Peter the Great St. Petersburg Polytechnic University

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[Institute of Computer Science and Technology](http://english.spbstu.ru/structure/institut_computernikh_nauk_i_tekhnologiy/)

Department of Computer Systems & Software Engineering

Lecture

*Introduction to Qsys*

student:

Ivanov Il’ya

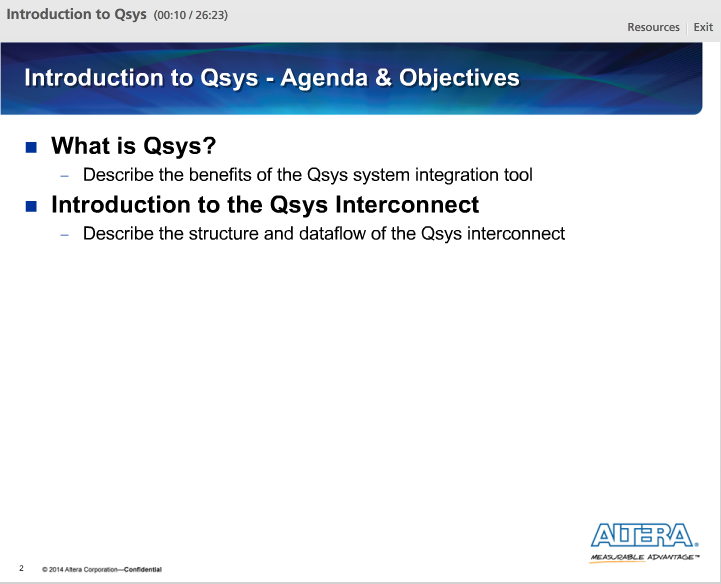
group: 3530901/70203

lecturer:

Antonov A.P.

Saint-Petersburg

2020

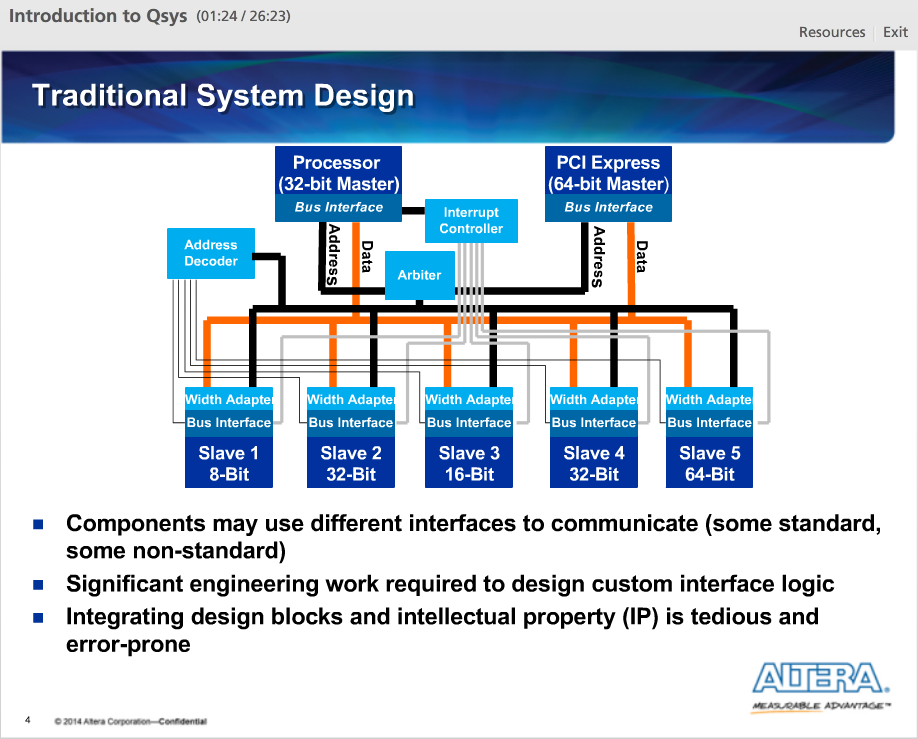


There are two main topics in this lecture:

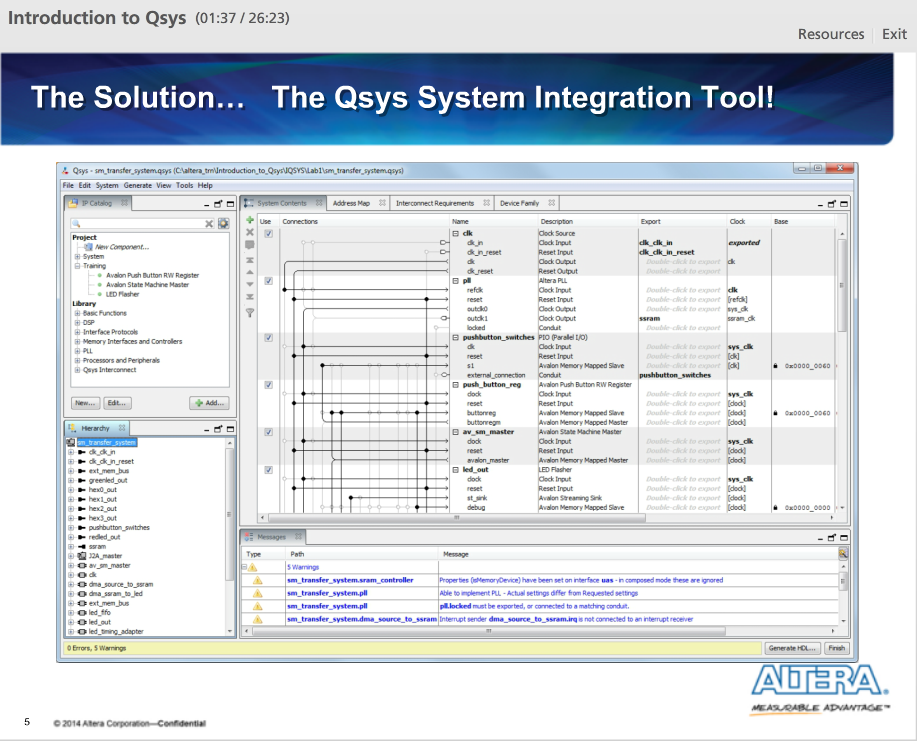
1) Introduction to the Qsys tool

2) Introduction to the Qsys interconnect

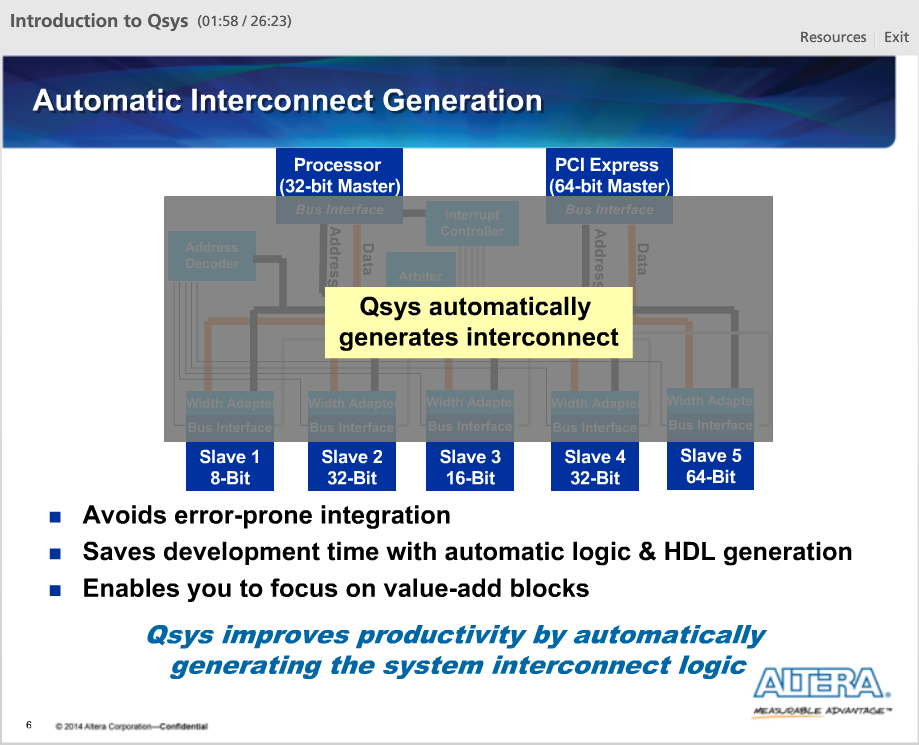
By the end of this lecture, we will be able to describe the benefits or advantages of using Qsys to build a system in our FPGA and we will be able to describe the structure and flow of data through the interconnect.



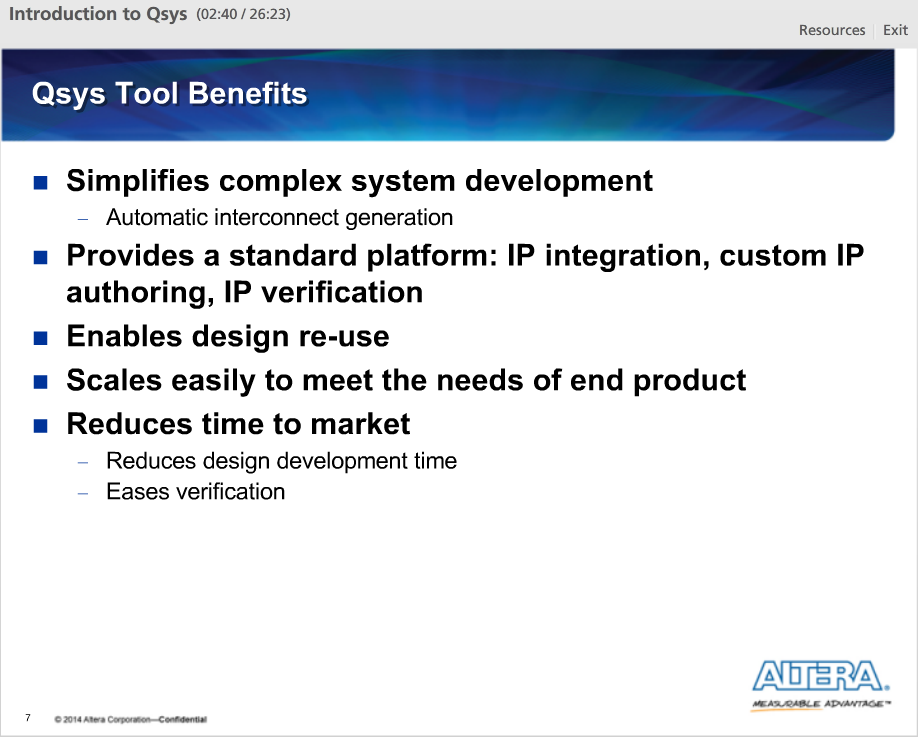
An example of a typical system. We have different devices in our system that have different interfaces that need to communicate in order for the system to work. This requires a lot of engineering effort to design custom interface, or glue, logic to enable this communication including things like arbitration and width matching. This type of logic does not add a lot of value, it is just necessary to get the system connected.



Qsys is the Altera solution to these system design problems. It provides a single interface for building simple to complex designs inside our target FPGA.



Qsys generates a custom-built interconnect structure that facilitates communication between all of our system components. This enables us to focus on value-add blocks.



