Exporting Data-Plane Energy-Aware Capabilities from Network Devices toward the Control Plane: The Green Abstraction Layer

(Invited Paper)

Diego Reforgiato[‡], Alfio Lombardo [‡], Franco Davoli[*], Leonardo Fialho[§], Martin Collier[§], Pasquale Donadio[¶], Raffaele Bolla[*], Roberto Bruschi[*]

[¶]Alcatel-Lucent, Milan, Italy. Email: pasquale.donadio@alcatel-lucent.com
[*]CNIT, Genoa, Italy. Email: {raffaele.bolla, roberto.bruschi, franco.davoli}@cnit.it
[§]Dublin City University, Ireland. Email: {leonardo.fialho, martin.collier}@dcu.ie
[‡]Lightcomm s.r.l., Catania, Italy. Email: econet@lightcomm.it

***Abstract*—Energy efficiency is well-known to have recently become one of the most important aspects for both today's and tomorrow's telecommunications infrastructures. To curb their energy requirements, next-generation hardware platforms of network devices are expected to include advanced power management capabilities, which may allow a dynamic trade-off between power consumption and network performance. At the same time, network protocols are going to evolve in order to carry energy-aware information, and to add them to classical performance indexes in network optimisation strategies. However, the question of how to map energy-aware indexes, often arising from low-level local hardware details, and the ones related to network performance is still an open issue. Starting from these considerations, we propose the Green Abstraction Layer (GAL), a device internal interface that provides a standard way of accessing and organising energy-aware information from the low-level hardware components to control processes. The GAL is specifically designed to hide the heterogeneous hardware implementation details, and to provide a simple, hierarchical, and common view of underlying power management capabilities to network control processes.**

I. Introduction

Network user traffic and router capacities are exponentially increasing. However, they are not compensated by a corresponding increase in silicon energy efficiency. Besides, the vast majority of currently deployed network links and devices are designed to operate (and, consequently, to consume power) constantly at their maximum capacity, irrespectively of the traffic load, even though their average utilisation lies far below the maximum [1], [2], [3].

These observations have suggested the possibility of adapting network energy consumption to the actual traffic profiles [4], [5]. Just similarly to general purpose computing systems, such possibility can be realised by including Power Management Primitives (PMPs) into the hardware platforms of networking devices, where energy absorption physically takes place.

PMPs allow aggressively modulating the energy consumption of networking devices or some parts of them, by putting them into standby states when not in use, or by decreasing their maximum performance in the presence of low incoming traffic volumes. On one hand, the best performance is provided when the device operates under no power limitation. On the other hand, it might be noted that the maximal power saving is obviously obtained when the equipment is turned off. Under such a condition the performance is actually zero.

With the above understanding, it is clear that power-managed devices need control loops able to dynamically tune hardware capabilities to provide the required Quality of Service (QoS) level to incoming traffic with the minimal power consumption.

It is also worth noting that PMPs are features locally available in network nodes, and their efficiency may heavily depend on the specific implementation and low-level details of the hardware platforms of devices; the latter may be quite heterogeneous, even when considering equipment of the same market segment or vendor.

Owing to such considerations, providing each network device with its own independent control loop - or Local Control Policies (LCP) - would appear an obvious choice. On the basis of the specific features of the local device, control loops may dynamically orchestrate the configuration of internal components (e.g., line-cards, link interfaces, network processors, etc.) to meet the desired QoS with the minimum power consumption. However, when energy optimisations are independently performed by each device, the overall network consumption might result much higher than in the case of cooperation among nodes. Along this direction, a number of approaches [6], [7], [8], [9] have been recently proposed in order to extend current routing and traffic engineering policies beyond classical network QoS metrics, and to explicitly consider also energy consumption of the whole network.

Even if potentially much more effective than LCPs, Network-wide Control Policies (NCPs) suffer from some drawbacks. First, NCPs can exhibit much higher feedback/convergence delays. Secondly, routing and traffic engineering frameworks generally may not have the ability of discriminating how logical network entities can be mapped on physical resources, which directly cause energy absorption. Finally, NCPs often represent a network device simply as a node in a graph, whose arcs are the (virtual/physical) network links.

