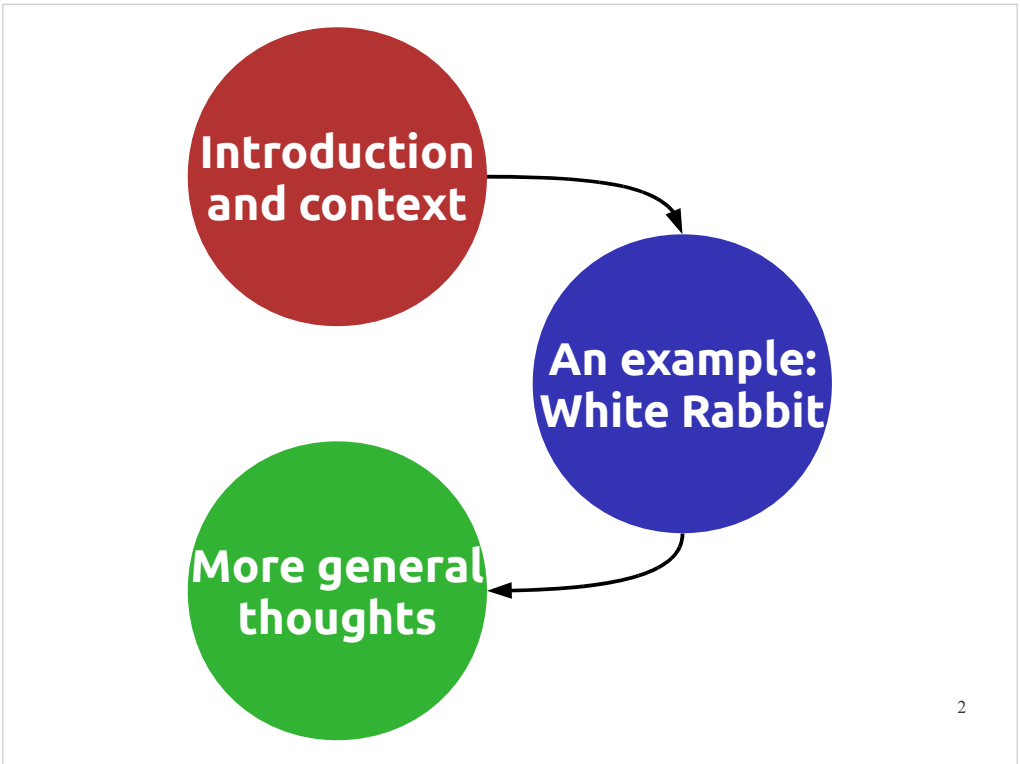
Although the OHR and the philosophy behind it are well known and endorsed by management and the Knowledge and Technology Transfer group at CERN, there are some parts of this talk which delve into domains such as how can OH be beneficial for society as a whole for which:
a) I am not mandated to represent CERN.
b) People at CERN involved in this might have different opinions.
I am therefore here representing nobody but myself, i.e. I am not on an official trip as far as CERN is concerned. I am on holidays.
Having said this, CERN is indeed a very cool place where the dissemination of information primes above economic (or other) considerations and all necessary conditions for the birth of OHR could be found.

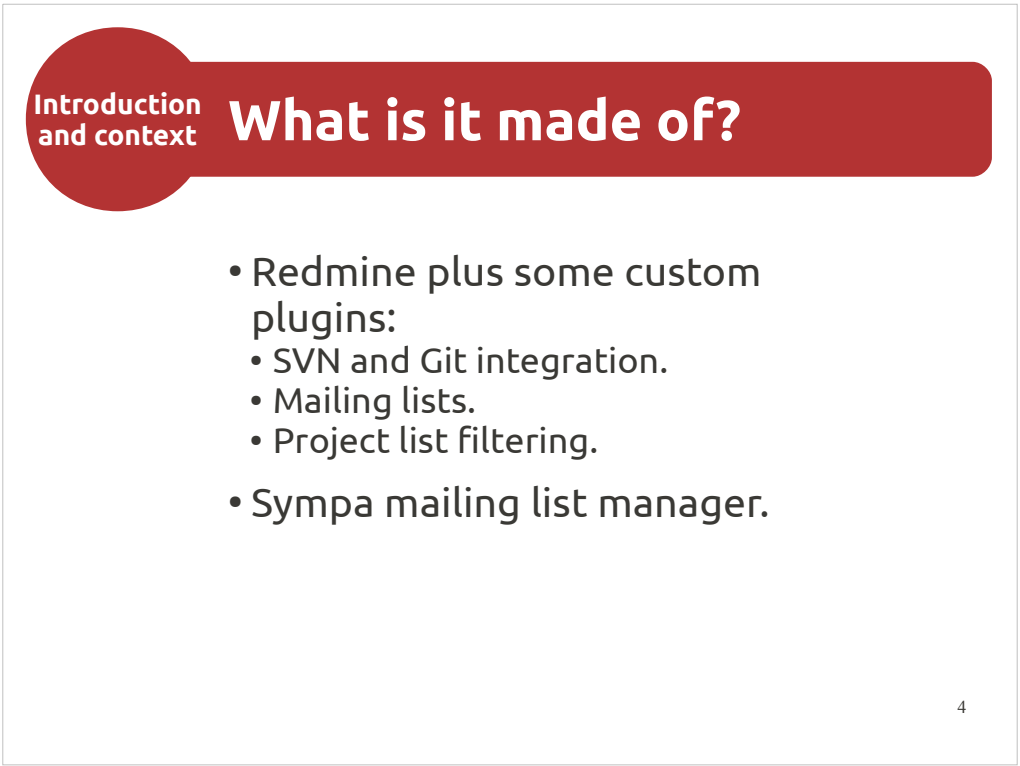


We will start with an introduction to the OHR, where it sits, what it is, why we are doing this and how we are doing it.