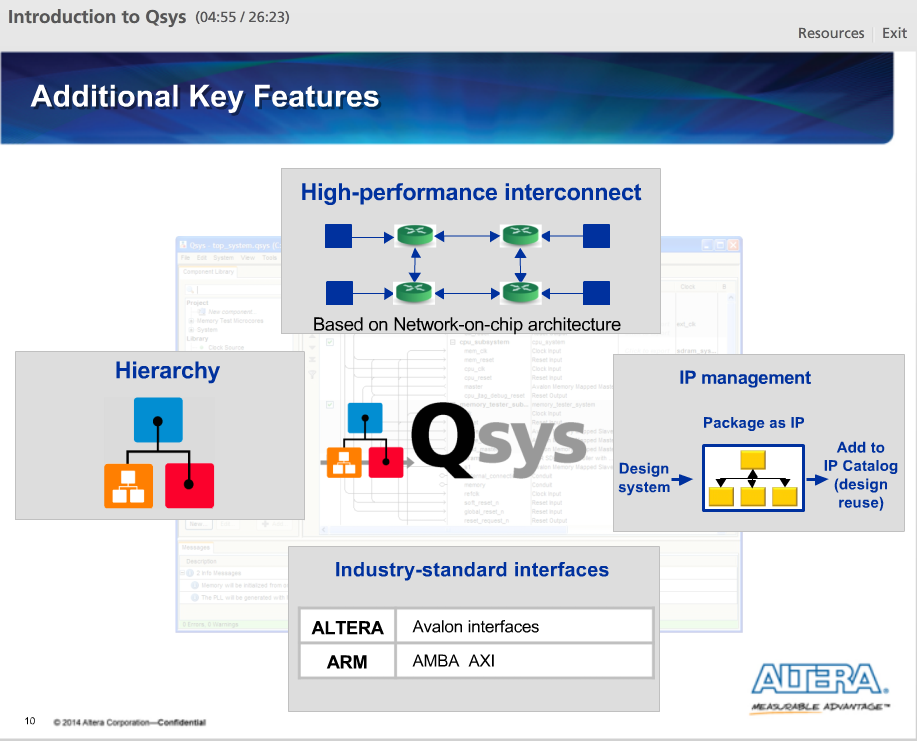
The main benefits of Qsys tool.



Qsys raises the level of abstraction during system development. Instead of designing at the signal level, Qsys allows a designer to design at the system component level.



The main features of Qsys.



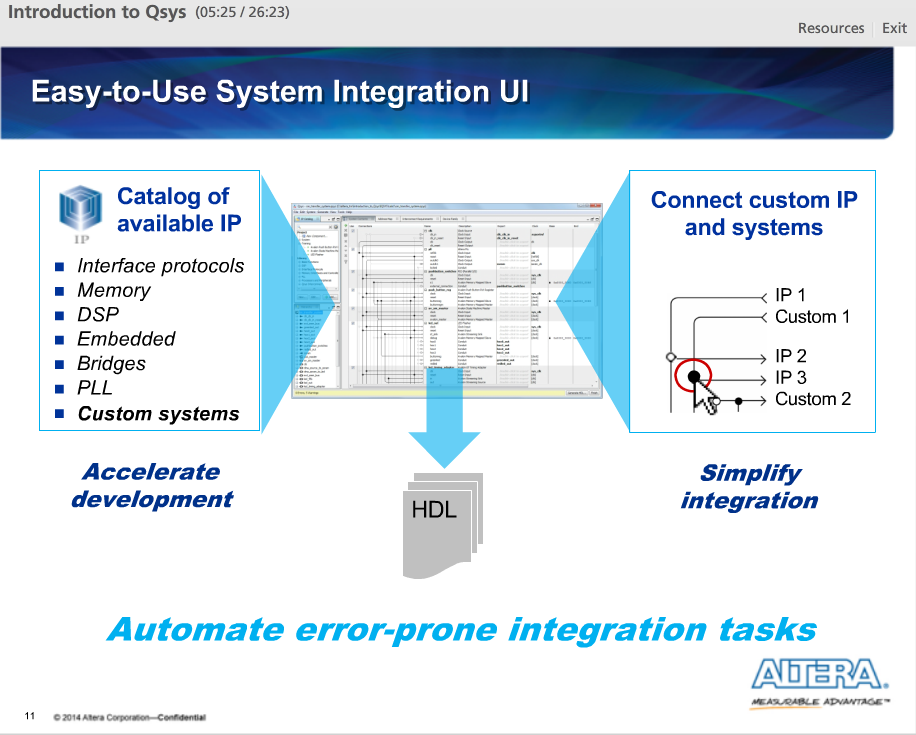
Qsys also has additional features to even further improve its usefulness:

1) Qsys uses for its system interconnect a high-performance interconnect modeled after networking protocols and chips

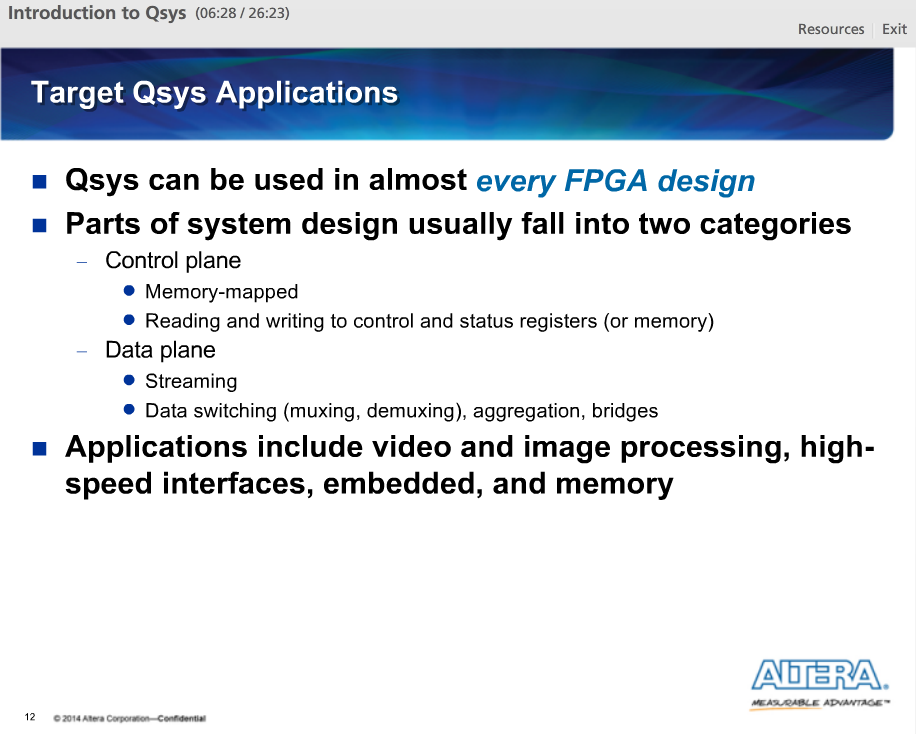
2) Qsys supports hierarchy allowing Qsys systems to be used inside other Qsys systems

3) Qsys supports industry standard interfaces allowing designers to easily connect and support various systems and data configurations.

