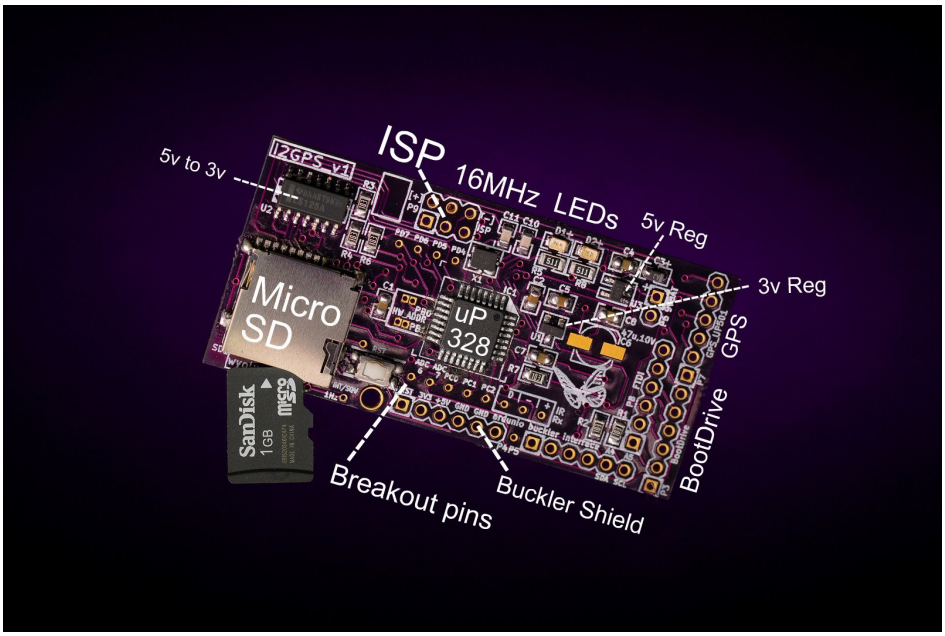
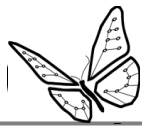


