ENABLING APPLICATIONS TO EXPLOIT SMARTNICS, FPGAS, AND ACCELERATORS

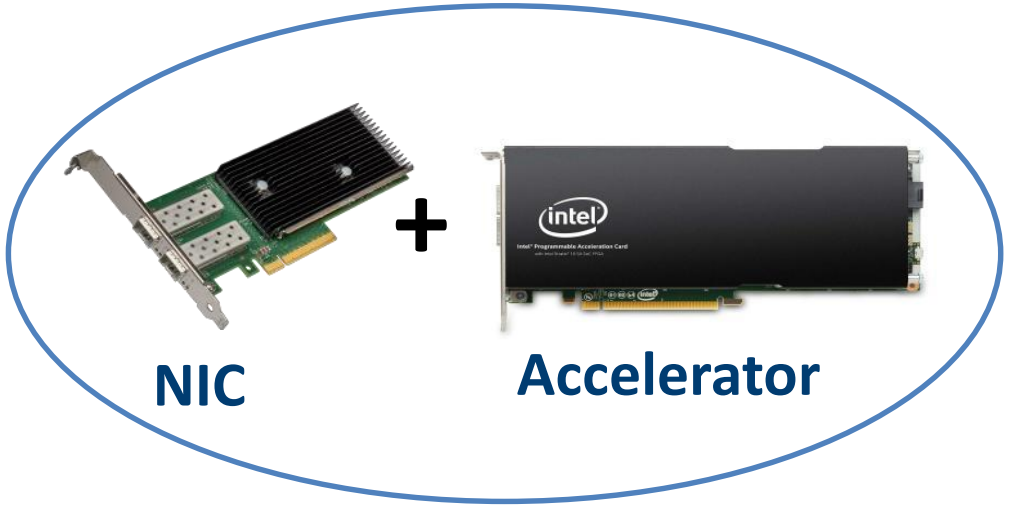
Venkata Krishnan, Sean Hefty

Intel Corporation

March2019

PROBLEM STATEMENT

WHAT IS A SMARTNIC?



SmartNIC = Network attached acceleration platform. Offloads compute from host processor.

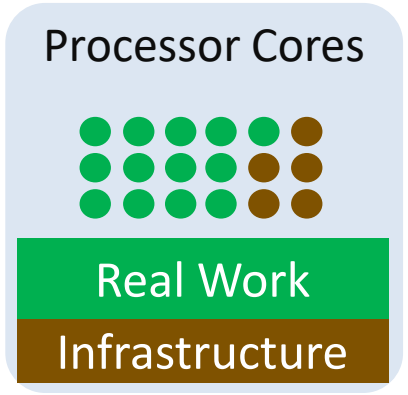
SmartNIC would ideally support the following:

- Traditional networking capabilities (e.g. RDMA)
- Integrates communication & computation in hardware
- Configurable for a particular application
- Software stack exposes networking & acceleration capabilities in a seamless manner to applications

WHY A SMARTNIC?