4) Qsys offers improved capabilities for IP management, allowing designers to package entire systems as IP for distribution to other users



All this is done by Qsys, while still providing an easy-to-use UI for system integration. Again, the goal is to build systems at a higher level of abstraction than manually writing code ourselves.



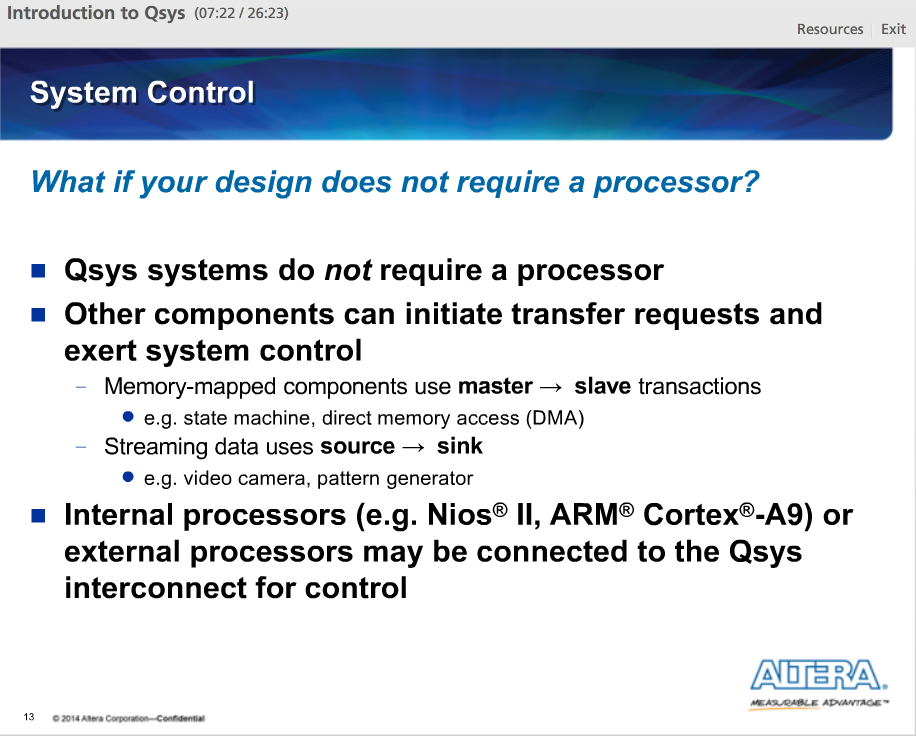
Pretty much any FPGA design can take advantage of Qsys.

We could divide the paths in typical designs into two types: the control plane and the data plane. Both are supported in Qsys.

The control plane uses memory mapped interfaces in which components present control and status registers to the system. These components are controlled by reading and writing to these registers.

The data plane uses point-to-point interfaces where packetized data continuously moves, or streams, through the system. The data plane then uses switching mechanisms like muxes and demuxes or data manipulators like aggregators or bridges to control the course of the data.

Qsys target applications include video and image processing, high-speed interfaces and protocols as well as embedded and memory applications.



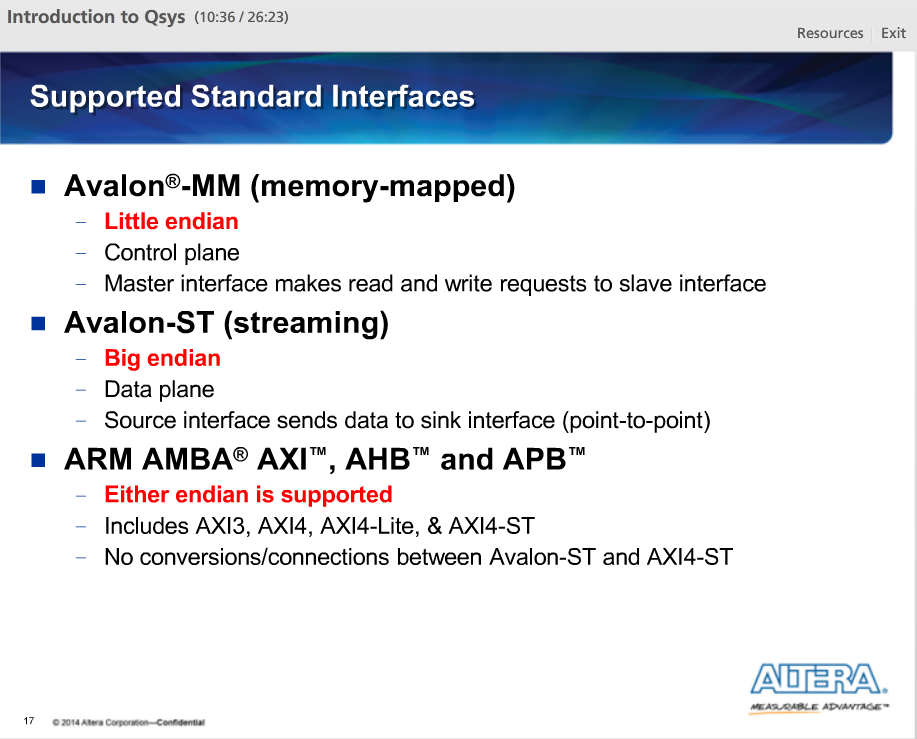
Qsys systems and processors like the Nios II processor are not synonymous. A Qsys system only requires components to initiate transfers and/or requests and components to receive those transfers or requests.

We can, of course, use internal processors, but we do not have to.

