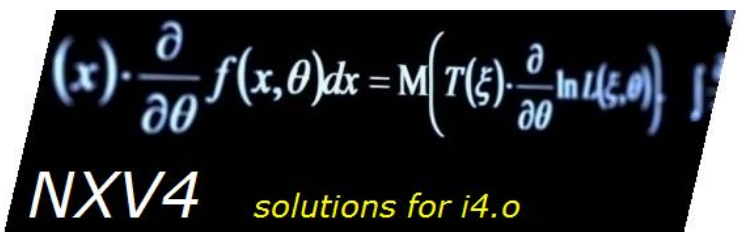
IND-4.0 portfolio

2022

NXV4 Solutions

tel.: +40-724-502-557
email: fftnxv4@gmail.com



IND - 4.0 consulting

(blue = in partnership)

EDU - 4.0 training

Product Lifecycle Management

Industry-4.0 (i4.0) follows the classical components industry, with extra emphasis on flexibility, single / small-number lot production, digitalisation, niche-market approach.

The product development sketch in figure 1 illustrates the steps required in fielding a product and seeing it off the production line. It is important to start off with this view (from early-stage VC) because the flexibility required for i4.0 is typically closer to the early-stage product, rather than that of an established production line (that re-invents and refines a product over a number of years).

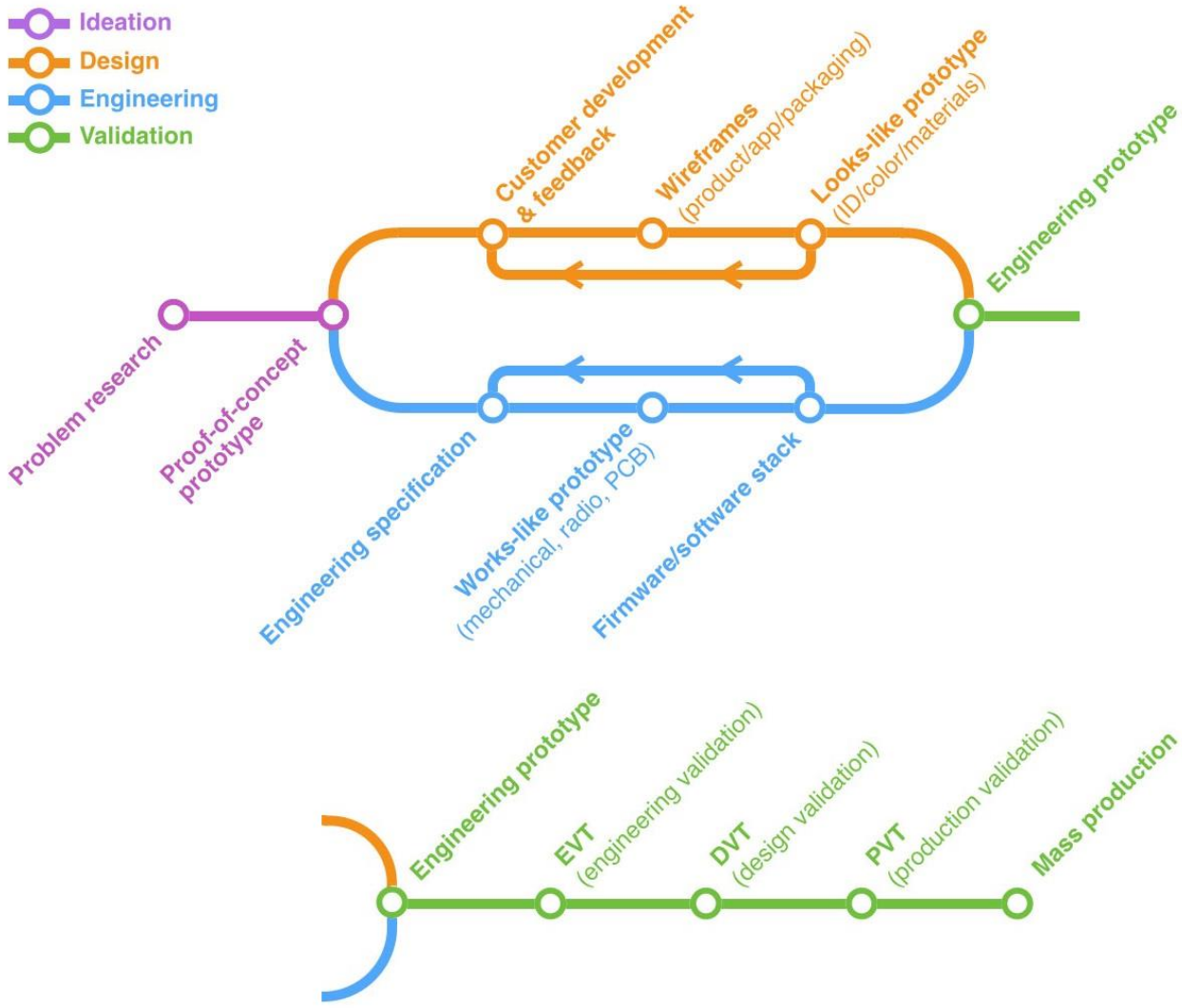


Figure 1 – Product Development flow (by Ben Einstein, Bolt - early-stage VC, Boston/SF).

The fundamental i4.0 question is: *where does Digital-Era speed-up / impact the above flow ?* The answer is one two-fold:

i. by digitalising most/all steps and connecting them together - and
ii. by developing and using a "***Digital Twin***"

This answer envisages ample digital infrastructure in the form of *"computers"_everywhere* (single-boards, SBC computers, etc) that by the sheer volume is difficult to maintain – and also difficult to interconnect, both volume wise and *need-to-know wise*.

The Digital Twin further adds to this infrastructure (in the software sector). The Twin is a piece of code that responds *like-a-hardware-prototype* (of the above flow chart). It parallelises production development by allowing downstream teams to work *as-if* the actual hardware prototype exists. This component in turn asks for the implementation of a "***Single Source of Truth***" mechanism.

By inception digitalisation is bi-directional, allowing product feedback both from the test-phases downline, or from the market upline to the design-teams. This enables global optimisation of all production sections, adding value and reducing costs.

This is precisely ***the point where you make the step ahead of your competition.***

You are able to develop the product faster, at lower costs and, above all, be able to afford single-lot-production. You flexibly ply your company to the market, picking up the small-number and unicate orders that nobody attends to. Today that means a large fraction of the market.