Accelerator for network & network related workloads

Desire for lower server overhead



SmartNICs reduce compute cycles doing infrastructure work

Infrastructure Offloads

Desire for improved performance and reduced latency

SmartNICs can provide better performance/watt than host-based apps

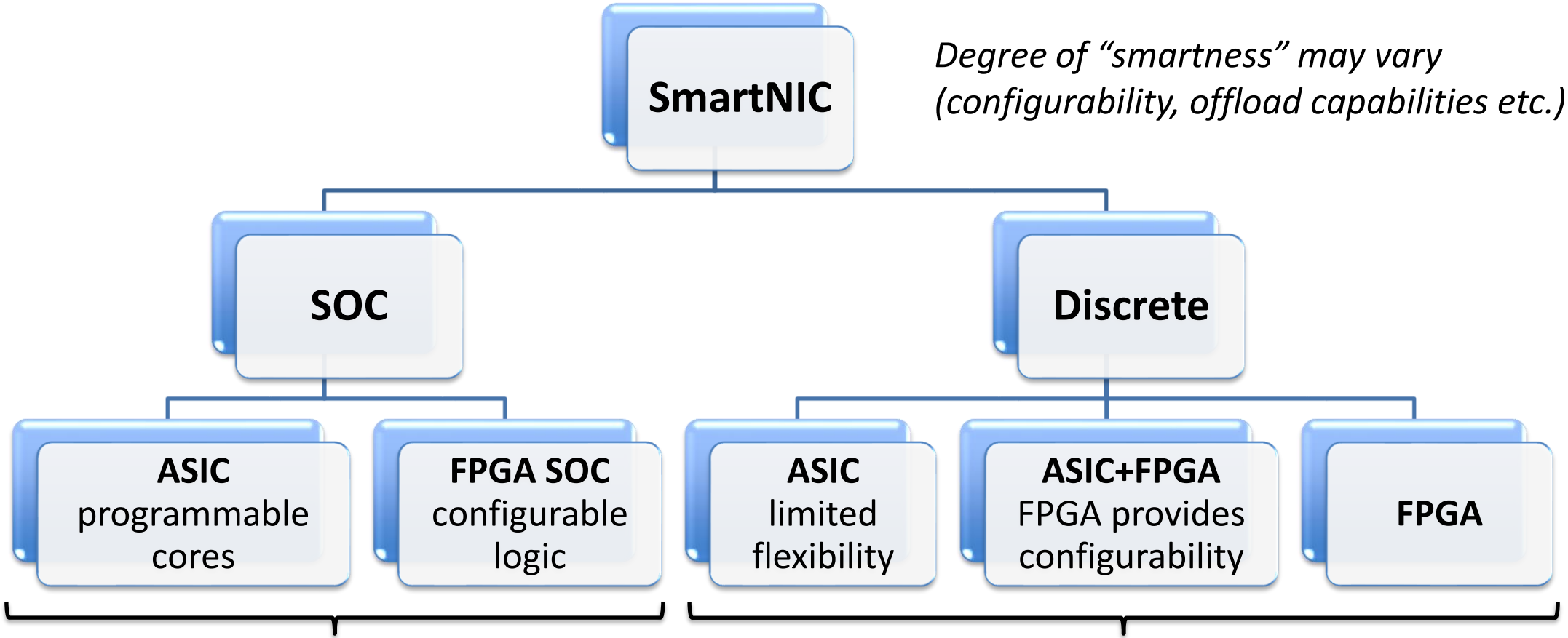
Application Acceleration

Desire for changes in network technology at the speed of software

SmartNICs provide programmable solutions

Agility

WHAT DO SMARTNICS LOOK LIKE?

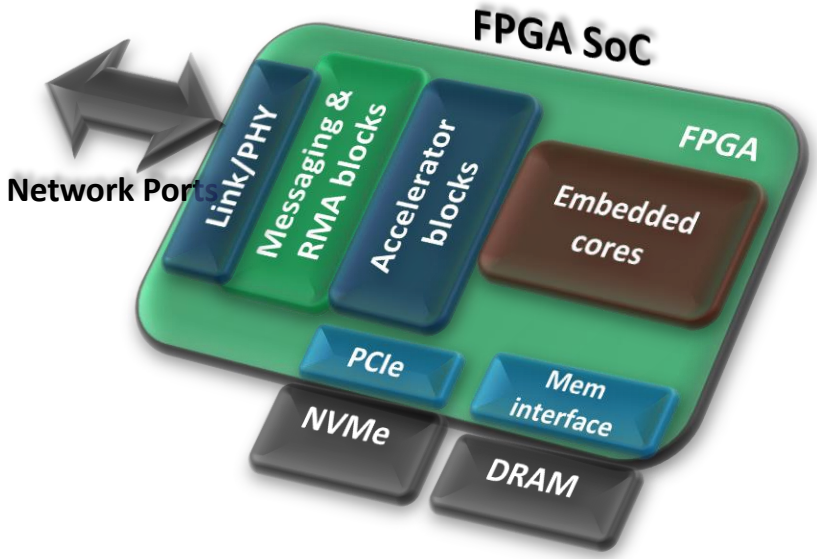


Degree of "smartness" may vary (configurability, offload capabilities etc.)