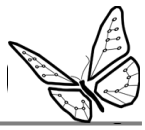
I2GPS v1

An easy to use RTC, GPS and SD memory card interface

brought to you by



CONTENTS



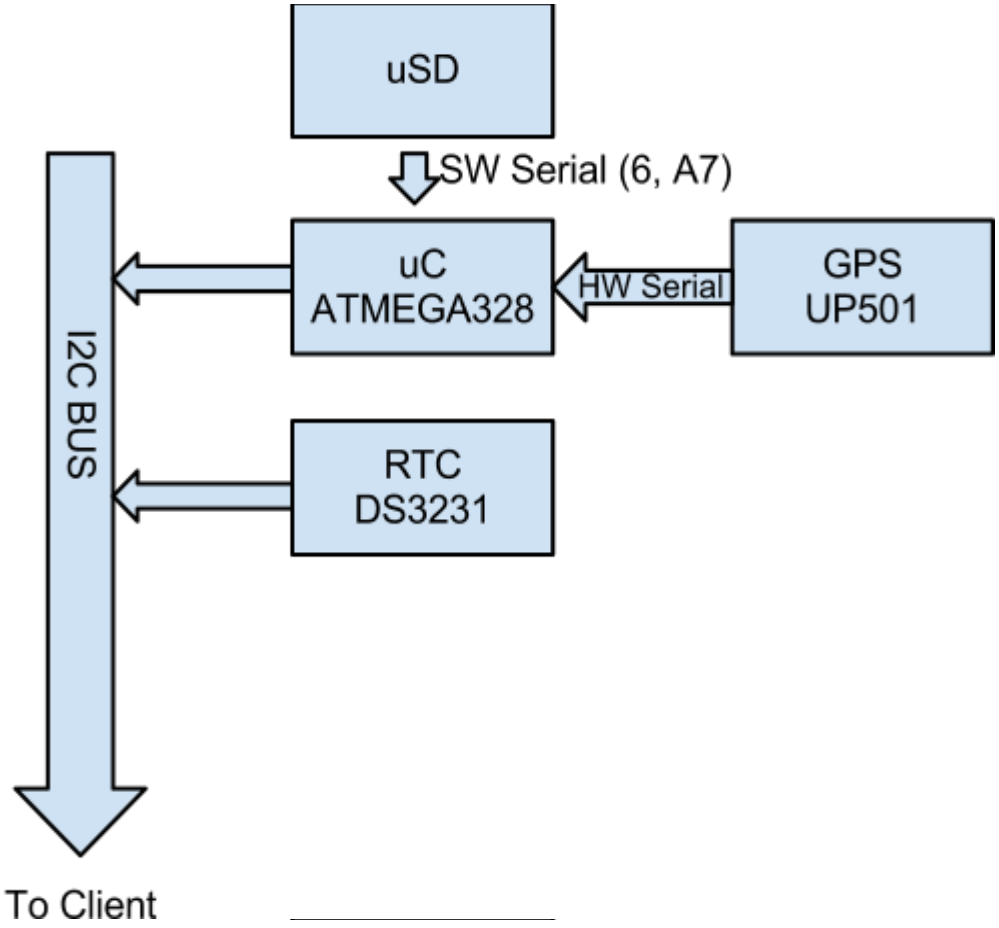
Features

micro SD card reader
Temperature controlled, precision Real Time clock, with battery backup
GPS interface for the Fastrax UP501 module
Arduino compatible
ChronoDot compatible
Two general purpose LEDs
IR interface
I2C interface requires no additional pins when used on an existing I2C bus.
Unused I/O pins are available via convenient breakout pads.

