TERMEX

ТЕРМОМЕТРЫ ЛАБОРАТОРНЫЕ ЭЛЕКТРОННЫЕ LTA

Протокол связи с компьютером

СОДЕРЖАНИЕ

1 Подключение и настройка 3
1.1 USB 3
1.2 Bluetooth 3
2 Общая информация 3
2.1 Принятые соглашения 3
2.2 Формат команды 4
2.3 Формат ответа 4
3 Поддерживаемые запросы 5
3.1 HELP — список доступных команд 5
3.2 T — значение температуры 5
3.3 R — значение сопротивления 5
3.4 TR — значение температуры и сопротивления 6
3.5 M — функция от температуры 6
3.6 GET_TCOEF — чтение значений коэффициентов для расчета температуры 7
3.7 SET_TCOEF — запись значений коэффициентов для расчета температуры 7
3.8 FILTER — управление фильтрацией 8
3.9 GET_FPARAM — чтение параметров фильтра 8
3.10 SET_FPARAM — запись параметров фильтра 9
3.11 TIME — управление временными периодами 9
3.12 TUNIT — размерность градуса температуры 10
3.13 DECIMAL — количество знаков после десятичной точки 10
3.14 LOG — управление логированием 11
3.15 SCRIPT — управление скриптом 12
3.16 UBAT — напряжение батареи 13
3.17 HAS2 — наличие второго канала 13
3.18 VERSION — версия встроенного ПО 14
3.19 SERIAL — заводской номер термометра 14

Настоящее описание распространяется на «Термометры лабораторные электронные LTA» и содержит сведения, необходимые для разработки прикладного программного обеспечения (ПО), предназначенного для управления работой термометра в составе программно-аппаратных комплексов.

Изготовитель оставляет за собой право вносить в протокол изменения, не затрагивающие описанные ниже функции.

1 ПОДКЛЮЧЕНИЕ И НАСТРОЙКА

1.1 USB

1.1.1 Термометр может быть подключен к USB порту компьютера с помощью стандартного кабеля для периферийных устройств с разъемами типа A и microB.

1.1.2 Так как термометр является стандартным HID-совместимым устройством, то настройка драйверов операционной системы выполняется автоматически при первом подключении термометра к USB порту компьютера.

1.1.3 Для облегчения написания прикладного ПО следует использовать библиотеку `lta.dll`, в которой находятся необходимые функции для обмена данными с термометром.

1.2 Bluetooth

1.2.1 Если термометр опционально оснащен Bluetooth интерфейсом, то он может быть подключен к любому устройству поддерживающему стандарт Bluetooth 4.0 Low Energy (BLE).

2 ОБЩАЯ ИНФОРМАЦИЯ

Обмен данными с термометром выполняется по инициативе компьютера (хоста) путем передачи соответствующей команды. Команда представляет собой строку ASCII символов заканчивающуюся символом новой строки ('\n', код 10). Ответ на команду термометр отправляет строкой, заканчивающейся парой символов '\nEOT' (код символа EOT = 0x04). Внутри строки ответа также могут встречаться символы новой строки.

2.1 Принятые соглашения

Далее по тексту при написании команд, завершающий символ команды ('\n') и ответа ('\nEOT') будут опускаться.

При описании синтаксиса команд в угловых скобках <> указываются обязательные параметры; в квадратных скобках [] необязательные параметры; в фигурных скобках {} допустимый набор параметров, разделенных символом '|', из которых выбрать нужно один.

2.2 Формат команды

2.2.1 Команда может быть записана как заглавными, так и строчными буквами.

2.2.2 Строка команды состоит из токенов, разделенных пробелами.

В общем случае команда имеет вид:

`'cmd prm1 prm2 prmN'`

где `cmd` — имя команды; `prmX` — параметры команды.

2.3 Формат ответа

Формат ответа зависит от конкретной команды. Обычно это число или набор чисел.

В случае возникновения ошибок ответ будет иметь вид:

`'[Exx]: сообщение об ошибке'`

где `xx` — код ошибки.

Список возможных ошибок.

[E01]: неизвестная команда. Введите 'help' для получения списка доступных команд.

[E02]: индекс вне допустимого диапазона.

[E03]: недопустимый номер измерительного канала.

[E04]: недопустимый параметр команды.

[E05]: недопустимый температурный коэффициент.

[E06]: недопустимые параметры фильтра.

[E07]: недопустимое значение.

[E08]: ошибка во время программирования flash памяти.

3 ПОДДЕРЖИВАЕМЫЕ ЗАПРОСЫ

3.1 HELP — список доступных команд

`help`

Выводит список доступных команд.

3.2 T — значение температуры

`t [n]`

Выводит значение температуры в обоих или указанном канале.

`n` — номер канала, число в диапазоне `[1, 2]`. Если не указано, то значение температуры выводится для двух каналов.

Если датчик не подключен или поврежден, значение будет `Inf`. Если неисправен измерительный преобразователь, то значение будет `NaN`.

Примеры.

```
Запрос: 't 1'
Ответ:  '37.235'
Запрос: 't'
Ответ:  '37.235 88.658'
```

3.3 R — значение сопротивления

`r [n]`

Выводит значение сопротивления в обоих или указанном канале.

