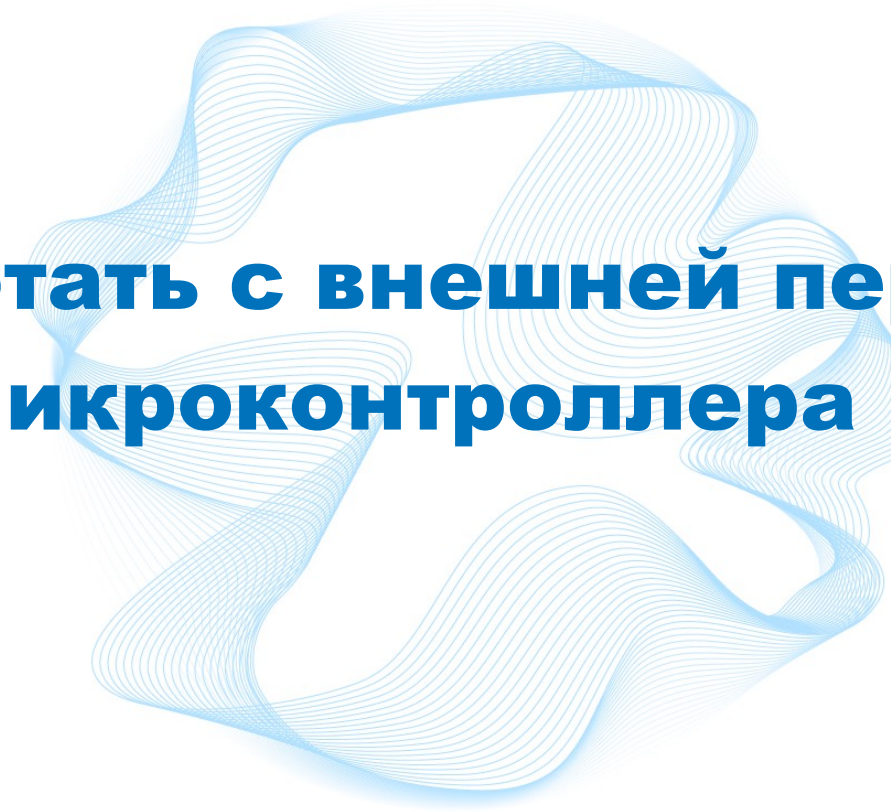


Как работать с внешней периферией микроконтроллера ?