We will then move on to a description of the biggest project so far in OHR: the White Rabbit network. This project aims at achieving sub-nanosecond synchronization of over 1000 nodes linked by typical fiber lengths of 10 km using an extension of Ethernet.

Finally, we will go through some considerations on whether this effort we initiated at CERN can be of any use to society at large.

All three-letter combinations for domain names are taken by (mostly) speculators so we used hw for hardware, although we still say “OHR” to refer to the repository, and not “OHWR”.



It could not be otherwise: the OHR is fully built using free software. The plugins which were developed are also completely public and free.

Our OH concept revolves about these three freedoms granted to the user:

- The capability of criticizing the design. This would typically be granted through a pdf of the schematics, and possibly some documentation.

- The capability of modifying a design. This means the source files for whatever EDA tool was used. Unfortunately the best tools are neither free nor free, but it's important to publish these source files anyway in order to allow users the possibility of modifying the design if they have the appropriate EDA tools. More later on the tools problem.

- The capability of manufacturing the hardware specified in the design files. This includes pick & place files, gerbers, BOM, etc.

With these three freedoms a user can take a design without having to build any unnecessary dependencies with respect to the developers. This seems very natural in the SW world, but it's unfortunately very rare in hardware.

Designs not satisfying these three conditions cannot be hosted in the OHR.

It looks reasonable to say that if a PCB is designed in a public institution its contents should not be held secret from the people who have paid for it, i.e. the tax payers. Granted, the design has been payed for by people from a certain country or group of countries, but there is today no practical way of publishing only for the American or the European people for example (if someone ever thought that would be a good idea). So designs performed in a public institution should be universally published unless there is a very good reason not to (like in the military).

Also note there is a difference between being public (put in the public domain like CERN did with the web) and being freely accessible for research purposes. Some people defend the latter and do not agree with the former. I don't quite see the problem with the former, but I must say this is a controversial issue and I don't represent CERN in this.

Governments could have an objection about funding open projects which can benefit companies from other countries. Our answer is: make sure your country has well-trained, skillful companies to quickly take advantage of this R&D. The advantage is always in the hands of the technology initiators.

This is well known for FOSS. It is not as easy in the case of hardware for several reasons:
- Well-commented code is sometimes self-documenting. HW designs need separate documents, and sometimes designers don't take the time to do it. Sharing forces you to do it (otherwise a schematic design is like encrypted code) and therefore to be a better designer.
- The amount of competent HW designers is nowhere near the amount of competent SW developers.
- It's not in the culture of most HW designers to share. The gain is not clear because there are no visible products (yet!) of how everybody wins when people play this game.

But even if the evidence is not as overwhelmingly compelling as in the case of SW, it still remains true that by publishing you increase the chances (however slim) of getting good feedback about your design. These chances will hopefully grow in the future.

Introduction and context

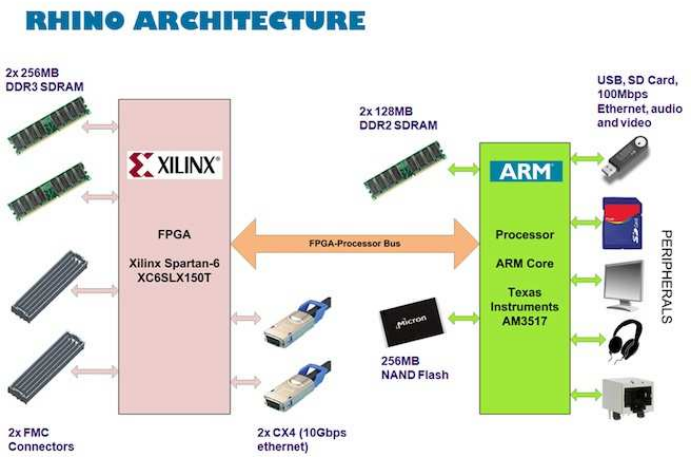
Reasons for OH (3)

Avoid unnecessary multiplication of effort

It's incredible how many design teams only in the High Energy Physics labs community are working right now on a 100 MS/s 14 or 16-bit ADC board. Many of them are probably making the same mistakes. One of them has the best SNR, another one has the best linearity, maybe one is best in all respects. In that case, all groups except one are wasting their time and could be doing something else, extra stuff to make the design even better, software support, etc. For not-for-profit organizations it would make a lot of sense.
To my great surprise I have been confronted with some very strange arguments for not publishing even in this context: some people seem to believe that secrecy is good to justify big development teams and say that their teams would be cut if their work was shown to be redundant with that of others, as if there was a shortage of things to do!

Introduction and context

An example: RHINO



The RHINO project is a perfect example of avoidance of multiplication of efforts. By publishing in the OHR, many people interested in Software Defined Radio can have a common platform and focus on gateware and software development. It is also a great teaching and learning tool.
The project is led by Alan Langman in the University of Cape Town. It was the first non-CERN project to join OHR. We hope the first in a long list!

