RF Convergence: From the Signals to the Computer

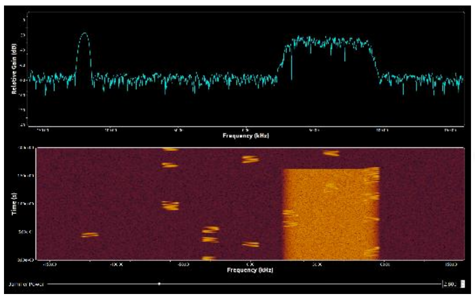
Tom Rondeau
DARPA/MTO

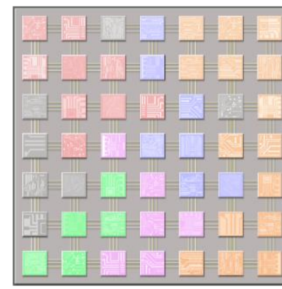
IEEE Workshop on 5G Technologies for Tactical and First Responder Networks
Columbia, MD

10/23/2018

RF Convergence

Spectrum ***Receiver/Exciter*** ***Processor*** ***Apps, Development***





C/C++
TensorFlow
GNU Radio
Python
Etc.

Access to unprecedented amount of spectrum and instantaneous bandwidth

Flexible, reconfigurable hardware enabling multiple missions from same hardware

- Comms
- Radar
- EW
- SIGINT

More bandwidth needs more processing and/or smarter math

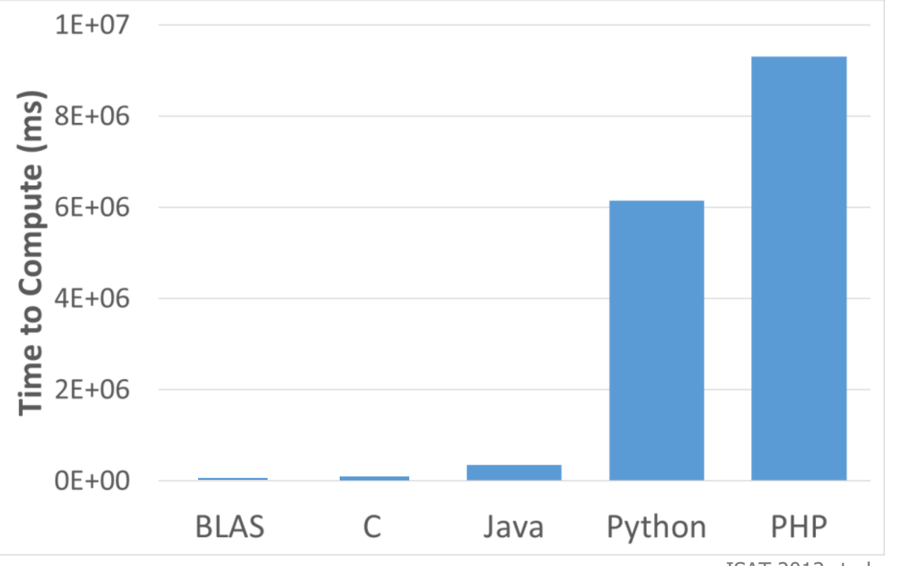
Development tools to quickly produce new applications and approaches to spectrum use

Dynamic responses and speed to new solutions need good programming models and efficient processors

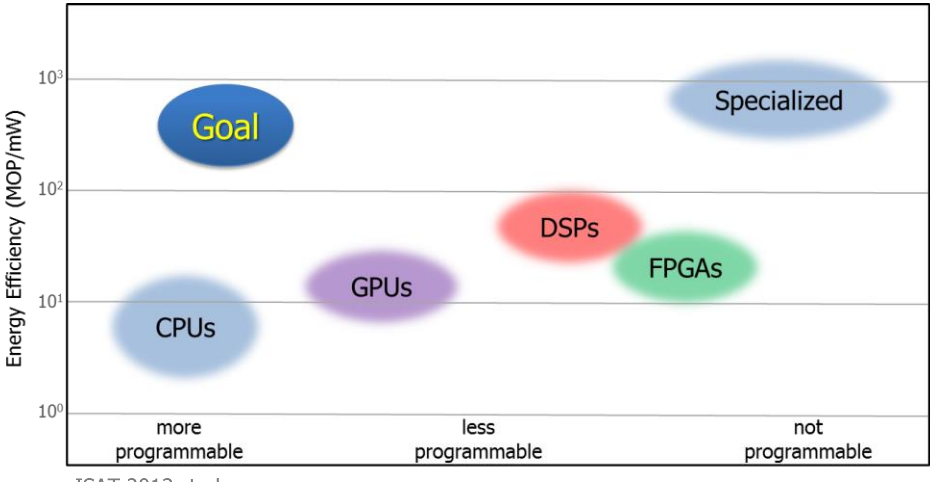
Programmability

- Productivity has come at the cost of compute efficiency
- Abstraction tends to ignore the underlying hardware

Matrix Multiply (ISAT 2012 study)



