Оглавление

Об авторах

Сай Яманур – старший инженер по приложениям интернета вещей (IoT) в компании по производству промышленных газов в Буффало, штат Нью-Йорк. Имеет более чем 10-летний опыт работы в качестве эксперта по встраиваемым системам, работая как над разработкой, так и над внедрением аппаратного и программного обеспечения. Соавтор двух книг об использовании Raspberry Pi для выполнения проектов «сделай сам». Представил персональную панель мониторинга здоровья на общенациональной выставке Maker Fair. В настоящее время Сай работает над проектами, направленными на улучшение качества жизни (QoL) людей с хроническими заболеваниями.

> *«Я хочу поблагодарить своих родителей и моего брата и соавтора Шри за всю помощь и поддержку; наших технических рецензентов Салмана Фариса и Джонатана Виттса за их проницательные комментарии и за тщательный анализ нашей работы; Рахула Наира за предоставленную нам возможность работать с издательством Packt. Я также хотел бы поблагодарить Роми Диас и Вайдехи Савант за их терпение и поддержку нашей работы».*

Шрихари Яманур – инженер-механик с опытом работы в области проектирования медицинских устройств, CAD/CAM, механотроники и надежной аппаратуры. В сотрудничестве со своим братом разрабатывает аппаратные продукты с открытым исходным кодом, направленные на повышение популярности в любительских кругах. Имеет множество сертификатов в области обеспечения качества, САПР и ВЭД. Помимо дизайна, производства и качества электронных изделий, его текущие интересы включают изменение поведения и активное самосовершенствование в борьбе с диабетом, инновационные парадигмы и методологии в области здоровья, а также влияние искусственного интеллекта на здравоохранение. Он является соавтором двух книг по приложениям Raspberry Pi и пишет блоги на различные темы.

> *«Я хочу поблагодарить своих родителей, наставников, друзей, кошек, а также моего брата и соавтора Саи за всю помощь и поддержку. Я хотел бы выразить благодарность своим наставникам Анну Тамуру и доктору Судхи Гаутама, а также моему другу Сатьяканту Тьягарадже за то, что он поддержал меня в трудные времена, и команде издательства Packt за их поддержку этой книги и других наших усилий на протяжении многих лет».*

О рецензентах

Салман Фарис – энтузиаст разработки и быстрого создания прототипов цифровых продуктов из Индии. Имеет степень бакалавра в области компьютерных наук и диплом цифровой разработки Академии Fab. В настоящее время работает инженером технической поддержки в английской компании Nebra и является ключевым участником сообщества разработчиков MakerGram, где занимается разработкой электронных и аппаратных продуктов.

Салман также является частью экспертной группы Edge Impulse, сообществ Qubitro, RAK и Seeed Studio, а также послом и основным участником крупнейшего в Индии собрания разработчиков Maker Faire (Хайдарабад) и соорганизатором фестиваля Maker Fest в Керале.

> *«Сначала я хотел бы поблагодарить Аллаха за Его всемогущее руководство в любых решениях, которые я принимаю. Я также хотел бы поблагодарить издательство Packt Publishing за возможность ознакомиться с этой замечательной книгой, особенно Шагуна и Эшвина, которые руководили рецензированием и помогали мне советом и поддержкой на протяжении всего процесса. Спасибо моим родителям, братьям и сестрам, родственникам, друзьям, наставникам и команде Nebra».*

Джон Виттс работает в области информационных технологий в сфере образования уже более 17 лет. Имеет ученую степень по изобразительному искусству, а также дизайну и разработке электронного обучения. В своей нынешней роли директора по цифровой стратегии Джон руководит всеми технологическими решениями в своей школе, а также преподает информатику учащимся в возрасте 11–16 лет. Джон также проводит мероприятия Hull Raspberry Jam в своем родном городе: бесплатные семинары по программированию для молодых людей, использующих компьютер Raspberry Pi. Джон рецензировал ряд изданий для Packt и написал свою собственную книгу «Wearable-Tech Projects with the Raspberry Pi Zero», опубликованную Packt. В свободное время Джон с удовольствием занимается объединением компьютерных технологий с искусством (generative art), используя библиотеки JawaScript и элементы физических вычислений на Raspberry Pi.

> *«Я хотел бы поблагодарить мою жену Салли и наших трех дочерей, Мейбл, Эмбер и Аду, за всю их поддержку, позволившую мне работать над этой книгой, а также авторов и всю команду издательства Packt за то, что они позволили мне принять участие в процессе создания этой замечательной публикации».*

Предисловие

Когда в январе 2021 года компанией Raspberry Pi Foundation был сделан анонс Raspberry Pi Pico, мы были поражены новыми возможностями, которые в плате за 4 доллара США открылись для любителей, специалистов широкого профиля, гражданских[1] и профессиональных ученых, преподавателей и студентов по всему миру. Доступный в различных формах, мощный, но недорогой микроконтроллер действительно может работать сам по себе и с другими инструментами, помогая людям разрабатывать очень мощные и элегантные решения. Мы ожидаем, что, подобно предыдущим поколениям продуктов от Raspberry Pi Foundation, Raspberry Pi Pico совершит еще одну революцию в области технологий, образования, развлечений и других массовых начинаний.

Основываясь на нашем опыте создания публикаций об **одноплатных компьютерах** (Single-Board Computers, SBC) Raspberry Pi, мы написали эту книгу, чтобы познакомить читателя с новыми и старыми проектами, для удовлетворения различных потребностей целевой аудитории: студентов, преподавателей, инженеров, ученых, художников и технических энтузиастов, которые хотят разрабатывать встроенные системы, предназначенные для экономичной автоматизации, устройств IoT[2], робототехники, медицинских устройств и художественных проектов.

Мы постарались сохранить разнообразие в проектах, а также представили различные датчики, способы программирования, описания интерфейсов и другие подробности, достаточные для того, чтобы как новички, так и продвинутые читатели могли создавать и реализовывать свои задумки на основе Raspberry Pi Pico.

Для кого предназначена эта книга

Как уже говорилось, мы разработали материалы к проектам, рассчитанным на широкий круг читателей. Возможно, вы опытный любитель или профессионал, заинтересованный в понимании того, как Pico может помочь вам в ваших проектах. У вас может быть маленький или большой опыт работы с электроникой, одноплатными компьютерами, микроконтроллерами или программированием. Вы можете обладать всеми необходимыми навыками и быть в поиске новых проектов, чтобы развлечь себя или преподавать своим ученикам. Эта книга рассчитана на людей с самым разнообразным опытом и направлением работы.

Тем не менее некоторый базовый опыт в программировании, электронике и смежных областях будет очень полезен при ознакомлении с материалами и проектами в книге. Если вы хотите начать свой опыт программирования на

---

[1] Гражданская наука (citizen science) – направление исследований с привлечением добровольцев из числа непрофессионалов. Термин «гражданский ученый» (citizen scientist) распространился на Западе в последнее десятилетие. Как правило, этим термином называют волонтеров, участвующих на добровольных началах в каком-либо профессиональном проекте, а не просто ученых-любителей. – *Прим. перев.*

[2] Internet of Things, «интернет вещей». – *Прим. перев.*

Python, то можете обратиться к другой нашей публикации *Python Programming with Raspberry Pi* («Программирование на Python с помощью Raspberry Pi»), также выпущенной издательством Packt[3].

О чем рассказывает эта книга

В **главе 1** «*Начало работы с Raspberry Pi Pico*» излагаются основы Raspberry Pi Pico, его разновидности, аксессуары и способы программирования. Мы также покажем, как создать классический пример «Hello World» и заставить мигать светодиод.

В **главе 2** «*Последовательные интерфейсы и их приложения*» мы исследуем, как использовать последовательные интерфейсы Raspberry Pi Pico для взаимодействия с датчиками, дисплеями и другим оборудованием. Мы также продемонстрируем, как заставить работать модуль Wi-Fi и подключить Raspberry Pi Pico к интернету.

В **главе 3** «*Проекты домашней автоматизации*» рассматриваются простые проекты домашней автоматизации, которые можно выполнить за выходные, а также приложения с последовательными интерфейсами. Мы также представляем Arduino RP2040 Connect и то, как его можно использовать вместо Pico.

Глава 4 «*Весело проводите время в саду!*» позволяет нам глубже разобраться в реализации проектов с помощью Pico. Мы подключаем датчик почвы к живому растению, измеряем температуру и влажность почвы, загружаем данные на аналитическую платформу IoT и визуализируем собранные данные.

Глава 5 «*Строим метеостанцию*» – это особое удовольствие для любителей погоды и гражданских ученых. Мы построим метеостанцию с различными датчиками и вариантами интерфейса с Raspberry Pi Pico.

Глава 6 «*Проектируем настенный семисегментный дисплей*» посвящена созданию средств отображения. Мы обсуждаем управление дисплеем через последовательный порт или из локальной сети.

В **главе 7** «*Разрабатываем устройство слежения за качеством воздуха*», продолжая предыдущую главу, мы демонстрируем другое применение средств наглядного отображения, на этот раз используя два разных подхода: один с использованием существующих источников данных, а другой – с использованием датчика двуокиси углерода для определения качества воздуха.

В **главе 8** «*Беспроводная связь*» мы выходим за рамки Wi-Fi и исследуем иные способы сбора и передачи данных по беспроводной сети, применяя LoRa, Sigfox и Bluetooth. Это позволит вам свободно разрабатывать беспроводные приложения с помощью Pico.

---

[3] На русском языке можно рекомендовать онлайн-курс «Программирование на Python на Raspberry Pi» (**https://myraspberry.ru/programmirovanie-python-na-raspberry-pi.html**). Для более подробного ознакомления с языком Python рекомендуется книга «Основы программирования на языке Python» (М.: ДМК Пресс, 2017), для общего введения в одноплатный компьютер Raspberry Pi – книга «Raspberry Pi. Руководство по настройке и применению» (М.: ДМК Пресс, 2014). – *Прим. перев.*

Глава 9 «*Строим робота!*» предназначена для энтузиастов робототехники. Мы демонстрируем, как можно управлять роботом с помощью Pico. В этой главе мы также представим MicroPython для тех, кто планирует использовать легкий код.

Глава 10 «*Знакомство с приложениями TinyML*» – это руководство для тех из вас, кто хочет разрабатывать приложения искусственного интеллекта (ИИ) с помощью Pico. Мы познакомим вас с TinyML – фреймворком[4], специально ориентированным на облегченные приложения ИИ. Покажем вам пример распознавания ключевых слов, который поможет подготовить почву для дальнейшего изучения этого направления.