Reasons for OH (4)

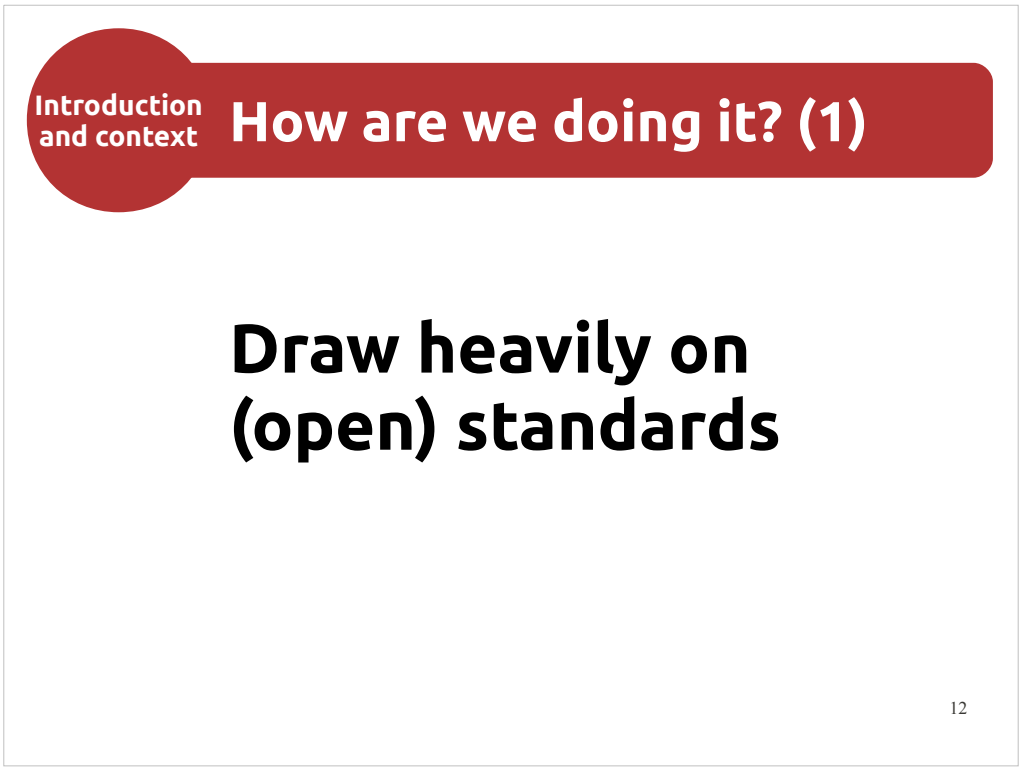
Better relationship with companies

As for everything in life, there are better companies and worse companies. There are also unforeseen difficulties for good companies. For clients it is sometimes problematic to be tied to a company through a dependency created by non-open design. Companies should be selected only on the basis of their competence, added value, good support and reasonable pricing. These conditions are best fulfilled with open designs. The notion that a company selling open hardware can make money off activities like support, testing, guaranteeing, etc. is largely inspired by the example of Linux and Red Hat (and others). Red Hat is one of the most profitable companies in the US, yet its product is completely open source. Red Hat clients are usually very happy and have no reason to stop their relationship with Red Hat. However, if there is a sudden change in policy, or Red Hat proves unable to sustain the level of support provided so far, clients have easy ways of looking for alternative support houses. This ensures that ultimately the client can never end up locked to a company for the wrong reasons.

Reasons for OH (5)

Fun!

It is this presenter's modest experience that having fun while designing invariable results in better designs than being miserable during a project. And designing openly is certainly more fun than doing it in a closed environment. You can get to know many interesting people with very useful ideas just by publishing your work. While this is not the original goal of any project, it often influences it for the better.



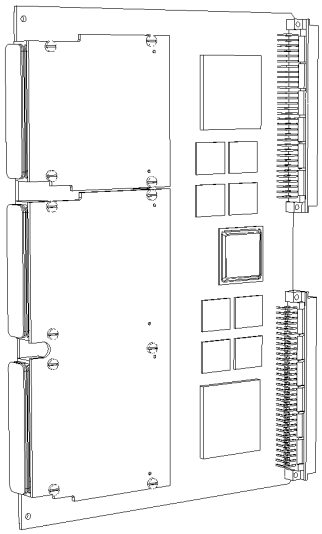
We have taken a number of strategic decisions, some conscious some more Darwinian:

I will describe three such choices: Ethernet, FMC (aka VITA 57) mezzanines and Wishbone.

- Ethernet was an obvious choice for White Rabbit. At the beginning it just sounded right. Now we know it was a very clever decision. We can test and diagnose our switches with standard software. We can buy cheap components and we know that the technology will still exist in 20 years.