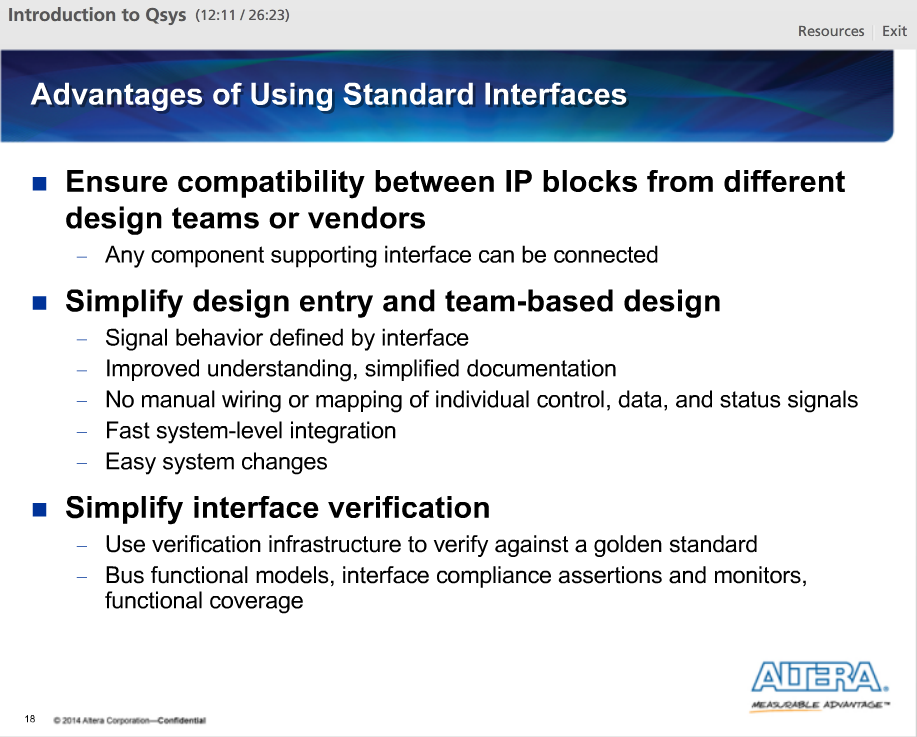


The Qsys interconnect is what connects all Qsys components together. It is a custom-generated interconnect structure created automatically by Qsys based on the connections and settings chosen in the Qsys UI.

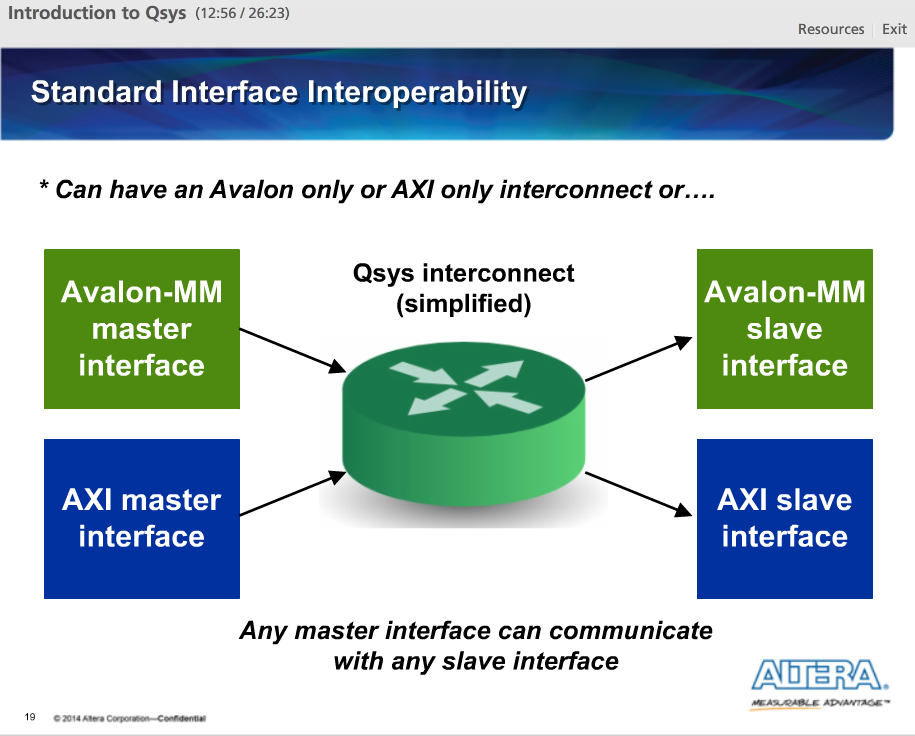
The Qsys interconnect architecture is modeled after networking chip architecture and thus it is optimized for performance, using a packet-based strategy for system components to communicate. This interconnect is made up of standard interfaces, embedded system structures, the Qsys packet and network on a chip routing resource.



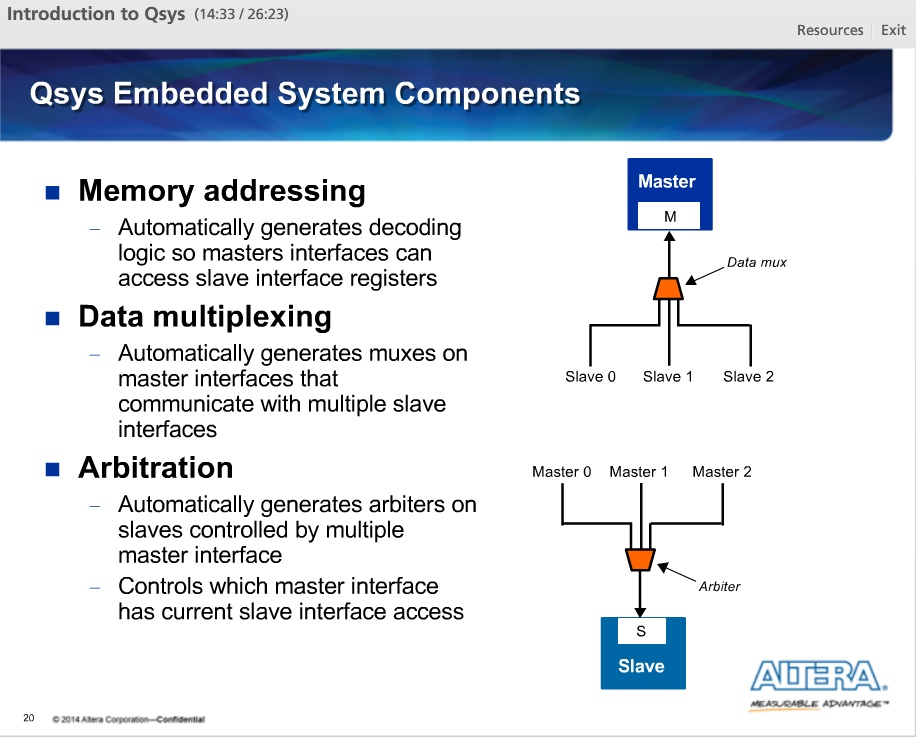
The first part of the Qsys interconnect architecture are standard interfaces. Standard interfaces are what components use to communicate with other components in the system. Qsys currently supports multiple standard interfaces, including the Avalon-MM (or memory mapped) interface, the Avalon-ST (or streaming) interface and the ARM AMBAAXI, AHB and APT interfaces.