Глава 11 «*Создаем готовый продукт*» демонстрирует, как еще больше ускорить процесс и создать готовый продукт. Мы демонстрируем метод создания несущей печатной платы для Pico, а также разъясняем, как использовать сотовый модуль для подключения.

В **главе 12** «*Дополнительные возможности работы с Pico*» мы завершаем книгу советами и рекомендациями, которые помогут продвинуть ваши проекты с помощью Pico дальше. Мы обсуждаем, как можно обновить прошивку Pico, как Arduino IDE можно использовать для программирования Pico, как провести измерение потребления Pico и программирование контактов ввода/вывода (PIO).

Мы надеемся, что продемонстрированные в книге проекты подготовят вас к будущим приключениям с Raspberry Pi Pico.

Как извлечь максимум пользы из этой книги

Программное/аппаратное обеспечение, описанное в книге	Требования к операционной системе
CircuitPython	Windows, macOS или Linux
MicroPython	Windows, macOS или Linux
Arduino IDE (C/C++)	Windows, macOS или Linux

Проекты, обсуждаемые в этой книге, требуют большого количества разнообразного аппаратного обеспечения. Чтобы сохранить единство повествования, мы использовали Raspberry Pi Pico во всех главах. Вам также понадобится беспроводной модуль ESP32 для подключения к сети. В каждой главе приведен список рекомендуемого оборудования, и мы по возможности перечислили альтернативные варианты. Мы оставляем за вами право заменять компоненты по своему усмотрению.

[4] Фреймворк (framework) – программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. – *Прим. перев.*

> Если вы используете цифровую версию этой книги, мы советуем вам вводить код самостоятельно или получить доступ к коду из репозитория книги на GitHub (ссылка доступна в следующем разделе). Это поможет вам избежать любых потенциальных ошибок, связанных с копированием и вставкой кода.
>
> Если вы обнаружите какие-либо конкретные проблемы, связанные с оборудованием или с примерами кода, размещенными в репозитории, пожалуйста, не стесняйтесь поднимать проблему на GitHub.

Загрузка файлов с примерами кода

Вы можете загрузить файлы с примерами кода для этой книги с GitHub по адресу **https://github.com/PacktPublishing/Raspberry-Pi-Pico-DIY-Workshop**. Если код будет обновлен, он будет обновлен в репозитории GitHub.

У нас также есть другие пакеты кода из нашего богатого каталога книг и видео, доступного по адресу **https://github.com/PacktPublishing/**. Загляните туда!

Код в действии

Видеоролики с кодом в процессе работы для этой книги можно посмотреть по адресу **https://bit.ly/3OZJb5Z**.

Загрузите цветные изображения

Мы также предоставляем PDF-файл, содержащий цветные изображения скриншотов и рисунков, используемых в этой книге. Вы можете скачать его здесь: **https://static.packt-cdn.com/downloads/9781801814812_ColorImages.pdf**.

Текстовые соглашения

В этой книге используется ряд текстовых соглашений.

`Код в тексте`: указывает кодовые слова и имена переменных в тексте, имена таблиц базы данных, названия программных библиотек. Пример: «Модуль `board` содержит определения контактов и периферийных устройств, специфичных для платы».

Имена файлов, *имена папок, расширения файлов, пути файловой системы*. Пример: «Файл *rp2040-datasheet.pdf* содержит документацию микроконтроллера RP2040».

Доменные имена и адреса в интернете, если они не представляют собой законченного URL, представляются полужирным курсивом: ***bit.ly***.

Блок кода задается следующим образом:

```
from machine import Pin
import utime
led = Pin(25, Pin.OUT)
while True:
```

Когда мы хотим привлечь ваше внимание к определенной части блока кода, соответствующие строки или элементы выделяются жирным шрифтом:

```
while True:
led.toggle()
utime.sleep(1)
```

Любой ввод или вывод из командной строки записывается следующим образом:

```
>>> print("Hello World")
```

Жирный шрифт: обозначает новый термин или важное слово. Названия в меню или диалоговых окнах также выделены жирным шрифтом. Пример: «Импульс можно создать на вкладке **Create impulse** слева».

> **Советы или важные примечания**
>
> Обозначаются вот так.

Связаться с нами

Обратная связь от наших читателей всегда приветствуется.

Несмотря на то что мы приложили все усилия для обеспечения точности нашего контента, ошибки случаются. Если вы нашли ошибку в этой книге, мы были бы признательны, если бы вы сообщили нам об этом. Если у вас есть вопросы по какому-либо аспекту этой книги, напишите нам по адресу **dmkpress@gmail.com** и укажите название книги в теме вашего сообщения.

Поделитесь своим впечатлением

Когда вы прочитаете «*Raspberry Pi Pico в любительских проектах*», мы будем рады узнать ваше мнение о книге! Пожалуйста, разыщите книгу в каталоге сайта издательства «ДМК Пресс» (**https://dmkpress.com**) и поделитесь своими впечатлениями.

Ваш отзыв важен для нас!

Часть I

Введение в Pico

Цель этой части – представить Raspberry Pi Pico, его варианты и периферийные устройства, доступные на Raspberry Pi Pico. Часть I начинается с проекта мигающего светодиода и описания последовательных интерфейсов, доступных на микроконтроллере RP2040. Затем мы будем продвигаться вперед, работая над простыми проектами домашней автоматизации и садоводства.

Эта часть содержит следующие главы:

- глава 1 *«Начало работы с Raspberry Pi Pico»*;
- глава 2 *«Последовательные интерфейсы и их приложения»*;
- глава 3 *«Проекты домашней автоматизации»*;
- глава 4 *«Весело проводите время в саду!»*.

Глава 1

Начало работы с Raspberry Pi Pico

В этой главе мы хотели бы углубиться в краткое введение в Raspberry Pi Pico и микроконтроллер RP2040. Мы рассмотрим функции Raspberry Pi Pico, периферийные устройства RP2040, дополнительное оборудование для Pico и платы на основе RP2040, разработанные другими производителями. Мы также обсудим варианты языка программирования, доступные для Pico, и дополним главу обсуждением простых примеров «Hello World!», где мы печатаем что-то на экране и мигаем светодиодом.

К концу этой первой главы вы освоитесь с Pico, будете готовы приступить к программированию микроконтроллера RP2040 и начать планировать реализацию проектов из последующих глав этой книги, а также продумывать, как вы можете создавать свои собственные проекты с помощью Raspberry Pi Pico!

Мы собираемся осветить следующие основные темы:

- представление Raspberry Pi Pico и RP2040;
- обсуждение вариантов платы Pico;
- пайка разъемов на плате Pico;
- реализация примера «Hello World!»;
- реализация примера мигания светодиода;
- определение полезного дополнительного оборудования для Pico.

Технические требования

Аппаратное и программное обеспечение, необходимое для этой вводной главы, будет использоваться на протяжении всей книги. В следующих главах мы также представим дополнительные или специфические для главы требования к оборудованию.

Требования к оборудованию для этой главы:

- ноутбук или персональный компьютер с портом **универсальной последовательной шины (USB)**;
- *дополнительно:* паяльное оборудование, включая электропаяльник, припой, защитные очки и разное оборудование;

- *дополнительно*: беспаечная макетная плата и комплект соединительных проводов.

Видеоролики с кодом в действии для этой главы можно посмотреть по адресу **https://bit.ly/3MRdYjx**.

Представление Raspberry Pi Pico и RP2040

Raspberry Pi Pico – это новый одноплатный микроконтроллер, представленный компанией Raspberry Pi Foundation. Плата микроконтроллера Pico, несмотря на невысокую цену, обладает неплохими характеристиками. Pico построен на RP2040, двухъядерном микроконтроллере с ядром Cortex-M0+. Плата поставляется в общей сложности с 40 контактами, по 20 контактов с каждой стороны, как показано на рис. 1.1. Pico поставляется с 2 МБ встроенной флеш-памяти и светодиодом на выводе GP25. GP (или полностью **GPIO**) означает **General Purpose Input/Output** – контакт ввода/вывода общего назначения.

Рис. 1.1. Raspberry Pi Pico

Спецификация для Raspberry Pi Pico доступна по адресу: **https://bit.ly/3cwv1lc**. В проектах, изложенных в этой книге, мы будем использовать различные периферийные устройства, доступные на Pico. Удобно распечатать разводку выводов, предоставленную Raspberry Pi Foundation (источник: **https://bit.ly/3wa0nwq**). Схема разводки может помочь с выбором выводов при планировании проекта. На рис. 1.2 показана схема разводки, представленная Adafruit Industries.

Плата Pico может использоваться в различных приложениях, связанных с роботами, дистанционным мониторингом, гражданской наукой и т. д. В этой книге мы познакомим вас с различными примерами применения, включающими в том числе и периферийные устройства микроконтроллера RP2040.

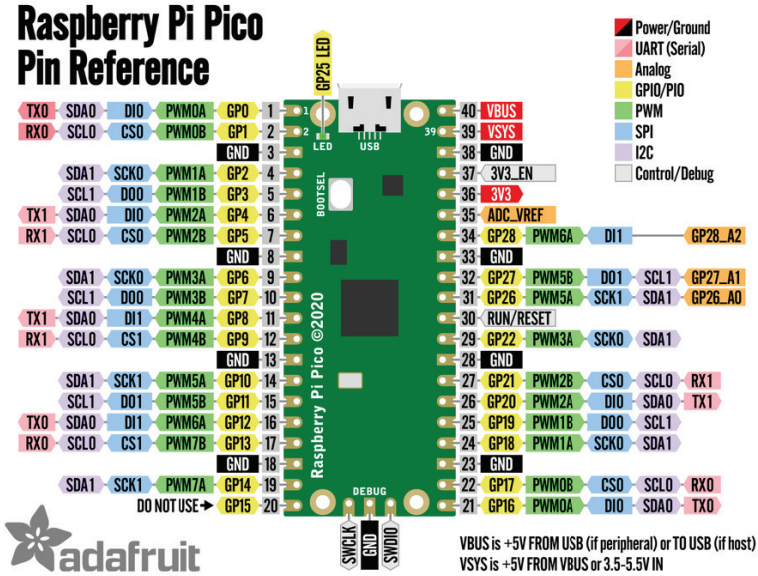

Рисунок 1.2. Разводка выводов Raspberry Pi Pico (источник: Adafruit Industries)

Микроконтроллер RP2040

RP2040 представляет собой двухъядерный микроконтроллер ARM Cortex-M0+ с 264 килобайтами (КБ) статической оперативной памяти (SRAM), но не имеет встроенной флеш-памяти. RP2040 включает в себя целый ряд периферийных устройств: двухпроводной последовательный интерфейс (I2C), последовательный периферийный интерфейс (SPI), универсальный асинхронный интерфейс приемника-передатчика (UART) и контакты с программируемым вводом/выводом (PIO). Выводы PIO микроконтроллера RP2040 позволяют вам создавать свой собственный интерфейс, такой как дополнительный UART или видеоинтерфейс. В главе 12 «*Дополнительные возможности работы с Pico*» мы обсудим использование PIO для подключения периферийных устройств.