Such simplistic representation does not allow maintaining the knowledge of some hardware peculiarities that may be very important for reducing consumption.

All these points suggest the possibility of jointly adopting LCPs and NCPs in order to optimise the device energy consumption in a hierarchical way.

Starting from the considerations mentioned above, it is easy to conclude that there is still a significant gap between hardware power management and NCPs, as well as some open issues on which control loops should be adopted, and on how effectively managing the relationships among multiple (local and/or network-wide) control loops. Besides devising new energy-aware LCPs and NCPs, an almost necessary condition for their effective development and adoption is that of representing management and control actions and device/network status information in some standard abstract form, independently of the details of the specific manufacturers' implementations.

The main contribution of the present paper consists in the specification of a layer that provides a way to expose green networking capabilities of devices toward the network control plane. Such layer consists of a novel standard interface, the Green Abstraction Layer (GAL) [10], conceived by the ECONET consortium [11]. Within the GAL we define the Green Standard Interface (GSI), which is a simple interface for exchanging power management data among data-plane elements and processes realising control plane strategies, in a standard and simplified way. This paper will identify and design standard interfaces to control, configure and monitor energy-aware elements in network devices.

In more detail, the paper is organised as follows: Section II will define the GAL architecture. Section III introduces the GSI, a platform-independent interface that works at different detail levels and that can be applied for control and data-planes intercommunication as well as for intra-data-plane notifications. Section IV describes the GSI command set suitable to manage network resources of the ECONET platform. A concrete example of mapping the GAL onto an NetFPGA environment is shown in Section V. Finally, Section VI ends the paper with conclusions and future research directions.

II. Green Abstraction Layer Architecture

The GAL architecture was conceived as a modular and easily extendable software framework. The GAL architecture aims at two main objectives:

- to provide interface capabilities towards heterogeneous hardware;
- to provide multiple hierarchical interfaces towards control processes in order to permit energy configurations at more detail levels of the internal architecture of a device.

In order to divide and conquer such complex aspects and internal interactions, the ECONET consortium designed the GAL to provide a hierarchical representation of network devices, by specifying their components at the various levels in terms of abstract entities.

As shown in figure 1, the GAL hierarchical architecture permits to address each resource of the ECONET platform by using a simple layering schema. Figure 1 depicts four possible layers:

- **Layer 1** groups and manages "Device entities", represented in the figure by network nodes;
- **Layer 2** groups and manages "Chassis entities" represented in the figure by shelves;
- **Layer 3** groups and manages "Line card entities" represented in the figure by boards;
- **Layer 4** groups and manages "Hardware components entities" represented in the figure by basic components, such as ports, fans, chips, etc.

Figure 1 also shows the GSI on top of each manageable entity, as well as LCPs operating towards the data-plane, depicted in the figure by the Convergence Layer.

III. Green Standard Interface

The GAL provides two interfaces: *a)* the GSI, which is the external or public interface used to interact with clients and applications, and *b)* an internal interface named Convergence Layer (CL) towards the data-plane. The GSI is the northbound interface of a GAL instantiation. It is expected to be used by three main sets of control plane processes:

- LCPs, conceived to optimise the configuration of the device in order to achieve the desired trade-off level between energy consumption and network performance according to the incoming traffic load;
- NCPs to autonomically control and optimise the behaviour of a network (conceived as a set of devices). Typical examples of these kinds of policies are traffic engineering, routing and signalling algorithms/protocols (e.g., OSPF-TE/RSVP-TE) with "green" extensions;
- Monitoring and Operation Administration & Management (OAM) to control and optimise the behaviour of a network manually, as Network Management systems with "green capabilities".

The GSI provides a set of commands to setup the power management and monitoring of a wide set of energy-aware resources and network devices. In other words, the GSI allows accessing energy-aware resources to discover their power characteristics and to deal with their provisioning, configuration, monitoring, and decommissioning.

The CL represents the platform-specific functionalities (in terms of messages, commands and attributes) that work directly with the device, and it offers the GSI a common basis upon specific architectures.