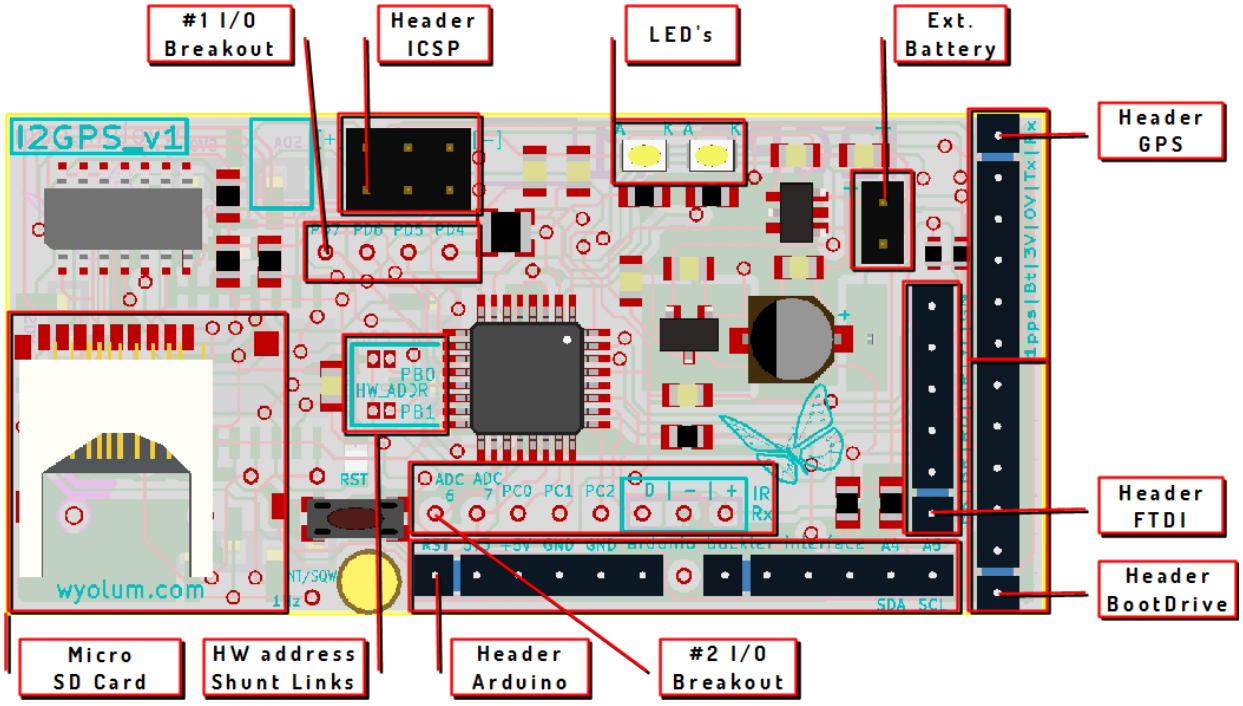
Potential Uses

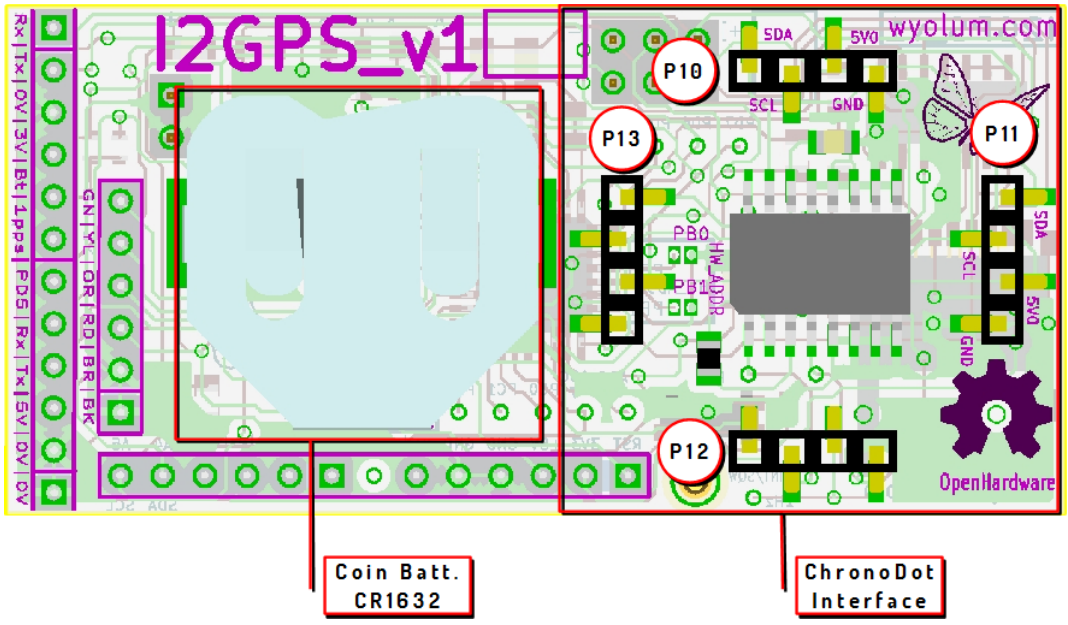
- Stand-alone data logger
- Simple-to-use, persistent storage
- Program loader for separate Arduino compatible
- Store events, animations, and sounds for ClockTHREE
- Automatic Time Zone adjustment. The ClockTHREE or ClockTHREE_Jr never needs to be set or adjusted.