Вот список ресурсов для более подробного изучения RP2040:

- технические характеристики RP2040 доступны по следующей ссылке: **https://bit.ly/3rw41x5**;
- техническое описание Raspberry Pi Pico доступно по следующей ссылке: **https://bit.ly/3cwv1lc**;
- видео от Raspberry Pi foundation о PIO в микроконтроллере RP2040 можно найти по следующей ссылке: **https://bit.ly/39ni6Xg**;
- документацию по RP2040 и Raspberry Pi Pico от Raspberry Pi Foundation можно найти по следующей ссылке: **https://bit.ly/3flFLv9**.

Мы рекомендуем вам загрузить технические описания Pico и RP2040. Они пригодятся в качестве справочного материала при разработке, и в определенных местах этой книги мы будем отсылать вас к этим документам для получения дополнительной информации.

> **О приобретении компонентов в России**
>
> Приобрести Raspberry Pi Pico и его варианты, а также комплектующие, необходимые для выполнения проектов из книги, в России можно в магазинах сети «Чип и Дип», в российских интернет-магазинах, торгующих электронными компонентами (например, **iarduino.ru**; **electronshik.ru**; **amperka.ru** и др.), а также на AliExpress. В этой главе далее и некоторых других разделах книги встречаются ссылки на производителей и интернет-магазины, доступные в США и Европе, но не работающие с Россией (например, **adafruit.com** или **digikey.com**). В большинстве случаев подобная ссылка сопровождается сноской с указанием российского источника напрямую. Если она отсутствует, то указанные там товары или их аналоги следует самостоятельно поискать в отечественных интернет-магазинах или через специальных посредников по приобретению.

Обзор вариантов платы Pico

С момента запуска Raspberry Pi Pico было выпущено несколько вариантов платы на основе RP2040 от различных компаний, выпускающих свободное аппаратное обеспечение. Опишем более подробно некоторые из них.

- **SparkFun Thing Plus – RP2040**: это плата с открытым исходным кодом от SparkFun (**https://bit.ly/2NS5vUn**). Thing Plus выпускается в форм-факторе **Feather** от Adafruit[5]. Уникальность этой платы заключается в том, что она поставляется с держателем карты microSD и индивидуально адресуемым RGB-светодиодом. На рис. 1.3 представлена плата RP2040 Thing Plus с нижней стороны.

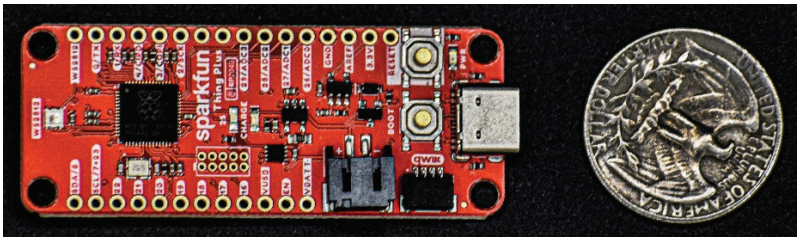

Рисунок 1.3. SparkFun Thing Plus (RP2040)

[5] Форм-фактор Feather – популярная типовая конфигурация платы размерами 60×23 мм (2,38×0,9 дюйма), разработанная компанией Adafruit. В форм-факторе Feather выпускается множество любительских электронных устройств по всему миру (среди самых популярных, например, ESP 8266 и ESP 32). Подробнее об этом стандарте см. **https://learn.adafruit.com/adafruit-feather/feather-specification**. – *Прим. перев.*

- Процессор **SparkFun MicroMod RP2040**. Это еще один вариант от SparkFun (**https://bit.ly/3clp0hG**). Он поставляется с 16 МБ встроенной флеш-памяти в форм-факторе MicroMod, использующем стандарт M.2[6]. На рис. 1.4 вы можете увидеть верхнюю сторону платы. Обратите внимание на выемку в форме полумесяца, которая используется для крепления модуля к несущей плате с помощью винта M2.5.

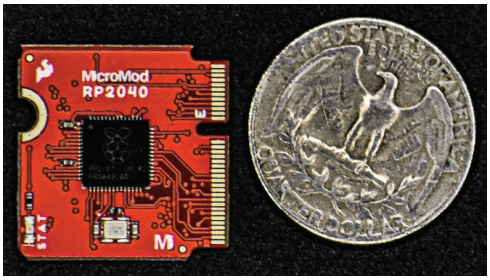

Рисунок 1.4. Процессор MicroMod RP2040

SparkFun также производит различные несущие платы для экосистемы MicroMod. Например, несущая плата (**https://bit.ly/3cnlrHF**), показанная на следующем рисунке, была разработана для управления дисплеем с интерфейсом **HDMI** (**high-definition multimedia interface**, *мультимедийный интерфейс высокой четкости*) с использованием RP2040.

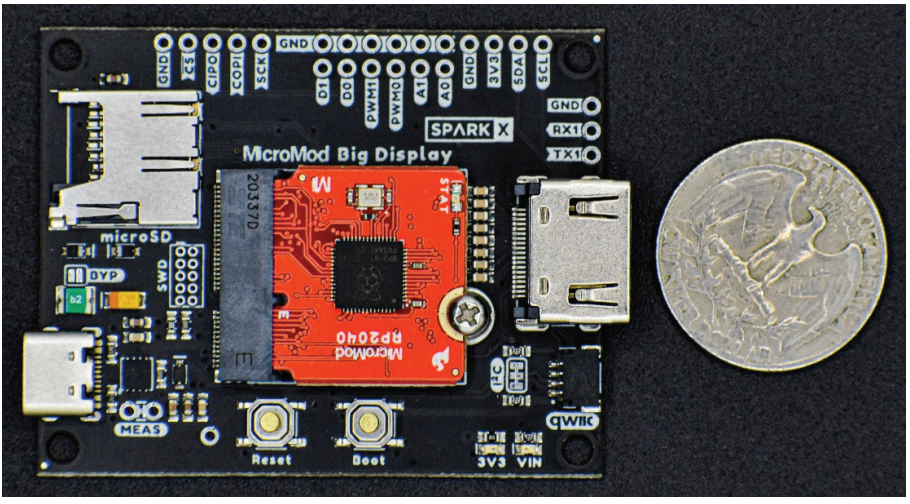

Рисунок 1.5. Большая дисплейная плата MicroMod для процессора RP2040

- **SparkFun Pro Micro – RP2040**. Плата Pro Micro – RP2040 (**https://bit.ly/3cnhVgH**) – это вариант, который принадлежит к относительно небольшой экосистеме семейства плат Pro Micro. На плате размещается 16 МБ

[6] Формат M.2 – спецификация компьютерных карт расширения и их разъёмов, расширяющая возможности PCI Express в сторону большей компактности. Плата стандарта M.2 имеет ножевой разъем, аналогичный PCI Express, но с уменьшенным шагом 0,5 мм (см. рис. 1.4). – *Прим. перев.*

флеш-памяти, индивидуально адресуемые светодиоды RGB и универсальные зубчатые площадки выводов, которые позволяют либо установить разъемы, либо припаивать модуль непосредственно к другой печатной плате. Универсальные контакты Pro Micro показаны на рис. 1.6.

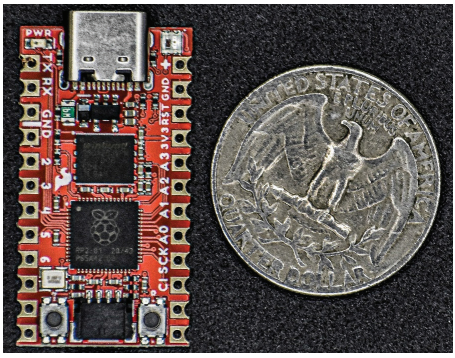

Рисунок 1.6. SparkFun Pro Micro – RP2040

- **Pimoroni Tiny 2040**. Эта плата от Pimoroni (**https://bit.ly/3d9f7Tf**) размером примерно с четверть дюйма имеет 8 МБ флеш-памяти и RGB-светодиод. Универсальные зубчатые площадки выводов позволяют припаять его либо к разъему, либо непосредственно к пользовательской печатной плате. Мы должны отметить, что вам понадобится вырез в вашей плате, чтобы припаять таким образом модуль. Это связано с тем, что микроконтроллер в плате Tiny 2040 находится на нижней стороне, как показано на рис. 1.7. Мы далее проиллюстрируем установку этой платы на пользовательскую печатную плату.

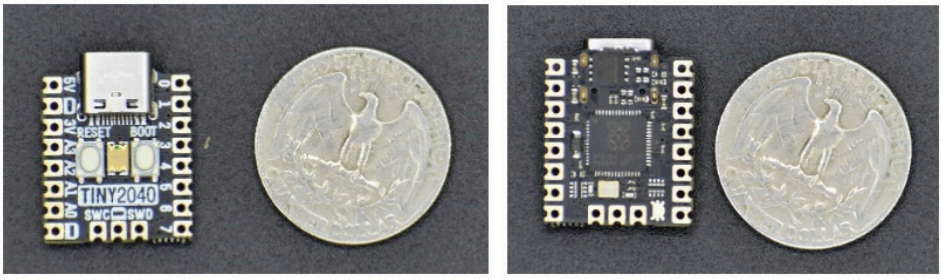

Рисунок 1.7. Pimoroni Tiny 2040

- **Adafruit Feather RP2040**. Как следует из названия, эта плата от Adafruit (**https://bit.ly/3cm3tW0**) представляет собой печатную плату с микроконтроллером RP2040, несущую 8 МБ флеш-памяти. Как и в случае с SparkFun Thing Plus, она отлично сочетается с разъемами Qwiic/STEMMA[7]. Плата показана на рис. 1.8.

