RTClib / RTClib.cpp
100644 186 lines (155 sloc) 5.277 kb

- raw
- normal view
- history

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

```
1  // Code by JeeLabs http://news.jeelabs.org/code/
2  // Released to the public domain! Enjoy!
3
4  #include <Wire.h>
5  #include <avr/pgmspace.h>
6  #include "RTClib.h"
7  #include <WProgram.h>
8
9  #define DS1307_ADDRESS 0x68
10 #define SECONDS_PER_DAY 86400L
11
12 #define SECONDS_FROM_1970_TO_2000 946684800
13
14 ////////////////////////////////////////////////////////////////////////////////
15 // utility code, some of this could be exposed in the DateTime API if needed
16
17 static uint8_t daysInMonth [] PROGMEM = { 31,28,31,30,31,30,31,31,30,31,30,31 };
18
19 // number of days since 2000/01/01, valid for 2001..2099
20 static uint16_t date2days(uint16_t y, uint8_t m, uint8_t d) {
21 if (y >= 2000)
22 y -= 2000;
23 uint16_t days = d;
24 for (uint8_t i = 1; i < m; ++i)
25 days += pgm_read_byte(daysInMonth + i - 1);
26 if (m > 2 && y % 4 == 0)
27 ++days;
28 return days + 365 * y + (y + 3) / 4 - 1;
```

```
29  }
30
31  static long time2long(uint16_t days, uint8_t h, uint8_t m, uint8_t s) {
32  return ((days * 24L + h) * 60 + m) * 60 + s;
33  }
34
35  ////////////////////////////////////////////////////////////////////////////////
36  // DateTime implementation - ignores time zones and DST changes
37  // NOTE: also ignores leap seconds, see http://en.wikipedia.org/wiki/Leap_second
38
```

1d04f0fd » adafruit
2010-05-14 now with zip file

```
39  DateTime::DateTime (uint32_t t) {
40  t -= SECONDS_FROM_1970_TO_2000; // bring to 2000 timestamp from 1970
41
```

f8a1ab0d » adafruit
2010-03-11 First commit

```
42  ss = t % 60;
43  t /= 60;
44  mm = t % 60;
45  t /= 60;
46  hh = t % 24;
47  uint16_t days = t / 24;
48  uint8_t leap;
49  for (yOff = 0; ; ++yOff) {
50  leap = yOff % 4 == 0;
51  if (days < 365 + leap)
52  break;
53  days -= 365 + leap;
54  }
55  for (m = 1; ; ++m) {
56  uint8_t daysPerMonth = pgm_read_byte(daysInMonth + m - 1);
57  if (leap && m == 2)
58  ++daysPerMonth;
59  if (days < daysPerMonth)
60  break;
61  days -= daysPerMonth;
62  }
63  d = days + 1;
64  }
65
```

```
66  DateTime::DateTime (uint16_t year, uint8_t month, uint8_t day, uint8_t hour, uint8_t min, uint8_t sec) {
67  if (year >= 2000)
68  year -= 2000;
69  yOff = year;
70  m = month;
71  d = day;
72  hh = hour;
73  mm = min;
74  ss = sec;
75  }
76
77  static uint8_t conv2d(const char* p) {
78  uint8_t v = 0;
79  if ('0' <= *p && *p <= '9')
80  v = *p - '0';
81  return 10 * v + *++p - '0';
82  }
83
84  // A convenient constructor for using "the compiler's time":
85  // DateTime now (__DATE__, __TIME__);
86  // NOTE: using PSTR would further reduce the RAM footprint
87  DateTime::DateTime (const char* date, const char* time) {
88  // sample input: date = "Dec 26 2009", time = "12:34:56"
89  yOff = conv2d(date + 9);
90  // Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
91  switch (date[0]) {
92  case 'J': m = date[1] == 'a' ? 1 : m = date[2] == 'n' ? 6 : 7; break;
93  case 'F': m = 2; break;
94  case 'A': m = date[2] == 'r' ? 4 : 8; break;
95  case 'M': m = date[2] == 'r' ? 3 : 5; break;
96  case 'S': m = 9; break;
97  case 'O': m = 10; break;
98  case 'N': m = 11; break;
99  case 'D': m = 12; break;
100 }
101 d = conv2d(date + 4);
102 hh = conv2d(time);
103 mm = conv2d(time + 3);
104 ss = conv2d(time + 6);
```

```
105 }
106
107 uint8_t DateTime::dayOfWeek() const {
108 uint16_t day = secondstime() / SECONDS_PER_DAY;
109 return (day + 6) % 7; // Jan 1, 2000 is a Saturday, i.e. returns 6
110 }
111
112 uint32_t DateTime::unixtime(void) const {
113 uint32_t t;
114 uint16_t days = date2days(yOff, m, d);
115 t = time2long(days, hh, mm, ss);
116 t += SECONDS_FROM_1970_TO_2000; // seconds from 1970 to 2000
117
118 return t;
119 }
120
121 ///////////////////////////////////////////////////////////////////////////////
122 // RTC_DS1307 implementation
123
124 static uint8_t bcd2bin (uint8_t val) { return val - 6 * (val >> 4); }
125 static uint8_t bin2bcd (uint8_t val) { return val + 6 * (val / 10); }
126
127 uint8_t RTC_DS1307::begin(void) {
128 return 1;
129 }
130
131 uint8_t RTC_DS1307::isrunning(void) {
132 Wire.beginTransmission(DS1307_ADDRESS);
133 Wire.send(0);
134 Wire.endTransmission();
135
136 Wire.requestFrom(DS1307_ADDRESS, 1);
```