IV. GSI Command Set

Even though each specific GSI implementation would depend on the internal architecture of the device (e.g. network

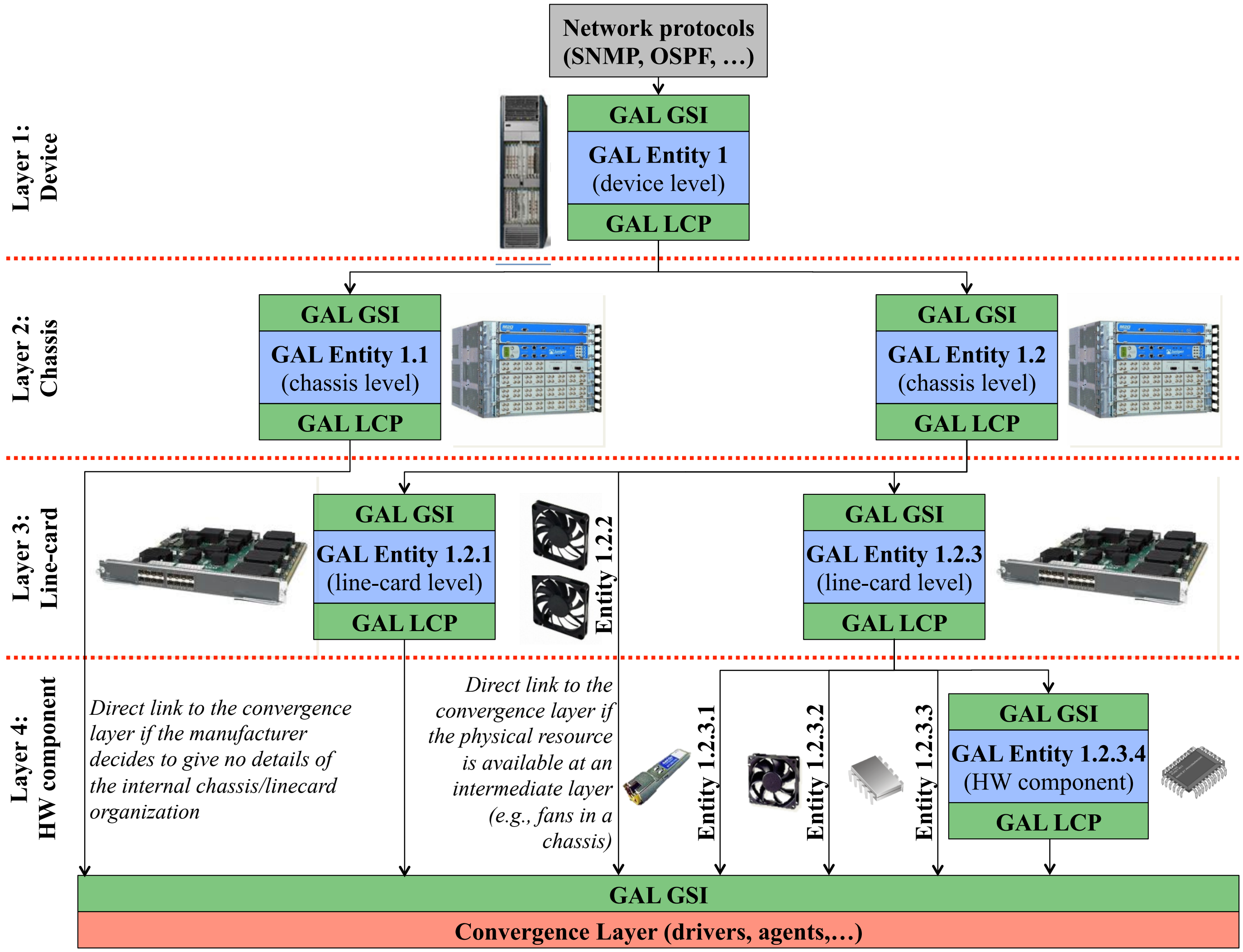

Fig. 1. The Green Abstraction Layer (GAL) hierarchical architecture depicting the different layers of network components, the Green Standard Interface (GSI), the Local Control Policies (LCP), and the convergence layer towards the data-plane.

elements, network cards, chips, fans, etc.), in this paper we propose a set of abstract commands to enable the GSI to operate. The main functionalities offered by the GSI commands are the following:

- **Discovery:** used to retrieve information about the state, the power consumption and the availability of the physical device;
- **Provisioning:** allows the energy-consumption configuration (activation) of a physical device through the selection of one among the energy-aware states (EASs) offered by the device driver (entity driver);
- **Monitoring:** permits monitoring of relevant parameters (state, power consumption, etc.) of the physical device;
- **Release:** allows the energy-consumption release (deactivation) of already configured physical devices. After release, the device should operate its default configuration.