Introduction and context

Carrier/mezzanine split



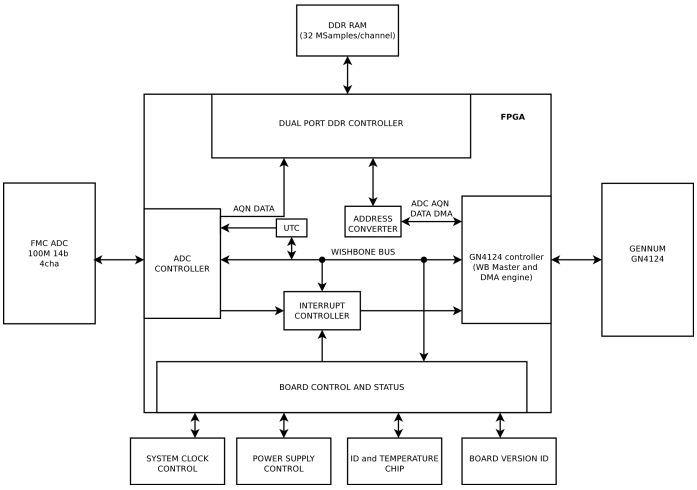
Courtesy of VITA

13

A carrier/mezzanine split allows re-use of carriers and mezzanines. FMC is the only standard which fits our needs, so it seemed to be a natural choice. Another big advantage of the carrier/mezzanine split is that people do what they do best: analog designers design analog mezzanines, digital designers design carriers and digital mezzanines. BTW, we suffered recently from a company which was good in analog design but struggled with a digital interface. Since the design was not open we could not help reduce the delay. This was bad for us, but also for the company. A carrier/mezzanine split combined with open design is good for small and medium design companies.

Introduction and context

Wishbone example



Wishbone is the choice of opencores.org for internal FPGA inter-connect. We have:

- Developed a wbgen tool to automate slave generation in VHDL, Verilog and C header files.
- Helped in the modification of the WB spec to better support high-latency high-throughput peripherals such as DDR RAM.
- Tried to cast all our FPGA designs into WB blocks.

Here's an example of a WB-based FPGA design. WB slaves for the DDR RAM and FMC control are re-usable in other designs. Same thing for the PCIe-WB bridge (WB master). Also, if we want to replace one block by another (say the PCIe-WB bridge by an Ethernet-WB bridge aka Etherbone) it is pretty simple. WB might or might not be the best possible interconnect, but it satisfies our requirements in terms of openness and it is a standard. It is also the choice of the opencores.org people.

How are we doing it? (2)

Push drivers to official Linux kernel

This is not yet happening but it is definitely in our vision. Some of our developments can be used in combination with a Linux box (e.g. our PCIe boards in an industrial PC). Others (e.g. the WR switch) contain embedded Linux systems themselves. Aligning with the official kernel has many advantages I am not going to explain to this audience. One of the options we would like to explore in the future is giving WB the status of a recognized bus in the kernel. Our WB slaves could then have kernel modules supporting them, and this code could be re-used from system to system. How realistic and effective this would be remains to be seen, but it looks like a natural trend to follow: FPGAs have more and more gates. What used to be a PCB is now a WB slave block inside an FPGA, so why not give it a driver?

How are we doing it? (3)

Working with lawyers for an OH license

Hardware is not like software. Copyright is easy to circumvent: just change names of resistors, shapes, etc. For better or worse, protecting ideas (instead of expressions of ideas) is the realm of patents. Copyright just protects expressions of ideas. There are some interesting efforts on the licensing front for OH, like the OHL of John Ackerman, which aims at creating a patent-free area by specifying that the licensee of an open design shall not sue the licensor for any kind of patent infringement related to that design. This license is “persistent” or “reciprocal” as GPL is, i.e. modified versions of the design must be licensed through OHL as well. Other people are using BSD-type licenses (non-persistent). The Creative Commons family of licenses is also popular. Current thinking at CERN revolves around a persistent copyright-based license with no provisions about patents. This is still a work in progress, but we need to converge fast because some of our designs are already in the prototyping stage!

Dealing with patents is a very hairy issue. One could say that by publishing designs in OHR one is making the life of patent trolls easier, i.e. you can get sued by people who hold a patent you did not know existed. These days there are patents for every conceivable piece of hardware. The patent system is completely abused. Long gone are the days when patents primarily served the purpose of fostering creativity. At this point in time OHR is not big enough money-wise for patent trolls to care about us, but this could become an issue in the future.

How are we doing it? (4)

Collaborating with companies

Companies are a key ingredient of our OH concept. They can be our partners in two ways:

- As developers: a company can be paid to develop OH. This has been tried at CERN and works very well.

- As commercial partners: a company takes designs from OHR and sells them with added value: manufacture, test, guarantee, support... We plan to be clients of such companies for our own designs. Some companies have already shown interest in this approach.

An example: White Rabbit

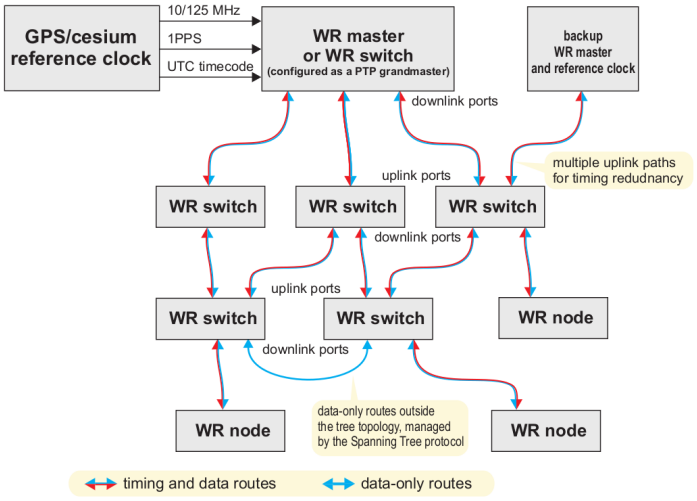
Project goals

- Sub-nanosecond sync through Ethernet.
- Upper bound in frame delivery time.
- ~1000 nodes.
- ~10 km fiber links.