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit
2010-05-14 now with zip file

f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit

2010-05-14 now with zip file

```
137 uint8_t ss = Wire.receive();
138 return !(ss>>7);
```

f8a1ab0d » adafruit
2010-03-11 First commit

```
139 }
140
141 void RTC_DS1307::adjust(const DateTime& dt) {
142 Wire.beginTransmission(DS1307_ADDRESS);
143 Wire.send(0);
144 Wire.send(bin2bcd(dt.second()));
145 Wire.send(bin2bcd(dt.minute()));
146 Wire.send(bin2bcd(dt.hour()));
147 Wire.send(bin2bcd(0));
148 Wire.send(bin2bcd(dt.day()));
149 Wire.send(bin2bcd(dt.month()));
150 Wire.send(bin2bcd(dt.year() - 2000));
151 Wire.send(0);
152 Wire.endTransmission();
153 }
154
155 DateTime RTC_DS1307::now() {
```

1d04f0fd » adafruit
2010-05-14 now with zip file

```
156 Wire.beginTransmission(DS1307_ADDRESS);
157 Wire.send(0);
158 Wire.endTransmission();
159
160 Wire.requestFrom(DS1307_ADDRESS, 7);
161 uint8_t ss = bcd2bin(Wire.receive() & 0x7F);
162 uint8_t mm = bcd2bin(Wire.receive());
163 uint8_t hh = bcd2bin(Wire.receive());
164 Wire.receive();
165 uint8_t d = bcd2bin(Wire.receive());
166 uint8_t m = bcd2bin(Wire.receive());
167 uint16_t y = bcd2bin(Wire.receive()) + 2000;
168
169 return DateTime (y, m, d, hh, mm, ss);
```

f8a1ab0d » adafruit
2010-03-11 First commit

```
170 }
171
```

```
172 ///////////////////////////////////////////////////////////////////////////
173 // RTC_Millis implementation
174
175 long RTC_Millis::offset = 0;
176
177 void RTC_Millis::adjust(const DateTime& dt) {
178     offset = dt.secondstime() - millis() / 1000;
179 }
180
181 DateTime RTC_Millis::now() {
182     return (uint32_t)(offset + millis() / 1000);
183 }
184
185 ///////////////////////////////////////////////////////////////////////////
```

1d04f0fd » adafruit
2010-05-14 now with zip file
f8a1ab0d » adafruit
2010-03-11 First commit

1d04f0fd » adafruit
2010-05-14 now with zip file
f8a1ab0d » adafruit
2010-03-11 First commit