GSI commands may be forwarded directly to the CL or to a GAL entity. As shown in figure 1, GSI commands should be forwarded directly to the CL if the manufacturer decides to give no details of the internal hardware architecture, or if the physical resource does not require or provide a LCP. Organising the internal hardware architecture at different levels permits the GSI commands to flow through different GAL entities, where LCPs are applied. At the very end of this hierarchical organisation, all GSI commands should reach the CL in order to perform the corresponding operation.

Attributes and parameters of the GSI command set, as well as the data types used by these commands, are deeply analysed in the following sub-sections.

A. Discovery Command

The discovery command permits to retrieve information related to the physical infrastructure organisation and energy management features. The command is called `GAL_Discovery` and performs a synchronous discovery of entity capabilities and characteristics. Implementations of this command should be aligned with the following reference model:

```
GAL_Discovery ( ResourceID, Profile ) :

 IN  ResourceID  GAL_ID   unique resource
                          identification
OUT   Profile   GAL_PROF discovery entity container
OUT   retval     GAL_INT  GAL_SUCCESS when it was
                          possible to discover the
                          entity
OUT   retval     GAL_INT  GAL_ERROR when an error
                          occurred during the
                          discovery operation
OUT   retval     GAL_INT  GAL_NOT_IMPLEMENTED if the
                          command is not implemented
```

The Profile output parameter provides information regarding the energy consumption profile of the queried entity as well as the existence of other entities attached (children).

B. Provision Command

The provision command permits to configure an individual resource or a group of resources. The GSI provides two provisioning commands: GAL_Provision, which performs the synchronous configuration of a single entity, and GAL_RProvision, which performs a synchronous and recursive configuration of all entities connected to the target entity identified by the ResourceID parameter. Implementations of these commands should be aligned with the following reference model:

```
GAL_Provision ( ResourceID, State ) :

 IN  ResourceID  GAL_ID  unique resource
                         identification
 IN     State    GAL_INT energy state the device
                         should operate in
OUT   retval     GAL_INT GAL_SUCCESS when it was
                         possible to provision the
                         entity
OUT   retval     GAL_INT GAL_ERROR when an error
                         occurred during the
                         provision operation
OUT   retval     GAL_INT GAL_NOT_IMPLEMENTED if the
                         command is not implemented
```

```
GAL_RProvision ( ResourceID, State ) :

 IN  ResourceID  GAL_ID  unique resource
                         identification to use as
                         root
 IN     State    GAL_INT energy state the devices
                         should operate in
OUT   retval     GAL_INT GAL_SUCCESS when it was
                         possible to provision the
                         entity
OUT   retval     GAL_INT aggregated error codes
                         when at least one error
                         occurred during the
                         provision operation
OUT   retval     GAL_INT GAL_NOT_IMPLEMENTED if the
                         command is not implemented
```

The aggregated error code returned by the GAL_RProvision command should represent the errors occurred on the target entity and on the other entities connected to the first one.