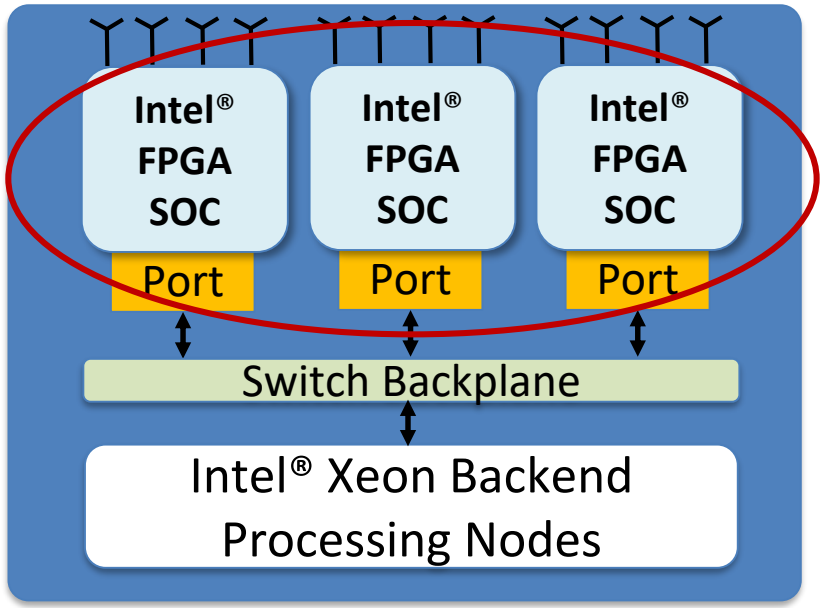
Integrates processor cores with embedded OS. Could function **autonomously** without a host platform.

Applications run on host.
SmartNIC offloads compute from host processor
Typically uses PCIe host bus interface to host.

WHAT COULD (AUTONOMOUS) SMARTNICS DO? EXAMPLE: COMPUTING NEAR SENSORS



FPGA SOC as Front End Processing Nodes

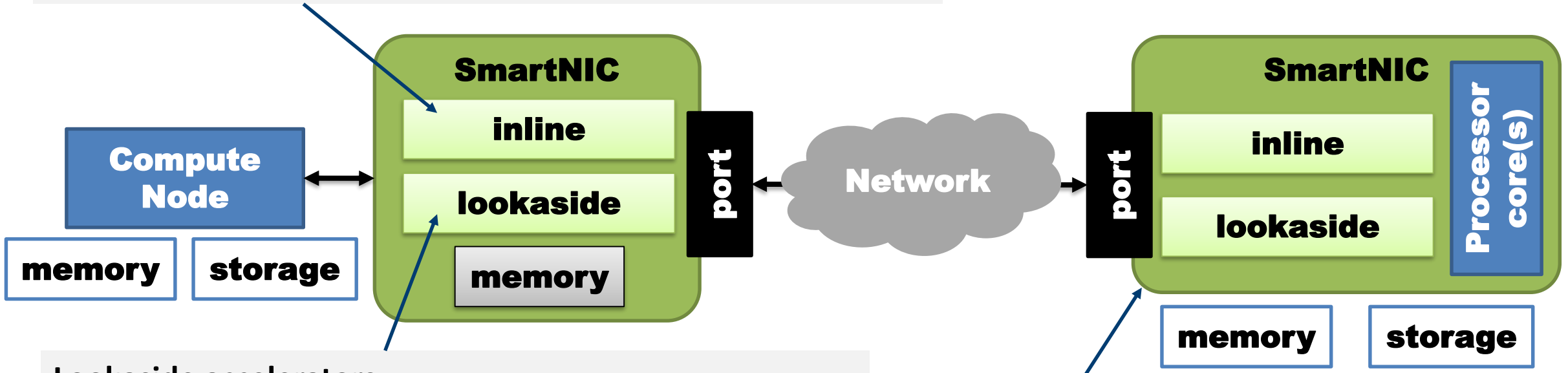


Frontend (Trigger) - Particle detectors, Radio Astronomy, Aerospace etc.