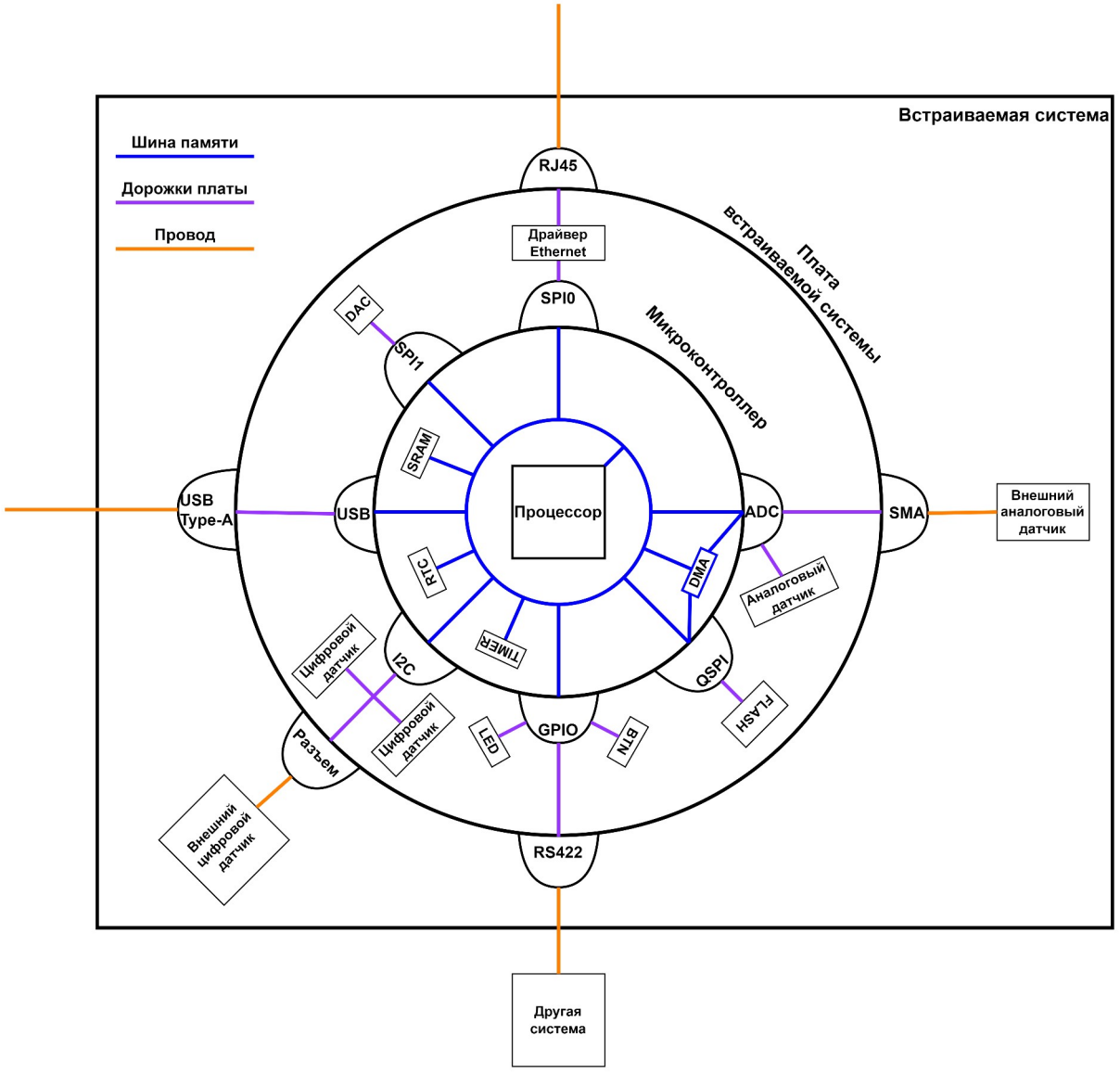
Цель занятия

- Разобраться в чем отличие внешней периферии от внутренней, в чем отличается взаимодействие с внешней периферией, как написать драйвер устройства

