APSCO - 2019

CubeSat Technology (in)capabilities

Leonardo M. Reyneri
Politecnico di Torino

Technological capabilities

What are the technological capabilities of CubeSat ?

Nowadays technology offers nearly incredible opportunities… But…

The effective use of these opportunities often depends on the design capabilities of the design team!!!

Design team capabilities → S/S capabilities

Players in the market

There are several categorie of p ye in the market:

- Amateurs (students or no of n w n limited knowledge and te no gic capability but with a lot o ntl us sm
- Students of courses in ae sp e ma
- Autonomous student tea
- Teacher-driven student te ns
- Application industries or en es
- Space industries

CAPABILITIES

COST & RESOURCES

RELIABILITY

KNOWLEDGE

Student-Level CubeSat Capabilities

No previous experience in S/C design!

Little interdisciplinarity

Students make a huge number of errors

Knowing the typical errors beforehands may help teachers to improve their teaching

Must learn from previous lessons. Often (e.g. first CubeSat) none or very little…

Need to share experiences!!!

Simplify → open CubeSats to high schools…

What do students usually miss?

What do students usually miss?

Software → radiation sensitivity

How to take a class-level piece of SW and make it rad-tolerant?

Steps: once SW is developed,

Measure its sensitivity

Identify spots which must be hardened

Harden selectively

Use user-transparent hardening C++ classes

What do students usually miss?

Hardware → incorrect spes

How to verify that specifications of a CubeSat module are complete and correct?

A Few Specification Errors

Students prepared the following requirements:

Requirement #1: the system shall count upwards the events on digital input CLK

Requirement #2: the system shall be capable of counting up to 100 events

Requirement #3: the system shall operate with 5V supply voltage

TTL 3.3V

Initial value ? 0-100 or 1-100 ?

Tolerance: from 4.75V to 5.25V

Lesson Learned

- Preparing correct specification is a quite hard task
- It requires a lot of experience, usually more than student-level
- Teacher shall teach “procedures” to lay down correct specs
- Such procedures are quite similar for HW, SW, mechanical, etc.

Procedural laying down of specs

Procedure #1: No single numeric value shall be unique, except boundaries. Examples:

- Supply voltage from A to B (value +/- tolerance)
- Power consumption less than A (boundary)
- Clock frequency less than A (boundary)
- Algorithm parameter from A to B (min/max allowed values)
- RAM usage shall be less than A (boundary)
- Supported vector sizes from A to B (range)
- As usual, exceptions exist… (e.g. some digital values)

Procedural laying down of specs

Procedure #2: Any variable shall have at least min, max, default and "reset" values:

- Counter shall count from A to B. At power on, it shall count C; when reset it shall count D.
- Output of DAC to drive a motor shall range from A to B. At power on, it shall be C; during emergency it shall be D .
- Output of a SW digital filter shall range from A to B. At power on, it shall be C; when reset it shall be D.

Procedure #1:
no value shall be unique..
➔ C±ΔC

But what other errors are present here?

What do students usually miss?

Hardware → incorrect spes

How to verify that specifications of a CubeSat module are complete and correct?

Steps: once you have specs,

Verify against a set of formal procedures

Identify specs which must be improved

Correct specs → verify again

Fill a check-list table and proceed only when fully compliant

What do students usually miss?

Complexity → who has got enough experience to manage it ???

Surely no student !

A CubeSat is too much for a single person

A CubeSat shall be designed hierarchically !

Hierarchy increases the number of individual modules and interfaces but makes each module feasible by individual students!

The key step

Go down to subsystem, or even subsubsystem level!!!

