This text is provided as is without warranties. Whatever you make out of it, you do on your own responsibility.

Let´s start this as a mini collaboration project. Everyone with the link to this document will be able to edit / comment. Eventually we will copy the result over to the arduino wiki.
You are free to comment with questions with the target of improving this document. Do not misuse commenting on this document to get problems solved specifically to your project - use a forum for that!
pls have a first look, put your comments in that and then we can decide to create a (sticky?) post in Arduino:Storage to promote this document and the ongoing work on it. As soon as we collaboratively decide, that it`s kind of ready for the wiki, we can copy or link it there. Maybe over time we can identify some named editors keeping the edit access and everyone else can only comment. Let´s see how this works, it´s also the first public worldwide community document editing for me
And everyone please feel free to edit, modify, restructure whatever you want in this document. In the worst case we can revert to any previous version thanks to the cloud)
As this document is stored under my account @ google I hereby reserve all rights to remove any inappropriate, abusive or whatsoever unhelpful stuff that may come in; but of course the technical content remains owned by the community.

Ok, so you decided to store some data on an SD card attached to your Arduino :)

This document is intended to help you get common problems resolved. Maybe some expert topics are covered by accident as well, but focus should be on how to efficiently solve basic problems.

Start with describing the problem you try to solve

Basically the following problem categories can be distinguished
1. writing / read SD card does not work at all
2. SD card writing / reading speed (most presumably you try to create some form of data logger)

Also spend some thoughts, if SD is really the right approach for your use case.

When SD card writing / reading does not work at all

Ok, men don't read manuals. But start here !!! Any women working with

Arduino: you too! :)

You should start with some basic reading. Or you will soon start to waste your time - and worse: others´ time as well.

For example LadyAda has a very good intro on SD Card usage with Arduino:
http://www.ladyada.net/products/microsd/#introduction

If you are more into moving pictures, start with this great video tutorial from Jeremy Blum:
Tutorial 11 for Arduino: SD Cards and Datalogging

Before getting started, be sure you have covered the basic hardware setup

1. Proper Hardware

If you built your own hardware, maybe you should write your own document on this :), I doubt that this one will help you.

2. Proper wiring of your hardware

There are lots of different shields and hardware out there, either specifically for SD or combined with other functionality.

You need to check with the provider of the shield you use, how to set up the wiring for your SD card. For example LadyAda has a very good description of the required wiring.

If you use any library, which is not a standard library delivered with the Arduino development environment, you need to check the library documentation for wiring guidelines as well.

3. Formatting of the SD Card

To my understanding the SD Cards must be <= 2GB in size and formatted with Fat16 file system or even with the SD Formatter for all the currently available libraries.

Error analysis in the software

1. First get a very simple example running - verify your hardware

If you run EXACTLY this sketch and this does not work, you need to recheck your hardware and wiring (or you set up your development environment incorrectly like selecting the wrong COM port, choosing the wrong target port or a wrong programmer. But this would mean that not any Arduino coding would have worked for you so far...).

There´s absolutely no point in continuing with any further software analysis, if this does not work.

The standard examples you need to check

1. Check general setup: run the CardInfo example from the Arduino SD standard library. Do not modify the example for the first check.
2. run the read/write example
3. run examples provided with the library you are using (in case you use a non-standard library)

2. Cut back your code and add things back: step by step!

This approach is like (model) race car setup: change one small thing at a time. Change exactly one thing, not two. Only by sticking to this you will learn, which changes have which impact.
by the way: this approach makes also sense with issues other than writing to SD.

(1) Take your code, which doesn´t write to SD or better a copy of it. Comment out or remove EVERYTHING from your code except the pure writing to SD. Also remove/comment out #includes of libraries. Remove any texts and variables you don´t need for the basic test (texts and variables block memory)
(2) get this remainder working with some test data. Remember that at this point you only want to see the data being written/read, nothing else.
(3) at back your other coding step by step. As soon as the overall thing doesn´t work anymore go back and take the last thing out again. If this helps: you found the part where to dig deeper.
(4) maybe you need to revisit your overall design and check, if the combination of libraries and hardware you want to use really fits the arduino you are testing with (memory consumption, available pins and usage thereof, …)

Example: ‘I took your suggestion to cut right back my code and slowly added things until it stopped working - turns out the problem was a conflict with the motor shield I'm using - it was making use of timer 1 and enabling both pins 9 and 10 for PWM output. Although the SD card is CS pin 4, pin 10 is the CS pin for the ethernet in my board and the PWM output was messing with the SPI somehow.’

