Documentation for drmLib

/\*

drmLib.h - Utility Library for drm

To use put #include <drmLib.h> in your source code

Created by drm 20151213

V1.0 --> First with EEPROM access and start print

V2.0 --> adding RTC access

\*/

#ifndef drmLib\_h

#define drmLib\_h

#define drmLibVer "drmLib V2.0 -- 20160207"

#include <avr/sleep.h> //

#include <avr/power.h>

#include <avr/wdt.h>

#include "Arduino.h"

#include <EEPROM.h>

#include <Wire.h>

#include <TM1637Display.h>

// Doug's Data structures, constants and enums

#define ER\_BADID -20 // bad ID on requested operation

#define ER\_BADOPEN -21 // error opening file

#define ER\_UNEXPFIO -22 // unexpected result with file IO

#define ST\_AOK 0 // Everything is good

#define ST\_NOERR 1 // No error occured

#define ST\_NOOP 2 // Nothing happened

#define ER\_ERR -1 // generic error

#define ER\_UNK -1001 // unknown error

#define FALSE 1!=1

#define TRUE 1==1

#define MAX\_LINE 180

#define NUMRTCREGS 19

enum rtc\_type

{

DS3231,

OTHER

}

This structure contains all the RTC data

typedef struct **parseTime** {

byte seconds;

byte minutes;

byte hours;

byte dow;

byte dom;

byte month;

byte year;

byte csr;

byte sr;

int int\_year;

unsigned long lsec;

long tempf; // F temperature \* 100 (poor mans float)

};

}

// Routines in this Library

byte **drmBcd2Dec**(byte inbyte);

Converts a two digit BCD number into a binary representation

unsigned short **drmSerialNo**();

Retrieves the serial number from the EEPROM of an Atmel processor

void **drmStartPrint**(const char \*drmversion);

Prints out the standard start message

void **drmPrtLead0**(long in, int places);

Prints a long integer with leading zeros

void **printTime**(unsigned long milli\_time);

Prints the Time in the format d-h:m:s given a millisecond value

int **initRTC**(rtc\_type type);

Sets up an RTC module

struct parseTime **readRTC**(rtc\_type type);

Parses out the elements of the date/time from RTC raw data

void **readClock**(byte \*readBytes);

Returns time from an RTC

struct parseTime **decodeTime**(byte \*readBytes);

Parses time data returned from RTC