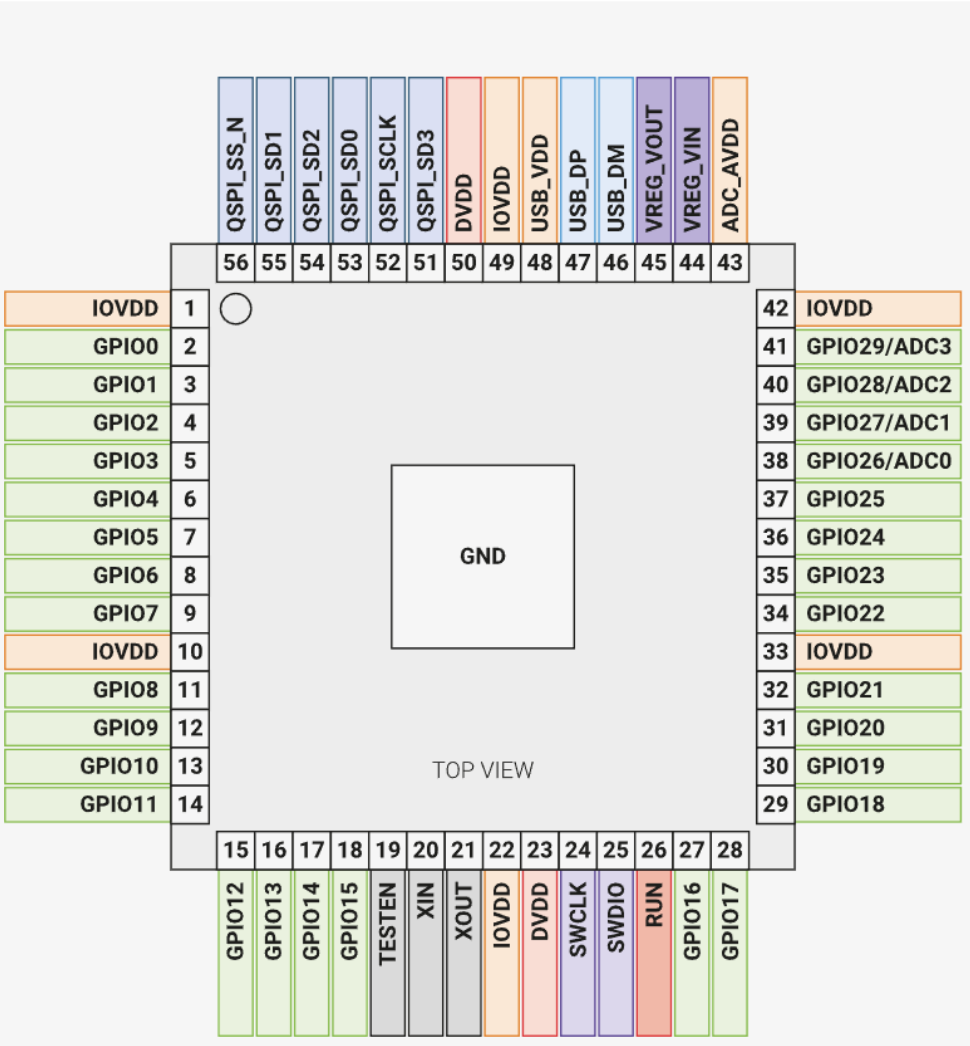
Внешняя периферия микроконтроллера

Определение

- Внешняя периферия (англ. external peripherals) — аппаратные устройства, соединённые с микроконтроллером при помощи физических интерфейсов обмена данными.
- Внешней периферией может быть что угодно: память (как флеш-память в Raspberry Pi Pico, подключённая через QSPI), вычислительные модули, сложные интерфейсы типа USB 3.0, Ethernet, WiFi, которые трудно разместить внутри микроконтроллера, а также аналоговые и цифровые датчики.


Встраиваемая система
Шина памяти
Дорожки платы
Провод
RJ45
Драйвер Ethernet
SPI0
Плата встраиваемой системы
Микроконтроллер
DAC
SPI1
SRAM
USB Type-A
USB
Процессор
ADC
SMA
Внешний аналоговый датчик
RTC
DMA
Аналоговый датчик
TIMER
Цифровой датчик
I2C
QSPI
FLASH
Разъем
Цифровой датчик
LED
GPIO
BTN
Внешний цифровой датчик
RS422
Другая система

- Каждый пин имеет своё назначение и может предоставлять одну или несколько функций
- Порт GPIO может работать как обычная ножка, в которую мы записываем логический 0 или 1 (как мы делали со светодиодом). Но можно подключить его к внутренней периферии: SPI, I2C, UART, PWM. Тогда ножка будет управляться соответствующим модулем.
- Не каждую ножку можно подключить к каждому модулю



Внеплатная периферия

- Периферия может не находиться на одной плате с микроконтроллером. Такую периферию будем называть **внеплатной**. Обычно это случается, когда встраиваемая система имеет большие размеры или когда необходимо производить измерения в удалённом месте.
- Например, на заводе может быть линия производства молока. По этой линии перемещаются пакеты, а множество датчиков и **актуаторов** (исполнительных устройств) следит за её состоянием и управляет ею. Но линия достаточно большая, и датчики разнесены — они находятся не на одной плате с микроконтроллером.
- Если мы говорим о простых датчиках, то главное их отличие от внутриплатных — способ подключения.

Основные характеристики интерфейсов

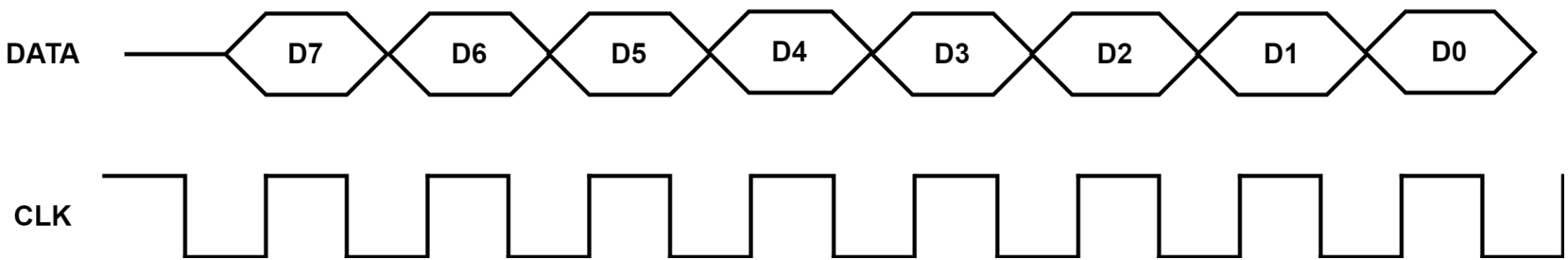
Последовательная и параллельная передача

- **Последовательная передача** — информация между устройствами передается порциями по байту
- **Параллельная передача** — информация между устройствами передается порциями по несколько байт за счет использования дополнительных проводников. Таким оразом увеличивается пропускная способность
- **Пропускная способность (англ. throughput)** — количество данных, передаваемых за единицу времени. Измеряется в битах в секунду (бит/с) или байтах в секунду (Б/с).