- "Filter" huge volume of data by performing compute *at point of data acquisition*
- Estimated reduction in backend nodes/fabric requirements **could be 10x-100x**
- Flexibility enables new/updates to algorithms

SMARTNIC ACCELERATOR USAGES

Inline accelerators perform compute on data during transmit/receive operation (streaming or bump-in-wire model)



Lookaside accelerators

Same as traditional accelerator model. However, output from accelerator can be directly transmitted to target over network. Similarly data received from network can be forwarded to accelerator block directly for processing. There is no data movement back/forth to host.

Triggered Accelerator

No host/OS involvement.
Inline and/or lookaside accelerators triggered by incoming packet (Disaggregated model)

SOFTWARE INTERFACES

OBJECTIVES

Expose common software APIs to apply data operations on network flows

- **Support offloaded accelerations in conjunction with network**
  - Smart NIC, FPGA, GPU, enhanced switches
  - Local and/or remote accelerations
  - Inline and look-aside
- **Discover available network functions**
- **Enable functions at specific points in network data flows**

This is NOT an FPGA development kit or a general API for executing on GPU kernels.

COMMUNICATION ACCELERATION API REQUIREMENTS

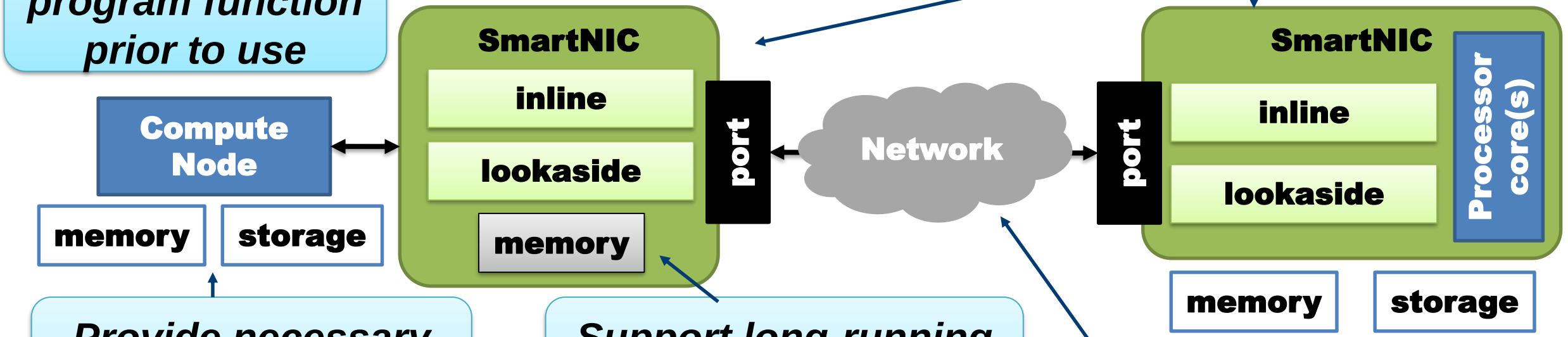
Discovery mechanism – available vs active