ISAT 2012 study



ISAT 2012 study

Specialization

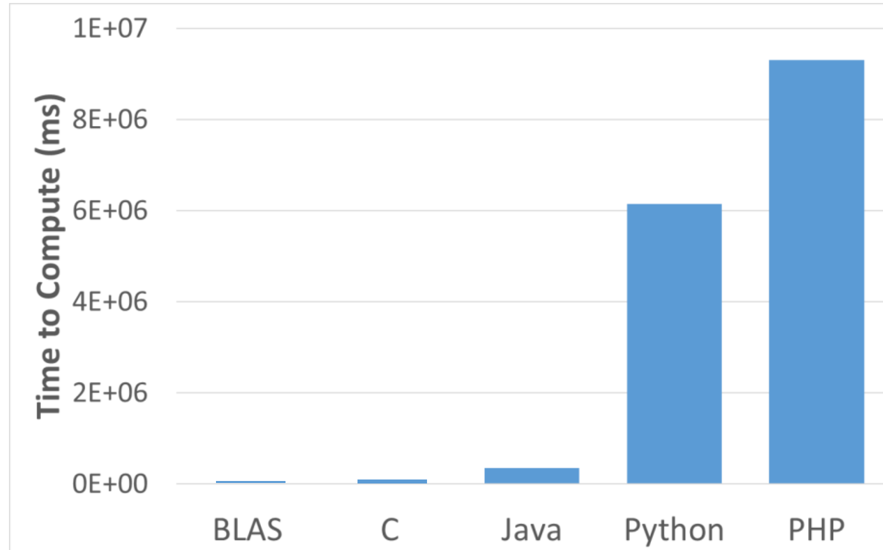
- Performance has come at the cost of usability
- Difficulty in programming and system integration

Developing new solutions for RF converged devices need flexibility in the hardware and support for programming

Programmability

- Productivity has come at the cost of compute efficiency
- Abstraction tends to ignore the underlying hardware

Matrix Multiply (ISAT 2012 study)



ISAT 2012 study

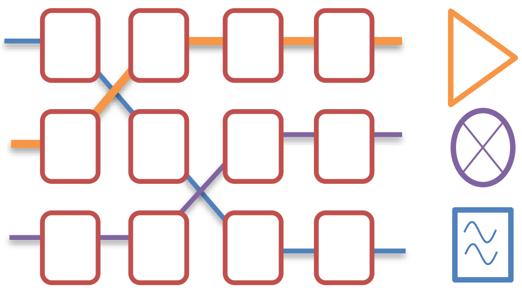
ISAT 2012 study

Specialization

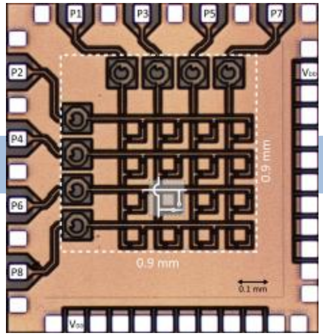
- Performance has come at the cost of usability
- Difficulty in programming and system integration

The DARPA RF-FPGA program

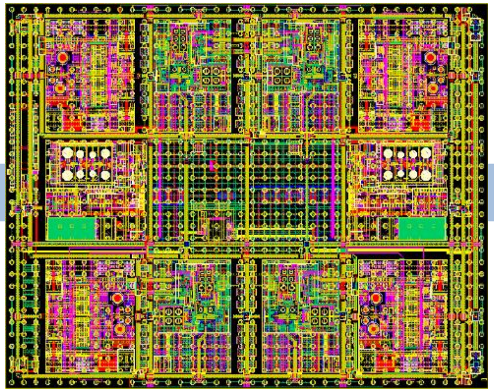
Can we create an FPGA-like fabric of switches and programmable RF components?



