько

- virtual bool isDone(uint64_t t) const override; - готов ли тест
- virtual uint64_t getDoneTime() const override; - время выполнения теста

Максимальное и минимальное значение напряжения

- virtual float getMaxValue() const override;
- virtual float getMinValue() const override;

Работа с JSON файлами

- virtual void getParam(JsonObject &jsonDat) override - Получить тек. параметры в JSON файл
- virtual ReturnStatus setParam(JsonObject &jsonMsg, JsonObject &jsonDat) override; - установить параметры из JSON файла

Методы

Внутренние:

Установить продолжительность и значение из JSON файла

- void setDurationFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void setValueFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);

ps_multistep_test

УНИВЕРСИТЕТ ИТМО

ps_multistep_test – это тест, где каждое изменение напряжения задается шагом

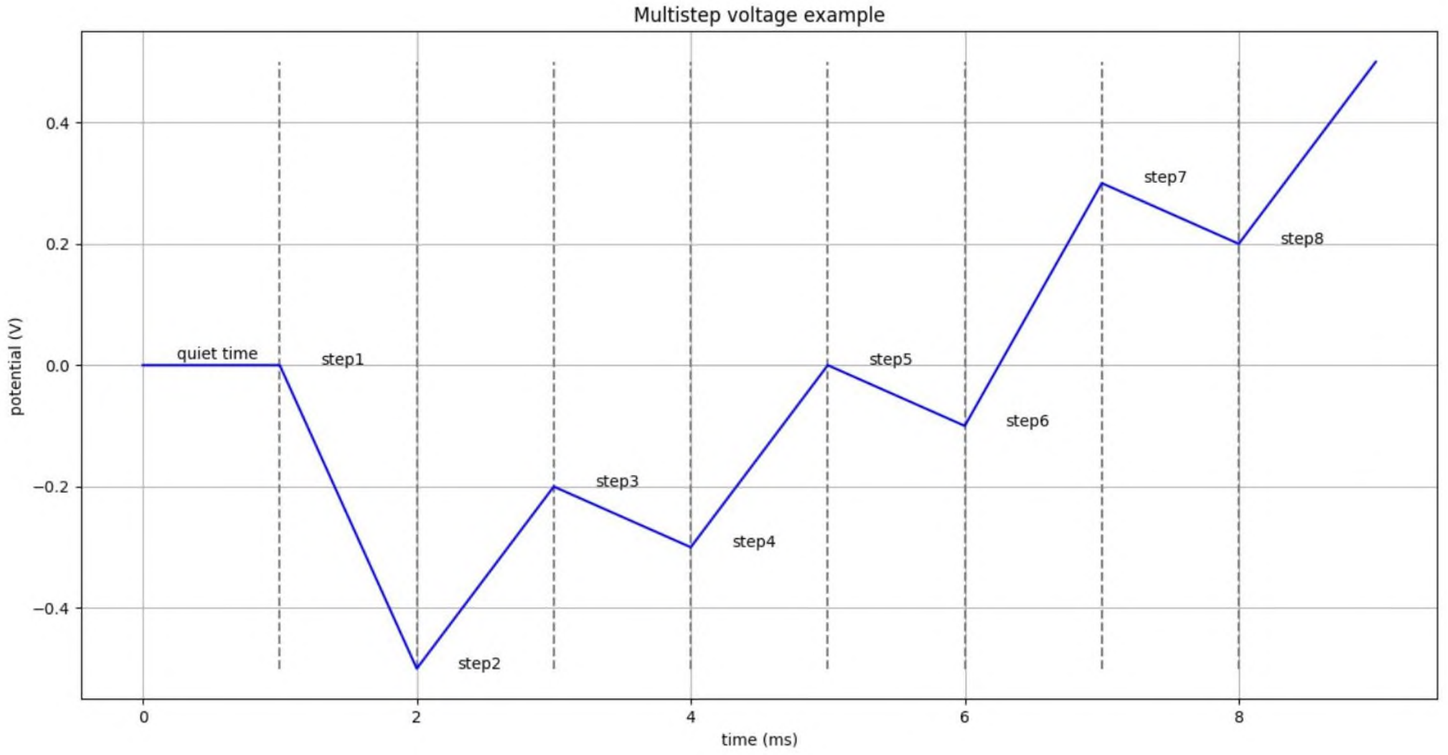
Параметры:

Параметр	Ед. изм	Тип данных
время молчания	мс	uint_64
напряжение молчания	В	float
шаг* (продолжительность, напряжение)	(мс, В)	(uint_64, float)

**макс кол-во шагов = 50*

ITMO more than a UNIVERSITY

ps_multistep_test



Методы

Для пользователя:

MultiStepTest(size_t numStep=5) - конструктор, в котором имя теста ставят multiStep, готовят массивы значений и продолжительности, устанавливают количество шагов (по умолчанию 5) и включают совместимость с мультиплексором.

Установить или получить значение в шаге

- virtual void setStepValue(size_t n, float value)
- virtual float getStepValue(size_t n) const

Установить или получить продолжительность в шаге

- virtual void setStepDuration(size_t n, uint64_t duration);
- virtual uint64_t getStepDuration(size_t n) const;

Методы

Получить продолжительность теста

- virtual uint64_t getDuration() const;

Установить или получить кол-во шагов

- virtual void setNumStep(size_t numStep);
- virtual size_t getNumStep() const;

Получить максимально возможное кол-во шагов

- virtual size_t getMaxNumStep() const;

Готов ли тест и за сколько

- virtual bool isDone(uint64_t t) const override; - готов ли тест
- virtual uint64_t getDoneTime() const override; - время исполнения теста

Методы

Сброс

- virtual void reset() override;

Получить текущее значение

- virtual float getValue(uint64_t t) const override;

Максимальное и минимальное значение

- virtual float getMaxValue() const override;
- virtual float getMinValue() const override;

Работа с JSON файлами

- virtual void getParam(JsonObject &jsonDat) override - получить параметры в JSON файл
- virtual ReturnStatus setParam(JsonObject &jsonMsg, JsonObject &jsonDat) override - установить параметры из JSON файла - value and duration

Методы

Внутренние:

Установить значение и продолжительность из JSON файла

Обработаны след. исключения:

- для кол-ва шагов:
  - не найдено кол-во шагов;
  - на вход подан не массив;
  - массив слишком большой
- **для value and duration:**
  - продолжительность не int;
  - значение не float;
  - размер массива не 2;
  - на вход подан не массив.

void setValueAndDurationFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);

ps_periodic_test

УНИВЕРСИТЕТ ИТМО

Параметры:

Параметр	Ед. изм	Тип данных
время молчания	мс	uint_64
напряжение молчания	В	float
амплитуда	В	float
смещение по y	В	float
период	мс	uint_64
кол-во циклов	шт.	uint_32
фазовый сдвиг (смещение по x)		float

ITMO more than a UNIVERSITY

Методы

Для пользователя:

PeriodicTest::PeriodicTest() - конструктор, в котором имя теста = periodic и рассчитывается перемещение в секундах.

Установить или получить амплитуду

- virtual void setAmplitude(float amplitude);
- virtual float getAmplitude() const;

Установить или получить смещение (относительно оси ординат)

- virtual void setOffset(float offset);
- virtual float getOffset() const;

Методы

Установить или получить период

- virtual void setPeriod(uint64_t period);
- virtual uint64_t getPeriod() const;

Установить или получить кол-во итераций

- virtual void setNumCycles(uint32_t numCycles);
- virtual uint32_t getNumCycles() const;

Установить или получить смещение относительно оси абсцисс

- virtual void setShift(float lag);
- virtual float getShift() const;

Методы

Текущая итерация

- virtual uint32_t getCycleCount(uint64_t t) const;

Готов ли тест и за сколько

- virtual bool isDone(uint64_t t) const override;
- virtual uint64_t getDoneTime() const override;
- virtual float getValue(uint64_t t) const override; - получение текущего значения. Создан для обязательной инициализации в классах-наследниках;

Максимальное и минимальное значение

- virtual float getMaxValue() const override;
- virtual float getMinValue() const override;

Методы

Скинуть параметры в JSON файл

- virtual void getParam(JsonObject &jsonDat) override;

Установить параметры из JSON файла

- virtual ReturnStatus setParam(JsonObject &jsonMsg, JsonObject &jsonDat) override;

Методы

Внутренние:

Установить амплитуду, смещение, период, кол-во циклов, перемещение из JSON файла. Прописаны исключения на тип и для перемещения на значение [между 0 и 1]

- void setAmplitudeFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void setOffsetFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void setPeriodFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void setNumCyclesFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void setShiftFromJson(JsonObject &jsonMsgPrm, JsonObject &jsonDatPrm, ReturnStatus &status);
- void updateShiftInUs() - смещение по x в секундах

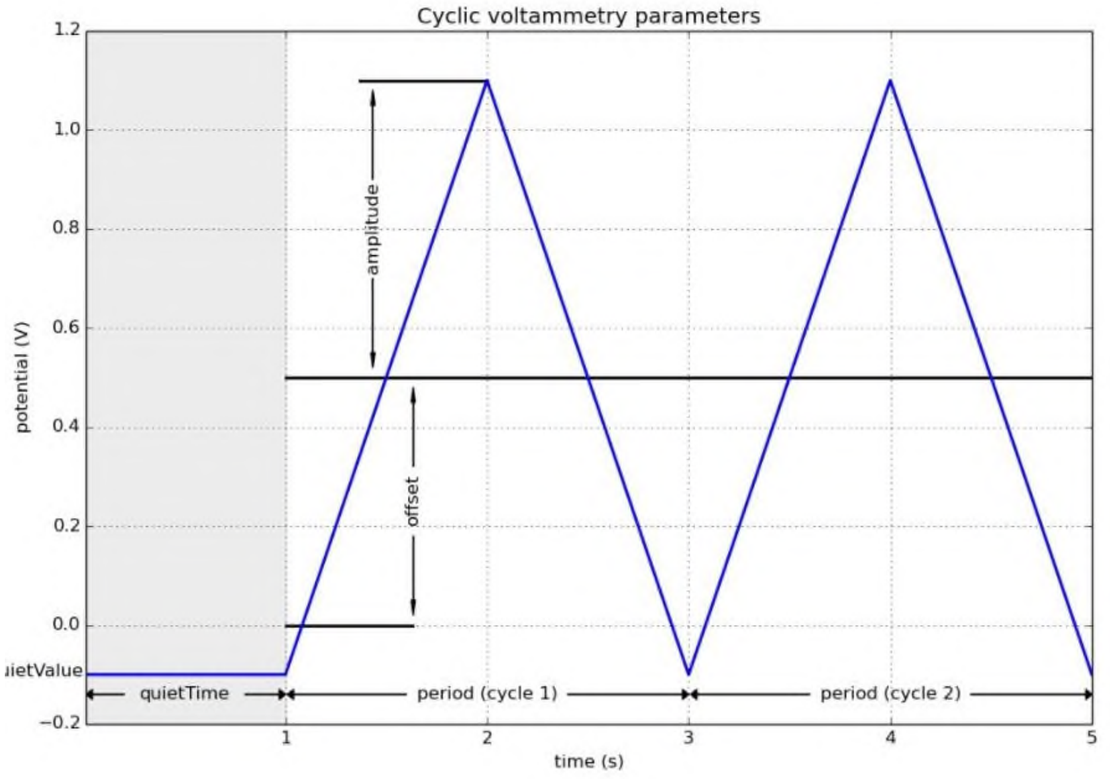
ps_cyclic_test

Наследник periodic_test. Тест, где потенциал между рабочим и опорным электродами циклически наращивается вверх и вниз кусочно-линейным способом - в треугольной форме волны.

Параметры:

Параметр	Ед. изм	Тип данных
время молчания	мс	uint_64
напряжение молчания	В	float
амплитуда	В	float
смещение по y	В	float
период	мс	uint_64
кол-во циклов	шт.	uint_32
фазовый сдвиг (смещение по x)		float

ps_cyclic_test



Методы

Для пользователя:

- CyclicTest() - - конструктор, в котором имя теста = cyclic и multiplexer вкл.
- virtual float getValue(uint64_t t) const override - получение текущего значения.

Методы

Для пользователя:

- CyclicTest() - - конструктор, в котором имя теста = cyclic и multiplexer вкл.
- virtual float getValue(uint64_t t) const override - получение текущего значения.


УНИВЕРСИТЕТ ИТМО
НАШИ КОНТАКТЫ
ITMO
UNIVERSITY