C. Monitor Command

The monitor set of commands permits to monitor the energy parameters and values of an individual resource or a group of resources. There are three commands to monitor entities: GAL_Monitor_Consumption, which returns the energy consumption of a single entity, GAL_Monitor_RConsumption, which returns the accumulated value of the energy consumption of a group of entities, and GAL_Monitor_State, which returns the current power state of the device. Implementations of these commands should be aligned with the following reference model:

```
GAL_Monitor_Consumption ( ResourceID, Power ) :

 IN  ResourceID  GAL_ID  unique resource
                         identification
OUT    Power     GAL_INT energy consumption in
                         mWatts
OUT   retval     GAL_INT GAL_SUCCESS when it was
                         possible to monitor the
                         entity
OUT   retval     GAL_INT GAL_ERROR when an error
                         occurred during the
                         monitor operation
OUT   retval     GAL_INT GAL_NOT_IMPLEMENTED if the
                         command is not implemented
```

```
GAL_Monitor_RConsumption ( ResourceID, Power ) :

 IN  ResourceID  GAL_ID  unique resource
                         identification to use as
                         root
OUT    Power     GAL_INT sum of the energy
                         consumption of all child
                         devices in mWatts
OUT   retval     GAL_INT GAL_SUCCESS when it was
                         possible to monitor all
                         the entities
OUT   retval     GAL_INT aggregated error codes when
                         at least one error occurred
                         occurred during the
                         monitor operation
OUT   retval     GAL_INT GAL_NOT_IMPLEMENTED if the
                         command is not implemented
```

The Power output parameter of the GAL_RConsumption is the sum of the power consumption of the entity target by the ResourceID parameter plus the power consumption of the entities connected to the first one. Moreover, the aggregated error code returned by the GAL_RConsumption command should also represent the errors occurred on the target entity and on the other entities connected to the first one.

```
GAL_Monitor_State ( ResourceID, State ) :

 IN  ResourceID   GAL_ID    unique resource
                            identification
OUT    State    GAL_STATE operational state of the
                            device
OUT   retval     GAL_INT    GAL_SUCCESS when it was
                            possible to monitor the
                            entity
OUT   retval     GAL_INT    GAL_ERROR when an error
                            occurred during the
                            monitor operation
OUT   retval     GAL_INT    GAL_NOT_IMPLEMENTED if the
                            command is not implemented
```

D. Release Command

The GAL_Release synchronous command permits to eliminate any special energy-aware configuration from a given entity. Implementations of the GAL_Release command should be aligned with the following reference model:

```
GAL_Release ( ResourceID ) :

 IN  ResourceID  GAL_ID  unique resource
                         identification
OUT   retval    GAL_INT GAL_SUCCESS when it was
                         possible to release the
                         entity
OUT   retval    GAL_INT GAL_ERROR when an error
                         occurred during the
                         release operation
OUT   retval    GAL_INT GAL_NOT_IMPLEMENTED if the
                         command is not implemented
```

E. Data Types

There are five data types the GSI commands use: GAL_ID, GAL_INT, GAL_PROF, GAL_STATE and GAL_CHILD. These data types have been specified in order to provide a compatibility layer to different implementations. This compatibility layer avoids common implementation errors such as integer sizes and architectural byte ordering, commonly known as Little-Endian and Big-Endian. Table I presents the basic data types and their sizes. GAL_PROF, GAL_STATE and GAL_CHILD complex data types are specified underneath:

```
GAL_PROF :

num_states    GAL_INT   number of states
states       GAL_STATE array of states
num_children  GAL_INT   number of children
children     GAL_CHILD array of children
```

```
GAL_STATE :

index      GAL_INT index of the state
perf_pkt   GAL_INT average packets per second
perf_bdw   GAL_INT average bandwidth in bytes per
                   second
perf_lat   GAL_INT average latency in μ sec
pwr_min    GAL_INT minimum power consumption in
                   mWatts
pwr_avg    GAL_INT average power consumption in
                   mWatts
pwr_max    GAL_INT maximum power consumption in
                   mWatts
trans_up   GAL_INT transition time to the state
                   above
trans_down GAL_INT transition time to the state
                   below
```

TABLE I
DEFINITION OF BASIC DATA TYPES USED IN THE GSI.

GAL Type	# of Bytes	Signal	C/C++ Equivalent
GAL_INT	32	unsigned	uint_32t
GAL_ID	256	—	char[256]

```
GAL_CHILD :

index    GAL_INT index of the child
child_id  GAL_ID  child's unique resource
                  identification
```

V. The NetFPGA 1G Board: a Case Study

This section provides a brief description of a possible integration of a NetFPGA [12] device into the GAL architecture.