Порт LPT (Line Print Terminal)

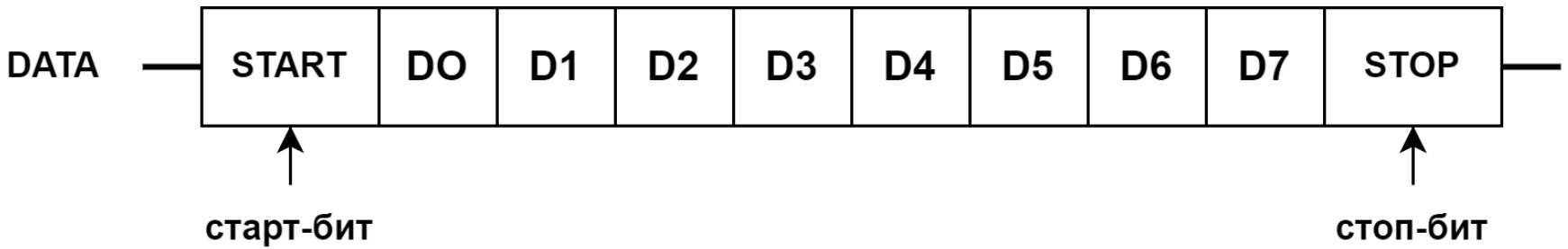
Синхронная передача

- **Синхронная передача** — Передатчик генерирует последовательность импульсов на специальной линии (обычно называется CLK или SCK), и каждый импульс говорит приёмнику: «сейчас читай». Эта последовательность импульсов называется тактовым сигналом. Приёмник считывает состояние линии данных по фронту или спаду тактового сигнала.



Асинхронная передача

- **Асинхронная передача** обходится без тактового сигнала. Вместо этого передатчик и приёмник заранее договариваются о скорости передачи — количестве бит в секунду. Эту скорость называют **бодовой скоростью** (англ. baud rate). Чтобы приёмник понял, где начинается и заканчивается байт, данные обрамляются специальными битами: стартовым в начале и стоповым в конце.

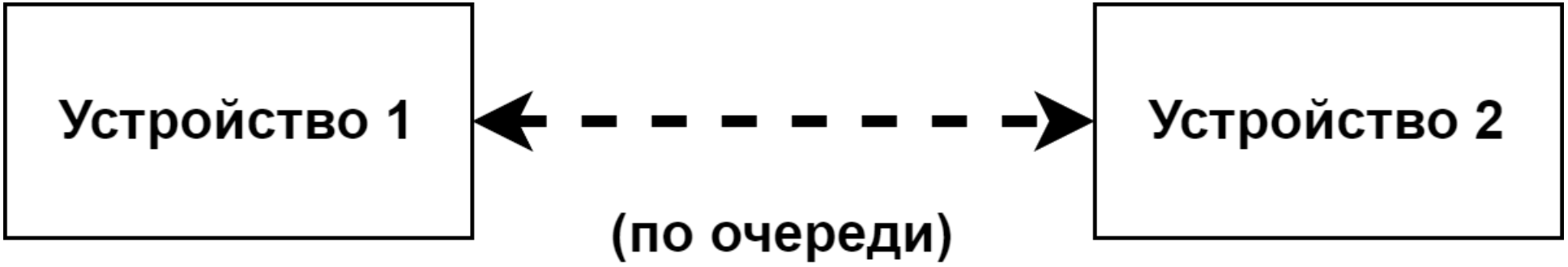


Симплекс

- **Симплекс** (англ. simplex) — передача данных только в одном направлении. Одно устройство всегда передаёт, другое всегда принимает. Пример — пульт дистанционного управления: он отправляет команды телевизору, но ничего не получает обратно.

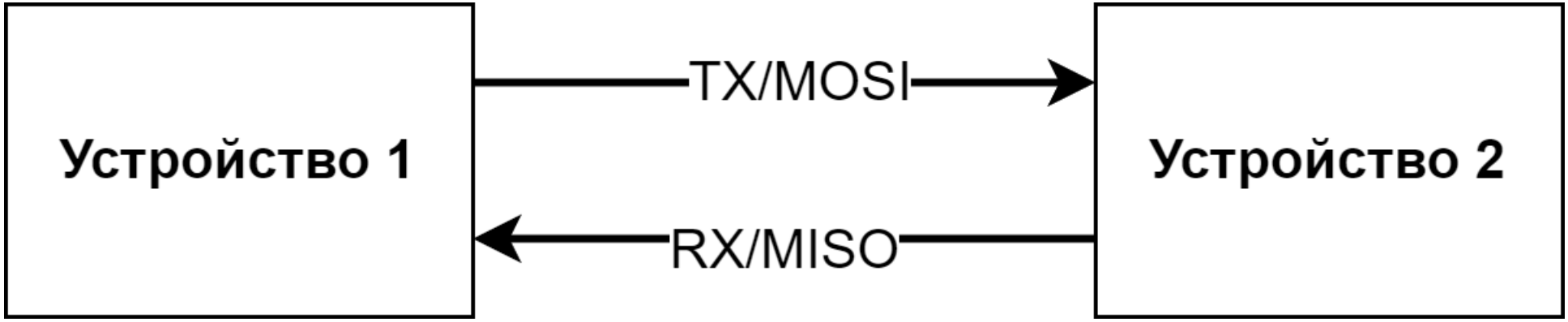
Полудуплекс

- **Полудуплекс** (англ. half-duplex) — передача данных в обоих направлениях, но не одновременно. Устройства «говорят по очереди», как в рации: пока один передаёт, другой слушает. Пример — I2C: линия данных SDA одна, и в каждый момент времени её использует либо ведущий, либо ведомый