Concept

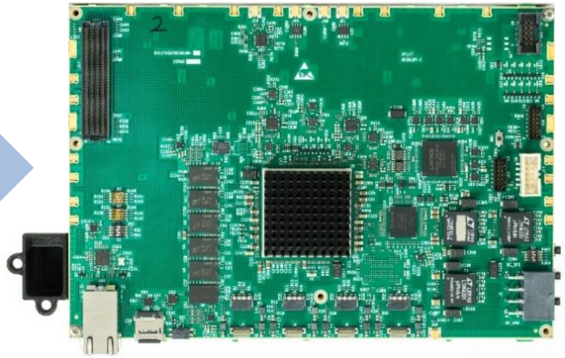


Proof of Concept



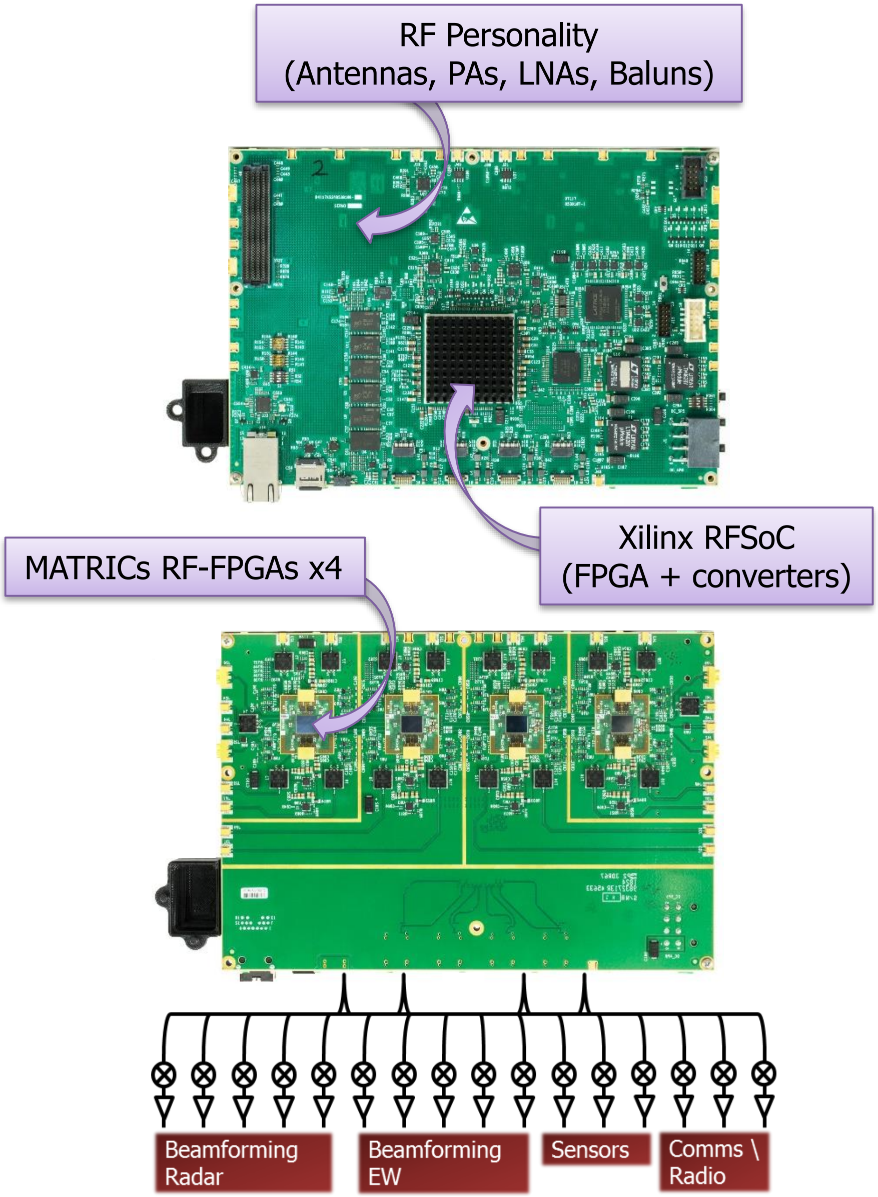
Prototype

Demonstration



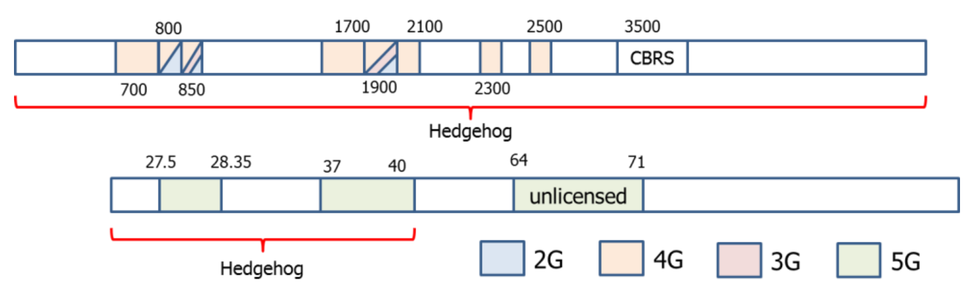
Integrated System

Hedgehog: Multifunction, multichannel RF Convergence device

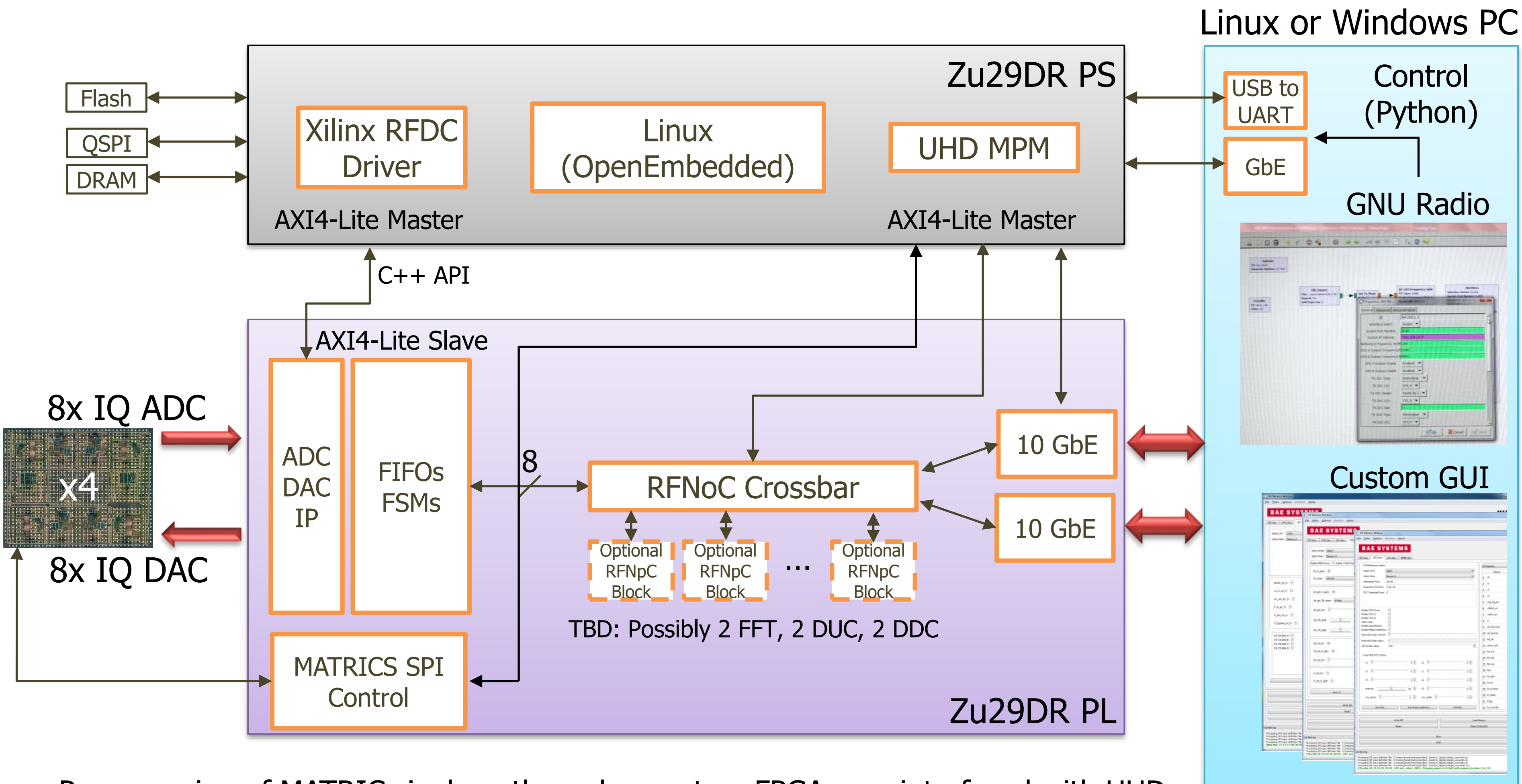


Parameter	Specification
Freq. Range	DC – 40 GHz
IBW	10 MHz - 2 GHz
Channels	8 Tx, 8 Rx
Integrated processing	GPP and FPGA Over 280 Gbps I/O
Converters	16 x 14 bit DACs 16 x 12 bit ADCs Integrated with processor

Covering all the G's



Programming on Hedgehog



- Programming of MATRICs is done through a custom FPGA core interfaced with UHD