[7] Qwiic и STEMMA – стандарты разъемов для подключения внешних модулей от компаний SparkFun (Qwiic) и Adafruit (STEMMA, см. также далее). Другие известные в нашей стране подобные стандарты – Grove (Seeed Studio), отечественные Trema (**iarduino.ru**) и Troyka (**amperka.ru**). – *Прим. перев.*

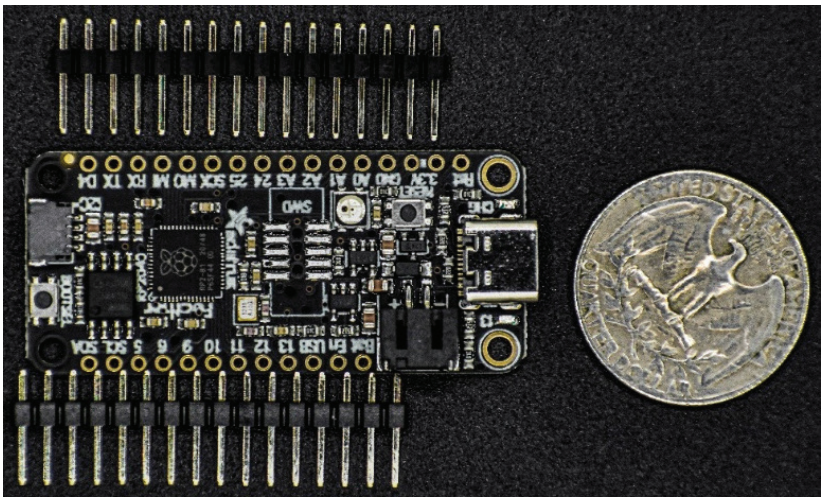

Рисунок 1.8. Adafruit Feather RP2040

- **Adafruit ItsyBitsy RP2040**. Эта плата от Adafruit (**https://bit.ly/3sqdB5R**) является дополнением к их линейке продуктов Itsy Bitsy. С точки зрения разводки выводов она идентична другим продуктам Itsy Bitsy от Adafruit. Эта плата поставляется с 8 МБ встроенной флеш-памяти. Вариант Itsy Bitsy, показанный на следующем рисунке, удобен для макетирования, что позволяет встроить плату в ваш проект.

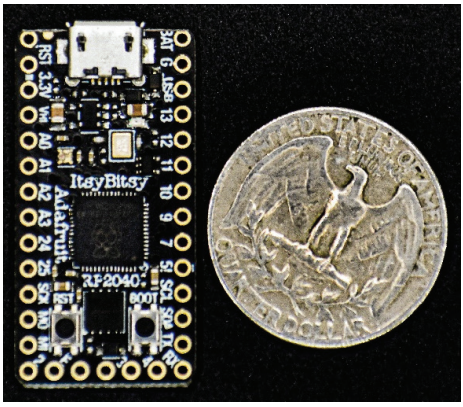

Рисунок 1.9. Adafruit ItsyBitsy RP2040

- **Adafruit QT Py RP2040**. Эта плата (**https://bit.ly/3lU201q**) является дополнением к семейству продуктов **QT Py** от Adafruit. Она также поставляется с 8 МБ встроенной флеш-памяти. Зубчатые контакты модуля, показанные на рисунке ниже, позволяют спроектировать несущую плату, с помощью которой плата QT Py RP2040 может быть встроена в ваш дизайн. Поскольку микроконтроллер RP2040 расположен на нижней стороне, вам необходимо убедиться, что в вашей конструкции есть вырез для размещения QT Py.

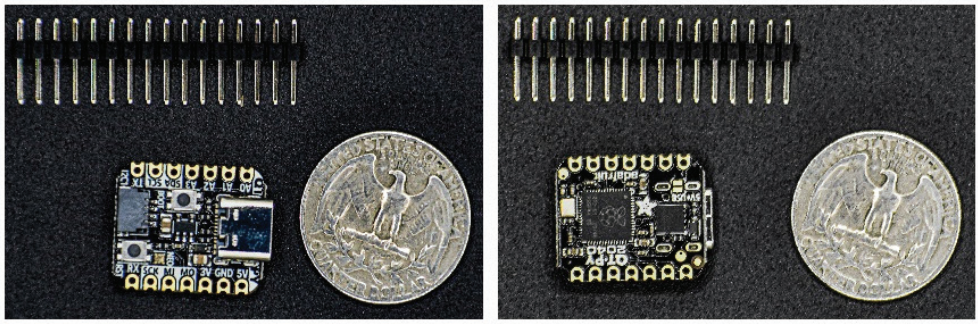

Рисунок 1.10. Adafruit QT Py RP2040

Варианты, которые мы обсуждали здесь, не исчерпывают все разнообразие плат на основе RP2040, но мы хотели представить некоторый выбор для начала работы с микроконтроллером. Например, если вы знакомы с форм-фактором Feather, то можете начать с платы Thing Plus от SparkFun или платы Feather от Adafruit, которые рассматриваются в этом разделе. Вы можете использовать любую плату по вашему выбору, но режим использования и интерфейс могут отличаться в зависимости от варианта. Мы постараемся подчеркнуть различия, где это возможно.

Пайка соединительного разъема Pico

В этом разделе мы рассмотрим настройку Pico для предстоящих проектов. Это включает в себя пайку штыревого разъема для внешних соединений и, возможно, изготовление корпуса с помощью 3D-принтера.

Пайка разъема

Pico выпускается с 40 отверстиями для разъема в два ряда по 20 контактов. Нам нужно припаять разъем, чтобы получить доступ к периферийным устройствам микроконтроллера RP2040 для нашего проекта.

> **Важное замечание**
>
> Пайка разъемов требует предварительной подготовки и присмотра взрослых. Не пытайтесь выполнять пайку без предварительной подготовки. Здесь вы можете просмотреть учебное пособие по пайке:
>
> **https://electricsexpert.ru/kak-pravilno-payat-payalnikom/**.

Вы можете приобрести разъемы из того же источника, что и Pico[8]. Шаги, которые необходимо выполнить для пайки, включают следующее:

[8] Требуется два штыревых разъема на 20 прямых контактов с шагом 2,54 мм (они показаны на рис. 1.8 и 1.10). Официальное название разъема – PLS-20. Если разъемы не входят в комплект поставки платы Raspberry Pi Pico, то в России их можно приобрести из источников, указанных в примечании на стр. 23. – *Прим. перев.*

1. Проще всего припаять разъемы с помощью макетной платы. Разместите их на макетной плате и уложите поверх нее Pico, как показано на рис. 1.11.

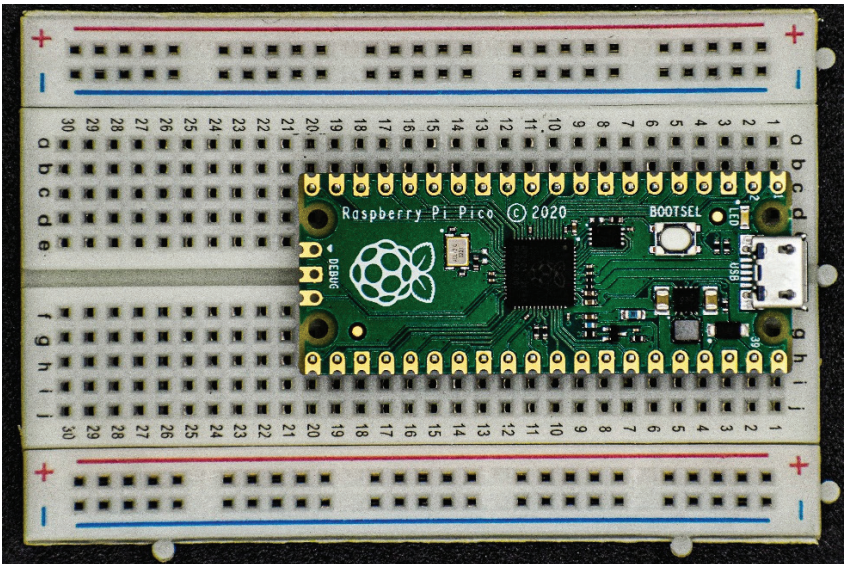

Рисунок 1.11. Pico с разъемами на макетной плате

2. Если вы будете паять отдельные контакты слишком медленно, то это может привести к повреждению макетной платы из-за избыточного тепла. На следующем рисунке показан Pico с запаянными контактами:

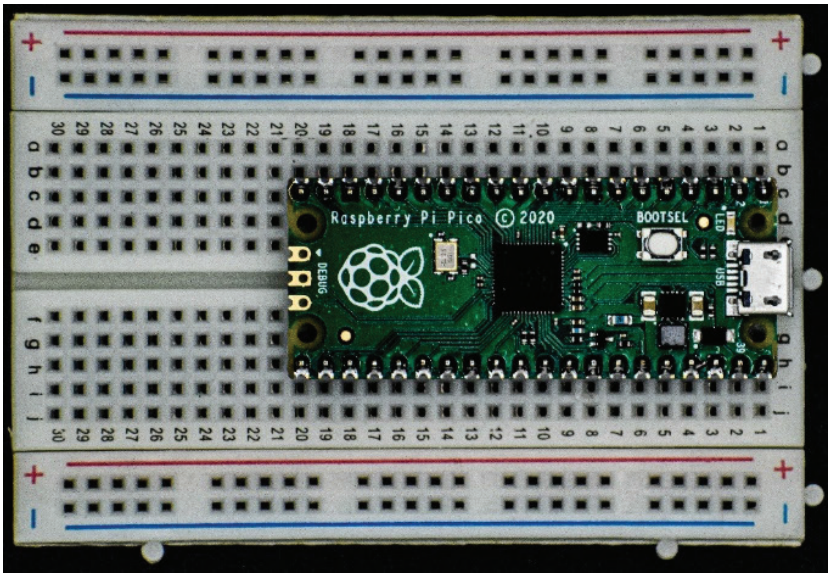

Рисунок 1.12. Pico на макетной плате

Теперь, когда мы припаяли разъемы, мы готовы попробовать. В следующем разделе мы рассмотрим необязательный шаг добавления кнопки сброса для Pico.

Реализация примера «Hello World!»

На любом языке программирования первое упражнение по кодированию состоит в том, чтобы вывести на экран «Hello World». Мы обсудим этот пример с использованием MicroPython и CircuitPython.

Кнопка сброса для Pico (дополнительно)

Pico не содержит кнопку сброса. Чтобы сбросить Pico, вам придется отсоединить и снова подключить USB-кабель, что может быть утомительно. Это можно преодолеть, добавив кнопку сброса между выводами платы 28 (GND) и 30 (Reset), как показано на следующем рисунке (вы можете получить такую кнопку по этой ссылке: **https://bit.ly/3w8AmO2**)[9]:

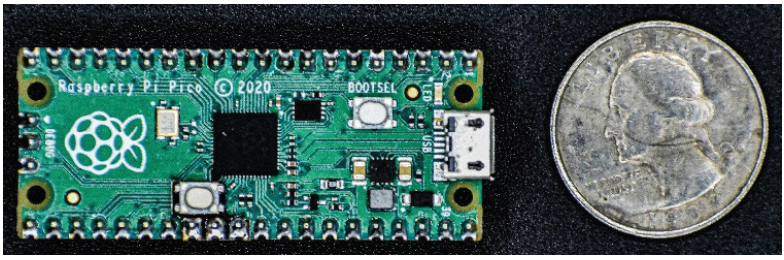

Рисунок 1.13. Кнопка сброса на Pico

Кнопка сброса упрощает перезапуск программы во время разработки. Добавление кнопки сброса необязательно, и не рекомендуется к этому приступать, если вы не чувствуете себя свободно в приемах пайки компонентов.