Полный дуплекс

- **Полный дуплекс** (англ. full-duplex) — передача данных в обоих направлениях одновременно. Каждое устройство может передавать и принимать в один и тот же момент времени. Для этого нужны отдельные линии для каждого направления. Примеры — SPI (отдельные линии MOSI и MISO) и UART (отдельные линии TX и RX).

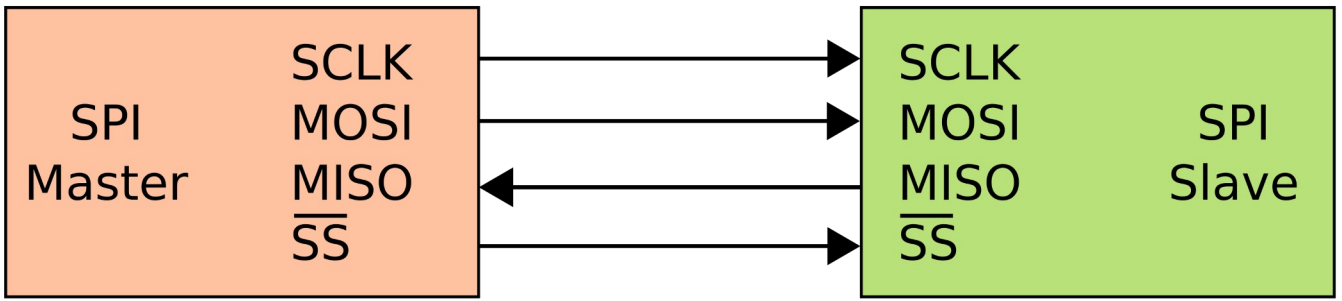


Интерфейсы для внутриплатной периферии

SPI

SPI работает по принципу «ведущий-ведомый» (master-slave). Ведущее устройство (микроконтроллер) генерирует тактовый сигнал и управляет обменом. Для этого используются четыре линии:

- MOSI (Master Out Slave In) — данные от ведущего к ведомому
- MISO (Master In Slave Out) — данные от ведомого к ведущему
- SCK (Serial Clock) — тактовый сигнал, генерируемый ведущим
- CS (Chip Select, также называют SS — Slave Select) — выбор ведомого устройства

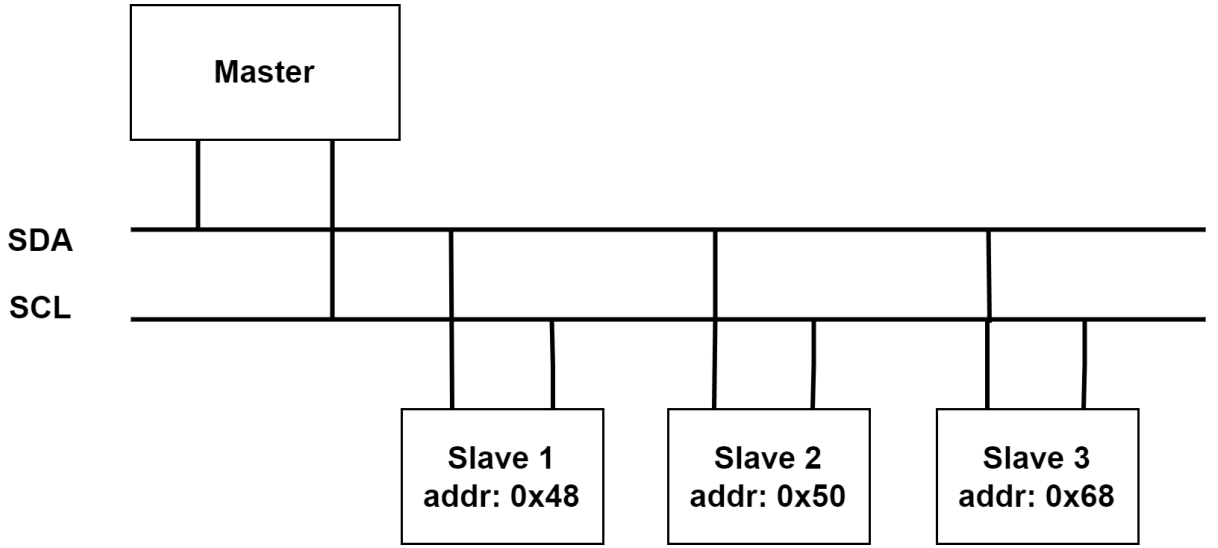


I2C

I2C — Главное преимущество I2C — для подключения множества устройств достаточно всего двух проводов.

