Project	Forest Recorder	**Page**	1/2
Version	FR-001	**Updated**	14/06/2018
Author	Juan Barbosa	**Reviewed**	
		Reviewer	

Tausand

1 Settings

Internal implementation of the arduino code requires the following libraries: `SPI`, `Adafruit_VS1053`, `SdFat`, `RTClibExtended` which are arduino only. The following AVR libraries are used as well: `avr/sleep`, `avr/power`, `avr/wdt`.

1.1 RTC DS3231

Connection with the real time clock is made using the I^2C protocol. Wires go as follows:

- `A4 (PC4)` → `SDA`
- `A5 (PC5)` → `SDL`
- `D2 (PD2)` → `SWQ`

The last of the connections is used for the alarm wake up, since the arduino is on `POWER_DOWN_MODE` signal needs to be set on one of the external interrupt pins, in this case `INT0`.

1.2 VS1053 breakout

The board uses SPI communication, connections are port to port. There are 3 chip select cables, one for the board (`CS`), one for the board memory (`XDCS`) and the SD select (`SDCS`).

- `D5 (PD5)` → `DREQ`
- `D8 (PB0)` → `XDCS`
- `D9 (PB1)` → `RST`
- `D10 (PB2)` → `CS`
- `D11 (PB3)` → `MOSI`
- `D12 (PB4)` → `MISO`
- `D13 (PB5)` → `SCLK`

1.3 SD

Connection is made using the breakout SPI ports which were listed before. Chip Select requires a wire from `D4 (PD4)` to `SDCS`.

MicroSD card **must** be `FAT32` formatted.

1.4 UART

Communication with the Arduino is made through UART protocol. Settings are set as follows:

- `baudrate = 9600`
- `bytesize = EIGHTBITS`
- `parity = PARITY_NONE`
- `stopbits = STOPBITS_ONE`

2 Scheduling

Recording activity is set on a file called `schedule.dat` which has the following structure:

```
d1-m1-y1-H1-M1; d2-m2-y2-H2-M2\n
```

Before the semicolon are the starting times, after it, the stop times. Each time descriptor is made as a 2 digit, zero padded number. That is day 4 must be placed as `04`.

- `d`: day
- `m`: month
- `y`: year
- `H`: hour
- `M`: minute

Each line **must** be 32 bytes.

3 Time setting

To set the current time to the RTC, there must be a UART connection active. On a computer as soon as a communication port is open, and connection is establish with the Arduino, this one automatically restarts. There is a 2 second window in which the micro will wait for incoming data from UART, this time starts as soon as the Arduino writes `Connection\r\n`. If no data is received, it will continue to load its recording protocol. If data is received it will start a UART only mode, which only ends with a hardware restart.

There are 3 functions implemented: `setTime`, `getTime` and `reset`.

3.1 setTime (0x00)

- Message start: `0x00`
- Message longitude: 13 bytes
- Message: [0x00, 'd', 'd', 'm', 'm', 'y', 'y', 'H', 'H', 'M', 'M', 'S', 'S'] **(zero padded)**

For example on ASCII datetime 14/06/2018 16:50:00 looks like this:

[0, '1', '4', '0', '6', '1', '8', '1', '6', '5', '0', '0', '0']

Or using hex notation:

[0x00, 0x31, 0x34, 0x30, 0x36, 0x31, 0x38, 0x31, 0x36, 0x35, 0x30, 0x30, 0x30]

- Answer: `getTime`

3.2 getTime (0x01)

- Message start: `0x01`
- Answer: "dd,mm,yy,HH,MM,SS" **(not zero padded)**
  For example datetime 14/06/2018 16:50:00 looks like this:

```
"14,6,18,16,50,0\r\n"
```

3.3 reset (0x02)

- Message start: `0x02`