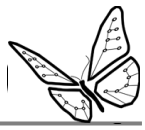
Block Diagram



Physical Interfaces







Physical Interfaces, Description

(RED Markers point to Pin # 1 of each header)

#1 I/O Breakout

1 = PD7
2 = PD6
3 = PD5
4 = PD4

Header ICSP

1 = MISO
2 = 5V0
3 = SCK
4 = MOSI
5 = RESET
6 = GND

LED's

D1 = PD4
D2 = PD7

EXT BATT

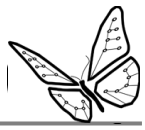
1 = + 9 V max.
2 = GND

HEADER GPS

1 = GPS_Rx , ADC7
2 = GPS_Tx , PD6
3 = GND
4 = 3V3
5 = GPS_BATT
6 = 1pps , PD2

HEADER BOOTDRIVE

1 = GND
2 = GND
3 = 5V0
4 = TXD , PD1
5 = RXD , PD0
6 = Boot , PD5



HEADER FTDI

1 = GND
2 = GND
3 = 5V0
4 = RXD , PD0
5 = TXD , PD1
6 = RESET

HEADER ARDUINO

1 = RESET
2 = 3V3
3 = 5V0
4 = GND
5 = GND
6 = NC
7 = NC
8 = NC
9 = NC
10 = NC
11 = NC
12 = SDA , PC4
13 = SCA , PC5

#2 I/O BREAKOUT

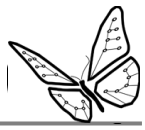
1 = ADC6
2 = ADC7 , GPS_Rx
3 = PC0
4 = PC1
5 = PC2
6 = Ir_Dat , PC3
7 = GND
8 = 5V0

NOTE : 6 , 7 , 8 for Ir Receiver

HW ADDRESS SHUNT LINKS

PB0
PB1

NOTE : To set HW Address, jumper one of these ports to GND



MICRO SD-CARD

SD_CS = PB2 , SS
SD_DATA_IN = PB3 , MOSI
SC_CLK = PB5 , SCK
SD_DATA_OUT = PB4 , MISO

CHRONODOT INTERFACE

P10 , # 1 = SDA
P10 , # 2 = SCL
P10 , # 3 = 5V0
P10 , # 4 = GND

P11 , # 1 = SDA
P11 , # 2 = SCL
P11 , # 3 = 5V0
P11 , # 4 = GND

P12 , # 1 = BAT (NOT CONNECTED)
P12 , # 2 = 32k
P12 , # 3 = SQW
P12 , # 4 = RST

P13 , # 1 = BAT
P13 , # 2 = 32k (NOT CONNECTED)
P13 , # 3 = SQW
P13 , # 4 = RST (NOT CONNECTED)

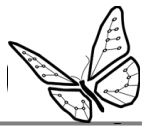
NOTE

Ref. designators are placed near Pin #1.
Use wire jumpers if NOT CONNECTED pins are required.
The four connectors allow two possible mounting positions.
Lateral , using P11 and P13. Longitudinal , using P10 and P12

Software Interface Options

I2GPS (I2C interface to SD and GPS).

<INSERT Memory MAP>



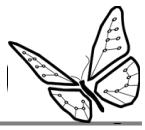
GPS INTERFACE

This interface provide time, position, and motion information when the GPS receiver is tracking without a large processing or memory burden on the host processor.

```
/*
 * Read n_byte from slave starting from offset address addr.
   Store result stored in dest (which must be at least n_byte
   long).
 * Return true if successful.
 *
 * Must be preceded with call to Wire.begin()
 */
bool gps_raw_read(uint8_t addr,
                  uint8_t n_byte,
                  uint8_t *dest){

  bool out = false;
  Wire.beginTransmission(I2GPS_I2C_ADDR);
  Wire.write(addr);
  Wire.endTransmission();
  // request n_byte bytes
  Wire.requestFrom((int)I2GPS_I2C_ADDR, (int)n_byte);
  if(Wire.available()){
    for(uint8_t i = 0; i < n_byte; i++){
      dest[i] = Wire.read();
    }
    out = true;
  }
  return out;
}
/*
 * Write n_byte bytes of data from source to I2GPS over I2C
   starting at offset addr.
 *
 * Must be preceded with call to Wire.begin()
 */
void gps_raw_write(uint8_t addr,
                   uint8_t n_byte,
                   uint8_t *source){
  Wire.beginTransmission(I2GPS_I2C_ADDR);
  Wire.write(addr);
  for( uint8_t i=0; i < n_byte && i < 32; i++){
    Wire.write(source[i]);
  }
}
```

Example GPS over I2C usage.