В следующем разделе мы напишем первую программу для Pico.

MicroPython

В этом примере мы будем использовать **Thonny IDE** (IDE означает **integrated development environment** – интегрированная среда разработки). Thonny IDE можно загрузить с **https://thonny.org**. Она доступна для операционных систем Windows, Mac и Linux.

Чтобы можно было сразу приступать к программированию Raspberry Pi, он выпускается с предустановленным Thonny. Вы можете запустить Thonny из **Menu** | **Programming** (*Меню* | *Программирование*) и прокрутить вниз раскрывающийся список, чтобы найти **Thonny Python IDE**, как показано на рис. 1.14.

[9] Вместо экзотической кнопки стоимостью, как указано по ссылке, почти два евро, можно применить обычную тактовую кнопку стоимостью около 10 рублей, доступную в источниках согласно примечанию на стр. 23. Ее не припаивают к плате Pico, а устанавливают на макетную плату между контактами платы 28 и 30. – *Прим. перев.*

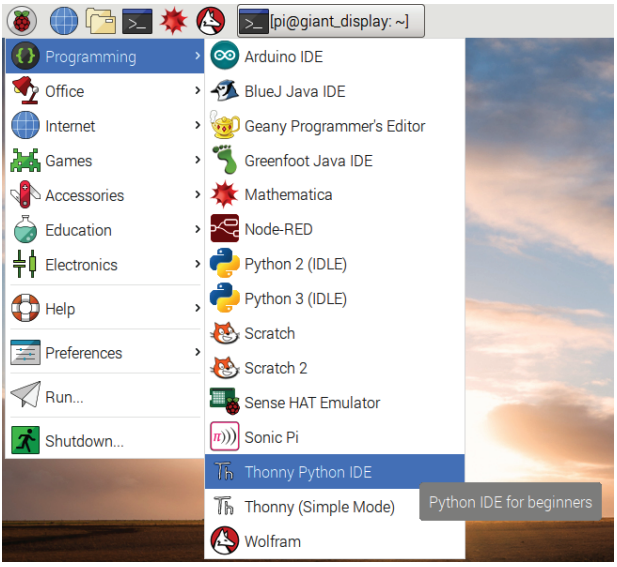

Рисунок 1.14. Расположение Thonny IDE на рабочем столе Raspberry Pi

Теперь давайте подготовим Pico, прошив двоичный файл MicroPython.

Прошивка двоичного файла MicroPython

Для прошивки файла Pico должен быть подключен к компьютеру определенным образом. Шаги по прошивке двоичного файла следующие.

1. Первый шаг – загрузить двоичный файл с **https://bit.ly/31nBMFW**.
2. Второй шаг: нажмите и удерживайте кнопку загрузки, показанную на рис. 1.15, при подключении кабеля microUSB к Pico. Убедитесь, что другой конец USB-кабеля подключен к компьютеру.

Рисунок 1.15. Кнопка загрузки Pico BOOTSEL

3. Pico должен появиться на вашем компьютере как запоминающее устройство, которое может выглядеть следующим образом:

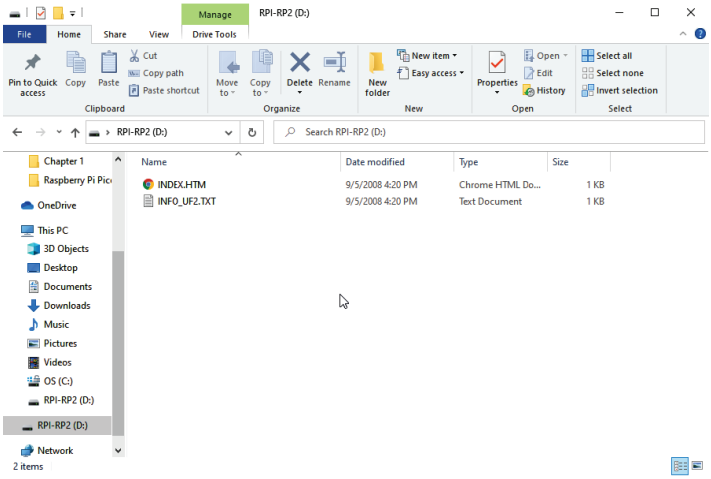

Рисунок 1.16. Raspberry Pi Pico как запоминающее устройство

4. Затем скопируйте двоичный файл на накопитель. Pico сбросит сам себя, и мы готовы писать программы на MicroPython.

Написание первой программы

Давайте запустим Thonny и отправимся на тест-драйв! Вам нужно будет предпринять следующие шаги.

1. Запустите Thonny, и вы должны увидеть окно, как показано на рис. 1.17.

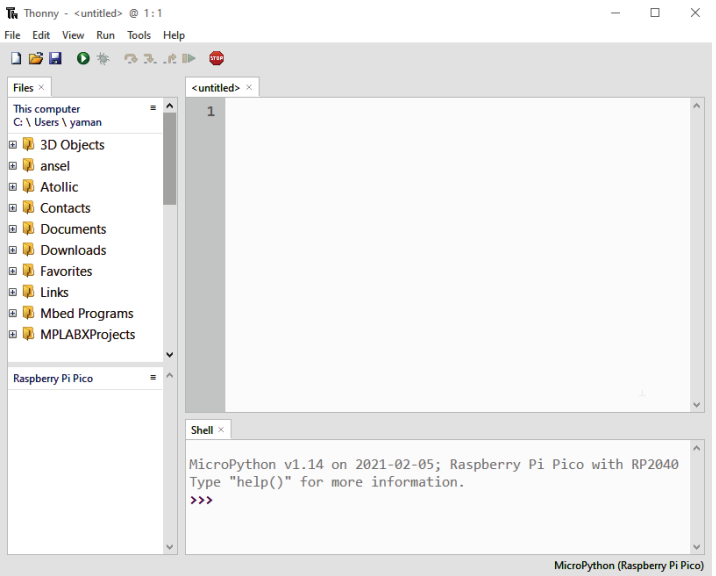

Рисунок 1.17. Thonny IDE

2. Мы собираемся запустить нашу первую программу, используя интерпретатор, работающий на вашем Pico. Интерпретаторы Python позволяют тестировать код по мере его написания. Перейдите в раздел **Run** (*Выполнить*), а затем нажмите кнопку **Select Interpreter** (*Выбрать интерпретатор*).

   Здесь выберите следующие параметры, как показано на рис. 1.18:

   - **MicroPython (Raspberry Pi Pico)**;
   - **< Try to detect port automatically >** (*Попробуйте автоматически определить порт*).

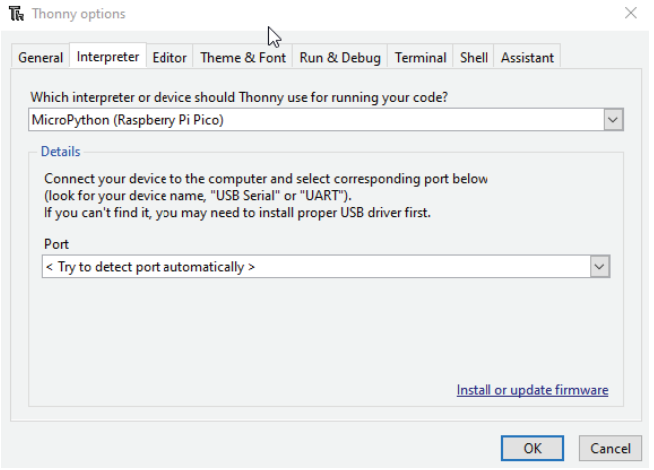

Рисунок 1.18. Выбор интерпретатора Python

3. Теперь должен загрузиться интерпретатор Python (см. следующий рисунок). Это интерпретатор MicroPython, работающий на Pico. Давайте проведем тест-драйв.

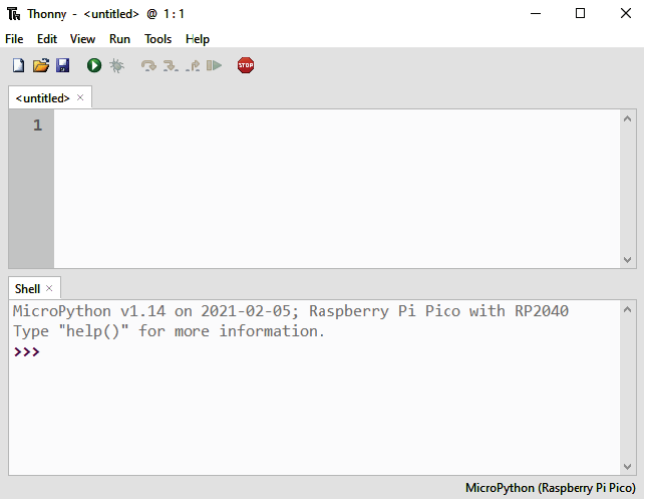

Рисунок 1.19. Интерпретатор Python

4. В командной строке после знака приглашения >>> введите следующую строку кода:

```
>>> print("Hello World")
```

Вы должны увидеть следующий результат:

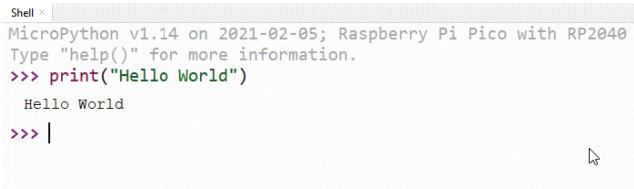

Рисунок 1.20. Результаты работы интерпретатора

Вы только что закончили писать свою первую программу на Raspberry Pi Pico! В следующем примере мы напишем сценарий, который начнет выполняться непрерывно, когда Pico питается от USB-кабеля.

Реализация примера мигания светодиода

В предыдущем разделе мы использовали интерпретатор для написания нашей программы. Интерпретатор может быть полезен при тестировании кода или получении дополнительной информации об импортируемых модулях. В этом примере мы обсудим написание скрипта, который автоматически запускается при включении Pico. Мы обсудим пример начала работы с электроникой в программе, аналогичной «Hello World», которая будет мигать светодиодом с интервалом в 1 секунду.