WR is a multi-lab multi-company project to deal with the problem of distributed node synchronization at the ns level in an Ethernet environment. At this level of precision, custom switches are necessary. Switch development is currently the main focus of the project. The project aims at synchronizing up to 1000 nodes with typical fiber lengths of 10 km (CERN site). This technology has applications in all sorts of distributed control and data acquisition systems: telescope arrays, particle accelerators, power grid monitoring, etc. Even projects which don't need the ns precision can consider WR instead of other solutions (e.g. NTP) if they need a complete guarantee for an upper bound in terms of time difference between any two nodes. Another important aspect of the project is achieving determinism through the definition of several traffic types with different switching priorities.

An example: White Rabbit

Network overview

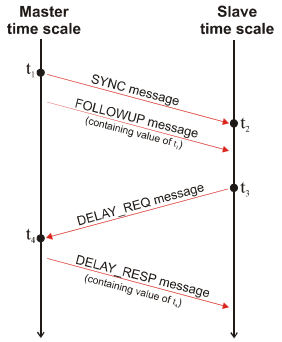


19

A WR network is hierarchical timing-wise and flat data-wise. The first switch in the hierarchy recovers its clock from an external clock source and uses it to encode all data going to its downlink ports. The next switch layer will sense data in its uplink port(s) and extract their clock from that data. Through this mechanism, syntonization is achieved throughout the network. For synchronization we still need to account for fiber or cable-induced delays (see next slide). Data-wise it's nothing different from a normal Ethernet network. We plan to implement 802.1q support for VLANs and prioritized traffic. With the combination of perfect synchronization and determinism we can take advantage of new ways of performing control actions in a time coherent way, like sending a message to all nodes at 11:59:59.999 saying “Take acquisition data at 12:00”. A whole millisecond is taken to account for (guaranteed) worst case delay, but this delay does not need to be the same for every node in order to guarantee coherent acquisition.

An example: White Rabbit

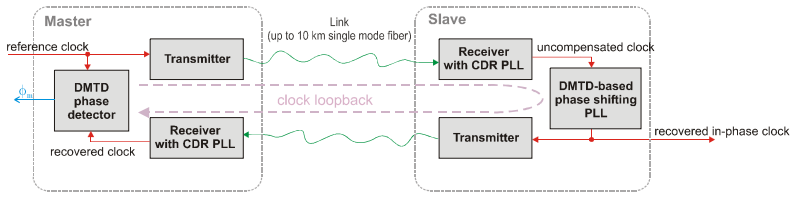
Normal PTP



PTP stands for Precise Time Protocol (aka IEEE 1588). Its main purpose is to evaluate transmission delays in a network in order to establish a common distributed time base. This figure illustrates the four time tags needed by PTP to evaluate transmission delay. PTP is itself an improvement over NTP to have a system where the assumption of symmetric delay is closer to reality (by time-tagging in hardware instead of calling gettimeofday).

An example: White Rabbit

Enhanced PTP

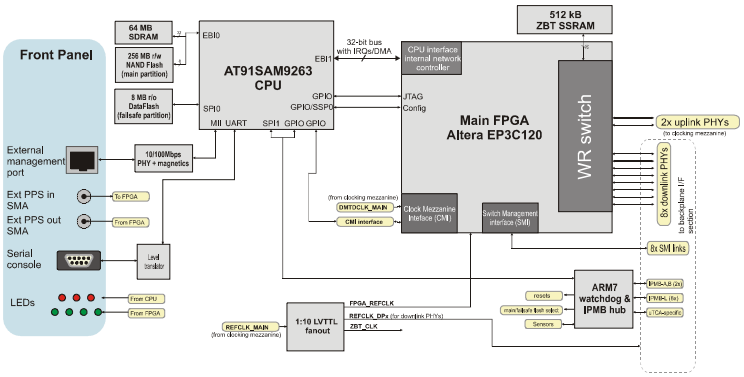


21

In a typical PTP implementation, local oscillators are free-running, making it necessary to send PTP messages quite often in order to keep the timebase from drifting too much, i.e. we have a trade-off between determinism (possibly put at risk by too many PTP messages) and quality of sync (improved by more frequent PTP messages). WR simplifies this situation by recovering the same frequency in all nodes. Now PTP exchanges can take place much more infrequently since they only have to compensate for fiber (and copper) delay changes. These are thermal effects, so a PTP exchange once a minute or so is enough. This means for all practical purposes 100% of the network bandwidth is available for the application at hand. Another nice consequence of syntonization is it opens the door to phase detector technology as a means of measuring delay. Since phase measurements are much easier and precise than time measurements, WR technology can easily fulfill the 1 ns spec.