- Programming of the ADCs/DACs is done through a Xilinx Driver with a C/C++ API
- Tools will be available under the open-source GNU General Public License version 3 (GPLv3)
    - Gov’t users may have early access to alpha/beta versions

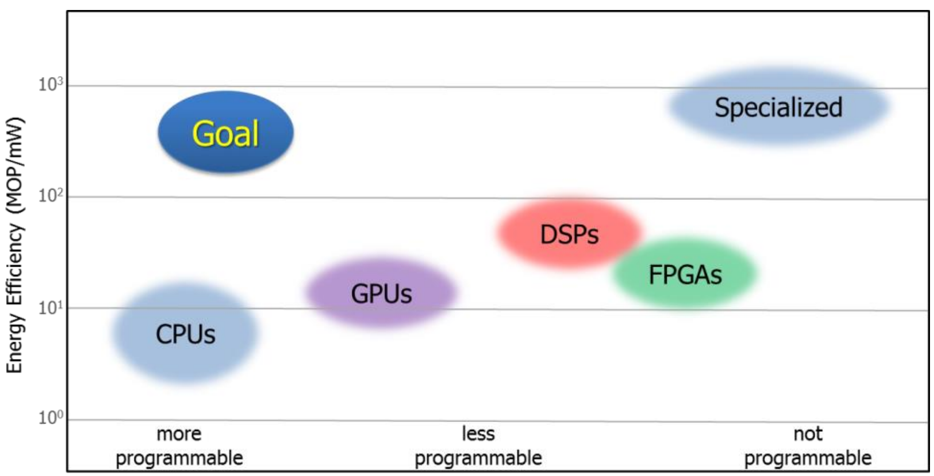
Efficient processing through use of specialized processors – and the development environment to support the complexity

Matrix Multiply (ISAT 2012 study)

ISAT 2012 study

Programmability

- Productivity has come at the cost of compute efficiency
- Abstraction tends to ignore the underlying hardware