Production scalability becomes ***your main marketing advantage*** – allowing you to market yourself through the number of sales you make, each account bearing your company's name. Many smaller contracts with numerous small players get your name mentioned in the branch everywhere. And we all know the difference between bulldozer-advertising and personal recommendations !

Allotting less to marketing (but significantly more efficiently) allows you to invest more technically in yourself and in your human resource (your most important asset) and let your accounts do the marketing for you. Advertising money is a one-time-burn, however investing into an i4.0 production line is a lifetime investment that rewards you through all your activity year after year.

Digital Twin

The *Digital Twin* is an unambiguous, technical and marketing-functional, equivalent in digital representation of a specific product.

Having a Digital Twin, even as a "*first-draft sketch*" (in digital form) of the product, is helpful in letting downstream teams (say packaging-and-casing for instance in semiconductors) work in the parallel development of the product, together with the other teams.

Modelling and continuously updating the "Digital Twin" will not affect downstream teams (as long as "contractual" product-borders are established in the product's parameter space). This allows you to cope with more complex features, defined in the twin's more advanced representations later in time.

This is a point to note that information is produced in large quantities (various versions, with different need-to-know levels for different teams) along the prod-dev chain – that will need to be correctly integrated in the SSoT system.

The reason for having a digital twin is thus not solely for not keeping waiting design teams, rather also for harmonising services and optimising resources.

The digital twin will likely have a tumultuous lifecycle, some important changes coming in after the first batch of prototypes appear and are evaluated. For semiconductors for instance this means the first real values of parasitic elements (capacitances, loss tangents, etc) will be known after batch-1 and be available firstly for the design of batch-2. The same for mechanotronic products: the first real tribometric values, fatigue and endurance tests are after batch-1. For chemical products these would be the figures for purity, mass spectrometry, etc.

Needless to say, in a highly competitive environment, such data demands a high-level need-to-know clearance. While the casing-and-packaging team will continue to be content with previous embodiments of the digital twin, performance-tuning teams will need to have access to up to date data.

Twin hierarchies are conceivable for complex (multi-component) products. In this case twins-of-systems are likely elaborated (which in turn can involve twins-of-components). Systems can also have various versions depending on the components' versions.

I showcase here my study of ground-bounce pacification for multi-module electronic systems (that due to geometrical considerations cannot afford a star-ground).

I twin-ed the MOSIS embodiment of the IBM-Burlington mixed-signal 7RF 180nm process (run T44H) as chip transistors (figure 2) and placed the twin in a SPICE simulation with intentional parasitic inductors in the ground distribution. I placed pacifying capacitors in parallel with the consumers to interact with the parasitic inductances (yielding lossless LC energy-flow, in contrast to

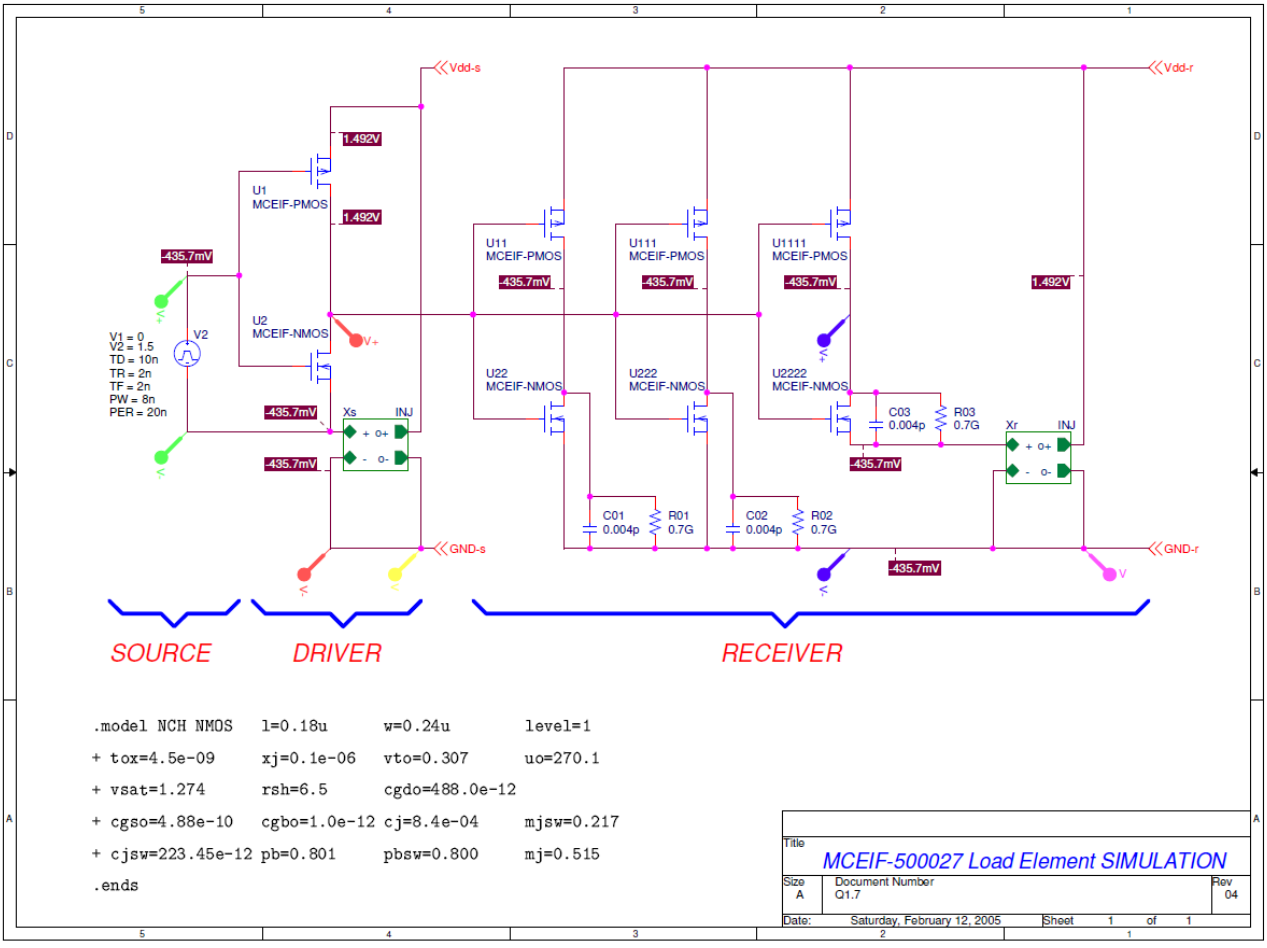


Figure 2 – Digital-Twin of a typical digital consumer using the MOSIS embodiment of the IBM-Burlington mixed-signal 7RF 180nm process (run T44H) as chip transistors.

the traditional RC E-flow, with αCV^2 losses).