I2C использует две линии:

- - SDA (Serial Data) — линия данных (полу-дуплекс)
- - SCL (Serial Clock) — тактовый сигнал

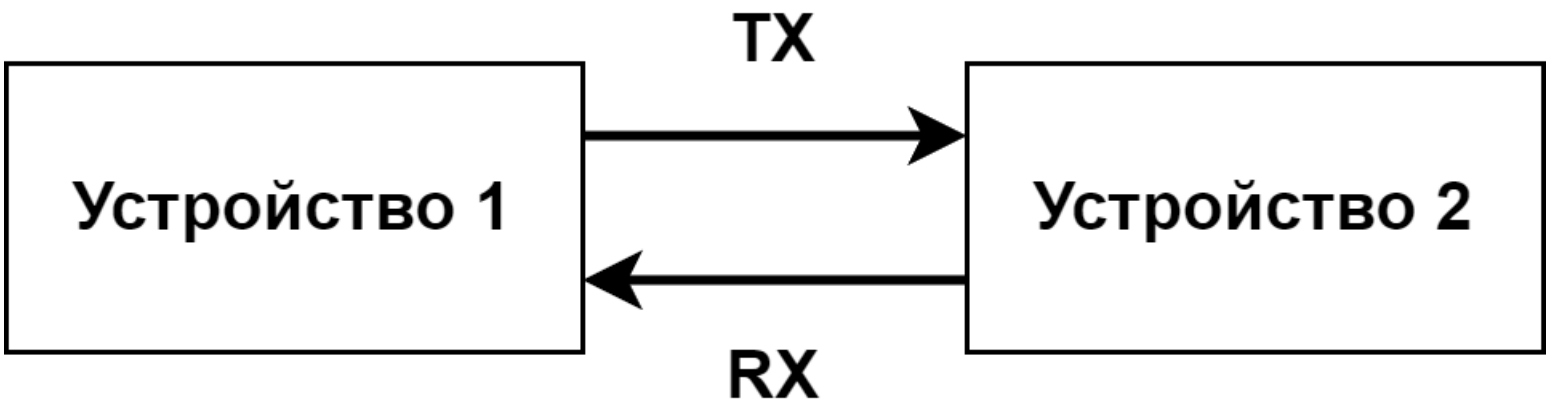


UART

UART — асинхронный интерфейс. В отличие от SPI и I2C, здесь нет тактового сигнала — устройства должны заранее договориться о скорости передачи.

UART использует две линии:

- - TX (Transmit) — передача данных
- - RX (Receive) — приём данных



Сравнение интерфейсов

Хар-ка	SPI	I2C	UART
Тип синхронности	Синхронный	Синхронный	Асинхронный
Кол-во проводов	4 + CS на каждое устройство	2	2
Макс скорость	До 62,5 МГц (RP2040)	До 400 кГц	До 115200 бод
Топология	Один ведущий, много ведомых	Один ведущий много ведомых	Точка-точка
Адресация	Отдельная линия CS	7- битный адрес	нет
Дуплекс	Полный	Полудуплекс	Полный
Применение	Дисплеи, флеш-память, быстрые АЦП	Датчики, EEPROM, RTC	Отладка, GPS, связь между платами

Интерфейсы для внеплатной периферии

Разъем

- **Разъёмное соединение** (англ. connector) — механическое соединение электрических контактов, которое позволяет надёжно удерживать соединение и при этом имеет механизм для его разъединения без инструментов.
- Примеры разъёмов, которые вы можете встретить во встраиваемых системах:
- Штыревые разъёмы (pin headers) — ряды штырьков, как на Raspberry Pi Pico
- Клеммники (terminal blocks) — винтовые зажимы для проводов, удобны для силовых цепей
- JST, Molex — компактные разъёмы для подключения батарей, вентиляторов, датчиков

Можно ли использовать I2C и SPI вне платы?

Основные проблемы при увеличении длины линии:

- Ёмкость кабеля — длинный кабель накапливает заряд и «сглаживает» фронты сигналов
- Помехи — длинный кабель работает как антенна и собирает наводки
- Задержка сигнала — на больших расстояниях сигнал приходит с задержкой, что нарушает синхронизацию
- На практике I2C надёжно работает на расстояниях до 1 метра, SPI — до 10–30 см при высоких скоростях. Для бóльших расстояний лучше выбрать интерфейс, изначально рассчитанный на длинные линии.

