1. NEVER loan out your copies of:
   - "The Art of Electronics" (Horowitz & Hill, Cambridge University Press) (you do have a copy don't you?)
   - "Illustrated Sourcebook of Mechanical Components" or Ingenious Mechanisms for Designers and Inventors
   - The Art of Computer Programming, Volumes 1..3 Boxed Set

2. Always quote at least twice the time you think it will actually take to do the job
   (if it's good enough for Scotty...) 90% of the code gets written in the first 10% of the time,
   the remaining 10% of the code will require more than the remaining 90% of the time.

3. Always have someone (or a group of) real pro(s) to fall back on for advice when you get stuck. But, never rely on someone else's circuit design to work as drawn or code to run as written.

4. Always document everything you do (why did you always see engineers and scientists with a log book?) and be ready to extract a complete history of actions at a moments notice. I don't care how sharp you are, at some point, while trying to solve a complex problem, you will realize that you don't remember exactly what you already tried... which means that you are duplicating effort, running in circles, and doomed...
   - When it really drops in the bucket, management WILL try to make you a scapegoat and being able to tell the customer exactly what you did may save your job (or get you a better manager or even a new job).
   - Watch your co-workers and boss... when they start to pull away from something, get your notes in order

5. Always understand that you may pick two of the following three, but not more:
   - It can be built well,
   - It can be built quickly,
   - It can be built cheaply.

6. **Ohms law**: Know it, look for it, use it. Very simple but often missed.
   - Ground planes, **power** busses, shields, etc... are not superconductors, they resist and a voltage will develop when current flows.
   - The heat dissipated from a component operating at its rated wattage might look small on paper but it can build up to a melt down in the real world. Use a higher rating and heat sink and ventilate, but don't depend on good ventilation; the customer has to set his paper work somewhere...
o Instantaneous currents can be huge in switching devices, even if the average is low. **Decouple** every major and sensitive component. Overbuild the **power** supply. **Power** traces are part of the **power** supply.

o **Noise** happens

7. Business WILL always be part of engineering. Get over it.

o Lots of people lie. You can't tell. You can't get along with out them. Hope it doesn't happen, be ready for those times when it does.

o Richard Stallman. "...People find ways of getting money by impeding society. Once they can impede society, they can be paid to leave people alone." 

o Salespeople will tell you anything they think will sell their product (see point 2.) Don't trust the Vendor, FAE, Distributor, Rep, etc... Ask around. On the other hand, if you want all the samples you can handle, make sure your title is "Senior Firmware Engr"

o Payment for contract work may be difficult to collect (see point 2.) *Fed: Oh! What to say... We can do volumes on this one. See: Consulting.*

o Never underestimate the stupidity of the end user of your product. Make it fool and idiot proof if possible. This is very hard to do. It takes infinite intelligence to anticipate boundless stupidity.

Artificial intelligence is no match for natural stupidity. From "The Tao of Programming": A [product] should follow the 'Law of Least Astonishment'. What is this law? It is simply that the [product] should always respond to the user in the way that astonishes him least.

Some people live lives of perpetual astonishment.

o "I have been consulting for 30 years and have come to understand that a stupid customer is a treasure, not a problem. If they are really good, they don't need me. If they are really "idiots", they don't need me to tell them that. They need me to help them for a fee."


o Make things that people want to buy. Not use, play with, see, understand, etc... **BUY**

o Design a product that has something better about it than any similar product.(Price, features, battery life, etc)

o Marketing is necessary. You must convince yourself that people need your product before you will be able to convince them to buy it. Try to see your product in the most favorable light possible:

  - A tag from a Duracraft electric fan brags that it is "Manually Reversible". So you can reverse the flow of air by flipping a switch? No. "Manually reversible" means you can pick the fan up and turn it around. -- Consumer Reports, Oct. 1991

o Have a life, don't neglect your spouse and kids in favour of your job. It really really really isn't worth it. I've never met an old man who said "God, I wish I'd spent more time on my work." If you boss doesn't understand that or care, start looking for one who does. If you find one who does understand, take good care of him or her.
8. K.I.S.S. The ideal design has zero parts. "An engineer is someone who can build for a dollar what a fool can build for twenty" - Robert A. Hienlein
   o If it ain't broke, it doesn't have enough features yet
   o To the optimist, the glass is half full. To the pessimist, the glass is half empty. To the engineer, the glass is twice as big as it needs to be.
   o In any product more complex than a few passive electronic components (resisters, caps, etc...), a little microcontroller with a clever program is worth the coding time because it reduces the component count. Of course, code is a component.

9. Hardware and Software (firmware) need to work together
   o Hardware should be viewed as a necessary evil, and minimized where possible.
     ▪ Code can always be modified, once the boards are made, the electronics is fixed, so programmers will always be expected to fix hardwares screwups.
     ▪ Any circuit, other than signal I/O and A2D / D2A, can be replaced by firmware giving a fast enough processor and clever enough coding, thereby reducing component counts and product cost.
   o Software should be viewed as a necessary evil, and minimized where possible.
     ▪ No amount of code can respond faster than the clock in the controller.
     ▪ Code almost always consumes more power than dedicated electronics.

10. "Thou shalt not design thy PCB before thou hast the components in thine hands."
    o And even if you DO have the parts in hand, make certain that you can order them in the quantities you need later. e.g. watch out for Maxim or Motorola (68HC11s) or TI (msp430) or Pericom.
    o When working with Atmel: "...until thou hast lifetime production quantity in hand, UNLESS thou art prepared to "slightly modify" code occasionally as new 'improved' processors are introduced and old ones are 'retired'."
    o When working with Zilog: "...until thou hast lifetime production quantity in hand, AND each part has been tested and shown to work." Actually, just don't use Zilog.
    o When working with Microchip: "...Olin has done at least one project with the chip(s), Microchip has acknowledged the problems he found, documented them in the errata sheet, you have read that sheet, and you really understand what is stated there."
    o 1st law of component engineering: only use devices that EVERYBODY makes EXACT replacements for.
    2nd law of component engineering: Exact replacements aren't.
    1st law of purchasing: Be sure to ask your vendors to recommend alternate parts which are identical to the specified part.
    2nd law of purchasing: Do not trust the vendor who tells you a replacement part is identical to the specified part. Have an engineer test the replacement parts before sending them to manufacturing.