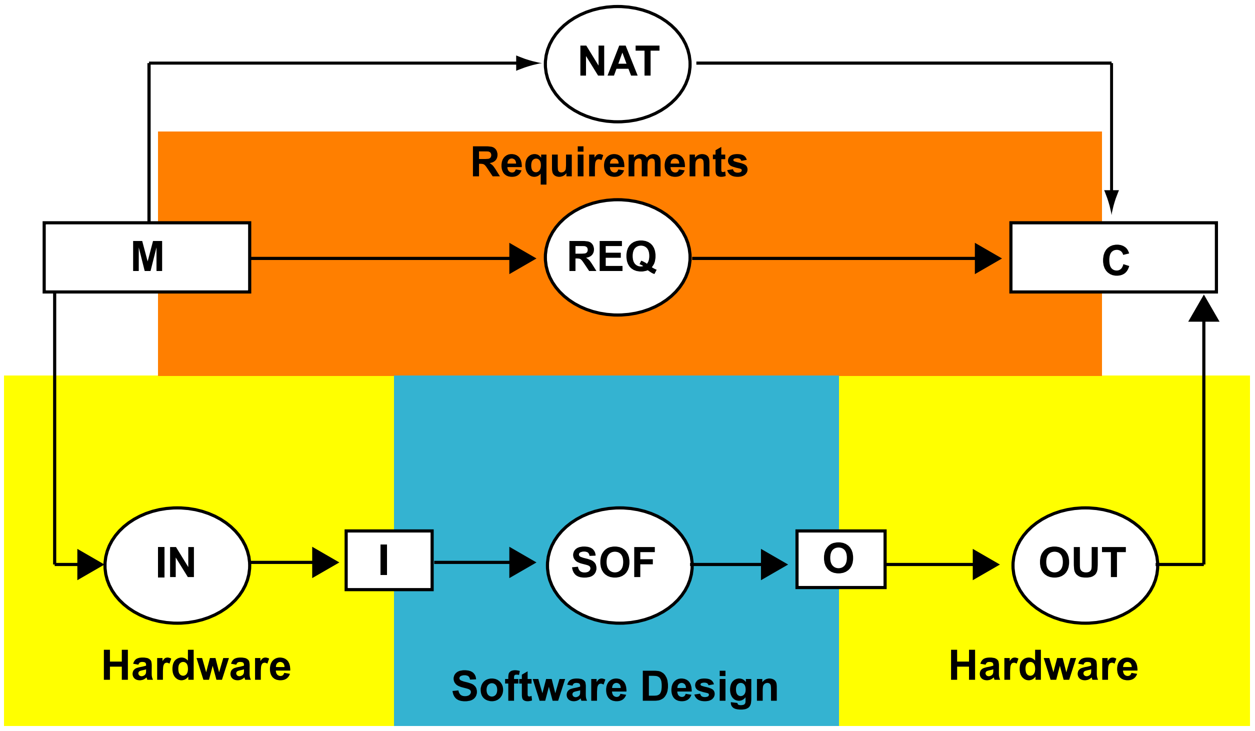
Monitored and Controlled Variables and the 4 Variable Model

There have been a number of questions about M-I and C-O mappings (rather than O-C). Please look at the OPG training modules for a more in-depth discussion (Module 2 as a start).

Briefly.

Monitored and Controlled Variables are the system inputs and outputs. There is a famous model, the “4 variable model” originated by Parnas & Madey that describes the relationship between requirements and software design.



The 4 variable model

REQ relates the vector of controlled variables, C, to the vector of monitored variables M. It is the requirements specification. (NAT represents the environment. It places constraints on C.)