The result showed how, in the traditional grounding, the ground line bounces given the digital power consumption of adjacent modules sharing the same GND line (figure 3). The top plot shows the module-under-test (MUT) output vs source input. This is relatively similar for all options considered (with a slight under-shoot for the GND with inductor version – in red). The mid plot shows the GND line with solely an inductor. The noise is rejected, but LdI/dt ground bounce (from own switching activity) is present. To keep the rejection, but alleviate bounce I placed a capacitor in parallel with the MUT. The bottom plot shows the GND line in this (final) configuration – exhibiting both good rejection of digital noise from neighbouring modules, as well as quiet GND for the MUT's own switching activity.

Note how the twin can be switched –in and -out, in order to test various values for the inductor-capacitor pair. This is typical for a digitalised product development flux.

This digital twin can also be used for a microprocessor placed on a motherboard with the afferent chip-set. The micro-processor can be the twin, from the point of view of the motherboard design-ers, important being that the motherboard design team not be held still while various microproces-sor options are tested – since most of the advanced functions of the microproces-sor need not be modelled by the twin.

For the mother-board design-team of importance are para-meters such as: fan-out, clock fre-quency, wave-front rise times and falling edges, which are known from the chip technology to be used, while the rest of architecture is largely irrelevant.

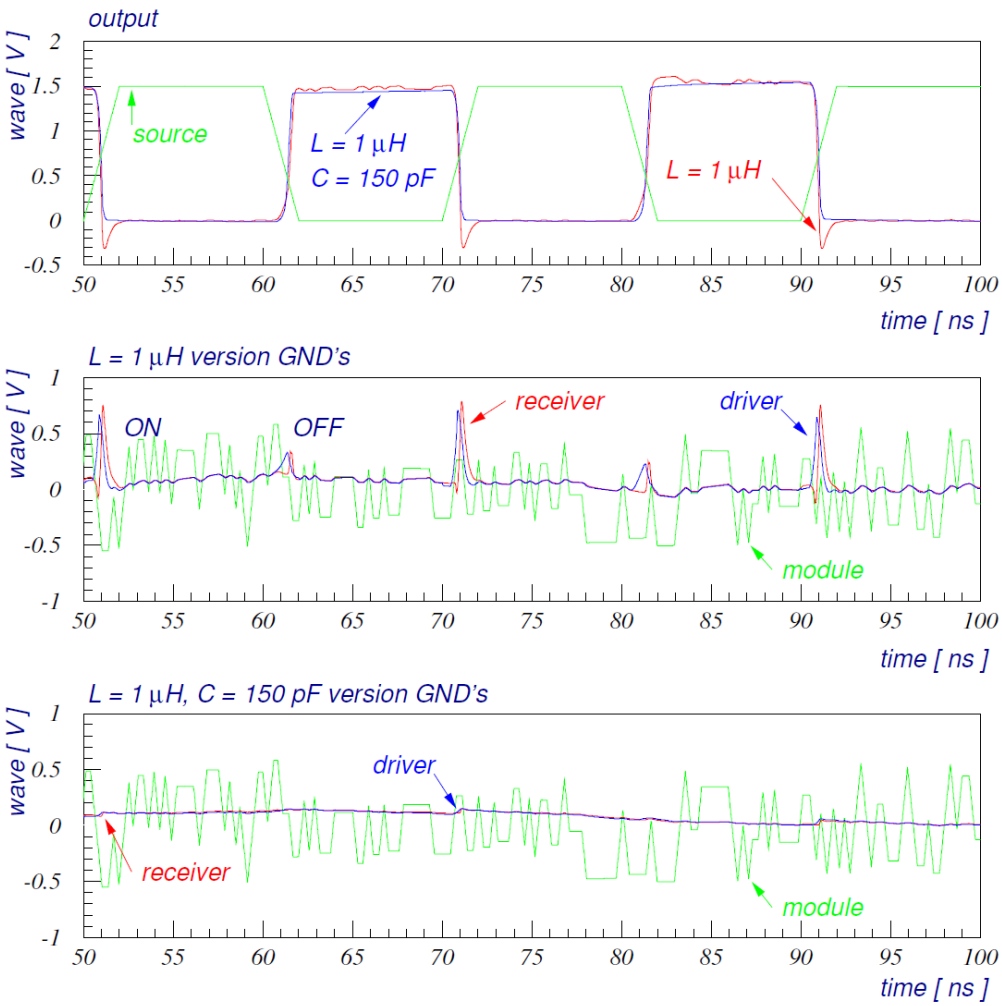


Figure 3 – waveforms for (top) module output vs. source input, (mid) GND line (1µH in the ground line) and (bot) in parallel with an 150 pF capacitor. Capacitance pacifies GND bounce, inductance rejects noise.

Single Source of Truth (SSoT)

As evidentiated above, design flows encompass a multitude of versions, as various teams work in parallel, and their versions need to be tested, or embedded into a bigger component under test.

Further more, each team has variants and older versions. Viewing privileges and need-to-know clearance are an important aspect in this web of versions.

Another aspect is that the versions produced, and the data in general, is stored locally, not dispatched throughout the enterprise.

Managing distributed data-storage is not new (for instance in GRID computing there are a multitude of file managing systems: Chirp, GRID-NFS, LegionFS, Ceph, L-Store, GFarm, GPFS-WAN, SRB, PUNCH, VegaFS, FT-NFS, WOW, IBM's legenday "Spectrum-Scale FS" GPFS, etc).

The maturity these systems is relatively good, each emphasizing compliance to certain needs. However, what is needed in an i4.0 environment is the addition of need-to-know privileges. Some of these privileges are at the production-line reliability level (industrial processes), while others are resend-request level (various users).

In such a vastly heterogeneous environment key is the specialised IT engineer to couple all storage management systems into one functioning entity, on different levels of need-to-know clearance and distribution locations.

In particular Industry-4.0 will hit this problem in a more dire way, as the versions are not just for informative purposes, rather they can be actual automation code implementing real time decisions, calibration data (needing updates), or live assembly line configurations – all with real time impact on production and costs.