An example: White Rabbit

WR switch main board



The main switching function resides in the Cyclone 3 main FPGA, helped by a low-latency external RAM for MAC-matching lookup.
For more complicated things (RSTP, SNMP, etc.) we use an embedded Linux system hosted in the ARM9 CPU. This includes a PTP daemon which has recently demonstrated PTP compliance in a plugfest (ISPCS 2010).
The ARM7 serves in principle as a watchdog (to avoid ever losing contact with the system) and also as an I2C hub for implementing management through IPMI (not done and not needed for the WR switch).
Fine timing management is done in a separate board.
This version (v2) of the MCH is the one we are working on for VHDL and software development so as to have a basic-switching implementation by the end of 2010. The next version (v3) will probably be Virtex-6 based and host 2 uplinks and 16 downlinks.

An example: White Rabbit

OH experience so far (1)

Project staffing much easier

We started in a collaboration with GSI. Then a couple of Spanish companies showed interest and applied for public funding to work on WR, which they got. Our friends in NIKHEF were working on fixed-latency Gb/s FPGA links for a neutrino telescope and joined the effort. There seems to be no end to the supply of “free” manpower, to the point that the bottleneck is often the management of all this rather than raw manpower.

An example: White Rabbit

OH experience so far (2)

Peer review working very well

This is project-internal (but CERN-external) peer review, and is linked with the capacity of this project to be of interest for many people, so potential and actual users/developers are the main source of reviewing. The WR mailing list has open archives, and the project is fairly well documented in OHR, so it's easy for people to join and start providing useful feedback quite rapidly.

An example: White Rabbit

OH experience so far (3)

Many companies interested

25

We have one company working on switch hardware and another one working on switch embedded software. Both of them will be ideally positioned at the end of development to start selling switches and/or using them for turn-key solutions. Another major company is starting development of a WR node in PXI format. Same thing for a WR time server (a box with GPS in and WR Ethernet time out) in the plans of yet another company.

Our attitude wrt black boxes that plug to our hardware is very liberal and anti-authoritarian. People should be allowed and able to design what they please: black, grey and white boxes. Customers should also be free to decide what they want to buy. We have a tendency not to buy closed designs for the reasons explained above, but that does not mean we despise closed designs.

The FOSS movement started when closed software “got in the way”. The reasons for OH are very similar. Closed hardware has basically the same disadvantages as closed SW: unnecessary dependencies wrt vendors, lack of freedom to understand and modify if need be, etc. Similarly to FOSS, this lack of freedom is most important to people who could actually use that freedom, i.e. designers. The average computer user (whatever that means) is mostly happy with a Windows PC or a Mac. Similarly, the average hardware user is happy with a closed-source ADC or DSP board. Between the average user and hardware specialists there is a gradation of profiles with increasing interest in OH, which we explore in the following slides. This situation is completely analogous to the one for FOSS, once a appropriate scaling factor is applied to the absolute numbers. Notice that for hardware we are limiting considerations to hardware typically used in control and data acquisition systems. We are not talking about toasters and cars for example. I don't know of anyone whose professional life is handicapped by lack of knowledge of the internals of his toaster.
An additional argument to consider FOSS and OH on equal footing is that it is becoming easier and easier to design hardware, and hardware has become more and more programmable itself thanks to HDLs.

According to wikipedia, Linux users represent less than 2% of the total desktop/laptop users: http://en.wikipedia.org/wiki/Usage_share_of_operating_systems. This does not keep it from being the best OS out there. Once your developer community is bigger than your needs or bigger than what can actually be reasonably managed, size is not a problem anymore.

If our sphere of action was only that of publicly funded labs and schools it would already be largely sufficient. Anything in addition will only make things easier.

These two problems are quite real if your sphere of action is the whole world of HW development, but we don't believe that this holistic approach is either possible or desirable.

If we reduce our action field to the (very big) subset of competent, publicly-funded (directly or indirectly) designers, then these two problems disappear. The fact that making HW takes money is a fact of life, completely unrelated to whether the design should be open or not. If you don't have money to build hardware, you are not in the hardware manufacturing community by definition. You can be a developer, but not a manufacturer.

Now that leaves us with a single problem: what about skilled developers outside of this community? Work on FOSS tools for HW design is the only answer I have at the moment. We absolutely need this to unleash these great design resources. But as I said, getting organized on the publicly-financed side would be already a game-changing decision.