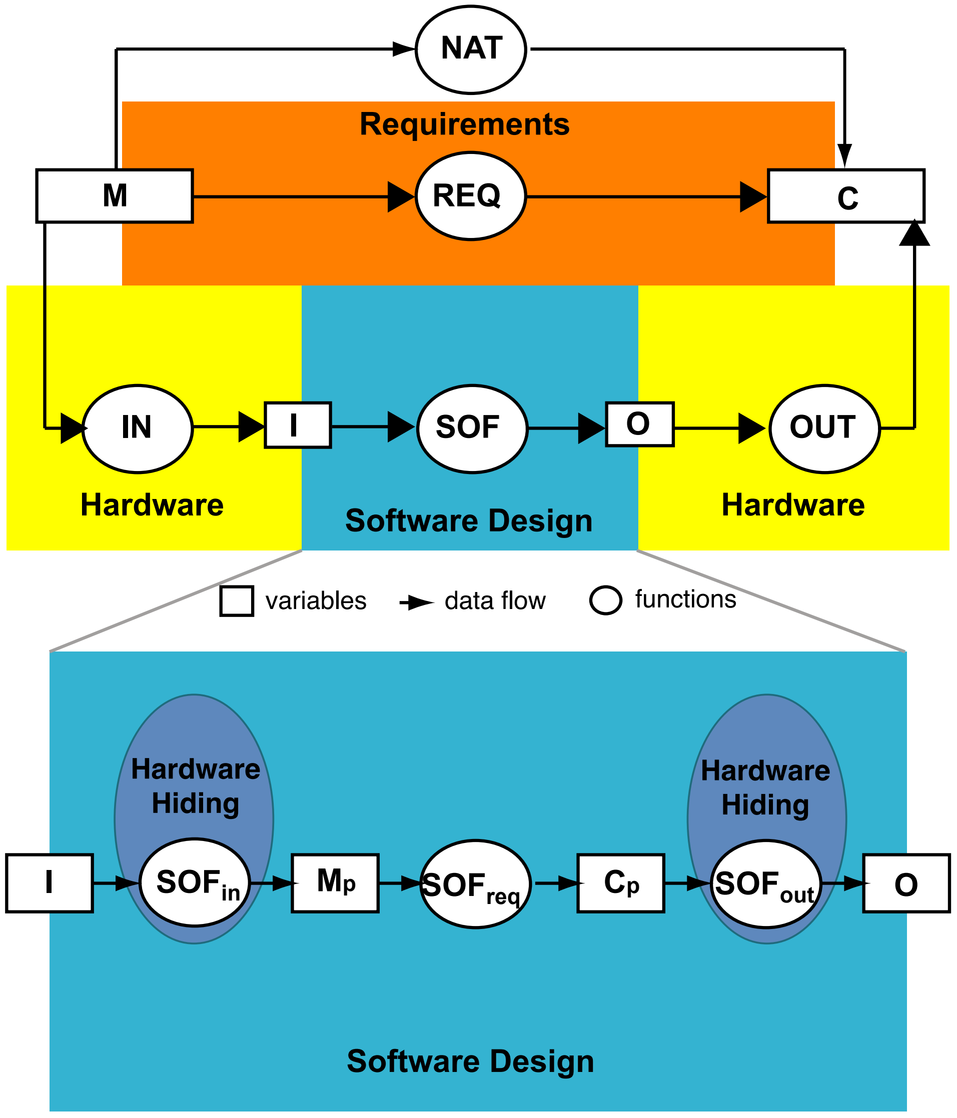
When we develop a software application, we use hardware (sensors, A/D converters etc) to transform M into a vector of Input Variables that the software has access to. Then the software design can be represented by SOF and it maps I to O, Output Variables. The Output Variables are then transformed by hardware again (actuators, D/A converters, D/D converters, etc) into the Controlled Variables.

In other words, the hardware, represented by a function IN, maps M to I. Also, hardware, represented by the function OUT, maps O to C.

Example:
We have m_1 representing temperature in degrees C, for example. The valid range may be -40^o to $+120^o$ C. We have hardware that maps the value of m_1 to i_1, such that $i_1 = round(4.0*m_1/5.0)$ And this is stored as a 12-bit fixed point number. I_1 is what is available to the software application.

So – this presents a problem. We want to use REQ as a specification for our software design, but REQ has inputs $m_1, m_2, \ldots, m_n$, whereas SOF has inputs $i_1, i_2, \ldots, i_n$. What do we do?

The simple answer is shown in the modified 4 variable model.



Modified 4 variable model

We use information hiding! In this case, what is likely to change? The hardware of course.

So, what we do is create some simple software to transform I into M_p (as close as possible back to M again). We call M_p “pseudo-M”.

Example:
If we return to our example above regarding temperature, all we need to do is calculate
$M_{p1} = I_1 * 5.0 / 4.0$

We cannot get back to M1 exactly, because we lost information in converting to a 12-bit integer. But we can calculate the error and see if it is within a given tolerance.

The C-O mappings work similarly.