Select accelerator and function

SmartNIC SW may need to program function prior to use

Persistent vs on-demand functions

Support local and remote accelerations

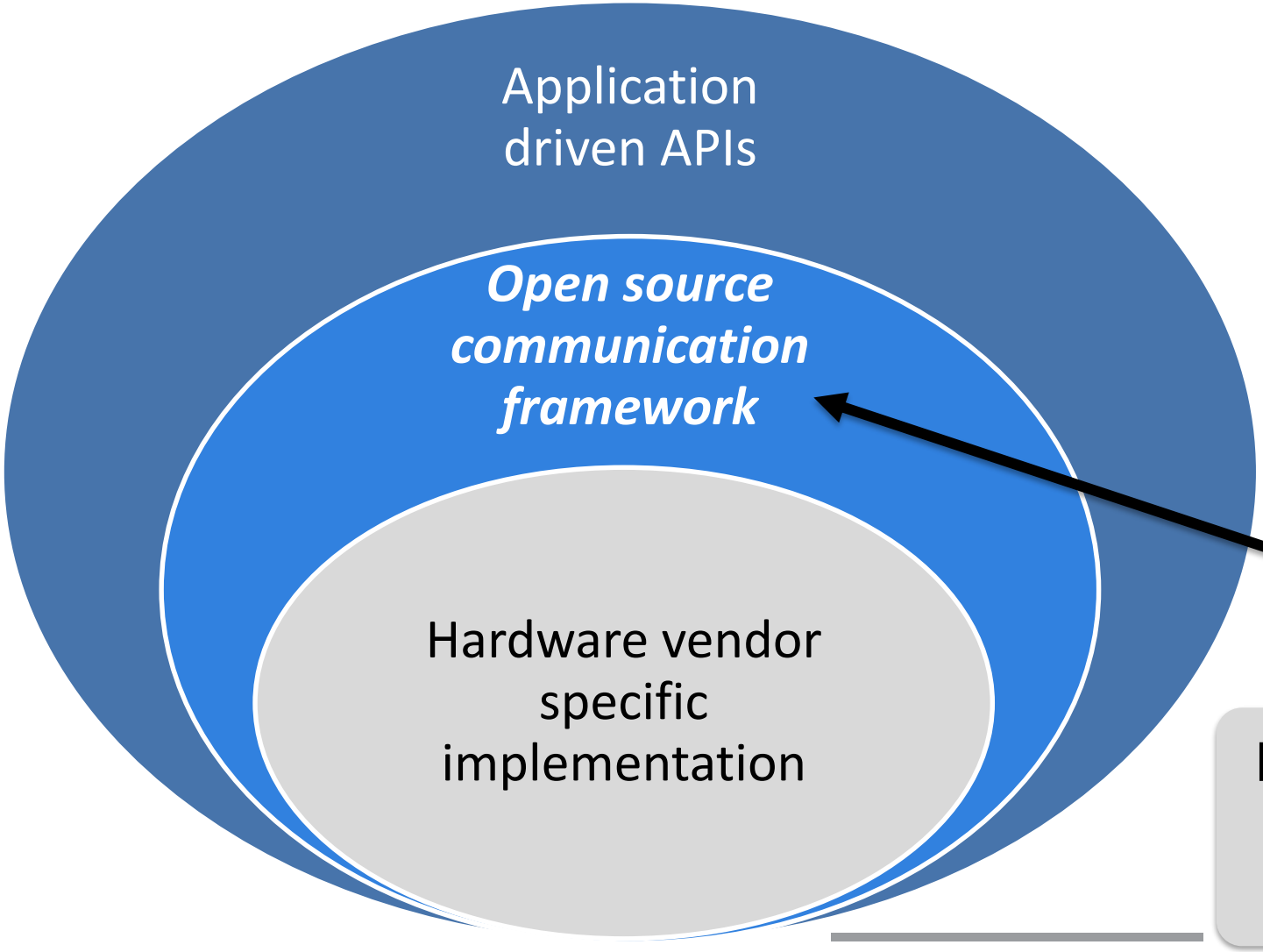


Provide necessary input parameters and output results

Support long-running functions (out-of-band execution)

Network protocol may need enhancements

PROPOSED VISION OF SOLUTION



Based on internal hardware prototyping

APIs targeting application use of specific accelerations

Extend existing communication framework to support acceleration functions

Define mechanism to pass input/output parameters and invoke acceleration

PROPOSAL (WORK IN PROGRESS)

- **Introduce new provider capability**
- **Extend attributes to request/report available accelerations**
- **Introduce new OFI object that corresponds to an acceleration**
  - Network function
  - Generic base definition
- **Specify network function with data transfers**
  - Apply to all transfers of a specific type
  - Specify per operation