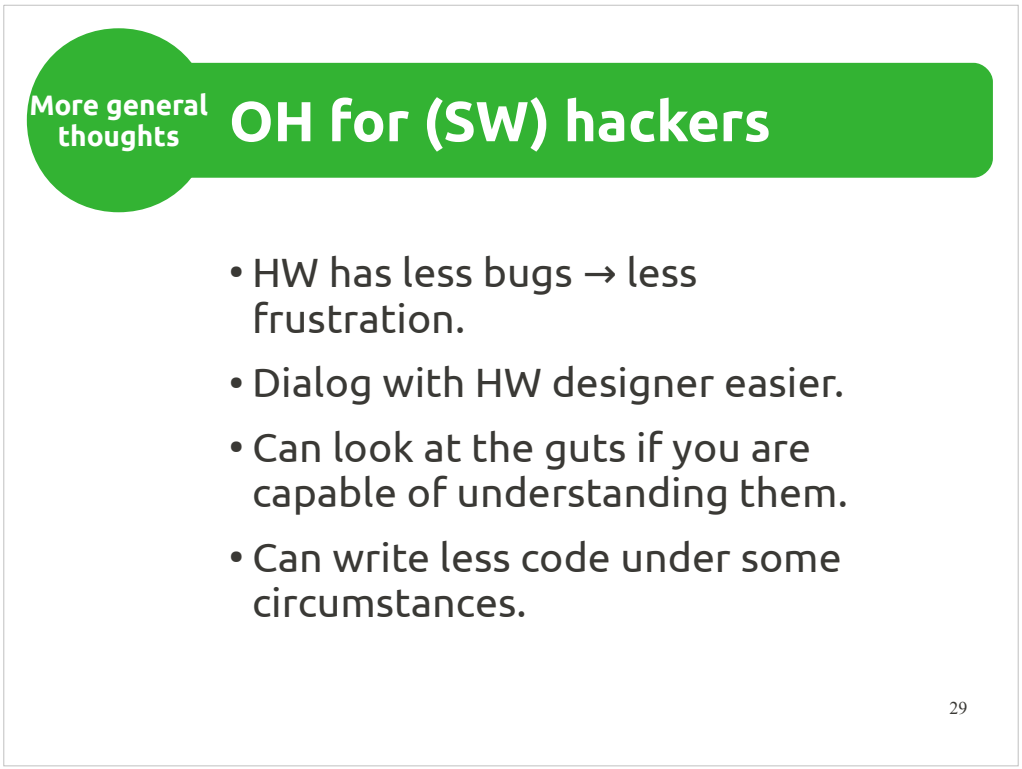
The advantages of OH are most obvious for hardware developers. Here is a quick list:

- Great learning tool for technologies you don't master.

- Avoiding unnecessary work if someone has designed what you need (or almost).

- If your role is to give support to others for that HW (as is ours at CERN), your life supporting black boxes can be quite complicated...

- Having your work reviewed increases its quality. In fact the mere fact of publishing your work pushes you to be better.

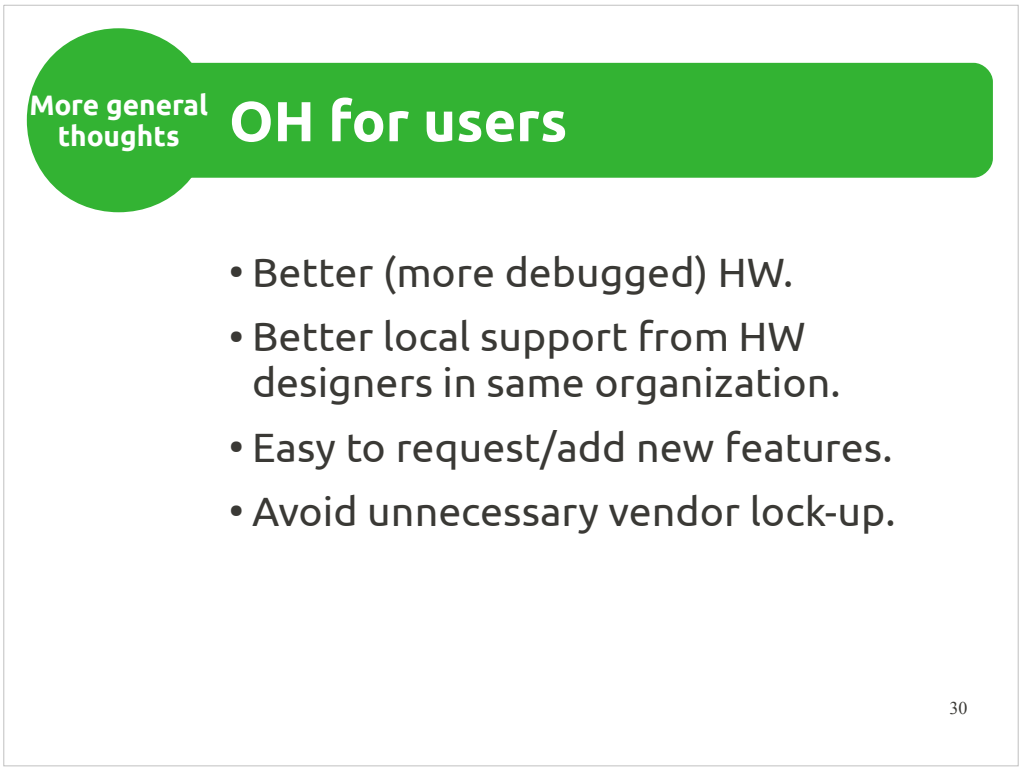


The second group in terms of appreciating the advantages of OH is the SW hackers. Here is an incomplete list:

- Better dialogue with the hardware designers when it comes to develop a device driver for a piece of HW. Less frustration and misunderstanding.