3. Typical Errors

- the sketch nowhere flush()es or close()s the file, ie data is never really written
- your sketch uses to much memory leading to memory overflows. Watch the free memory, there are library functions to check this. Typical symptoms are: arbitrary rebooting of the arduino (setup() is being executed ->place some log message in setup() to see this effect or blink an LED), the sketch just halting)

4. Dig into the libraries

If you try something special, be sure you have understood the available documentation and searched the Web for available answers. In particular if you are using multiple libraries there may be conflicts in memory usage and conflicts in usage of hardware I/O pins, SPI, ….

Examples:

- http://arduino.cc/en/Reference/SDCardNotes

SD card writing/reading speed

Start with defining your project goals by

- volume of data to be written to / read from the SD
- SD writing and reading speed you require
- How many read/write cycles will you require ?
  There are limits to SD cards for read/write cycles. After you hit those you won´t be able to write anymore (or even get corrupted data ?)
- For writing speed: which ratio of gaps in data you can accept?
  with increasing writing/reading speed, the probability of lost data increases. Example measurement to show this effect is described here:
  http://robertgetzner.com/PersonalWordpress/?p=1523
- Memory consumption of your sketch combining various libraries and tasks. There are some funny effects from memory overconsumption

Describe your goals separately for acquiring the data

... to be written (e.g. through ADC or other sensors) and writing the data to the SD. Keep in mind that these processes have to be run on the same arduino controller! Getting fast ADC (or other input) reading needs to be tackled separately, although some of the SD card examples combine this very well with fast SD card writing.

Understand the clash of targets

Fast data logging needs to combine mainly two things

- fast input
  e.g. ‘tricks’ for fast ADC like not using analogRead() but direct ADC control through ADC registers, interrupts etc.
- fast writing
  With increasing writing/reading speed, the probability of lost data increases; this is caused by SD card latency and writing speed.
  Example measurement to show this effect is described here:
  http://robertgetzner.com/PersonalWordpress/?p=1523

So you need to make the right software design using timers and interrupts to control the overall logging flow. In this design you need to balance the frequency of input readings and writing output. You must make a decision how fast you want to collect data AND how fast to write it to SD. As an overview check the documentation of NilRTOSlib https://github.com/greiman/NilRTOS-Arduino, where two threads (one producing data, the other consuming data) are described with a diagram.

The current libraries buffer data before it's actually being written, because SD cards have optimal sizes of data to be written during one write cycle. So the actual writing time is increased with the buffer size, but the write latency only hits you once per write cycle. But during this writing you can not collect further readings from the input (e.g. ADC); so during this time you will 'lose' data.

You need to decide depending on all these considerations

(1) which Arduino is capable to meet your requirements in terms of hardware sizing and which combination of libraries may work for your use case.
If you want to say log 1TB of data every 10 nanoseconds, you for sure need to look somewhere else ;)
(2) If SD card is the right medium for your purposes, or if other media fit better

- If you have low requirements (volume, speed), immediate transfer to a PC or mobile might be more efficient and even better fit your use case.
  You could also have SPI memory chips attached to your arduino, to non-permamently store your data (e.g. fast logging and writing to memory for some timeframe and then transmitting it to PC or mobile). Unfortunately I did not see any ready-to-use SPI memory chip shield for arduino yet, but there are descriptions out there (first hit: http://hackaday.com/2011/09/07/want-2-megabytes-of-sram-for-your-arduino/. There is a RAM extension board for Mega, but I don´t have any experience on this (http://ruggedcircuits.com/html/megaram.html)
- Use an EEPROM, e.g. have a look in the playground

Examples

- New fast data logging sketches
  Combination of fast ADC (leveraging the ADC timed interrupt) and fast writing to SD
- Fast data logger for multiple analog pins
- Girino - Fast Arduino Oscilloscope
  A great description, but I must admit I did not spend sufficient time to understand it yet...

Get help in a forum and report success

Of course there are so many combinations of libraries, hardware and different use cases that you may always step into some very specific trap.

To get you help fast please help the helpers on the forums: ensure upfront that your hardware works (if not: go to the hardware part of the forum), do the basic testing examples (the ones that work ALWAYS, except if your hardware setup is wrong).
Then continue with the step-by-step cutback and -addback procedure.

When seeking for help, describe as exactly as possible, which change lead to the error, ie made your sketch 'stop to work'.

When finally everything worked out well: report back to the forum that it worked. Only by this last step you will help others to identify the thread as having been solved and the descriptions therein as valid solution approach.

For example you can go to the arduino.cc:Storage forum