NETWORK FUNCTIONS

New capability

Define well-known functions, allow for extensions

‘Chain’ groups multiple functions together as a single larger function

Generic structure to request/report available functions

Returned by existing fi_getinfo() call

Extend domain attributes

```
#define FI_NETWORK_FUNC    (1ULL << ?)

enum {
    /* well known functions */
    fi_nf_noop,
    fi_nf_chain,
    …,
    /* OR in FI_PROV_SPECIFIC for
     * vendor specific functions
     */
};

struct fi_nf_info {
    struct fi_nf_info *next;
    int                type;
    uint64_t           caps;
    uint64_t           mode;
    uint64_t           flags;
    void              *data;
    size_t             data_len;
};
```

NETWORK FUNCTIONS

Open a network function

```
int fi_network_func(domain,
    struct fi_nf_info *nf_info,
    void * context,
    uint64_t flags,
    struct fid_nf **nf);
```

Associate function with endpoint

- Support providers that must configure function and endpoint prior to use
- Can specify types of data transfers to apply function to
- Or indicate that function will be specified when submitting the data transfer

```
fi_ep_bind(ep, nf, flags);
e.g. flags = FI_SEND | FI_RECV
e.g. flags = FI_WRITE | FI_REMOTE_WRITE
e.g. flags = 0
```

NETWORK FUNCTIONS

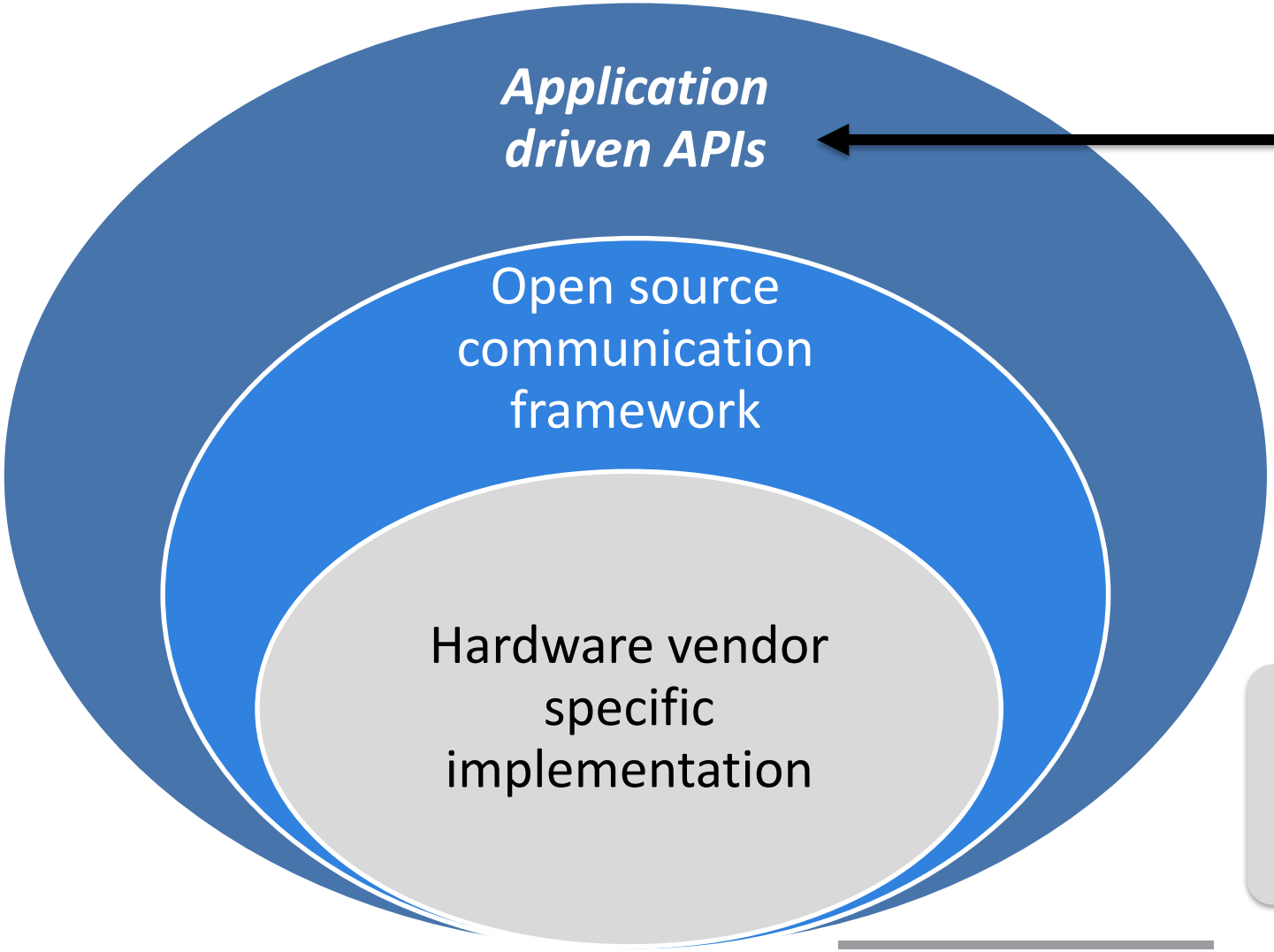
- **Specify function to apply to the current data transfer via existing context parameters**
  - Provide any needed input/output parameters