Pico поставляется с зеленым светодиодом на плате, и его расположение показано на рис. 1.21.

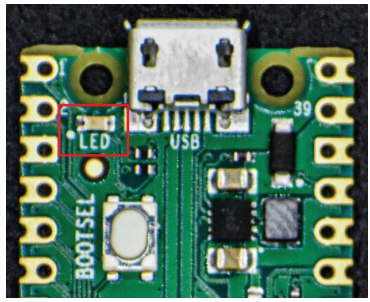

Рисунок 1.21. Расположение светодиода на плате Pico

Мы заставим этот светодиод мигать с интервалом в 1 секунду, то есть будем каждую секунду его включать и выключать. Для того чтобы написать эту программу, нам понадобятся модули `machine` и `utime`.

Согласно спецификации Pico, встроенный светодиод подключен к универсальному вводу-выводу (**general-purpose I/O, GPIO**) номер 25. Мы будем действовать следующим образом:

- для управления светодиодом используем класс `Pin` из модуля `machine`;
- для введения задержки между включением и выключением светодиода используем модуль `utime`.

Теперь вам нужно будет выполнить такие действия.

1. Давайте взглянем на следующий пример кода:

```
from machine import Pin
import utime

led = Pin(25, Pin.OUT)

while True:
led.toggle()
utime.sleep(1)
```

> **Упражнение**
>
> Выработайте практику активного использования интерпретатора во время написания кода. В интерпретаторе импортируйте модули `machine` и `utime`, попробуйте выполнить `help(machine)`, `help(utime)` и разберитесь в их возможностях.

2. Создайте новый файл и введите фрагмент кода, показанный на шаге 1. Установите место для размещения файла *Raspberry Pi Pico*, как показано на следующем рисунке:

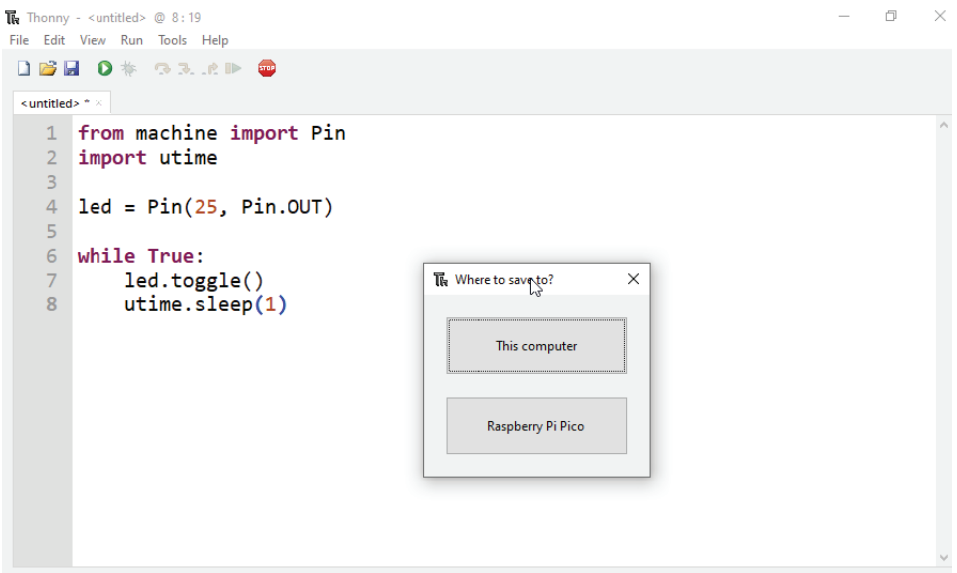

Рисунок 1.22. Расположение файла

3. Сохраните файл как *main.py*, и ваш код должен автоматически начать выполняться.
4. Попробуйте отсоединить USB-кабель и снова подключить его. Вы заметите, что скрипт запускается автоматически.
5. Чтобы остановить выполнение, нажмите на кнопку **STOP**, расположенную на панели инструментов, как показано на следующем рисунке:

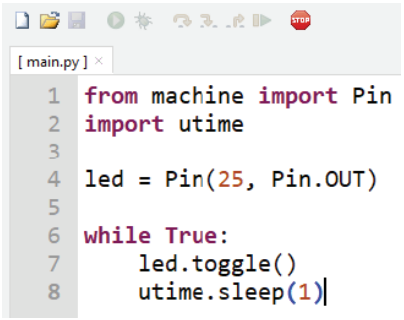

Рисунок 1.23. Кнопка **STOP** для прерывания выполнения

6. Чтобы возобновить выполнение, нажмите кнопку **Run Current Script** (*Запустить текущий сценарий*), на которую указывает зеленая кнопка воспроизведения, указанная в верхней части страницы, как показано на следующем рисунке:

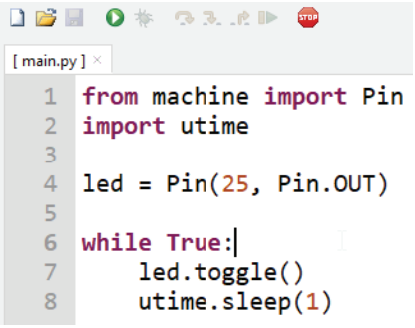

Рисунок 1.24. Кнопка **Run Current Script** для запуска выполнения скрипта

В следующем разделе мы более подробно рассмотрим код.

Описание примера кода

В этом разделе мы подробно рассмотрим пример кода, показанный ранее.

1. Мы начинаем с импорта модуля `utime` и класса `Pin` из модуля `machine` следующим образом:

```
from machine import Pin
import utime
```

GPIO

Вывод GPIO можно использовать в качестве входного или выходного вывода. Когда вы используете вывод GPIO в качестве выходного вывода, вы можете установить его в HIGH (высокий) или LOW (низкий) уровень. Аналогично, когда вы используете его в качестве входного вывода, вы можете определить, имеет ли сигнал на входе HIGH (высокий) или LOW (низкий) уровень. В этом примере мы включаем/выключаем светодиод, чередуя состояния HIGH и LOW.

2. Далее мы объявляем вывод 25 GPIO в качестве выходного вывода, используя объект, принадлежащий классу `Pin`. Первый аргумент в следующем фрагменте кода ссылается на номер вывода 25, в то время как второй аргумент устанавливает вывод в качестве выходного:

```
led = Pin(25, Pin.OUT)
```

3. Теперь нам нужно мигать светодиодом с интервалом в 1 секунду. Мы собираемся сделать это внутри бесконечного цикла.
4. Объект `led` имеет функцию переключения `toggle()`, которая переключает вывод между включенным (ON) и выключенным (OFF) состояниями.
5. Мы введем задержку в 1 секунду, вставив функцию ожидания после очередного переключения следующим образом:

```
while True:
    led.toggle()
    utime.sleep(1)
```

Поздравляем вас с вашим первым шагом с Raspberry Pi Pico! Теперь мы обсудим тот же пример в CircuitPython.

Пример CircuitPython

В этом разделе мы обсудим использование **CircuitPython** для программирования Pico на том же примере мигания светодиода. Мы также установим среду *Mu IDE* в программу с помощью CircuitPython.

Прошивка двоичного кода CircuitPython

В этом разделе мы будем прошивать двоичный файл CircuitPython на Pico. Процесс установки такой же, как и для MicroPython. Двоичный файл можно загрузить с сайта **https://bit.ly/31pnLI4**.

После того как вы загрузили двоичный файл, следуйте инструкциям из раздела «*MicroPython*».

Кодирование с помощью Mu

В этом разделе мы установим среду разработки Mu IDE. Ее можно загрузить с **https://bit.ly/3ruxDKW**. Mu IDE доступна для операционных систем Windows, Linux и Mac.

Установка на Raspberry Pi

На Raspberry Pi Mu IDE можно установить следующим образом:

1. Перейдите в **Menu | Preferences | Recommended Software | Select Mu** (*Меню | Настройки | Рекомендуемое программное обеспечение | Выбрать Mu*), чтобы выбрать Mu из списка устанавливаемого программного обеспечения.
2. Mu IDE можно запустить из **Menu | Programming | Mu** (*Меню | Программирование | Mu*).

В следующем разделе мы рассмотрим написание нашей первой программы с помощью Mu.

Запуск Mu

Шаги по написанию программы с Mu включают в себя следующее.

1. Подключите Pico к компьютеру / Raspberry Pi с помощью USB-кабеля.
2. Как только Mu будет запущен, нам нужно установить режим программирования. Поскольку мы программируем на CircuitPython, мы установим для него значение **CircuitPython**, как показано на следующем рисунке:

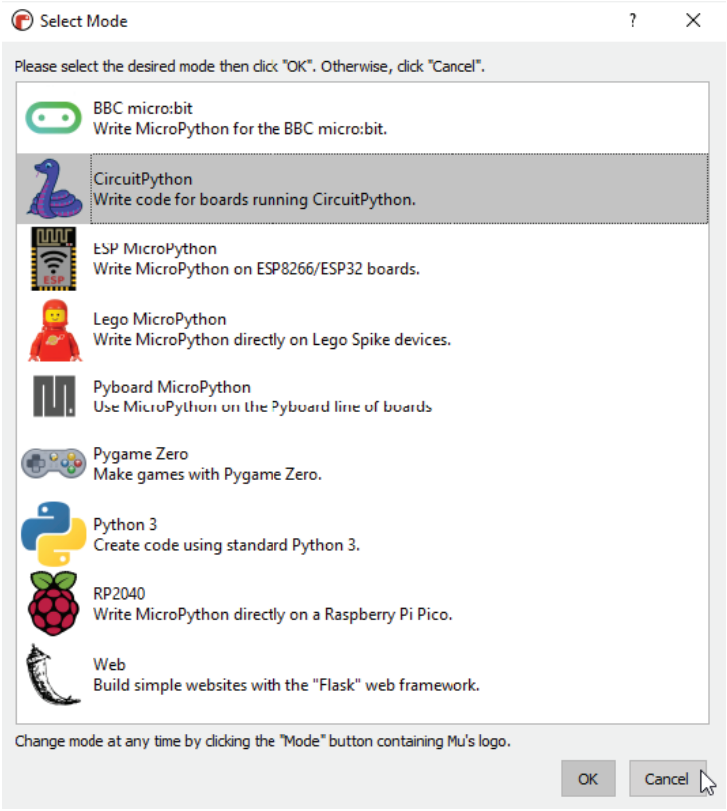

Рисунок 1.25. Выбор режима программирования

3. Вы также можете вернуться к установке режима программирования из главного окна IDE, как показано на следующем рисунке:

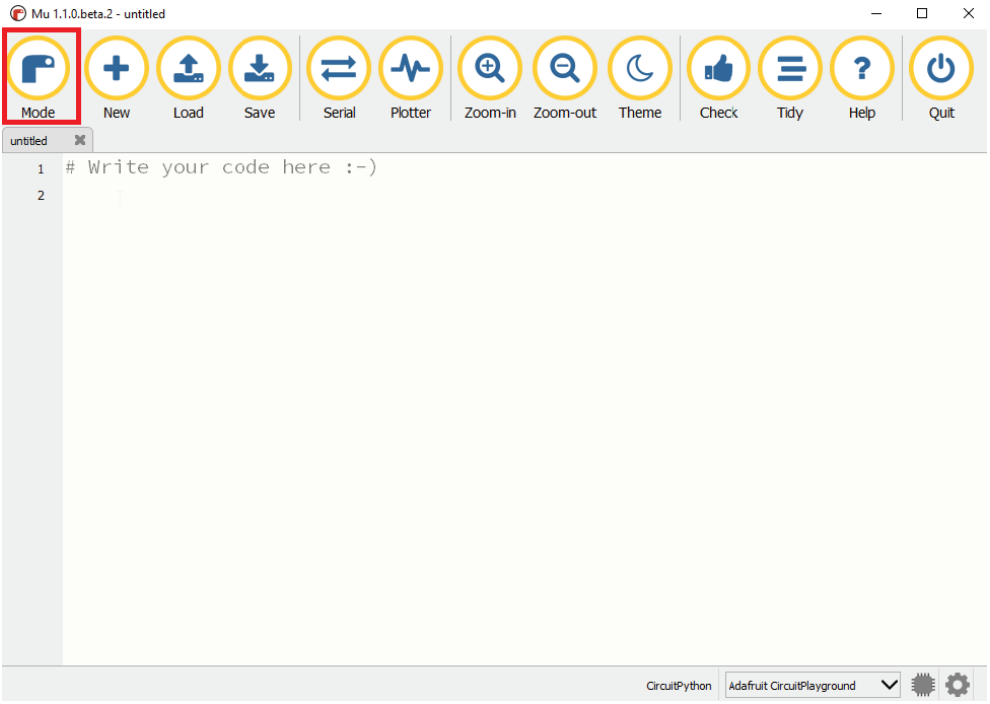

Рисунок 1.26. Изменение режима Python

Обратите внимание, что на рис. 1.25 Pico обнаруживается автоматически.

4. Нажмите кнопку **Serial**, чтобы запустить доступ к интерпретатору Python, как показано здесь:

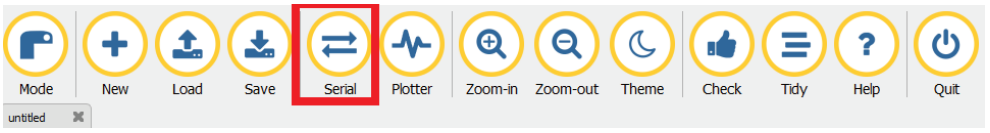

Рисунок 1.27. Расположение кнопки **Serial**

5. Запустите интерпретатор Python, нажав <**Ctrl**+**C**>, и он должен привести вас к приглашению REPL >>>, как показано в нижней части следующего рисунка:

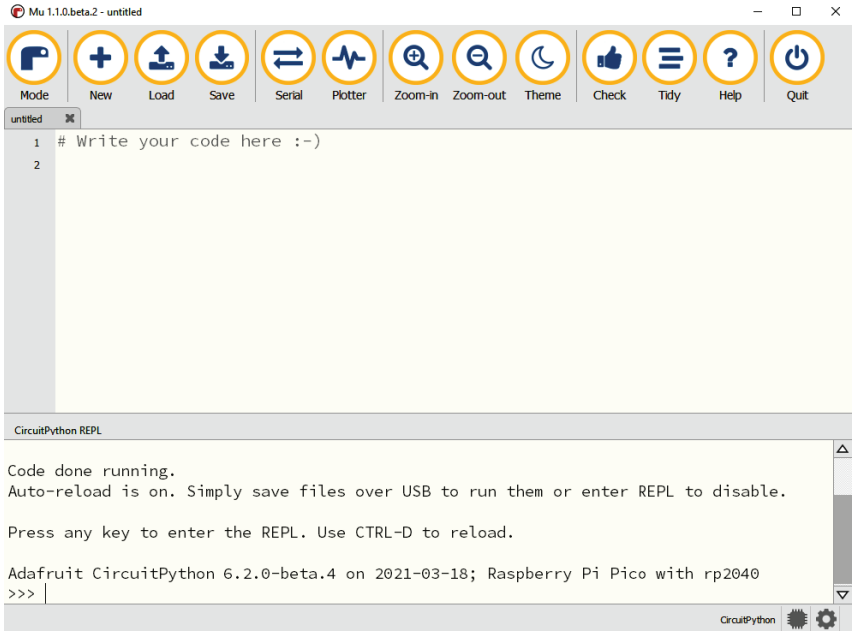

Рисунок 1.28. Интерпретатор CircuitPython на Pico

Наконец, в этой среде повторите пример «Hello World» из раздела MicroPython с помощью интерпретатора.

Второй пример мигания светодиода

Мы обсудим тот же пример мигания светодиода с использованием CircuitPython и рассмотрим различия между двумя вариантами Python. Взглянем на следующий фрагмент кода:

```
import time
import board
import digitalio

led = digitalio.DigitalInOut(board.LED)
   led.direction = digitalio.Direction.OUTPUT

   while True:
   led.value = True
   time.sleep(1)
   led.value = False
   time.sleep(1 )
```

Давайте рассмотрим пример кода следующим образом:

1. Мы начинаем с импорта модулей `time`, `board` и `digitalio`. Модуль времени `time` используется для введения задержки между включением и выключением светодиода.

2. Модуль платы `board` содержит определения контактов и периферийных устройств, специфичных для выбранной платы. В этом примере мы используем из модуля `board` константу `LED` для управления встроенным светодиодом на Raspberry Pi Pico.
3. Модуль `digitalio` обеспечивает доступ к периферийным устройствам Pico. В этом примере нам нужно объявить вывод светодиода (вывод GPIO 25) в качестве выходного вывода:

```
led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT
```

4. В первой строке предыдущего фрагмента кода мы объявляем `led` экземпляром класса `DigitalInOut`.
5. Во второй строке мы устанавливаем вывод светодиода в качестве выходного вывода.
   Мы используем класс `Direction` из модуля `digitalio`.
6. Далее мы входим в бесконечный цикл, в котором включаем/выключаем светодиод следующим образом:

```
while True:
    led.value = True
    time.sleep(1)
    led.value = False
    time.sleep(1 )
```

7. В первой строке цикла `while` мы для `led.value` устанавливаем значение `True`. Это включает светодиод. За этим следует задержка в 1 секунду, что достигается вызовом `time.sleep(1)`.
8. В третьей строке цикла `while` мы устанавливаем для `led.value` значение `False`. При этом индикатор гаснет. За этим также следует задержка в 1 секунду.
9. Мы хотим, чтобы скрипт запускался после сброса. Загрузите файл *code.py*, расположенный на Pico, в данный момент указанном на вашем компьютере в качестве устройства хранения. Кнопка *Load* (*Загрузка*) расположена на верхней панели инструментов.
10. Введите пример кода, который мы обсуждали, в *code.py* и сохраните его.
11. Нажмите <**Ctrl**+**D**> в интерпретаторе CircuitPython, и вы должны заметить мигание светодиода на Pico.

Поздравляем вас с написанием вашей первой программы CircuitPython для Raspberry Pi Pico!

CircuitPython или MicroPython?

В этой главе мы рассматривали примеры как с CircuitPython, так и с MicroPython. Примеры в чем-то идентичны и имеют схожую структуру. В чем их различия и какой вариант Python вы должны использовать для своей разработки?

Короткий ответ заключается в том, что это зависит от вас. Для единообразия мы будем обсуждать все примеры в CircuitPython с использованием Mu IDE.

Обе реализации имеют широкую пользовательскую базу и библиотеки для дополнительного оборудования. CircuitPython был выделен из MicroPython компанией Adafruit. CircuitPython может быть полезен при использовании интерфейсных модулей от Adafruit Industries.

> **Thonny и Mu IDE**
>
> В этой главе вы, возможно, заметили, что мы использовали редактор Thonny для примера MicroPython и Mu для примера CircuitPython. Мы хотели продемонстрировать различные инструменты, доступные для Raspberry Pi Pico. Вы даже можете использовать простой текстовый редактор для своей разработки. Мы покажем вам, как сохранить и загрузить ваш код в Pico.

В следующем разделе мы обсудим дополнительное оборудование для Pico.

Подключение полезного дополнительного оборудования для Pico

В этом разделе мы покажем варианты решений для выполнения макетов с дополнительным оборудованием, выпускаемые для Pico. Отметим, что мы обсуждаем только комплекты, разработанные специально для Pico. Этот список не является исчерпывающим, и мы выбрали ряд примеров, руководствуясь их различиями. Вы можете выбрать любой вариант по вашему выбору, но сразу скажем, что это необязательно. Простой беспаечной платы может быть достаточно, и макет будет выглядеть, как показано на рис. 1.29.

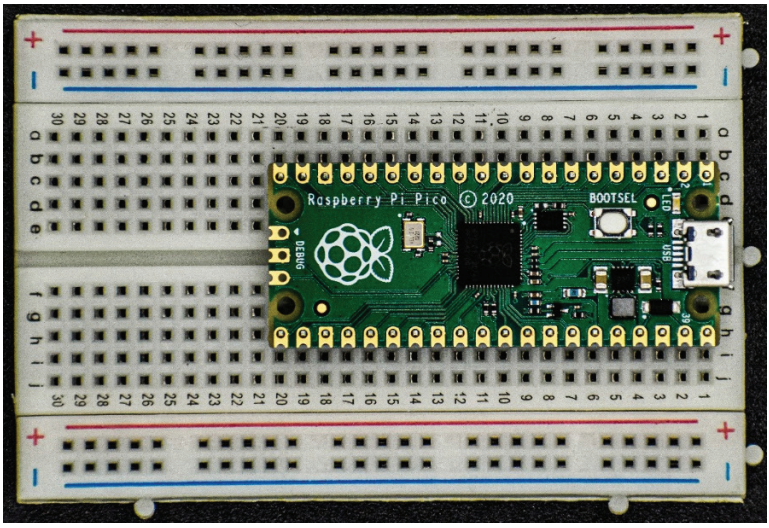

Рисунок 1.29. Pico на макетной плате

Далее мы рассмотрим варианты, в которых Pico поставляется в комплекте с различными макетными платами, облегчающими подключение.