Дифференциальная передача

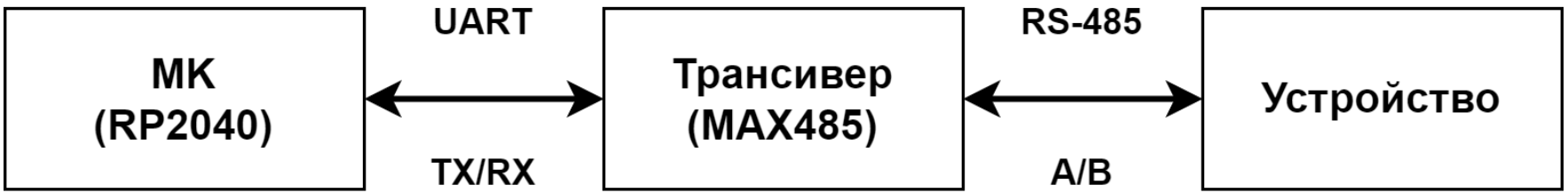
- Дифференциальная передача (англ. differential signaling) — способ передачи, при котором сигнал передаётся по двум проводам как разность напряжений между ними. Помехи воздействуют на оба провода одинаково и взаимно компенсируются при вычитании. Это обеспечивает высокую помехоустойчивость.

Проводные интерфейсы для длинных линий

Характеристика	RS-485	CAN
Максимальная длина	1200 м	40 м (1 Мбит/с) – 1 км (50 кбит/с)
Кол-во устройств на шине	До 32	до 110
Протокол	Физический уровень	Физический + канальный уровень
Применение	Промышленная автоматизация, умный дом	Автомобили, промышленное оборудование

Способ подключения

- Трансивер (англ. transceiver — от transmitter + receiver) — микросхема, объединяющая передатчик и приёмник для преобразования сигналов между разными физическими интерфейсами.



Драйверы

Драйверы внешней периферии

- Программный драйвер (англ. device driver library) — программный модуль, который инкапсулирует детали взаимодействия с устройством и предоставляет набор функций для работы с ним.
- В драйвере внешней периферии, подобно драйверу внутренней, должна содержаться информация о всех регистрах, параметрах и логике работы устройства. Всё это скрывается от пользователя драйвера за конкретными функциями инициализации, чтения, записи и управления устройством.

Логика и платформа

Логика устройства (платформонезависимая):

- Протокол: какие регистры читать и писать, в какой последовательности
- Вычисления: как преобразовать сырые данные в физические величины
- Состояние: инициализация, режимы работы, обработка ошибок

Аппаратный интерфейс (платформозависимая):

- Функции чтения и записи через SPI или I2C
- Управление пинами (CS, reset)
- Задержки

BME280

Table 18: Memory map

Register Name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Reset state
hum_lsb	0xFE	hum_lsb<7:0>								0x00
hum_msb	0xFD	hum_msb<7:0>								0x80
temp_xlsb	0xFC	temp_xlsb<7:4>				0	0	0	0	0x00
temp_lsb	0xFB	temp_lsb<7:0>								0x00
temp_msb	0xFA	temp_msb<7:0>								0x80
press_xlsb	0xF9	press_xlsb<7:4>				0	0	0	0	0x00
press_lsb	0xF8	press_lsb<7:0>								0x00
press_msb	0xF7	press_msb<7:0>								0x80
config	0xF5	t_sb[2:0]			filter[2:0]				spi3w_en[0]	0x00
ctrl_meas	0xF4	osrs_t[2:0]			osrs_p[2:0]			mode[1:0]		0x00
status	0xF3					measuring[0]			im_update[0]	0x00
ctrl_hum	0xF2						osrs_h[2:0]			0x00
calib26..calib41	0xE1…0xF0	calibration data								individual
reset	0xE0	reset[7:0]								0x00
id	0xD0	chip_id[7:0]								0x60
calib00..calib25	0x88…0xA1	calibration data								individual

Registers:	Reserved registers	Calibration data	Control registers	Data registers	Status registers	Chip ID	Reset
Type:	do not change	read only	read / write	read only	read only	read only	write only

BME280

6.2.2 I²C read

To be able to read registers, first the register address must be sent in write mode (slave address 111011X0). Then either a stop or a repeated start condition must be generated. After this the slave is addressed in read mode (RW = '1') at address 111011X1, after which the slave sends out data from auto-incremented register addresses until a NOACKM and stop condition occurs. This is depicted in Figure 10, where register 0xF6 and 0xF7 are read.