`n` — номер канала, число в диапазоне `[1, 2]`. Если не указано, то значение сопротивления выводится для двух каналов.

Если датчик не подключен или поврежден, значение будет `Inf`. Если неисправен АЦП, то значение будет `NaN`.

Примеры.

```
Запрос: 'r 1'
Ответ:  '115.2354'
Запрос: 'r'
Ответ:  '115.2354 130.6758'
```

3.4 TR — значение температуры и сопротивления

`tr [n]`

Выводит значение температуры и сопротивления для обоих или указанного канала.

`n` — номер канала, число в диапазоне `[1, 2]`. Если не указано, то значения выводятся для двух каналов.

Если датчик не подключен или поврежден, значения будут `Inf`. Если неисправен АЦП, то значения будут `NaN`.

Примеры.

```
Запрос: 'tr 1'
Ответ:  '37.235 115.2354'
Запрос: 'tr'
Ответ:  '37.235 115.2354 88.658 130.6758'
```

3.5 M — функция от температуры

`m { dt | min | avg | max } [n]`

Выводит значение указанной функции от температуры для обоих или указанного канала.

`dt` — разность температур между первым и вторым каналом; параметр `n` игнорируется;

`min` — минимальное значение температуры с момента включения термометра;

`avg` — среднее значение температуры с момента включения;

`max` — максимальное значение температуры с момента включения;

`n` — номер канала, число в диапазоне `[1, 2]`. Если не указано, то значение выводится для двух каналов.

Примеры.

```
Запрос: 'm dt'
Ответ:  '0.014'
Запрос: 'm avg 1'
Ответ:  '37.235'
Запрос: 'm max'
Ответ:  '39.173 91.874'
```

3.6 GET_TCOEF — чтение значений коэффициентов для расчета температуры

`get_tcoef <n>`

Выводит значения коэффициентов функции Каллендара-Ван Дюзена, используемых для расчета температуры в указанном канале, в виде: `'R0 A B C'`

`n` — номер канала, число в диапазоне `[1, 2]`.

Пример.

```
Запрос: 'get_tcoef 1'
Ответ:  '100.0004 3.9083E-3 -5.775E-7 -4.183E-12'
```

3.7 SET_TCOEF — запись значений коэффициентов для расчета температуры

`set_tcoef <n> <R0> <A> <B> <C>`

Устанавливает новые коэффициенты функции Каллендара-Ван Дюзена для расчета температуры в указанном канале. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`n` — номер канала, число в диапазоне `[1, 2]`.

`R0, A, B, C` — значения соответствующих коэффициентов.

Примеры.

```
Запрос: 'set_tcoef 1 100.0004 3.9083E-3 -5.775E-7 -4.183E-12'
Ответ:  'OK'
Запрос: 'set_tcoef 1 100.0004 3.E9083E-3 -5.775E-7 -4.183E-12'
Ответ:  ' [E05]: Invalid temperature coefficient'
```

3.8 FILTER — управление фильтрацией

`filter [val]`

Если параметр `val` не указан, то выводит признак (0 или 1) включена ли фильтрация результатов измерения.

Если параметр `val` указан, то выключает (`val=0`) или включает (`val=1`) фильтрацию. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`val = [0, 1]`.

Значение по умолчанию 0, фильтрация выключена.

Примеры.

`Запрос: 'filter'`

`Ответ:  '0'`

`Запрос: 'filter 1'`

`Ответ:  'OK'`

3.9 GET_FPARAM — чтение параметров фильтра

`get_fparam <n>`

Выводит значения параметров фильтра для указанного канала в виде: 'size level', где

`size = [1, 100]` — глубина фильтра (значение по умолчанию 10);

`level = [0, 100] °C` — порог фильтра (значение по умолчанию 0.2°C);

`n` — номер канала, число в диапазоне `[1, 2]`.

Пример.

`Запрос: 'get_fparam 1'`

`Ответ:  '10 0.2'`

3.10 SET_FPARAM — запись параметров фильтра

`set_fparam <n> <size> <level>`

Устанавливает новые параметры фильтра в указанном канале. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`n` — номер канала, число в диапазоне `[1, 2]`.

`size = [1, 100]` — глубина фильтра (значение по умолчанию 10);

`level = [0, 100] °C` — порог фильтра (значение по умолчанию 0.2°C);

Пример.

`Запрос: 'set_fparam 1 50 0.2'`

`Ответ: 'OK'`

3.11 TIME — управление временными периодами

`time { sample | shutdown | log } [val]`

Если параметр `val` не указан, то возвращает значение соответствующего периода времени.

Если параметр `val` указан, то устанавливает новое значение соответствующего периода времени. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`sample` — период измерения. Диапазон допустимых значений `[1, 3600] c`. Значение по умолчанию 1.

`shutdown` — период автовыключения термометра при бездействии. Диапазон допустимых значений `[0, 64800] c`. Значение по умолчанию 0, что соответствует отсутствию автовыключения. Значение 64800 соответствует 45 дням: 64800 = 45·24·60.