Much more manageable and reliable

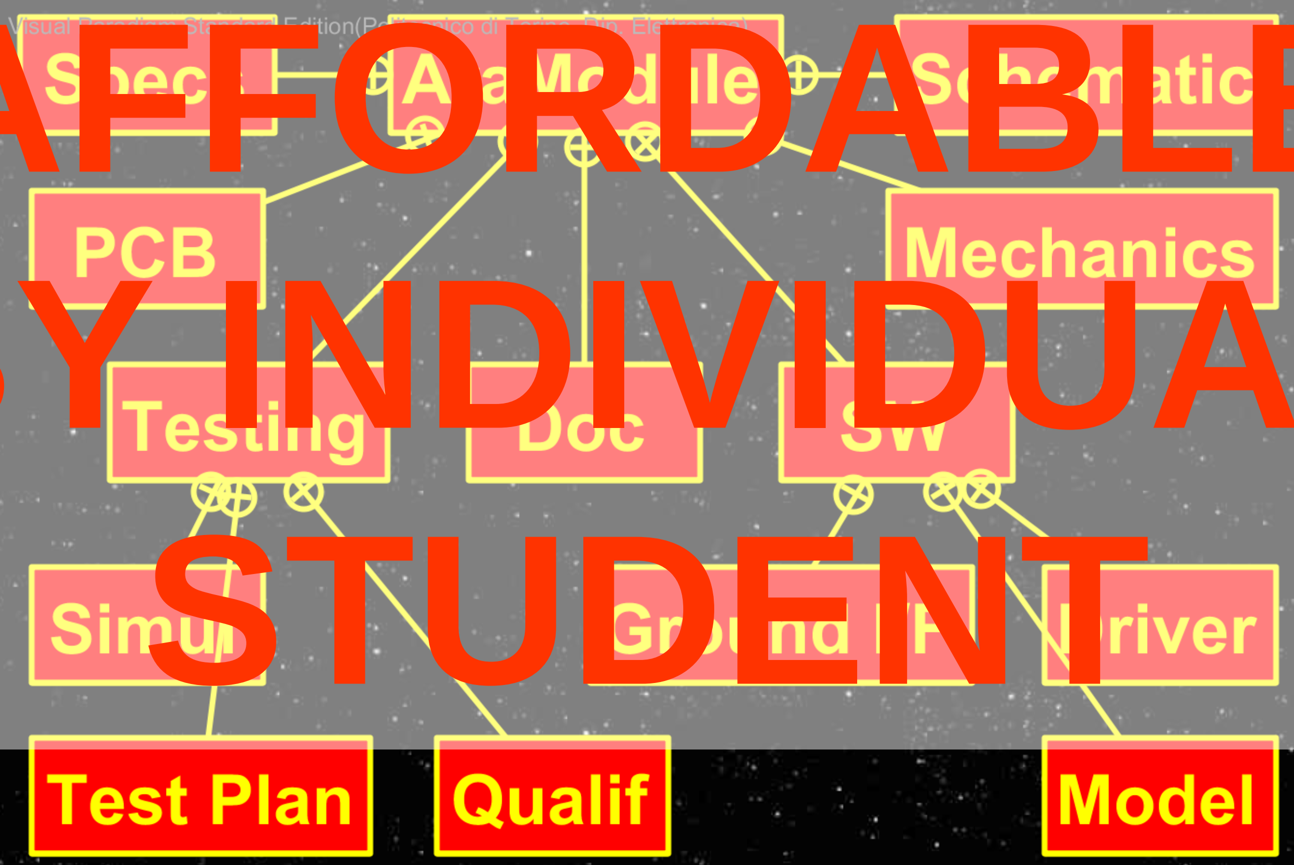
The key step

Manageability and reliability come from a simpler system →

- Easy and complete specifications
- Full testing → more fault coverage
- Complete and detailed documentation
- Model development

AraMiS → “AraModules” and “Tiles”

An AraModule is a tiny P&P subsystem



AFFORDABLE BY INDIVIDUAL STUDENT

What do students usually miss?

Hardware → interface coherence

How to verify that specifications of two interacting CubeSat modules are coherent?

Steps: once you have specs already verified for completeness,

Verify boundary of source with boundari(es) of destination(s) of signale

One boundary shall include the other one (and often viceversa)

Identify incoherent interfaces

Coherence among subsyst. parameters

When a system is composed of interacting subsystems, a major source of errors is the incoherence among system parameters.

This is particularly evident in the SW subsystem, as SW variables cannot store physical units and/or other ancillary information.

