stamp.peripheral.rtc

Class DS1302

```
java.lang.Object
  |
  +--stamp.peripheral.rtc.DS1302
```

public class **DS1302**
extends Object

This class encapsulates the capabilities of the Dallas DS1302 3-wire real-time clock including the RAM and Super Cap charging.

This class only needs a small main class. For example add the constructor and the following lines of code:

```
DS1302 t = new DS1302(CPU.pin1,CPU.pin2,CPU.pin3);  // your pins may vary
t.updateTimeDate();
System.out.print("Time: ");
System.out.println(t.printTime(true));
System.out.print("Date: ");
System.out.println(t.printDate());
```

Memory overhead with the small example above you should see at least 25K free

Day of week is represented by 1-7 (Sun-Sat). Zero is not supported by the DS1302. The time set method will change a 0 for the day of week to a 1 instead of throwing an error.

The constructor does not reset the time/date if invalid from power failure.

Use of RAM: The RAM is very useful to store rapidly changing information like seconds or voltage that would otherwise wear out EEPROM either external or inside the Javelin chip. The RAM routines are set up so the RAM address wraps around and therefore any address can be used. Forgiving but still only 31 bytes in size.

Backup power is through pin 8 of the DS1302. By using the `charge(true);` function a cap can be used instead of a battery. This feature is valuable for reliability, power failure monitoring or logging in remote locations.

The clock routines whenever called check the timer t1. If t1 has elapsed, a time request to the DS1302 chip is made otherwise the time values from hour, minute, second... are used. This makes processing faster and keeps the clock and data lines free for any other chips on the "bus". The timer is adjusted via: `UPDATE_PERIOD`.

Future revisions will be made by creating a super class that will abstract individual chips such as DS1307, DS1302 or the Pocket Watch B. So the same code could work with different hardware.

```
Revision History:
07/29/02: Class originally submitted to Parallax Inc.
          by customer Tim Constable of Boston, MA
          Original Version 0.81
08/22/02: Class modified, enhanced and tested by Steve Dill of Parallax Inc.
          Version 1.0 of DS1302 class approved.
```

Constructor Summary

DS1302(int data, int clock, int enable) Initialize DS1302 chip for r/w (write protect register) clock is not checked for valid time	

Method Summary

void	**charge**(boolean data) Charge the Super Cap to allow backup power.
void	**halt**(boolean b) Halts the dsa1302 chip
String	**numToString2**(int num) Takes an integer and formats it to two byte string with a leading zero
void	**protect**(boolean b) Protects the ds1302 chip from accidential writes
String	**readDate**() Format the date to US style, useful for date stamp on screen
String	**readDay**(boolean b) Returns the day of the week.
int	**readRam**(int location) Get one RAM register from the DS1302 chip.
int	**readRawTD**(int comand) Read a specific DS1302 chip register contents in the RAW, unformatted form.

int	**readTD**(int index) Read a DS1302 chips' time/date register, for sake of speed, this data is unformatted.
String	**readTime**(boolean ap) Format the time (12hr/24hr).
int[]	**readTimeDate**() Read all time/date variables from DS1302 chip.
void	**updateTimeDate**() Forces the public time variables to be updated.
void	**writeRam**(int location, int data) Save one byte to RAM in the DS1302 chip.
void	**writeTime**(int hr, int min, int sec, int mo, int date, int yr, int dayOfWeek) Set all the time variables in the DS1302 chip.

Methods inherited from class java.lang.Object
equals

Constructor Detail

DS1302

```
public DS1302(int data,
              int clock,
              int enable)
```

Initialize DS1302 chip for r/w (write protect register) clock is not checked for valid time

Throws:

If - clock chip does not exist

Method Detail

numToString2

```
public String numToString2(int num)
```

Takes an integer and formats it to two byte string with a leading zero

Parameters:
num -
Returns:
two character string

---

charge

```
public void charge(boolean data)
```

Charge the Super Cap to allow backup power. Charge at least an hour to fully charge the Super Cap. This feature is very useful for keeping a valid time without external power. When using batteries instead of a Super Cap DO NOT set charge(true); this feature is not ment to charge batteries.

Returns:
Nothing

---

writeRam

```
public void writeRam(int location,
                     int data)
```

Save one byte to RAM in the DS1302 chip. Allows all integers to map to 31 bytes

Parameters:
location - (full int range), Data to be stored
Returns:
Nothing

---

readRam

```
public int readRam(int location)
```

Get one RAM register from the DS1302 chip. Allows all integers to map to 31 bytes. This method will autowrap RAM values above 31 bytes.
Example: RAM location 32 = Ram location 0

Returns:

Ram data

---

readRawTD

```
public int readRawTD(int comand)
```

Read a specific DS1302 chip register contents in the RAW, unformatted form. See DS1302 docs for more detail. getRawTime will not generally be used except to further explore how the DS1302 works.

```
Sample DS1302 chip Commands:
READ_YEAR   = 0x8d;
READ_MONTH  = 0x89;
READ_DAY    = 0x8b;
READ_DATE   = 0x87;
READ_HOUR   = 0x85;
READ_MINUTE = 0x83;
READ_SECOND = 0x81;
```

Returns:

DS1302 chip register contents

---

readTD

```
public int readTD(int index)
```

Read a DS1302 chips' time/date register, for sake of speed, this data is unformatted. The RAW data's format follows the DS1302 data sheet.

```
array[0] = second;      // 0-59
array[1] = minute;      // 0-59
array[2] = hour;        // 0-23
array[3] = date;        // of Month 1-31
array[4] = month;       // 1-12
array[5] = dayOfWeek;   // 1-7 (Sun-Sat)
array[6] = year;        // 0-99
```

Returns:

Contents of time array

---

readTimeDate

```
public int[] readTimeDate()
```

Read all time/date variables from DS1302 chip.

```
0 = hour;        // 0-23
1 = minute;      // 0-59
2 = second;      // 0-59
3 = month;       // 1-12
4 = date;        // of Month 1-31
5 = year;        // 0-99
6 = dayOfWeek;   // 1-7 (Sun-Sat)
```

Returns:
array size 7

---

updateTimeDate

```
public void updateTimeDate()
```

Forces the public time variables to be updated. This method is also called from within the DS1302 class

Returns:
Nothing

---

readTime

```
public String readTime(boolean ap)
```

Format the time (12hr/24hr). Useful for a preformatted time stamp.

Returns:
Time String

---

readDate

```
public String readDate()
```

Format the date to US style, useful for date stamp on screen

Returns:
Date String

---

readDay

```
public String readDay(boolean b)
```

Returns the day of the week. TRUE - returns long format FALSE - returns short format

Returns:
string

---

writeTime

```
public void writeTime(int hr,
                      int min,
                      int sec,
                      int mo,
                      int date,
                      int yr,
                      int dayOfWeek)
```

Set all the time variables in the DS1302 chip.

```
hour      - hr (0-23)
minute    - min (0-59)
seconds   - sec (0-59)
month     - mo (1-12)
date      - date (1-31)
year      - yr (0-99)
dayOfWeek - dayOfWeek (1-7), Sun=1 - Sat=7
```

Force any Zero values for dayOfWeek, date, and month(mo) to a '1' Values exceeding maximums will be changed to minimum values
Example: if sec > 60 then sec=0.

Returns:
Nothing

---

halt

```
public void halt(boolean b)
```

Halts the dsa1302 chip

Returns:
nothing

---

protect

```
public void protect(boolean b)
```

Protects the ds1302 chip from accidential writes

Returns:
nothing

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

Javelin Stamp

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