- **Re-use deferred work queues to execute long-running functions separate from current data transfer**
  - Assumes results will be used by future transfer(s)

```
struct fi_nf_context {
    struct fid_nf *nf;
    void          **params;
    size_t         param_cnt;
    size_t         *param_len;
    void           *reserved[4];
};
```

```
struct fi_deferred_work { … }
fi_control(…)
FI_QUEUE_WORK
FI_SUBMIT_WORK
FI_CANCEL_WORK
FI_FLUSH_WORK
```

NEXT STEPS

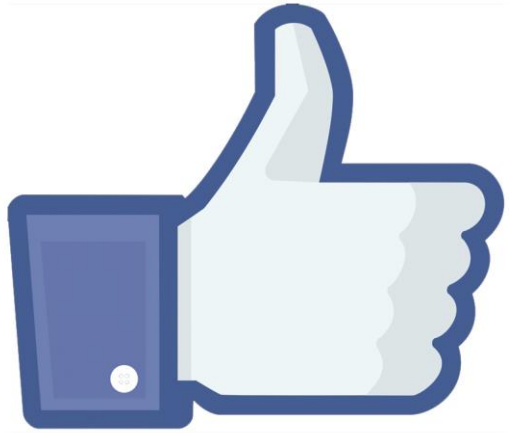


Identify common accelerations to drive ‘friendlier’ APIs

Coordinate changes with heterogeneous memory support

Expand solution to include Smart Networks (e.g. collectives offloads)

AND I CAN'T FORGET TO GIVE A CALL-OUT



THANK YOU

LEGAL DISCLAIMER & OPTIMIZATION NOTICE

- **Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.**
- **INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.**
-

Optimization Notice

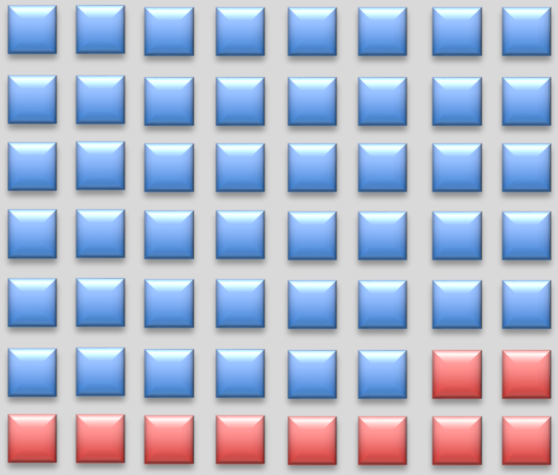
Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