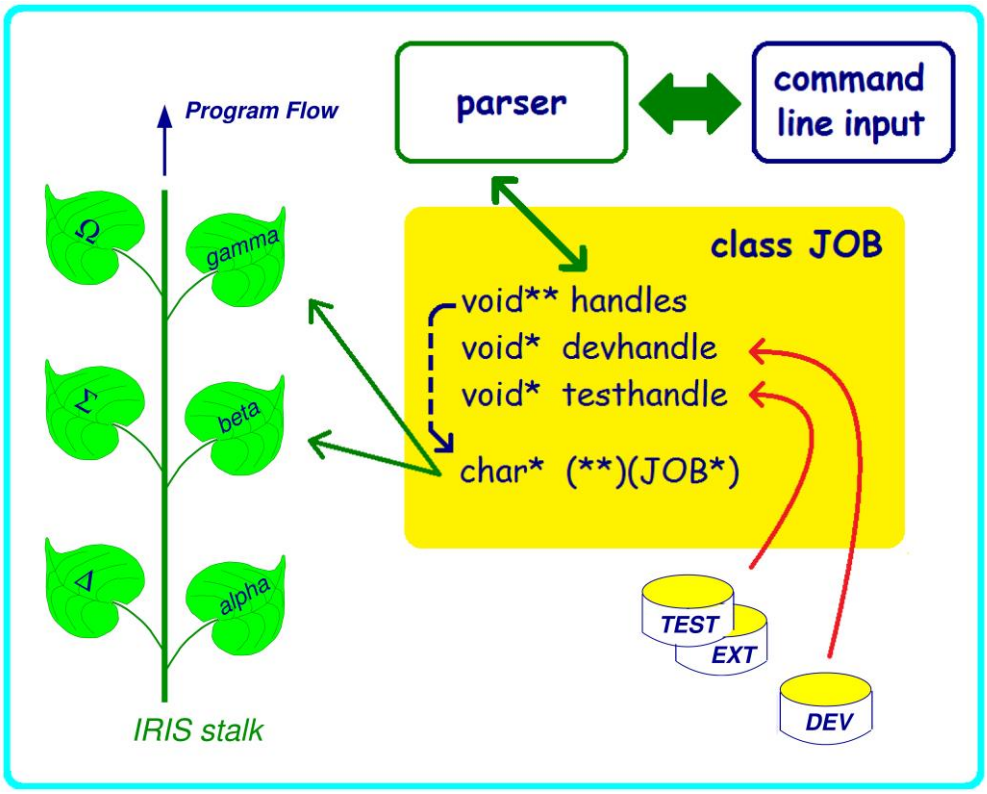


Figure 4 – Resource assembly for executable build depending on need-to-know priviledges within my *IRIS* framework. The input (code or data) resources are PROD (default), DEV, TEST and EXT – each with own clearance levels.

Data consistency across teams and versions will require the data to be encoded in a ledger (available from the FMS' timestamp) and be made available – based on the need-to-know clearance level provided by a request.

Overall, as mentioned at the beginning, adapting existing FMS technology to a *Single Source of Truth* (SSOT) environment will rely heavily on the IT Systems' Engineer.

I showcase in this respect the *IRIS* framework I designed to solve the problem of concurrent code development by various programmers, for data-acquisition trigger-software (that I present below in figure 6). The package under development demanded extensive compilation times, grinding development to a halt – as small code pieces had to be (hot)-swapped in/out of the package to test various hypotheses and approaches.

Figure 4 shows how *IRIS* builds an executable, picking the right leaf from the correct resource and assembling the (dynamically loaded) exec. The framework allowed simple assembly, compilation and hot swap-in for a set of "leaves", with assembly for the rest, compilation of the full complement, add-in of a new "leaves", etc.

Typical i40 SSoT applications will demand similar technology, only that the source can also be data, the sources will need clearance levels for various logical-units (production, engineering, design, management, accounting, marketing, etc) and the exec will not be necessarily only code execution, but also presentation of requested data to a given port with a certain encryption.

Certain elements of the distributed file system need to have in view the SCOR Value chain Operations Reference Model Vocabulary.

Flexible automation

Industry-4.0's flexibility is advertised as *today build a car, tomorrow a jet-plane*, an obvious exaggeration, yet reconfigurable tool & test equipment production lines lie at the heart of i40.
In this respect I can provide experience of impact in:

1. **SCADA systems** – be it a simple LabVIEW based system, or multi-modules from Rockwell Automation / Allen Bradley, "old" setups will continue to be the backbone of data-acquisition and actuation.

   However, while short distance communications pose little security threat (as they are confined to a controlled environment), the spread of the data in computing systems must be encrypted and more so if the data is transferred across networks.

   Reconfigured to i40 the above mentioned hardware will have a simpler role, responding to 2 needs:

   - *fast DAQ* – the traditional data processing routines can be transferred to 1 or more local SBC's that perform also encryption / decryption. The DAQ-boards may be allowed to just handle data-fluxes, not to flood buffers.

   - *secure data-transfers* – the SBC's (digital)-signal condition, process, compress and communicate the data with central architecture over standard protocols, for which security issues have been solved reliably and an ample pool of experience exists internationally – for in-stance no Stuxnet infiltration.

To exemplify these concepts, I show in figure 5 a data conditioning capability for an ECG-wave project I worked on, for which electrical shielding was not available due to the nature of the DAQ setup.

I performed the data conditioning (removal of 50 Hz hum) digitally, upon data-acquisition. Hardware wise this would be difficult because high-Q factor filters at these frequencies require bulky inductors, and active filters require multiple-pole circuitry (with more noise and signal phase-shift).

While such a task may be simple in words, it is in reality highly demanding in CPU power, which would inadvertently be leaked from the DAQ board CPU capability (as many boards today tend to have on-board processing capability), resulting in data-shipment blackouts.

Due to the high-quality of the digital filter (using Fourier space apodisation, allowing for precise cancellation of spectrum-leakage terms), the signal (blue) is very well recovered.

This happens in spite of the evident sparse sampling of the DAQ system.

It is also worth noting that in this case digital data-conditioning performs significantly better than a hardware analog signal-conditioning circuit.

This type of signal conditioning is possible when the AD converter of the DAQ board has a high dynamic i/p range (16 bit or higher).