`log` — период логирования. Диапазон допустимых значений `[1, 3600]`. Значение по умолчанию 10. Значение 1 соответствует сохранению результата при каждом измерении. Значение 2 соответствует сохранению результата каждого второго измерения и т.д.

Примеры.

`Запрос: 'time sample'`

`Ответ: '1'`

`Запрос: 'time shutdown 60'`

`Ответ: 'OK'`

3.12 TUNIT — размерность градуса температуры

`tunit [val]`

Если параметр `val` не указан, то возвращает текущую размерность градуса — символ 'C' для градуса цельсия, символ 'F' для градуса фаренгейта.

Если параметр `val` указан, то устанавливает новую размерность градуса. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`val = C или F`.

Значение по умолчанию 'C'.

Примеры.

```
Запрос: 'tunit'
Ответ:  'C'
Запрос: 'tunit F'
Ответ:  'OK'
```

3.13 DECIMAL — количество знаков после десятичной точки

`decimal [val]`

Если параметр `val` не указан, то возвращает текущее количество знаков после десятичной точки, используемых при отображении температуры.

Если параметр `val` указан, то устанавливает новое количество знаков после десятичной точки. При отсутствии ошибок выводит `'OK'`, в противном случае соответствующее сообщение об ошибке.

`val` — число знаков после точки, число в диапазоне `[1, 3]`.

Значение по умолчанию 2.

Примеры.

```
Запрос: 'decimal'
Ответ:  '2'
Запрос: 'decimal 3'
Ответ:  'OK'
```

3.14 LOG — управление логированием

`log { count | clr | at <idx> | range <idx0> <idx1> | enable [val] }`

Осуществляет управление логированием.

`count` — выводит количество записей в логе на данный момент.

`clr` — стирает все записи в логе.

`at <idx>` — выводит значение из лога по указанному индексу. **`idx`** должен находится в диапазоне **`[1, size]`**, где **`size`** — количество записей в логе (результат выполнения запроса **`'log count'`**).

`range <idx0> <idx1>` — выводит значения из лога в указанном диапазоне индексов. **`idx0, idx1`** должны находиться в диапазоне **`[1, size]`**, где **`size`** — количество записей в логе (результат выполнения запроса **`'log count'`**).

`enable [val]` — если **`val`** не указан, то выводит признак (0 или 1) включено ли логирование. Если **`val`** указан, то выключает (**`val=0`**) или включает (**`val=1`**) логирование.

Если у термометра два канала, то значение из лога выводится в виде двух чисел через пробел.

Примеры.

`Запрос: 'log enable 1'`

`Ответ:  'OK'`

`Запрос: 'log count'`

`Ответ:  '17'`

`Запрос: 'log at 3'`

`Ответ:  '24.275'`

`Запрос: 'log range 2 4'`

`Ответ:  '24.123\n 24.275\n 24.567'`

3.15 SCRIPT — управление скриптом

`script { info | start | stop | autostart [val] | pass <bla-bla> }`

Осуществляет управление скриптом.

Скрипт — это пользовательский код, загружаемый в термометр и выполняемый встроенной виртуальной машиной.

`info` — выводит информацию о состоянии скрипта в виде 3-х чисел: **`size started error`**. Где **`size`** — размер кода скрипта в байтах. Если скрипт не обнаружен, то равен нулю. **`started`** — запущен скрипт (значение равно 1) или нет (значение равно 0) на исполнение. **`error`** — код ошибки, возникшей при выполнении скрипта.

`start` — запускает скрипт на исполнение, если он есть и еще не запущен. Иначе ничего не делает.

`stop` — останавливает выполнение скрипта, если он есть и уже выполняется. Иначе ничего не делает.

`autostart [val]` — если **`val`** не указан, то выводит признак (0 или 1) включен ли автостарт скрипта при включении термометра. Если **`val`** указан, то выключает (**`val=0`**) или включает (**`val=1`**) автостарт.

`pass <bla-bla>` — завершает строку **`<bla-bla>`** символом **`'\n'`** и передает ее во входной буфер виртуальной машины. Если места во входном буфере не хватает, то лишние данные отбрасываются. Пользовательский скрипт может по своему желанию читать данные из буфера и обрабатывать их в соответствии с некоторым алгоритмом.

3.16 UBAT — напряжение батареи

```
ubat
```

Возвращает значение напряжения батареи, В.

Пример.

```
Запрос: 'ubat'
Ответ:  '2.542'
```

3.17 HAS2 — наличие второго канала

```
has2
```

Возвращает признак наличия второго измерительного канала.

Значению 0 соответствует отсутствие, значению 1 — наличие второго канала в термометре.

Пример.

```
Запрос: 'has2'
Ответ:  '1'
```

3.18 VERSION — версия встроенного ПО

`version`

Выводит номер версии встроенного ПО.

Пример.

```
Запрос: 'version'
Ответ:  '1.0.0-3'
```

3.19 SERIAL — заводской номер термометра

`serial`

Выводит заводской номер термометра.

Пример.

```
Запрос: 'serial'
Ответ:  '354232'
```