Pico Breadboard Kit (набор Pico с макетной платой)

Как следует из названия, здесь Pico поставляется с макетной платой, а также с четырьмя светодиодами, четырьмя кнопками и зуммером. Имеется пара разъемов для установки Pico на плате и два ряда разъемов для доступа ко всем контактам, имеющимся на плате Pico. Изображение комплекта можно увидеть на рис. 1.30. Комплект можно приобрести по адресу **https://bit.ly/3tV7ala**.

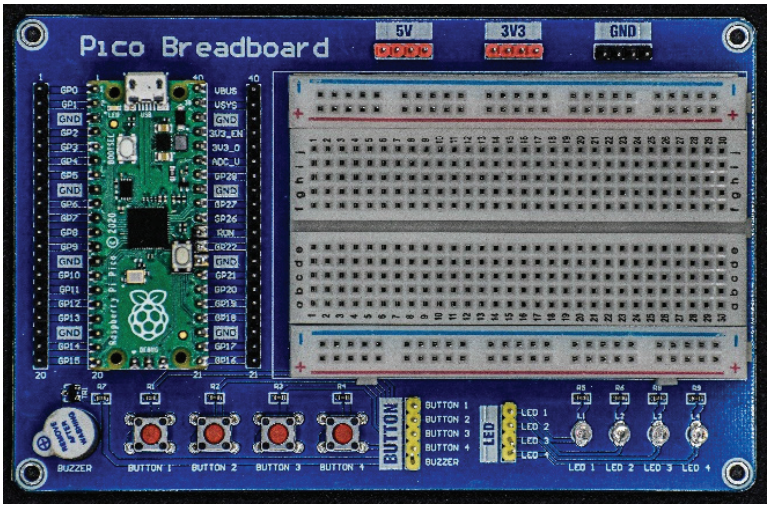

Рисунок 1.30. Pico Breadboard Kit

Pico Breadboard Kit может быть полезен абсолютному новичку в электронике.

Pico GPIO Expansion Board (плата расширения выводов GPIO Pico)

Этот комплект обеспечивает доступ ко всем выводам Pico. С обеих сторон есть два ряда штыревых и гнездовых разъемов, как показано на рис. 1.31. Эту плату можно приобрести по адресу **https://bit.ly/3rprobq**.

Два ряда штыревых и гнездовых разъемов позволяют использовать соответствующие соединительные кабели для создания прототипов.

Pico HAT Expansion (расширение Pico HAT)

Эта плата позволяет подключать любую плату Raspberry Pi **HAT** (что означает **Hardware Attached on Top** – аппаратура, устанавливаемая сверху) к Pico. В комплекте поставляется разъем 2×20, который позволяет подключать к плате оборудование HAT (см. рис. 1.32 далее). Плата также обеспечивает доступ к выводам платы Pico, которые подключены к выводам разъема HAT. Плату можно приобрести здесь: **https://bit.ly/3lYRfpu**. На указанной веб-странице также предоставляется сопоставление выводов Pico и HAT.

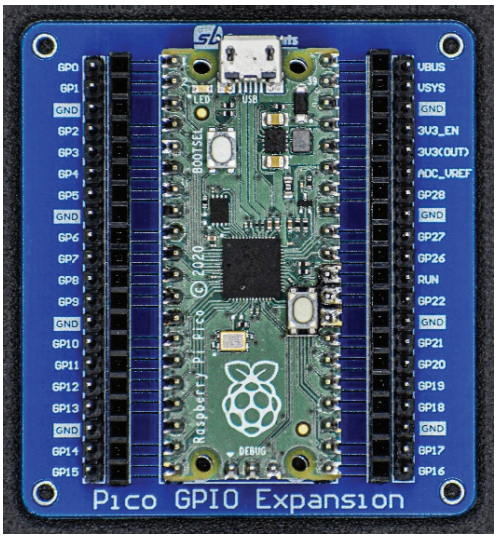

Рисунок 1.31. Pico GPIO Expansion Board

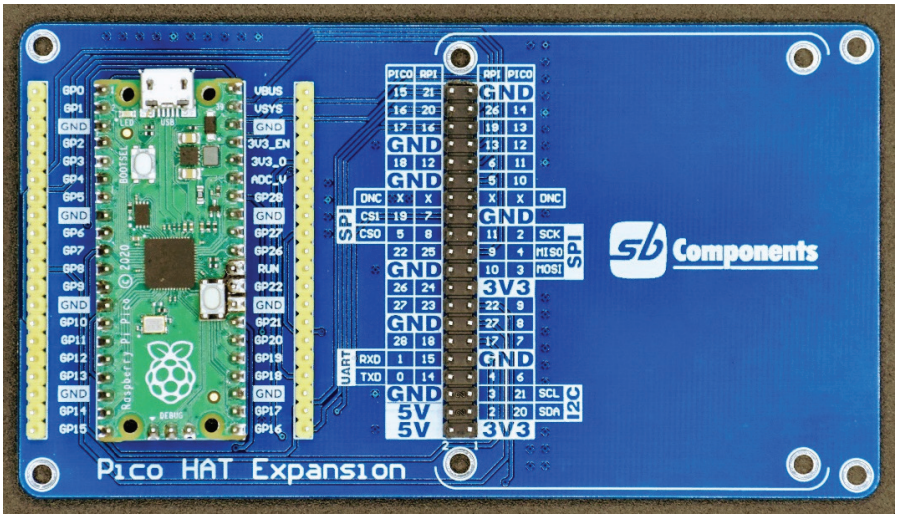

Рисунок 1.32. Pico HAT Expansion

Эта плата позволит добавить Pico в существующий проект Raspberry Pi. Это также может позволить вам использовать HAT-оборудование Raspberry Pi с Pico.

Grove Shield for Pi Pico (плата расширения Grove для Pi Pico)

Плата расширения Grove Shield for Pi Pico помогает подключить Pico к экосистеме Grove от Seeed Studio. На случай, если вы с ней незнакомы: экосистема Grove состоит из модульных плат для создания макетов и прототипов. Как вы

можете видеть на рис. 1.33, плата состоит из ряда разъемов, которые позволяют подключать ее к датчикам и исполнительным механизмам. Плату можно приобрести по адресу **https://bit.ly/2NZG3MO**.

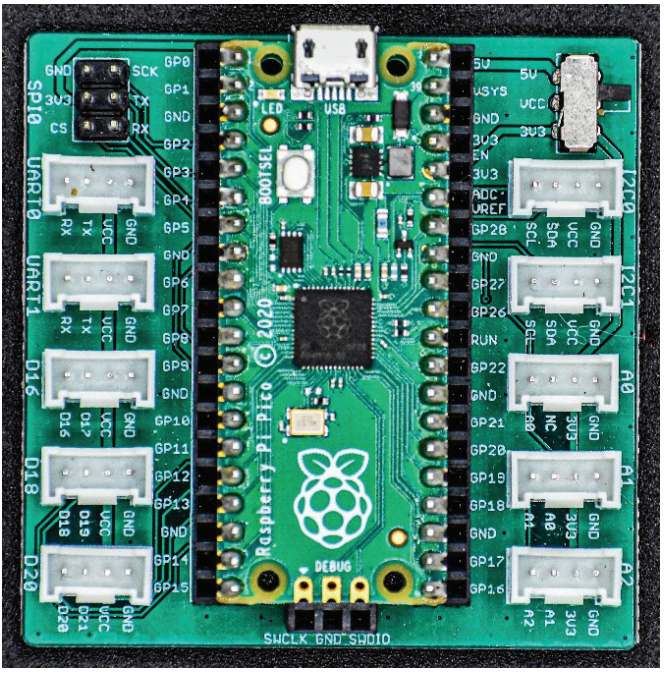

Рисунок 1.33. Grove Shield for Pi Pico

В примерах далее мы обсудим макетирование с помощью подобных плат расширения.

Pimoroni Pico Decker (четырехкратный расширитель)

Эта плата позволяет подключать до четырех плат расширения к Pico (см. рис. 1.34), но важно убедиться, что между дополнительными платами нет конфликтов контактов. Плату можно приобрести на сайте **https://bit.ly/3lXpWMt**.

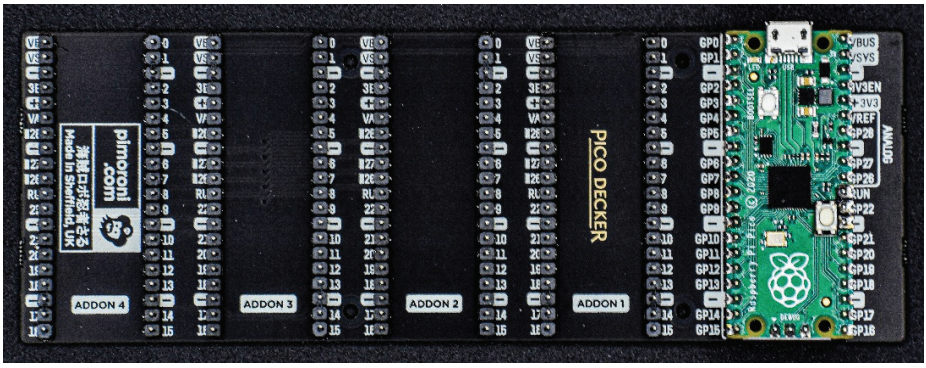

Рисунок 1.34. Pimoroni Pico Decker (четырехкратный расширитель)

В этом разделе мы рассмотрели различные комплекты, доступные для Raspberry Pi Pico. Список не является исчерпывающим – например, платы Feather/Thing Plus поставляются с совместимым оборудованием, известным как FeatherWings. Аналогичным образом платы MicroMod поставляются со своей собственной экосистемой дополнительного оборудования.

Итоги

В этой главе мы познакомили вас с Raspberry Pi Pico. Мы обсудили периферийные устройства микроконтроллера RP2040 и варианты платы Pico различных производителей оборудования. Мы также обсудили дополнительные комплекты для макетирования, доступные для Pico. Мы припаяли разъемы для Pico и добавили кнопку сброса. Мы также обсудили пример «Hello World!» и пример мигания светодиода с использованием CircuitPython и MicroPython.

Теперь, когда вы закончили настройку Pico и ознакомились с опциями, доступными для программирования Pico, мы обсудим особенности микроконтроллера RP2040 на практических примерах.

В следующей главе мы рассмотрим коммуникационные интерфейсы, доступные на Pico.