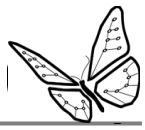


Arduino Client (I2C Master)	I2GPS Server (I2C Slave)	Comment

`int32_t lat;`
 	
`gps_raw_read(`
`  I2GPS_LAT_OFFSET,`
`  4,`
`  (uint8_t*)&lat);`
	Send 4 bytes of data stored at I2GPS_LAT_OFFSET.	Request current latitude. The result is in 0.001 Degrees.

`uint8_t gps_data[32];`
 	
`gps_raw_read(`
`  0,`
`  32,`
`  gps_data)`
	Send 32 bytes of data stored at offset 0.	Data block contains, time, position, motion values. See table XX for details.

`gps_raw_write(`
`  I2GPS_LOGRATE_ADDR,`
`  2,`
`  1);`
	Write time, and position data to SD card.	Format TBD, not implemented.



SD INTERFACE

```
Insert SD Interface Code here
```

Example SD file access over I2C

Arduino Client (I2C Master)	I2GPS Server (I2C Slave)	Comment

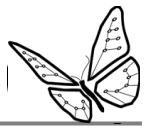
`gps_raw_write(`
`I2GPS_FILENAME_ADDR,`
`  8,`
`  "FILE.DAT");`
`gps_raw_write(`
`  I2GPS_FILESTAT_ADDR,`
`  1,`
`  I2GPS_FILE_WRITE &&`
`  I2GPS_FILE_ENABLE);`
	Receive filename at filename offset
	
	Enable writing to file.
		Filename is stored. When file status is "enabled" file is opened for writing.

`char* dat = "some text";`
`gps_raw_write(`
`  I2GPS_FILE_DATA_ADDR,`
`  strlen(dat),`
`  (uint8_t*)dat);`
	Store "some text" (10 bytes) in open file.	data is immediatly stored to open file. At most 31 bytes can be written. **See XXX for a method to transfer more data.**

`gps_raw_write(`
`  I2GPS_FILESTAT_ADDR,`
`  1,`
`  I2GPS_FILE_CLOSE);`		

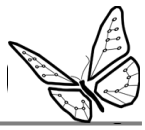
`gps_raw_write(`
`  I2GPS_FILESTAT_ADDR,`
`  1,`
`  I2GPS_FILE_CLOSE);`
	Close open file	

`gps_raw_write(`
`I2GPS_FILENAME_ADDR,`
`  9,`
`  "FILE2.DAT");`
`gps_raw_write(`
`  I2GPS_FILESTAT_ADDR,`
`  1,`
`  I2GPS_FILE_READ &&`
`  I2GPS_FILE_ENABLE);`
	Receive filename at filename offset
	
	Enable reading from file.
		Filename is stored. When file status is "enabled" file is opened for reading.



`char dat[32];`
`uint32_t pos = 123;`
`gps_raw_read(`
`  I2GPS_SEEK_ADDR,`
`  4,`
`  (uint8_t*)&pos);`
 	
`gps_raw_read(`
`  I2GPS_FILE_DATA_ADDR,`
`  32,`
`  (uint8_t*)dat);`
	Seek to position 123
	
	Read and send 32 bytes from open file to client;
		Can only read 32 bytes at a time. See XXX for reading more data.

`gps_raw_write(`
`  I2GPS_FILESTAT_ADDR,`
`  1,`
`  I2GPS_FILE_CLOSE);`
	Close open file.	



LINKS

- website : www.wyolum.com
- e-mail : info@wyolum.com
- forum : http://wyolum.com/forum/forumdisplay.php?fid=12
- SVN Repo : http://code.google.com/p/clockthree/
- ChronoDot : http://macetech.com/store/index.php?main_page=product_info&cPath=5&products_id=8
- Arduino : http://www.arduino.cc/