HockeyTek Referee

протокол синхронизации с внешними устройствами

Версия 1.0

Hockey Tek Referee

Коммуникация.

Данная реализация поддерживает TCP/IP подключение. Подключаемое устройство выступает в роли **клиента,** а **HockeyTek Referee в роли сервера**. Настроить порт и хост можно в настройках приложения. Максимальное количество подключенных клиентов **3**.

✦ Структура команды.

- 1 байт – команда
- 2 байта – длинна данных (не включает номер команды и CRC)
- Массив данных команды (длина зависит от команды)
- 2 байта – CRC (только данные не включая команду и длину) стандарт CRC16 ISO 3309

Команда	Длина данных		Данные	CRC*	
0 byte	1 byte	2 byte	(2 + data 0) byte + … + (2 + data size) byte	2 + (data size) + 1 byte	2 + (data size) + 2 byte
byte	short		bytes	short	

✦ Описание команд.

Счет - 0x09

Генерируется при изменении счета и при инициализации

Данные:

- 2 байта - счет левой команды, тип short
- 2 байта - счет правой команды, тип short

Пример: счет 1:2			Данные					
Команда	Длина данных		Счет левой команды		Счет правой команды		CRC*	
0 byte	1 byte	2 byte	3 byte	4 byte	5 byte	6 byte	7 byte	8 byte
0x03	0x00	0x04	0x00	0x01	0x00	0x02		

Период - 0x08

Генерируется при изменении периода и при инициализации

Данные:

- 1 байт - номер периода:
  - 1 - первый период
  - 2 - второй период
  - 3 - третий период
  - 4 - овертайм
  - 5 - буллиты
- 4 байта - полное время периода, миллисекунды

Игровой таймер - 0x0A

Генерируется при изменении времени периода. Если таймер запущен генерируется с интервалом **0.1 сек,** если таймер остановлен, то интервал **1 сек.**

В случае работы режиме «Расписания», если период закончен или еще не начинался, вместо данной команды будет генерироваться команда расписания (0x0B) с интервалом 1 сек.

Данные:

- 8 байт – текущее время с точностью до миллисекунды, в формате UNIX
- 4 байта – обратный отсчет времени периода, миллисекунды
- 4 байта – полное время периода, миллисекунды
- 1 байт – статус таймера:
  - 1 – таймер запущен
- 0 - таймер остановлен

Таймер расписания - 0x0B

Генерируется при изменении времени таймера, смены тайм-слота или статуса. Генерируется с интервалом **1 сек**.

Команда не генерируется, когда работает игровой таймер (0x0A), а также когда расписание остановлено.

Данные:

- 8 байт – текущее время с точностью до миллисекунды, в формате UNIX
- 4 байта – обратный отсчет времени слота, миллисекунды
- 4 байта – полное время слота, миллисекунды
- 1 байт – статус таймера:
  - 1 – таймер запущен
  - 0 - таймер остановлен
- 1 байт – номер (тип) слота:
  - 0 – слот не определен
  - 1 – слот пред матчевых событий
  - 2 – слот «первый период»
  - 3 – слот «после первого периода»
  - 4 – слот «второй период»
  - 5 – слот «после второго периода»
  - 6 – слот «третий период»
  - 7 – слот «после третьего периода»
  - 8 – слот «овертайм»
  - 9 – слот «после овертайма»
  - 10 – слот «буллитная серия»
  - 11 – слот «послематчевых событий»

➡ Расчет CRC16 ISO 3309

```
  /*!
   Returns the CRC-16 checksum of data.

   The checksum is independent of the byte order (endianness) and will
   be calculated accorded to the algorithm published in standard.
   By default the algorithm published in ISO 3309 is used.

   This function is a 16-bit cache conserving (16 entry table)
   implementation of the CRC-16-CCITT algorithm.
*/

static const unsigned short crc_tbl[16] = {
   0x0000, 0x1081, 0x2102, 0x3183,
   0x4204, 0x5285, 0x6306, 0x7387,
   0x8408, 0x9489, 0xa50a, 0xb58b,
   0xc60c, 0xd68d, 0xe70e, 0xf78f
};

unsigned short crcCalculator(const unsigned char *data_p, unsigned char
length)
{
   uchar c;
   unsigned short crc = 0xffff;

   while (length--)
   {
      c = *data_p++;
      crc = ((crc >> 4) & 0x0fff) ^ crc_tbl[((crc ^ c) & 15)];
      c >>= 4;
      crc = ((crc >> 4) & 0x0fff) ^ crc_tbl[((crc ^ c) & 15)];
   }

   crc = ~crc;

   return crc & 0xffff;
}
```