Advantages of using standard interfaces over custom ones.



Using Qsys, our system can be all Avalon interfaces or all AXI interfaces, but it does not have to be. With Qsys, we can have a mixture of supported interface types and Qsys will manage the communication between them.

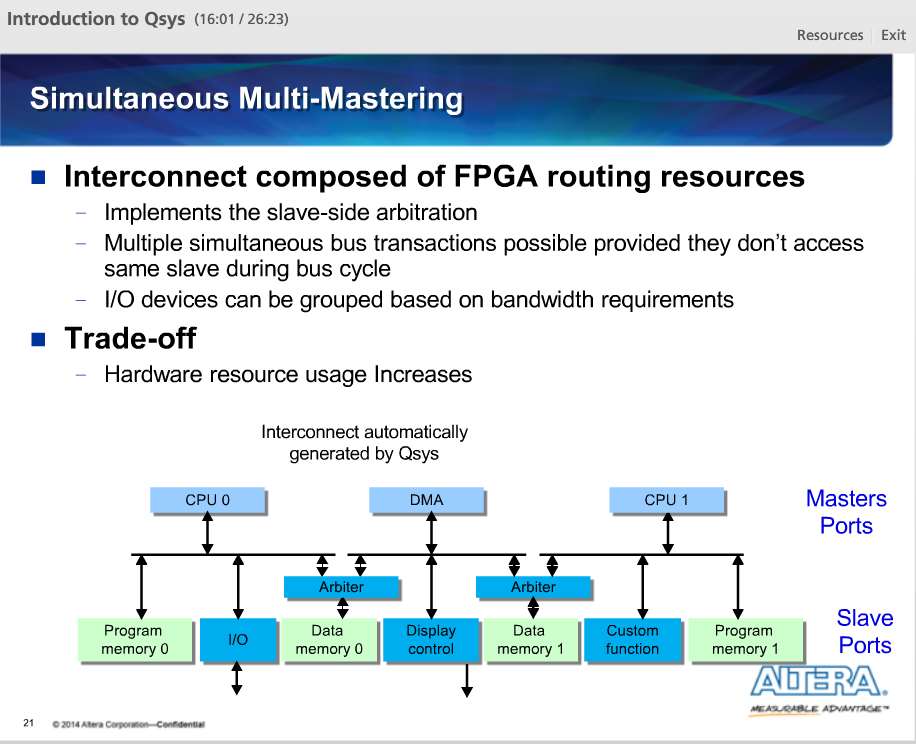


In addition to standard interfaces, the Qsys interconnect makes use of embedded system structures like memory addressing, data multiplexing and arbitration. This structure facilitate the communication between the potential multiple masters and slave interfaces that might make up a system.

With memory addressing, each slave interface in a non-streaming connection is assigned an address in memory space. A master interface communicates with that slave interface by issuing read and write transfer requests to that slave interface's addresses. Qsys automatically generates decoding logic to make this happen.

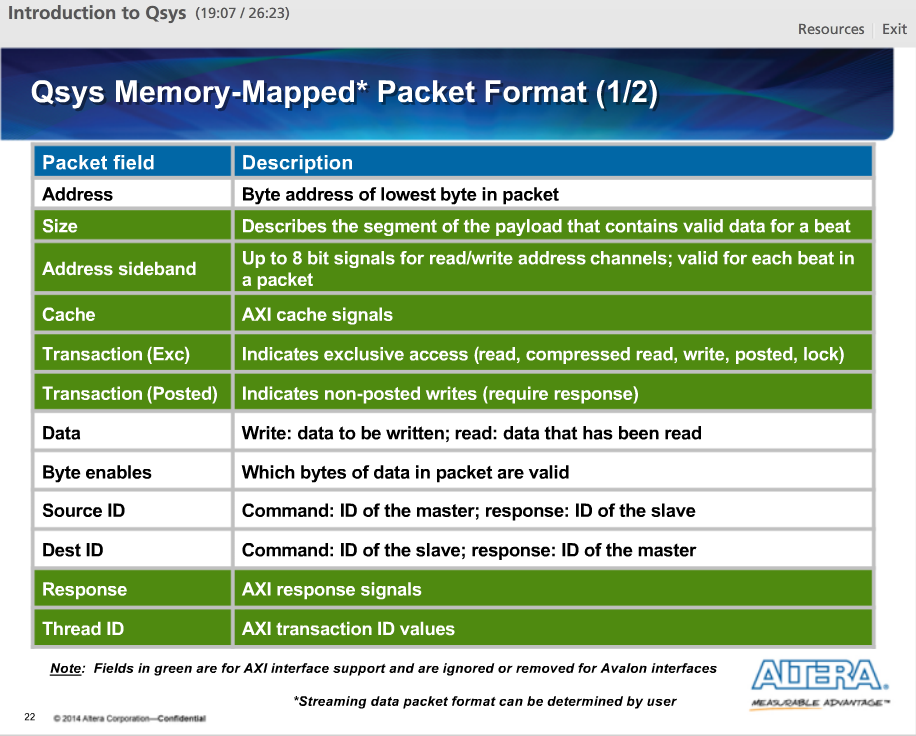
With data multiplexing, Qsys automatically creates muxes on the front side of master interfaces so they can communicate with multiple slave interfaces. A master interface can issue transfer requests with multiple slave interfaces and the interconnect handles the return of data from the slave interfaces back to the master interface by controlling these data muxes.