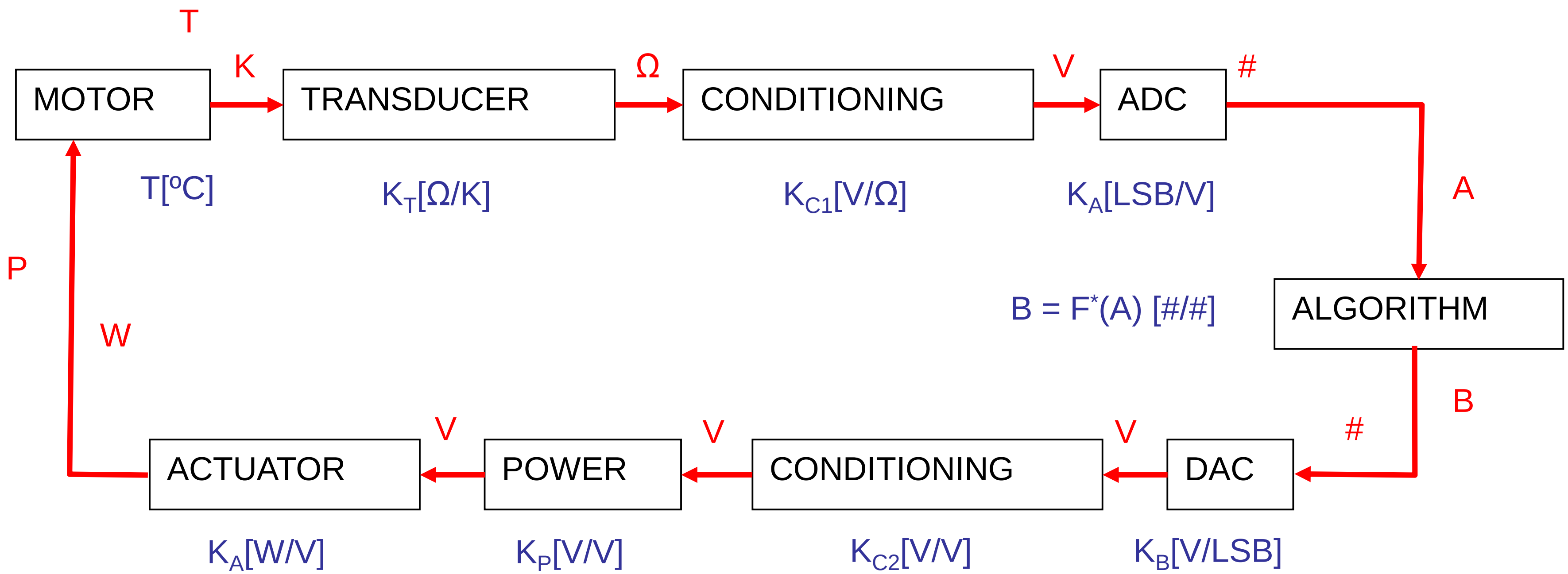
A Classical SW error

A classical mistake: to turn on a fan when temperature overpasses, say, 70ºC:

```
int temp = ADCread(); \\ read ADC value
if (temp > 70) \\ check if temp > 70ºC
  FanOn();       \\ turn on fan
```

What NUMERIC value is returned by ADC when TEMPERATURE is 70ºC ???

Coherence among subsyst. parameters



Specs:

Real world: we need $P = f(T)$ ➔

SW algo: $B=F^*(A) = (K_A*K_P*K_{C2}*K_B)^{-1}*(T*K_T*K_{C1}*K_A)$

Coherence among SS parameters

- ALL coefficients (K_A,K_P,K_{C2},K_B,K_T,K_{C1},K_A) do depend on possibly 5/10/20 mechanical, electrical, physical, chemical, etc. components… (maybe 20/100 altogether!)
- What if ANY of these components either is updated or modified or improved, or specs are changed?
- …it may happen AT ANY TIME during design phase or for new systems releases!
- A quite strict version tracking procedure is required.

Coherence among SS parameters

- Quite simple in SW (versioning tools) but…
- Quite complex among independent mechanical, electrical, electronic, software, system engineers within geographically distributed design teams
- ANY change in any component must “warn” ALL other components, elements, subsystems which may interact with it…
- Who keeps track of all these relationships?

Coherence in university projects

- QUITE TOUGH TASK!!!!
- Students usually come, go away, often very quickly
- Many students are very good and do excellent jobs; others are lazy or not experts and their technical quality is low
- They are NOT prone to a reasonable documentation ➔ major error source
- Model Based Engineering is a viable solution

What do students usually miss?

Testing → how many parameters can/do you really test ???

50% in the best cases… 20% in the worst!!!

Typically forgotten:

Radiation hardness (even in simulation)

and total dose (need real testing)

ADCS !!!

Do you have a real sun simulator for solar cells?

Thermovacuum: who has reasonable equip't?

Conclusion

The achievable technology capabilities in non-professional environments

strongly depends on the proper design techniques, procedures and testing used during design…

You can therefore “decide” to approach 10/20/30/40/50/60/70/80/90/100% of state of the art capabilities…