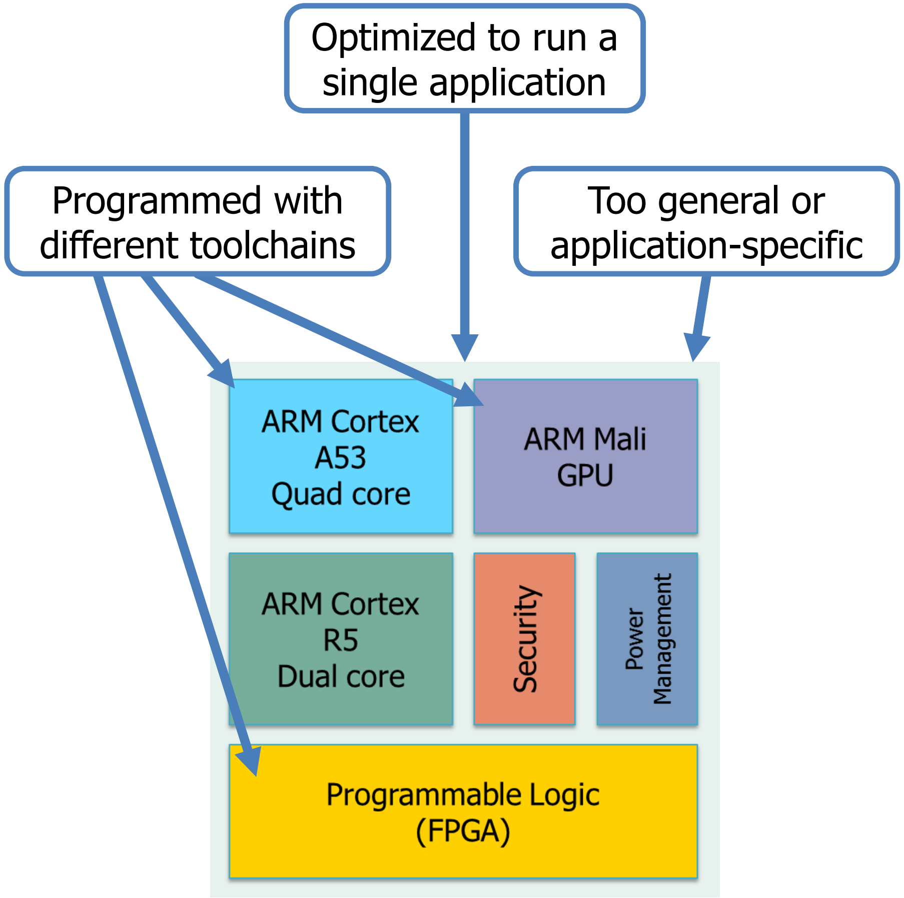


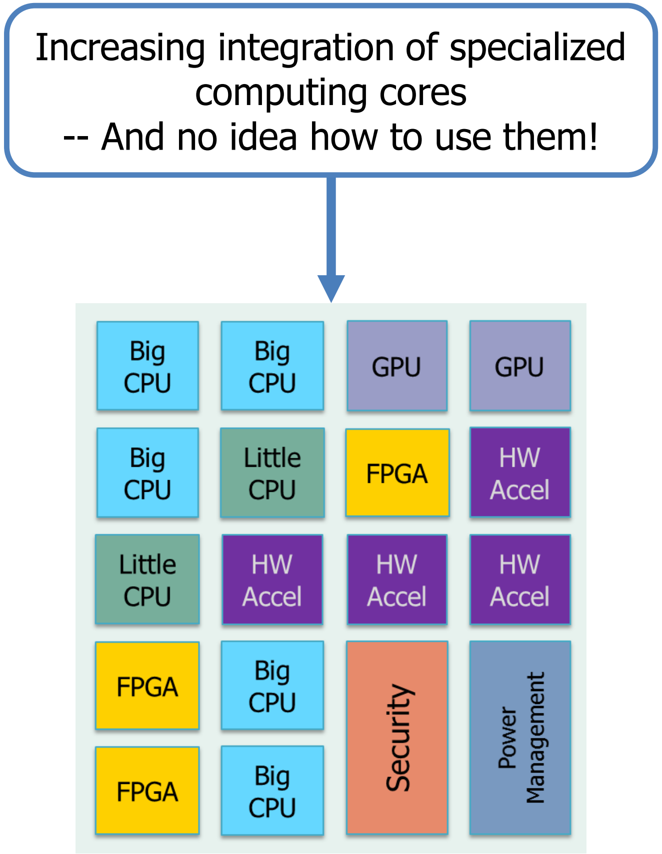
ISAT 2012 study

Specialization

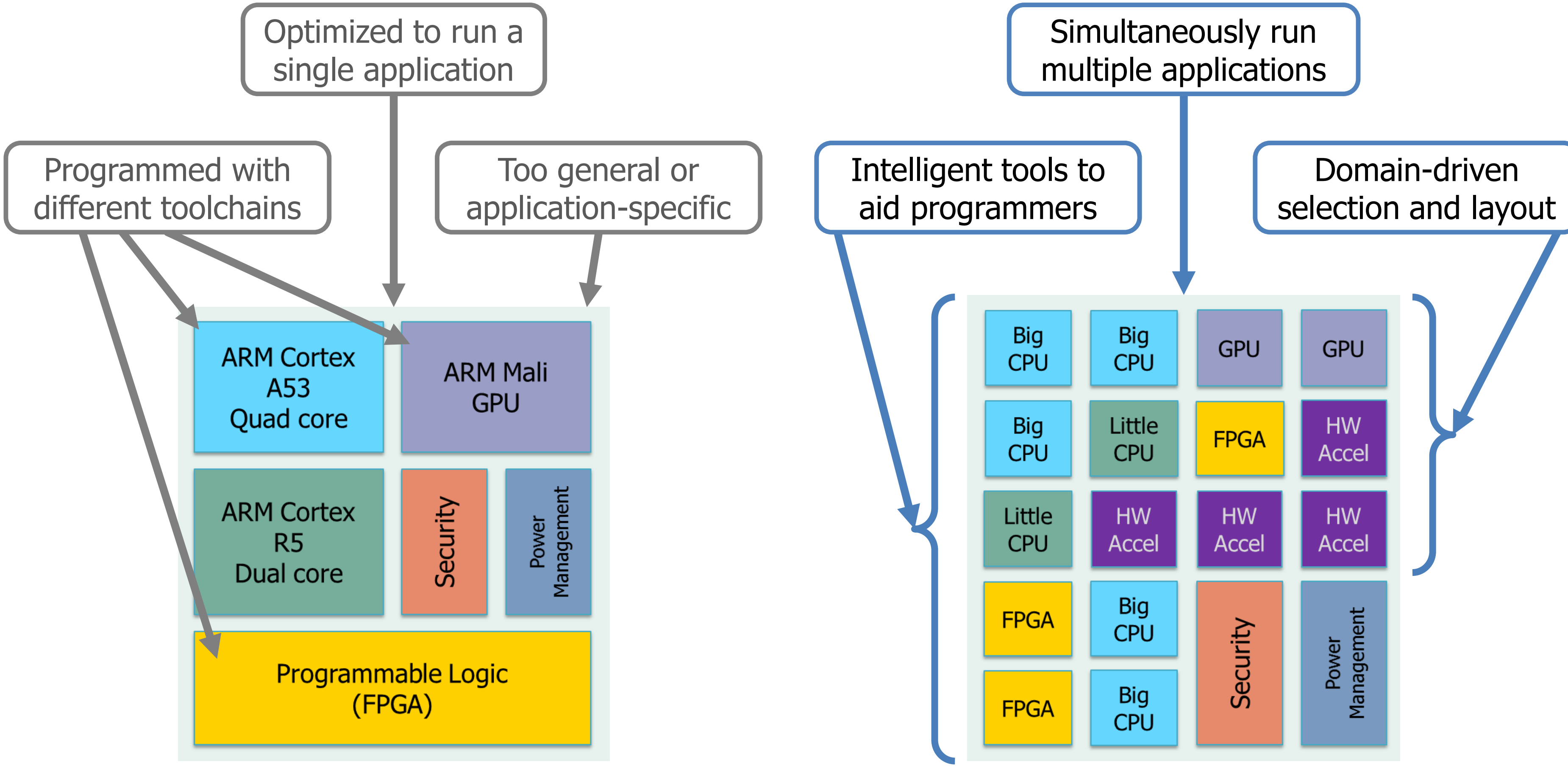
- Performance has come at the cost of usability