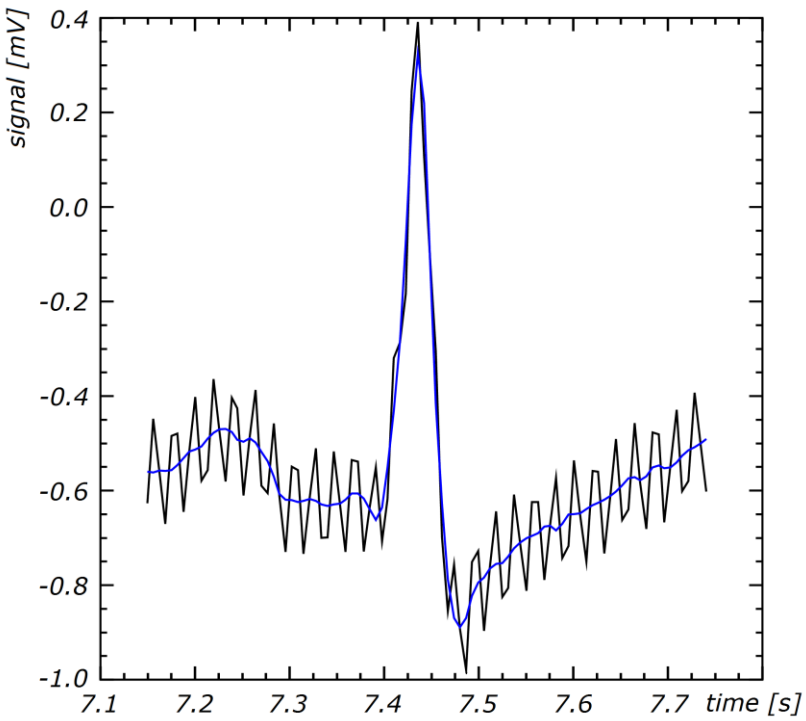


Figure 5 – ECG wave digital data-conditioning: (black) with 50 Hz hum, and (blue) digitally filtered with my *FoxLima* package.

For the setups for which this is not the case I offer *pure analog electronics solutions* – also of outstanding quality.

In the same vein, figure 6 shows an example of a fast decision data-acquisition mechanism that I designed. It is a flash-algorithm (used as trigger for data-acquisition). Figure 6 shows the CPU time consumption for each step and the load for each section of code.

The end-result of the trigger is seamless data-shipment (concurrently with data conditioning, compression and encryption), without data-shipment blackouts.

In terms of secure data-transfers, hosting security on so many low-level nodes is indeed a *security-distribution problem* – for which I field centralised update and control solutions.

This allows higher *DAQ bandwidth availability* for subsequent upgrades in data reading speed.

Unlike IOT-SCADA, I do not adopt a (virtual) data-model, rather opt for *Neo4j like data-scalability*, as I am convinced in the future this solution will bypass bandwidth problems.

2. **Decision systems** –data wired from "under the hood" of a data-acquisition system needs to be further processed on central computing platforms, performing analyses that flag into control algorithms, or simply provide analytics for human-factor decisions (such as

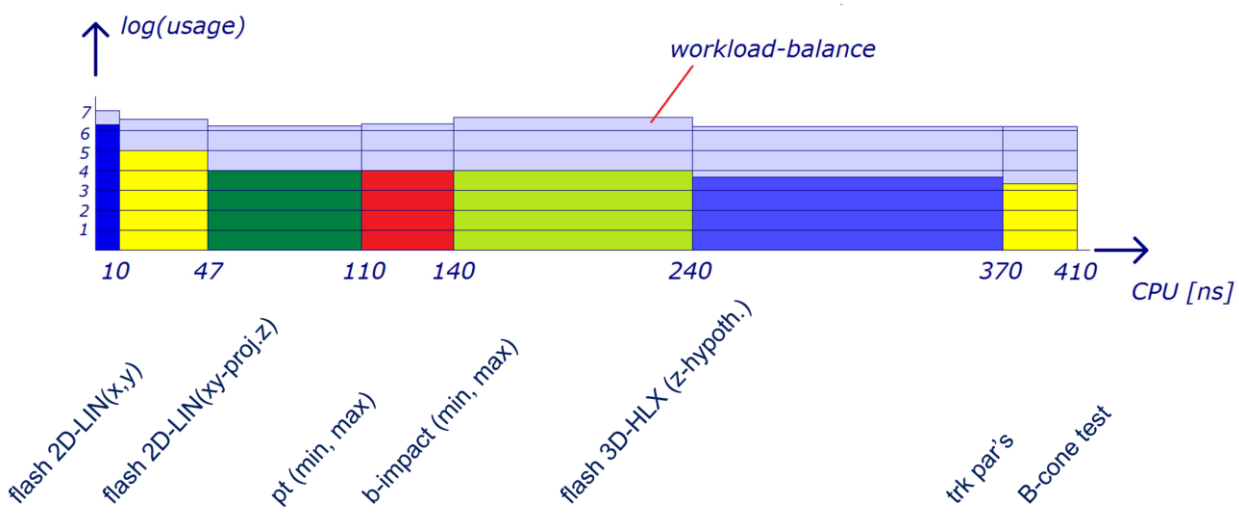


Figure 6 – structure of one of my trigger flash-algorithms, by section. Note the good homogeneity of the final work-load (in ciél-blue).

maintenance, spare parts, replacements). On the same platforms data collected from Sales & Marketing / Purchases provides likeso analytics for human decision.

3. **Artificial Intelligence systems** – unlike SCADA algorithms, which are typically machine-oriented (register / CPU level), the analytics of high-level decision making relies on high-level algorithms, often Artificial Intelligence assisted.