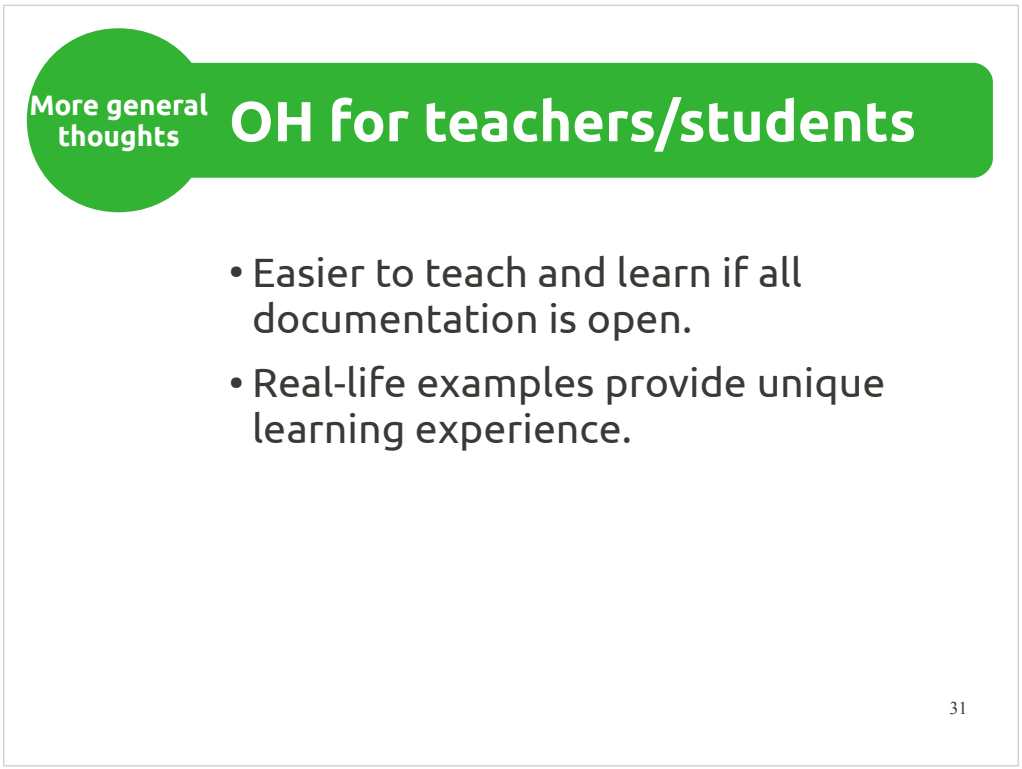
- Less buggy hardware, and bugs easier to find. Many hackers can read VHDL and Verilog, and to a certain degree can understand schematics and PCB layouts.

- Less code to write: if the guts of a system are open it can become apparent that some internal components are the same as for other designs (see considerations on WB some slides ago) and this software support can be re-used. For black boxes, development always starts from scratch. Another mechanism helping in reducing number of lines of code is the reduction in different HW solutions itself, as explained earlier.



By users we mean the engineers, scientists, researchers, etc. who use these devices for their professional activity. They can benefit from better (debugged) hardware, better support from local hardware designers and can request/add new features easily. Concerning quality it could be argued that open design establishes some kind of baseline for quality, since closed products can only be better (otherwise nobody would buy them) at least in some respects (clearly not in openness). The capability of easily adding (or having someone add) features is very important for power users, of which there are many in our type of environment.

Now, what about other users? I can only have a wild guess at the moment, but one of the things OH could do is avoid undesirable vendor lock-up situations. Look at what's happened with cars. It used to be that you could fix your car anywhere. Now you can only do it at places which have appropriate closed SW to talk with your closed HW. This is not in the interest of the final user. Cars don't get designed in public labs, so they are outside of our scope, but you see the point.



It is clearly easier to teach and learn hardware design when there is a wealth of open designs to learn from. Here again, the experience of Linux and how people have learned Operating Systems by looking at the source code of a real-life OS is very inspirational for OH proponents. The RHINO board, for example, can be a great platform to learn about embedded systems in general and Software Defined Radio in particular.

Teachers/students can of course also become users and/or developers.

This is one of the places where OH has quite a lot of potential to improve things. Some international organizations get funded by many member states and are supposed to buy stuff from them in return, respecting roughly the percentages of contribution from each country. The problem is some less-technologically-developed countries might contribute say 2% but there exists no company in those countries with a good enough R&D department for some projects. OH provides a common R&D department for small/medium companies so they can be considered for the supply and support.
National agencies often have a similar problem: they would like to spend tax payer money in their own countries but are often forced to go abroad to procure high-tech hardware. OH can also help in this as explained.
Finally, there is an awful lot of time spent on writing specifications about what these agencies want. If this product already exists in OHR, they can just point potential producers at it and save a lot of time and effort. The perfect competition conditions offered by OH also prevent artificially high prices.

Companies can benefit from OH in various ways:

- They can reap the results of published R&D therefore lowering their entry effort to a given technology.

- They can get help when in trouble, or choose to do what they do best and have someone do the rest.

More general thoughts

How can you help?

- Hardware designers: do OH.
- Hackers: help with drivers, support SW and tools.
- Others: document, test, advocate, buy OH.

This slide is mostly self-explanatory. One important aspect is development tools. Contrary to what happens in FOSS, the people suffering the inconveniences caused by closed HW design tools do not have in principle the talent for programming open design tools. We are collaborating with the Icarus Verilog community to have a high quality mixed-language (VHDL/Verilog) simulator. PCB design tools are another problem. The most promising programs in our view are GEDA and Kicad, but they remain a long way behind commercial products in some key aspects like the PCB layout tool. With the help of SW hackers, this very important limitation for OH can be tackled. The lack of free tools is indeed one of the biggest problems for OH to take off at a global scale.

Conclusions

- Effort justified even inside CERN alone.
- Looks like it's taking off though!
- Exciting times ahead both for designs and tools.

Thanks!