The NetFPGA router is an open reprogrammable hardware platform that is increasingly being used by networking researchers world-wide to rapidly prototype and evaluate new mechanisms. Hosted on the board are: a user-programmable FPGA (with two PowerPC processors), SRAM, DRAM, and four 1 Gbps Ethernet ports[1]. The FPGA directly handles all data-path switching, routing, and processing operations of Ethernet frames and Internet packets, leaving software to handle control-path functions only.

The GSI is in charge of providing the command set necessary to setup the power management and monitoring of a wide set of NetFPGA energy-aware resources and components. In other words, the GSI is used for: *a)* accessing and discovering NetFPGA resources and their power characteristics, *b)* autonomic provisioning and manual configuration of NetFPGA resources, *c)* monitoring, and *d)* decommissioning the NetFPGA components from energy-aware configurations.

The set of messages, their functionalities and workflow of the command set exposed by the GSI to the GSI clients are in line with the NetFPGA architecture. The NetFPGA resources manageable through the GSI and shown in figure 2 are the following:

- **Entity 1:** an abstraction to all the physical entities of the NetFPGA board;
- **Entity 1.1:** the FPGA chip, which provides frequency scaling capability [13];
- **Entity 1.2:** an abstraction to the four 1 Gbps Ethernet ports;
- **Entities 1.2.1, 1.2.2, 1.2.3 and 1.2.4:** the physical 1 Gbps Ethernet ports.

By benchmarking energy performance of this platform, it becomes possible to quantify energy saving achieved by applying new mechanisms for improving power efficiency [13]. This information is also used to fill the LCPs internal tables that provide the NetFPGA power consumption profile.

VI. Conclusions

In this paper we proposed a novel architecture and interfaces enabling the Green Abstraction Layer, to export data-plane

[1] 10 Gbps Ethernet ports are already available in the newer version.

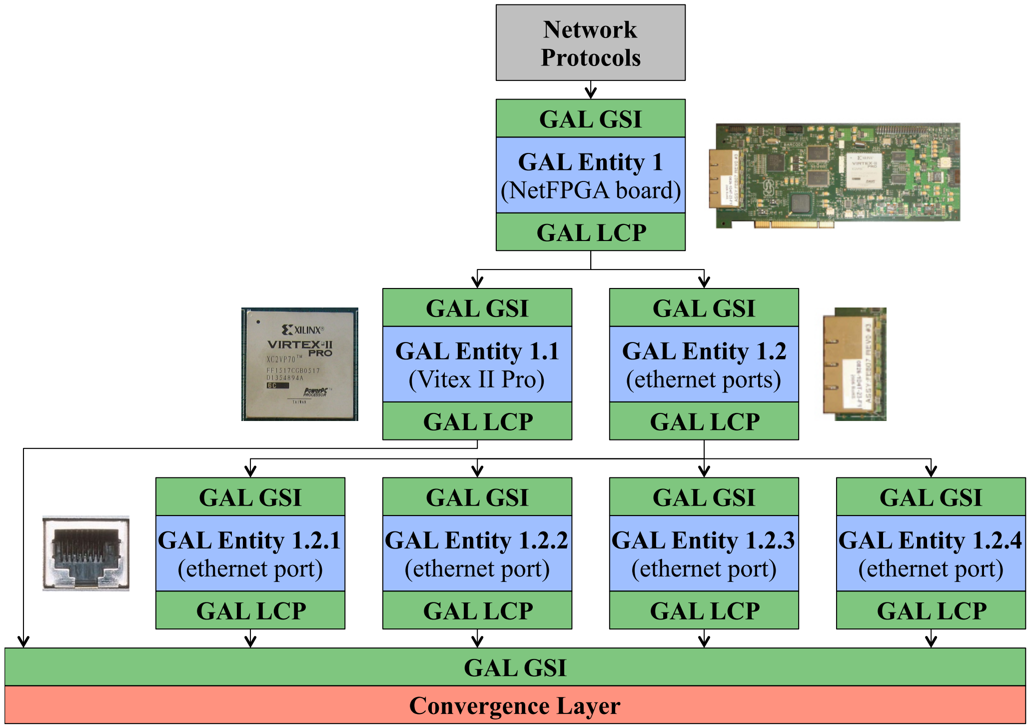

Fig. 2. NetFPGA 1G board according to the GAL architecture perspective.

energy-aware capabilities from network devices toward the Control Plane.