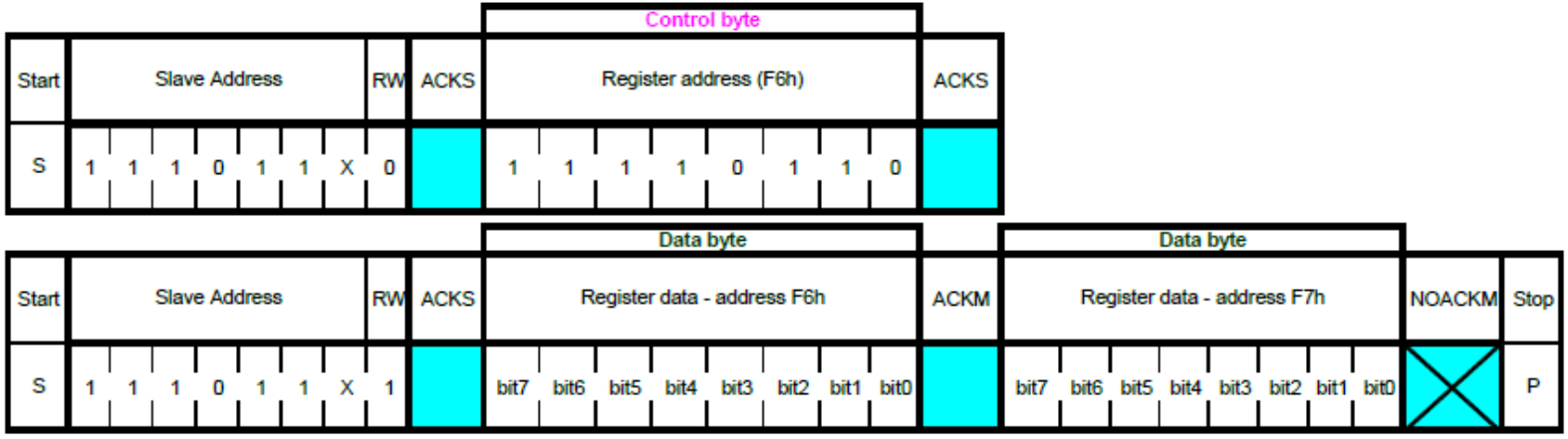


Figure 10: I²C multiple byte read

BME280

6.2.1 I²C write

Writing is done by sending the slave address in write mode (RW = '0'), resulting in slave address 111011X0 ('X' is determined by state of SDO pin. Then the master sends pairs of register addresse and register data. The transaction is ended by a stop condition. This is depicted in Figure 9.

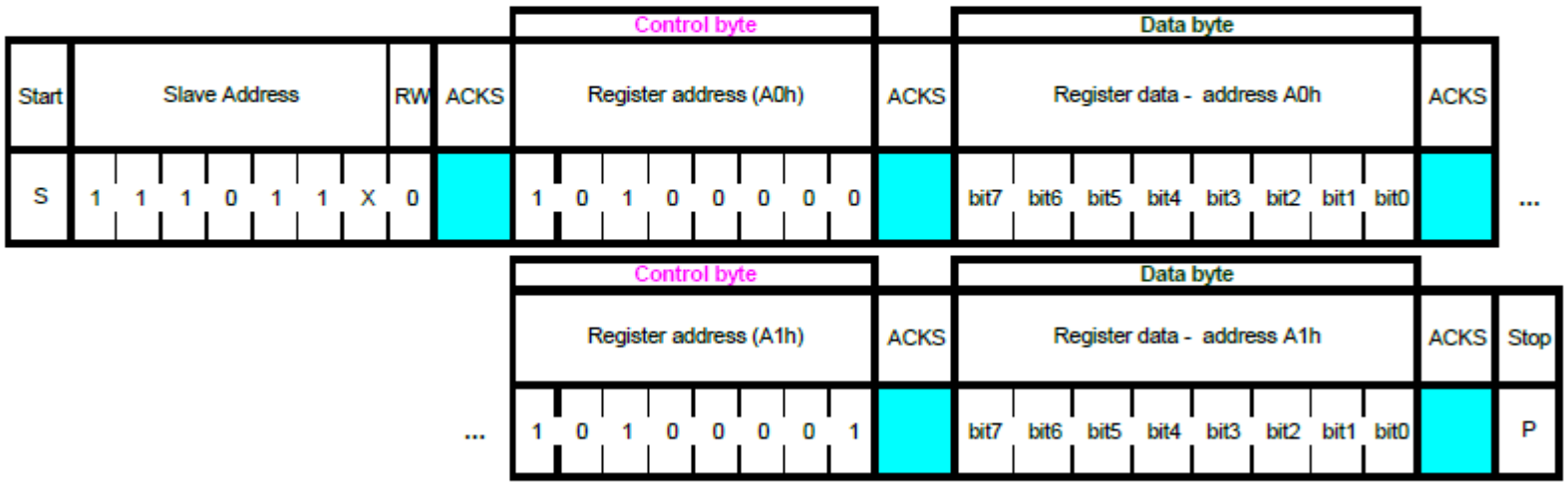


Figure 9: I²C multiple byte write (not auto-incremented)