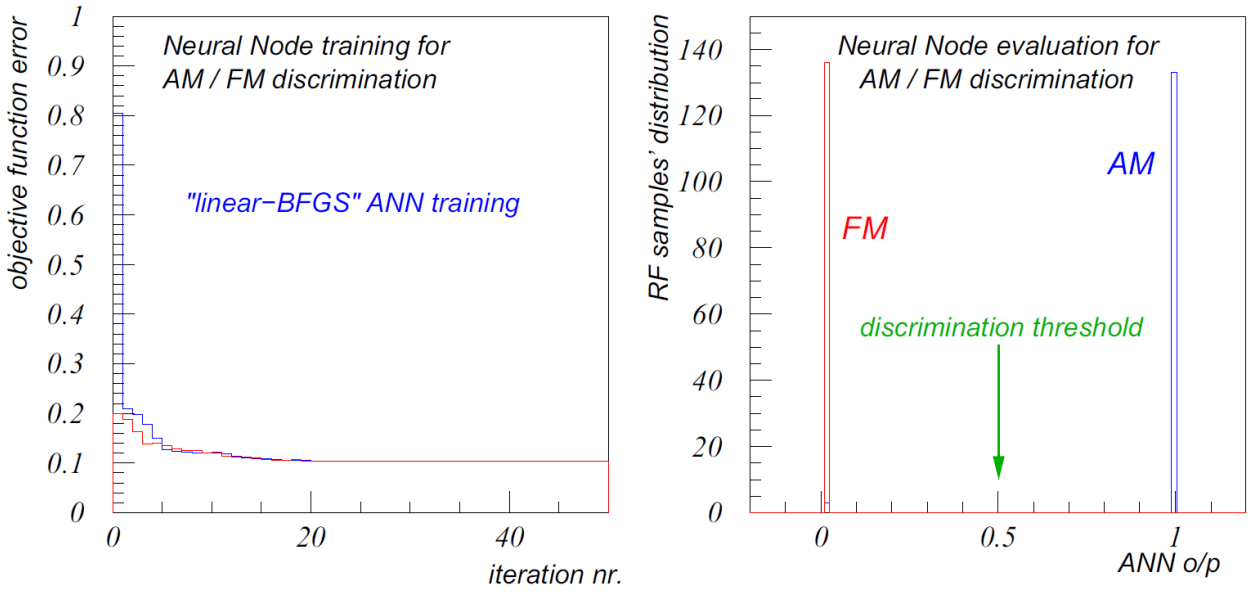


Figure 7 – training (left) and performance (right) of a neuro-soft classifier for AM vs. FM RF-modulation type, used in one of my SIGINT (field)-applications. Note the near perfect discrimination with only a single AM packet (blue) identified as FM (red).

To give a flavour of such an algorithm I showcase in figures 7 & 8 – AI classifiers for automated RF modulation type recognition used in one of my SIGNIT (field)-applications.

The classifiers are presented intermediate frequency data that they need to classify into AM, AM-LSB, AM-USB, FM and PM type modulation. I trained binary classifiers that distinguished between pairs of modulation types, which fed into a master network that flagged the appropriate type.

While the AM vs. FM case is "easy" (figure 7), the AM-LSB vs. AM-USB is known as particularly difficult (from traditional stochastic-moments classifiers), the two being essentially the same modulation. Yet, as seen in figure 8, the neural classifier distinguishes the two (to within ca. 95% accuracy), which is remarkable for this set !

I field the same top notch neuromorphic decision-making algorithms for any other technical, market, or supply chain analysis.

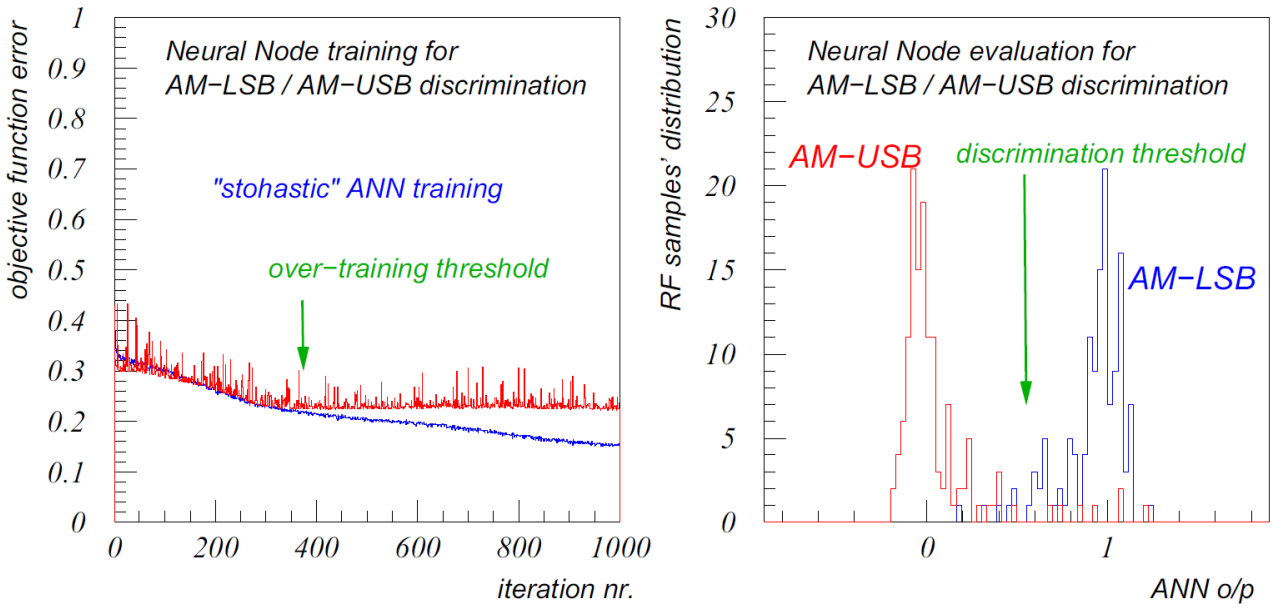


Figure 8 – training (left) and performance (right) of a neuro-soft classifier for AM-USB vs. AM-LSB RF-modulation type, used in one of my SIGINT (field)-applications. I used stochastic training for this particular application due to its very demanding nature. Note that these are essentially the same RF modulation, traditional classifiers used by other groups (statistical moments) failing with congratulations. My code's performance is in the 95% range.

Communications

Industry-4.0 applications may contain sensitive data / control commands, hence i-IOT needs dedicated lines and like so, performant encryption.

In this direction I designed the SXV4 package for remote-process communications, which brings the very detailed technicalities of socket communications to a user-friendly level – for the client-server paradigm, see figure 9.

From the `skt` base-class are derived the classes server and client implementing the communication protocols (see figure 9).

On the basis of the classes server and client I implemented practical applications (such as asynchronous TX / RX in the aplic_server and aplic_client applications). For this mode of operation the code interacts with the system's exception handling, for which I implemented an except C++-class. Although naively thought as a connection between 2 computers, this type of software can also link 2 processes on the same computer, as well as between the computer and itself (via its eth interface however).

Encryption

With respect to performant encryption, the highest to date encryption standard is the *Advanced Encryption Standard* (AES, FIPS-197-NIST). I designed an AES encryptor / decryptor (AXV4) acting stand alone, or in conjunction with my SXV4 package (figure 9), for hopping-ports full-duplex (much like military grade hopping-frequency transmissions).

Figure 10 shows the timing performance of my AXV4 package implementing AES.