OFFLOADS CAN PROVIDE SIGNIFICANT TCO SAVINGS

Not all cores are available for server workloads

Multi-core Server Node

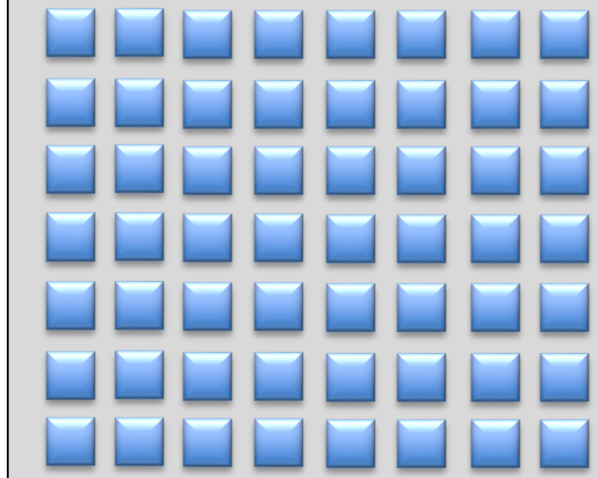


Infrastructure functions running on the host

Better utilization of general purpose cores

SmartNICs can provide full functional offload

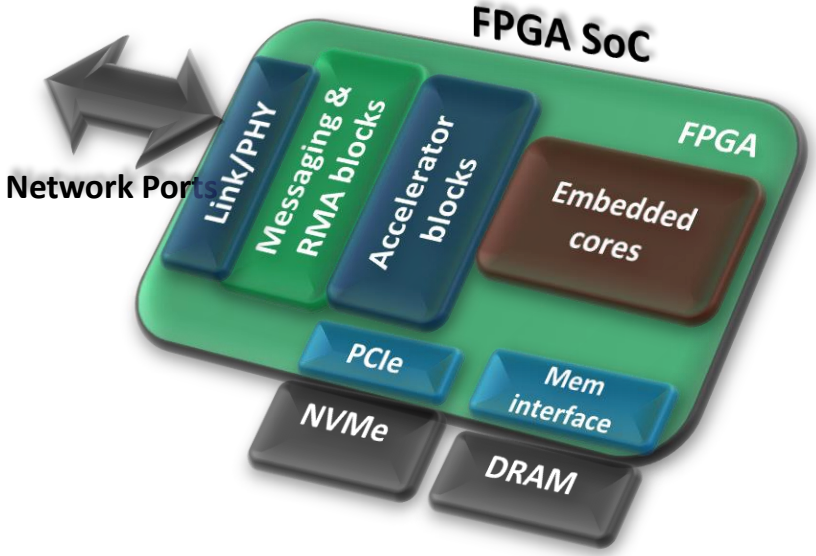
Multi-core Server Node



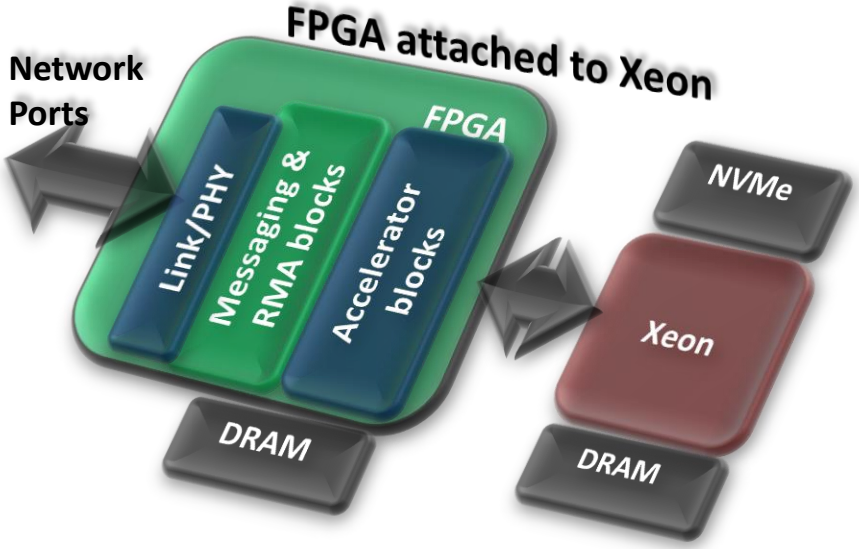
Standard NIC

SmartNIC with functional offloads

WHAT DO SMARTNICS LOOK LIKE? FPGA EXAMPLES…



SOC version



Discrete version