With arbitration, Qsys automatically creates arbiters for slave interfaces so they can communicate with multiple master interfaces. The arbiter controls which master interface has access to the slave interface at any one time. The users can control some aspects of the arbiters.



Qsys supports simultaneous multi-mastering. Simultaneous multi-mastering allows multiple master ports to transfer data simultaneously. Unlike traditional host-side arbitration architectures in which each master must wait until it is granted access to the shared bus, multiple memory mapped master ports can simultaneously perform transfers with independent slave ports. Arbitration logic stalls a master port only when multiple master ports attempt to access the same slave port during the same cycle.

So, we can make trade-offs between when and where multi-mastering is employed to provide balance.



Qsys packets are used by the Qsys interconnect to stream requests through the system using the Avalon-ST protocol. All memory-mapped transactions are converted into Qsys command and response packets. The size of the Qsys packet is not explicitly defined. This allows Qsys to set the size of the packet based on the system components in order to minimize the resources needed to construct the interconnect.

This packetizing mechanism is what allows different interfaces types to be connected together.

This table shows the makeup of a Qsys packet. The fields in white are used by all packet types. The fields in green are used when AXI requests or responses are moved through the interconnect. They are ignored or removed when converting to Avalon interfaces.

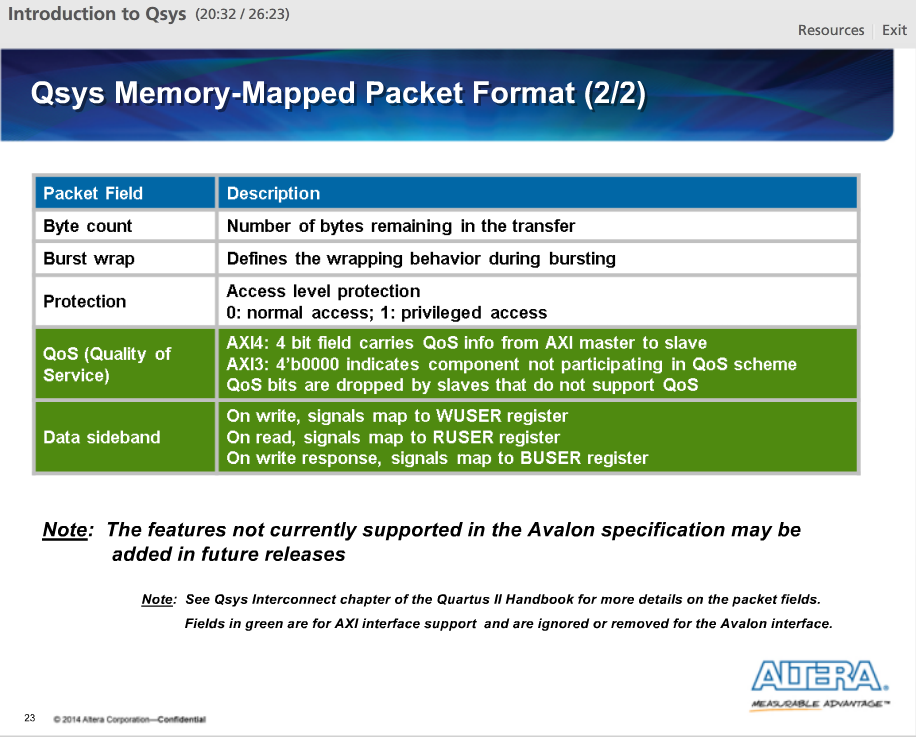
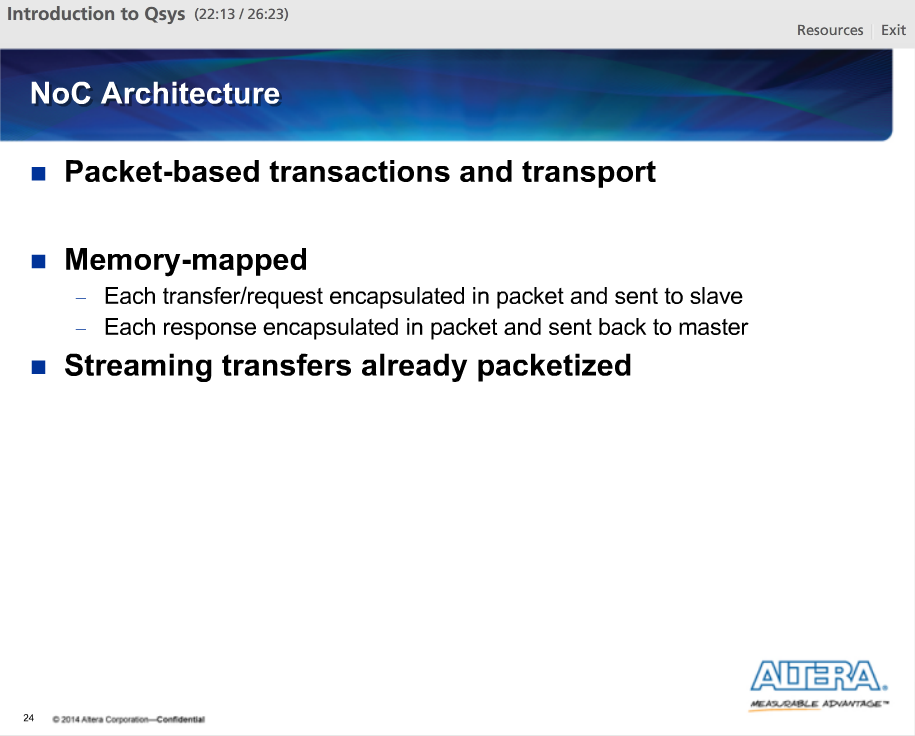


Table continuation.