In this work, we have outlined the definition of two interfaces, the Green Standard Interface, which is the "external" interface used to interact with clients and applications and an internal interface (Convergence Layer).

Finally we have described the hierarchical architecture of the GAL, and provided a functionally complete set of primitives, useful for the management of resources in the ECONET platform.

ACKNOWLEDGMENT

The work described in this paper was supported by the ECONET project (low Energy COnsumption NETworks), funded by the EU through the FP7 call.

REFERENCES

[1] R. Bolla, R. Bruschi, A. Carrega, F. Davoli, D. Suino, C. Vassilakis, and A. Zafeiropoulos, "Cutting the energy bills of Internet Service Providers and telecoms through power management: An impact analysis," *to appear in Computer Networks*, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2012.04.003

[2] The Climate Group and Global e-Sustainability Initiative, "SMART 2020: Enabling the low carbon economy in the information age," *http://www.theclimategroup.org/*, pp. 1—87, Jan 2008. [Online]. Available: http://www.theclimategroup.org/_assets/files/Smart2020Report.pdf

[3] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing Network Energy Consumption via Sleeping and Rate-Adaptation," *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, pp. 323—336, 2008. [Online]. Available: http://static.usenix.org/event/nsdi08/tech/full_papers/nedevschi/nedevschi.pdf

[4] R. Bolla, F. Davoli, R. Bruschi, K. Christensen, F. Cucchietti, and S. Singh, "The Potential Impact of Green Technologies in Next-Generation Wireline Networks: Is There Room for Energy Saving Optimization?" *IEEE Communications Magazine*, vol. 49, no. 8, pp. 80—86, 2011. [Online]. Available: http://dx.doi.org/10.1109/MCOM.2011.5978419

[5] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 2, pp. 223—244, 2011. [Online]. Available: http://dx.doi.org/10.1109/SURV.2011.071410.00073

[6] L. Chiaraviglio, M. Mellia, and F. Neri, "Minimizing ISP Network Energy Cost: Formulation and Solutions," *IEEE/ACM Transactions on Networking*, vol. 20, no. 2, pp. 463—476, 2011. [Online]. Available: http://dx.doi.org/10.1109/TNET.2011.2161487

[7] A. Cianfrani, V. Eramo, M. Listanti, and M. Polverini, "An OSPF Enhancement for energy saving in IP Networks," *Proceedings of the 2011 IEEE Conference on Computer Communications*, pp. 325—330, 2011. [Online]. Available: http://dx.doi.org/10.1109/INFCOMW.2011.5928832

[8] J. C. C. Restrepo, C. G. Gruber, and C. M. Machuca, "Energy Profile Aware Routing," *Proceedings of the 2009 IEEE International Conference on Communications*, pp. 1—5, 2009. [Online]. Available: http://dx.doi.org/10.1109/ICCW.2009.5208041

[9] R. Bolla, R. Bruschi, A. Cianfrani, and M. Listanti, "Enabling Backbone Networks to Sleep," *IEEE Network Magazine*, vol. 25, no. 2, pp. 26—31, 2011. [Online]. Available: http://dx.doi.org/10.1109/MNET.2011.5730525

[10] R. Bolla, R. Bruschi, F. Davoli, L. D. Gregorio, P. Donadio, L. Fialho, A. Lombardo, D. Reforgiato, and T. Szemethy, "The Green Abstraction Layer: A Standard Power Management Interface for Next-Generation Network Devices," *submitted to the IEEE Internet Computing Magazine*, 2012.

[11] "The ECONET Project," http://www.econet-project.eu.

[12] "The NetFPGA Project," http://www.netfpga.org.

[13] A. Lombardo, C. Panarello, D. Reforgiato, and G. Schembra, "Power control and management in the NetFPGA Gigabit Router," *to appear in the Proceedings of the Future Network and MobileSummit 2012 Conference*, 2012.