- Difficulty in programming and system integration

There is a trend to integrate more capabilities into a System-on-Chip





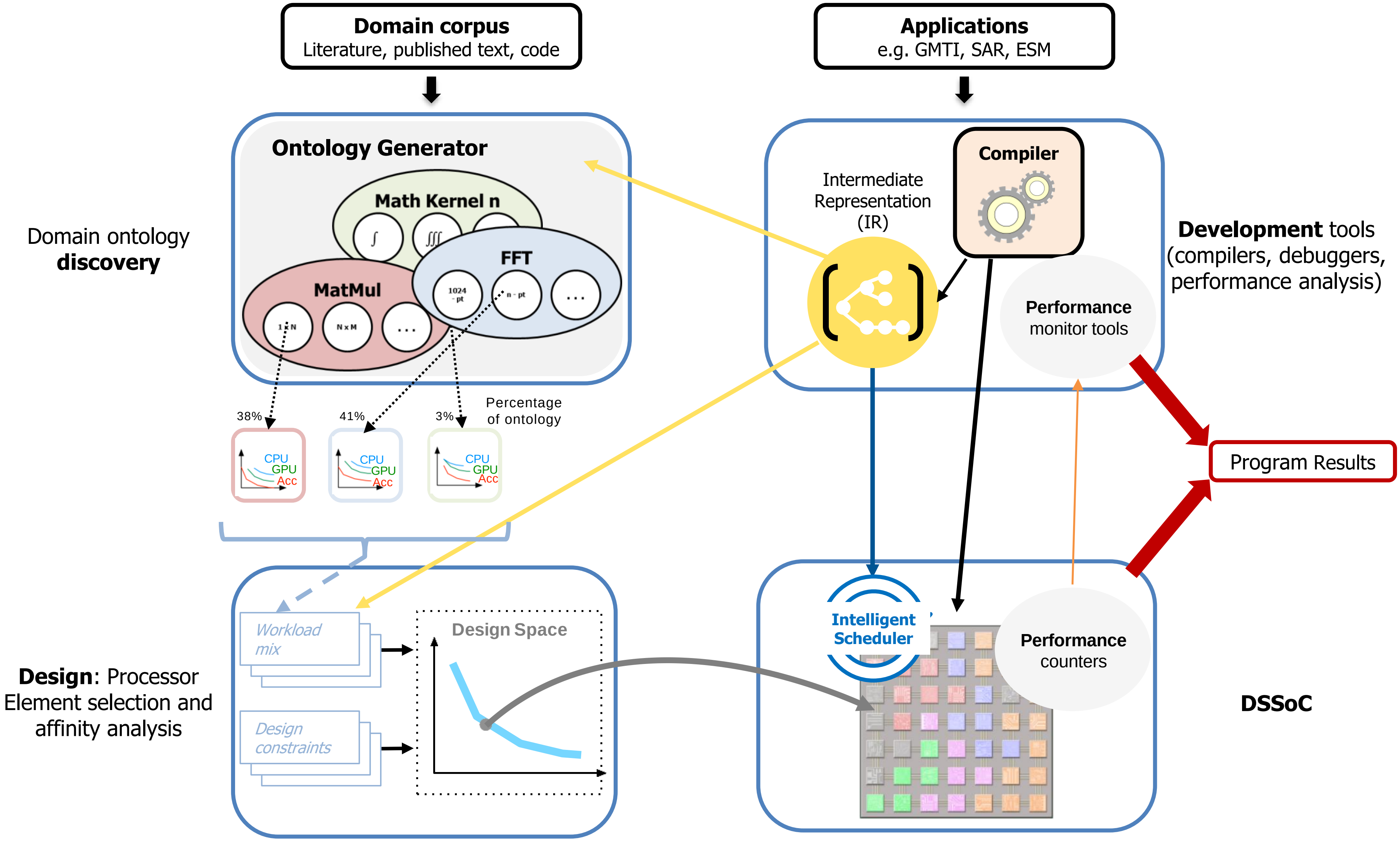
Domain-Specific System on Chip (DSSoC)