The last part of the Qsys system interconnect is the Network-on-a-chip, or NoC. The NoC serves to transfer packets of data through the system to facilitate communication.

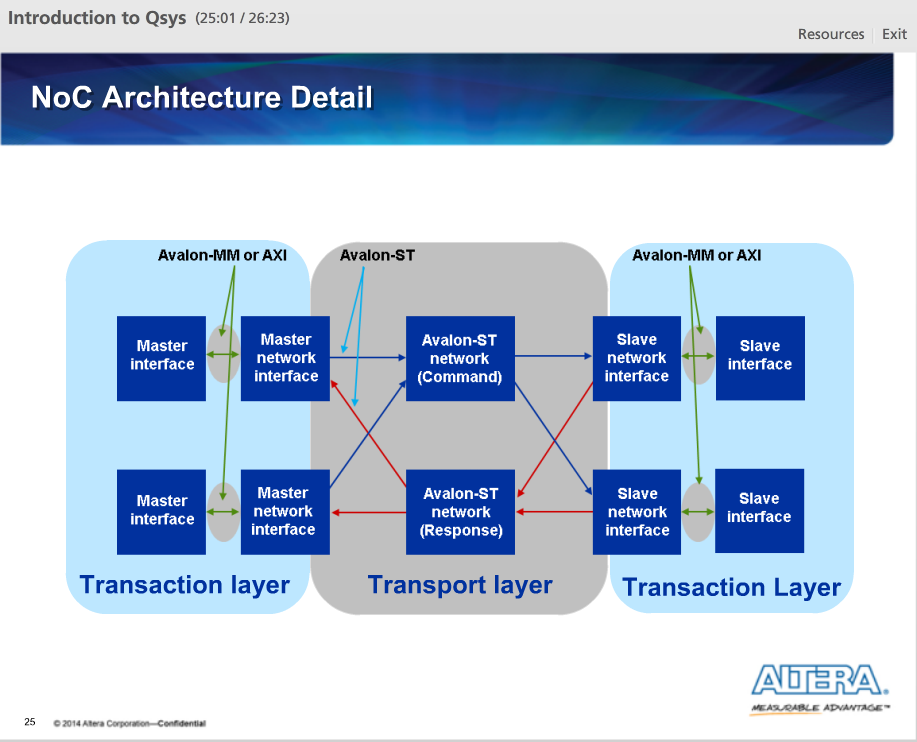
For memory-mapped interfaces, read and write transfers are converted into packets and sent through the system from master to slave where they converted back into memory mapped transfer requests. Responses from slaves are again converted into packets and sent through the system back to the requesting master where they are again converted back into memory-mapped transfer responses.

Since streaming transfers are already packetized, they are transferred through the system as is.

Latency is resolved in two ways:

1) The packet format used by Qsys is wide enough that a complete transaction can be contained in a single packet. This means that memory mapped write requests can be completed in one cycle and that memory mapped read requests can be completed in as little as one cycle each direction.

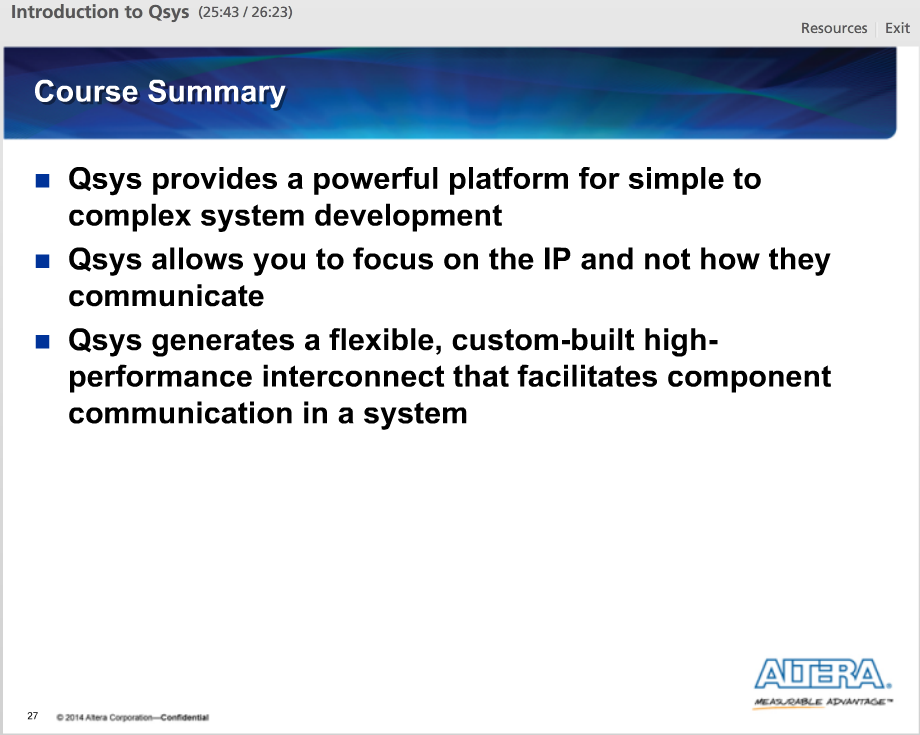
2) The command and responses packets use different paths.



Here is a block diagram example of the Qsys system showing the major parts of the NoC. This system contains two master ports and two slave ports. As indicated by the shaded regions in the back, the interconnect is made up of both memories mapped and streaming paths.

There are three sections to the NoC: the master network interface, the Avalon-ST network and the slave network interface.

There is one master network interface per master interface connected to the Qsys interconnect. There is one slave network interface per slave interface connected to the Qsys interconnect.



After this lecture, we could see that Qsys is a design tool that provides for us a powerful platform that we can use to build FPGA-based systems, from the simple to the very complex. Using Qsys allows us as the designers to focus on the actual IP in our systems and not worry about the logic needed for our IP to communicate.