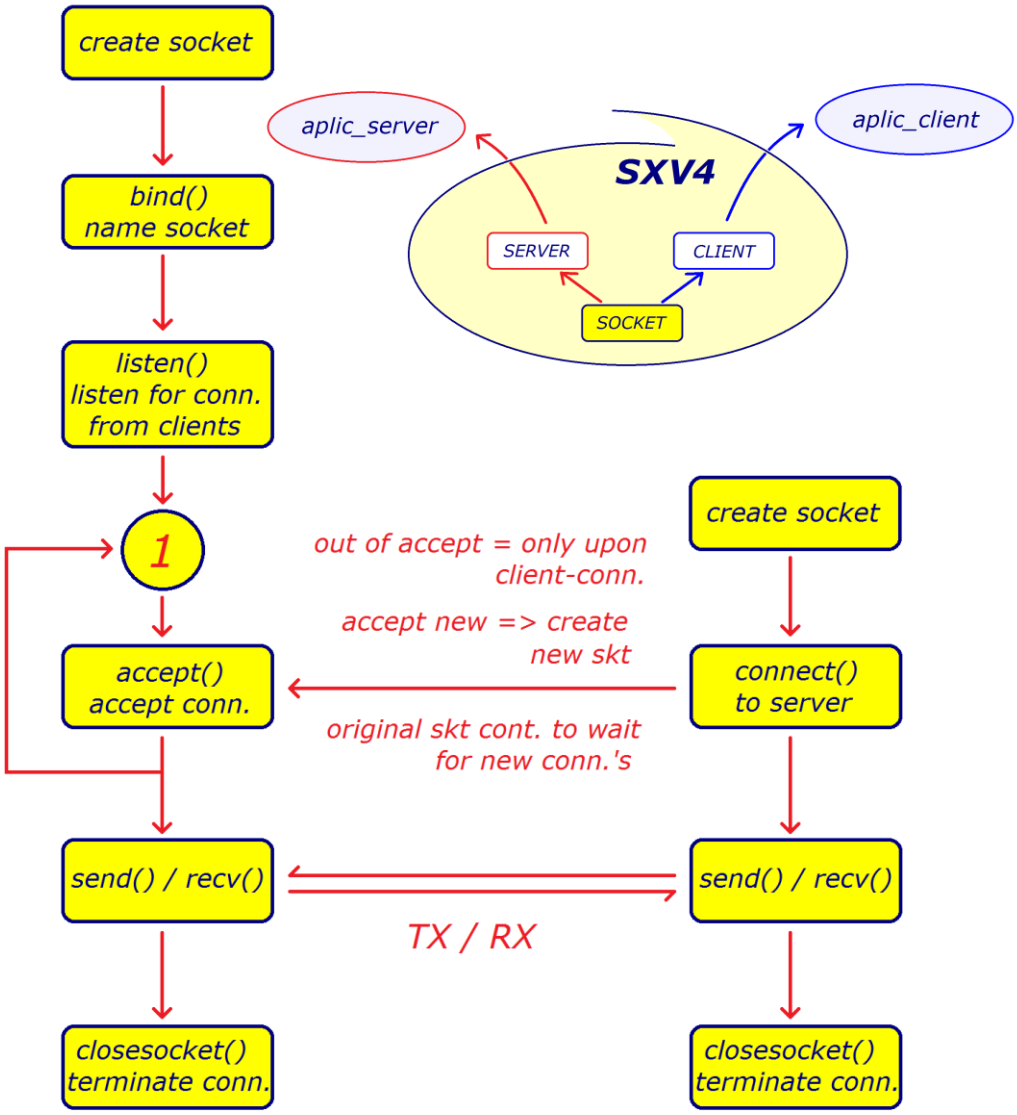


Figure 9 – schematic of my *SXV4* sockets package, which I used in conjunction with AES encryption for hopping-port full-duplex (much alike military grade hopping-frequency transmissions).

The problems with AES are typically practical implementation issues – which I solved under my proprietary AXV4 (C++ package), such as against side-channel attacks (by using tabled functions and eliminating functions whose exec time depends on data and could reveal by exec time the encryption mechanism).

SSoT distribution within an i40 enterprise may not always be throught i-IOT, sometimes the data being distributed via wireless communications. Such comm channels are known for eavesdropping and hacking problems. Nonetheless, sotimes industrial processes and floor-geometry may restrict communication to RF communication, case in which a reliable wireless encryption is necessary.

Although digital-encryption is available, as already stated, this is prone to eavesdropping and hacking, hence a physics-based protocol would be preferable. The reason for this is three-fold:

1. compression is needed at the same time with encryption;
2. the coding principle must be new, with no current hacking scheme, nor a foreseeable one;
3. the transmission must not be jammable.

In this respect I developed the method of *Maximal Phase-Space Compression*, which is a sequence-by-sequence maximal compression in an intermediate-space ("rotated") between real-space and Fourier-space.

Given the small size of the encrypted data, the signal can afterwards be spread in spectrum, to both lower its intensity (detectability) and towards making it less jammable (by spreading it in the spectrum).

The insert to the right details the conclusions of my study.

5. Signal Compression

Both classical and quantum signals contain information that behaves "quantum" in nature. Examples thereof would be a 2D laser field at its focal plane, for the quantum, and an RF signal (in 1D), for the classical. In both cases the "quantum" nature is evidentiated by the (complex) quantum correlation coefficient α. This coefficient allows the design of a "rotated" Fourier transform (between real space and Fourier space) in which the signal is best compacted.

Said compactification brings also non-statistical encryption of the signal, rendering obsolete traditional code-breaking methods [5-10].