We cannot fully utilize our current multi-processor systems

We want to manage an increase of capabilities with integrated specialized accelerators

DSSoC: Domain-driven design

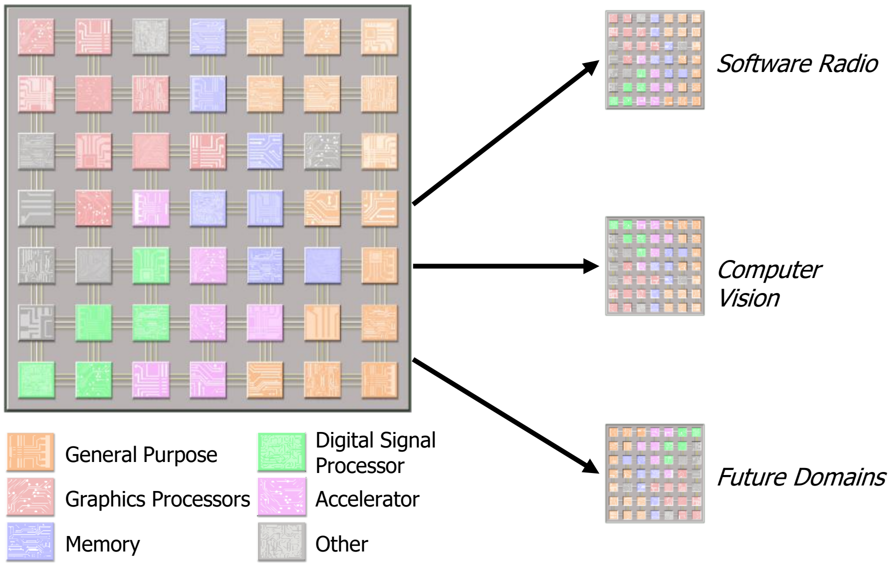


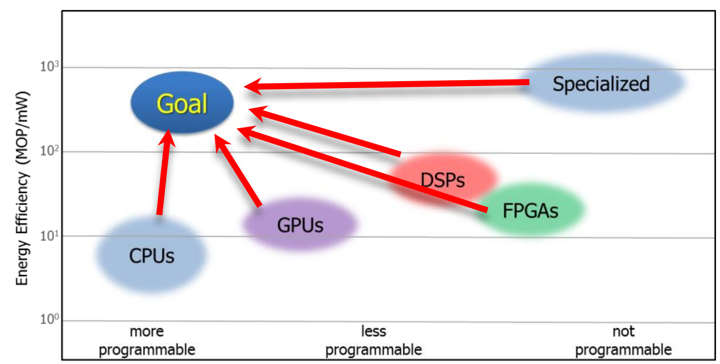
The DSSoC program will…

- Create a development ecosystem that takes advantage of the specialized hardware with no added burden to the programmer

- Design an intelligent scheduler for efficient data movement between DSSoC processor elements

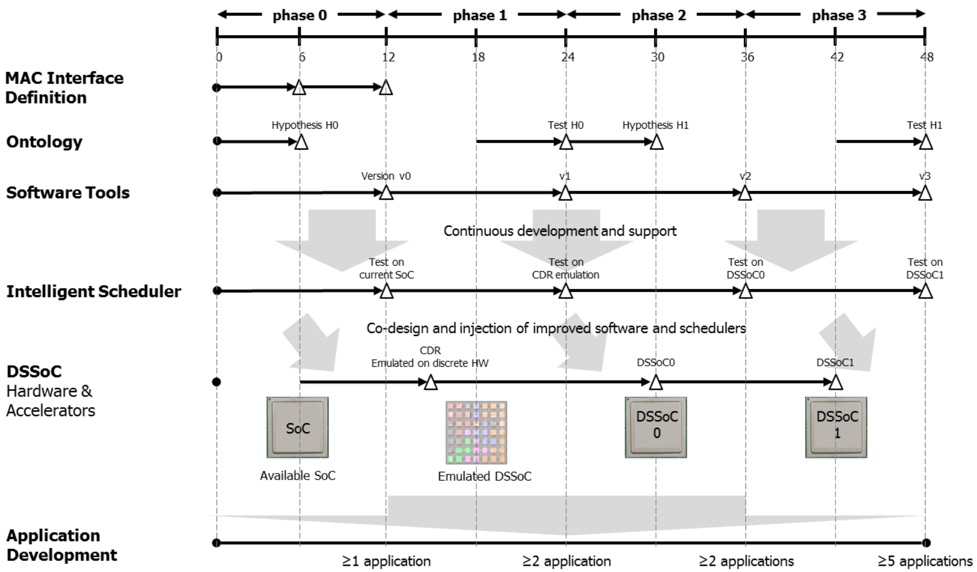
- Build a DSSoC of advanced, heterogeneous processors and accelerators for software radio