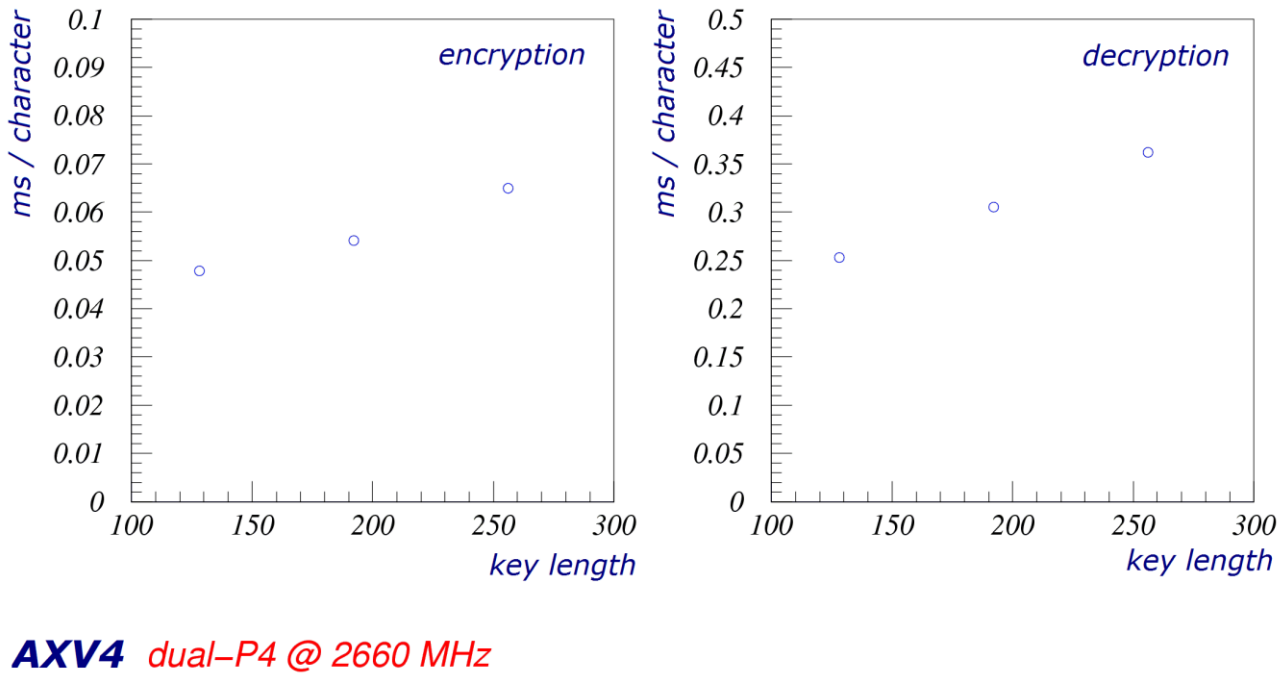


Figure 10 – timing performance of my AES encryption package *AXV4* for different key lengths.

Special hardware

Single Board Computers (SBC's) – more ubiquitous by the day, SBC's are replacing the last 2 decades' hardware of DSP's and embedded systems with reliable, standardised, cost-effective solutions packing a big IT-punch.

Today's high end automobile is vested with order of 0.1 billion lines of code, figure expected to hit 0.2 - 0.5 billion in the next decade, with autonomous driving and "*smart roads*". Compare this to the Space Shuttle with 0.0004 and the B-747 with 0.004 billion.

In this respect the paradigm of embedded software (and afferent electronics) is rapidly changing from dedicated assembler (or in certain cases C-asm) to ARM systems.

SBC's do not communicate by dedicated, expensive LOCs (manually crafted - debugged, delayed), rather with standard, internationally tested, protocols and security measures of reliable and quantifiable quality.

My partner's experience in ARM sys-admin (resource organisation and network) will bring the project building speed and reliability that your company needs.

Communications hardware – to date a number of non-standard communication channels exist in industrial environment, depending on the requirements set by the perturbing factors. Apart from optical fibre, which is preferred, in cases of remote accesss IR / VIS communication seems to be taking ground, as well as old workhorses such as ZigBee (wireless IEEE 802.15.4) and WLAN (wireless IEEE 802.11). Other times electrical networks are used – of which I showcase here a test board for evaluating the level of noise perturbation on an LVDS (Scalable coherent interface extension IEEE 1596.3-1996) communication channel – (the noise simulated with a Zenner diode, then amplified).

I offer *30 years* of electronics experience to approach any optical / electrical channel type of communication and securing it to standards (signal conditioning, signal processing, communication channels, micro-controllers).

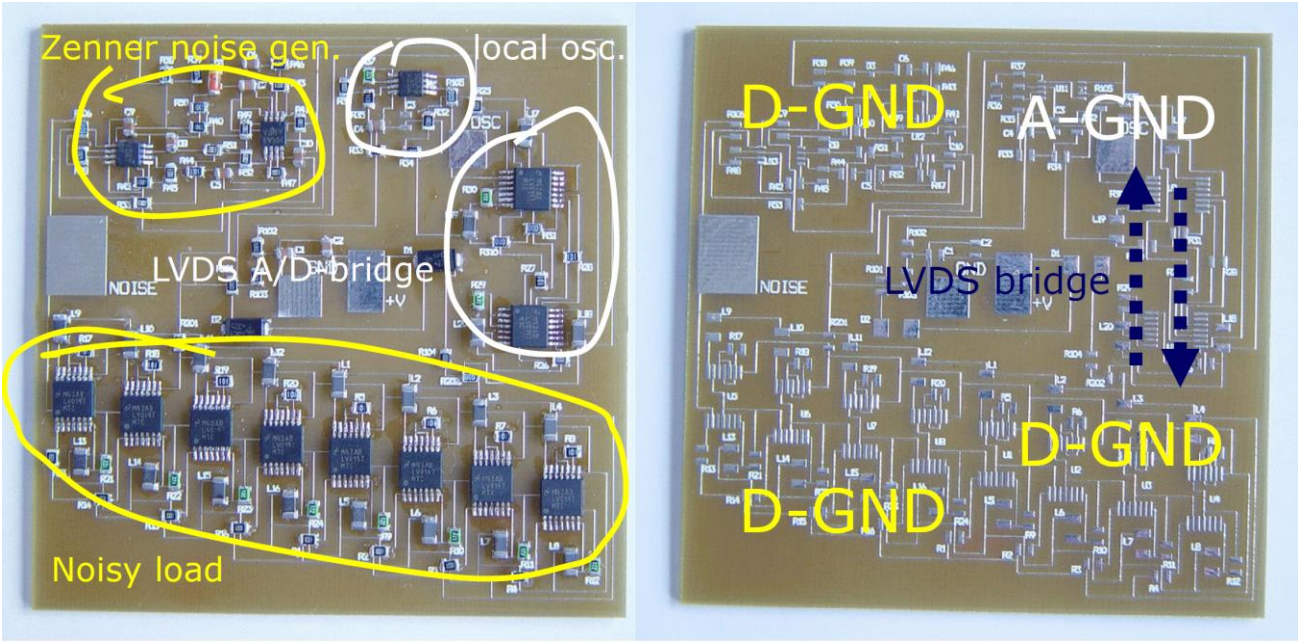


Figure 12 – custom design test-board for special LVDS communications under noisy digital environment.

i-IOT & Cyber-Security

- protocols, transports, VPN connections, wire/less authentication
- network management (DNS / DHCP)
- VLAN's and port trunking
- attack rejection and monitoring software

For more details visit: http://horia.pw/?p=me

Personnel training

On a per availability basis, the above chapters can be covered under *personnel training*, within an academia-industry partnership programme for C++ training.

For the structure of the *Applied C++ programme*, please visit my institutional page:

http://cern.ch/modima/WXX