DSSoC will enable rapid development of multi-application, heterogeneous systems through a single programmable device

DSSoC program and performer overview



DSSoC Performers

Arizona State University

IBM

Oak Ridge National Laboratory

Stanford University

	Phase 1	Phase 2	Phase 3
Chip & Scheduler			
Number of simultaneous apps	≥2	≥2	≥5
Integration time for new accelerators[1]		≤3 months	≤3 months
Power savings relative to previous phase		≤80%[2]	≤80%[3]
Utilization of PEs[4]	≥80%		≥90%
Max. time per scheduler decision	≤500 ns	≤50 ns	≤5 ns
Medium Access Control (MAC)			
Latency (PE to PE)	≤500 ns	≤50 ns	≤5 ns
Throughput (PE to PE)	≥25 Gbps	≥50 Gbps	≥100 Gbps
Power	≤50% of chip	≤40% of chip	≤20% of chip

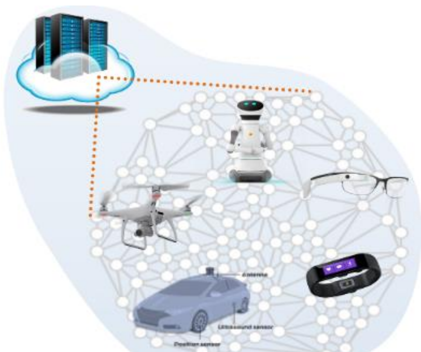
Power Constraints	
Embedded System (cell phone)	≤ 5 W
Portable System (laptop)	≤ 25 W

1. Three months to integrate new accelerators into DSSoC; enforced by program timeline
2. Compare the intelligent scheduler on DSSoC0 to the intelligent scheduler controlling the commercial SoC from phase 0.
3. Compare the intelligent scheduler on DSSoC1 to the intelligent scheduler on DSSoC0.
4. Ontology explains the required PEs and utilization; measure average utilization over developed apps.

DSSoC performer domains and applications

IBM T. J. Watson Research Center
Pradip Bose
Columbia University, Harvard University, Univ. of Illinois at Urbana-Champaign

CV+SDR
- Multi-domain application
- Multi-spectral processing
- Communications



IBM

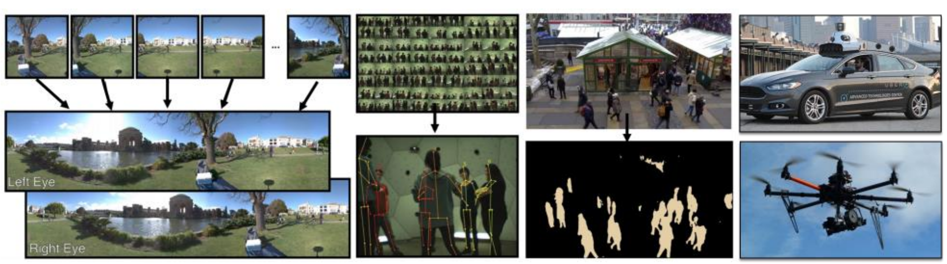
Arizona State University
Daniel W. Bliss
Univ. of Michigan, Carnegie Mellon University, General Dynamic Mission Systems, Arm Ltd., EpiSys Science

SDR
- Unmanned aerial
- Small robotic & leave-behind
- Universal soldier systems
- Multifunction systems

PlastyForma

Stanford University
Mark Horowitz
Clark Barrett, Kayvon Fatahalian, Pat Hanrahan, Priyanka Raina

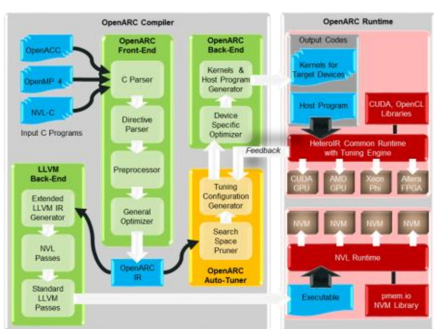
Computer Vision
- Still image and video processing
- Autonomous navigation
- Continuous surveillance
- Augmented reality



Stanford Google/YouTube

Oak Ridge National Laboratory
Jeffrey Vetter

SDR
- Communications and signal processing focused
- Up-front processing / data cutdown
- Improving understanding of processing systems



ORNL

DARPA

www.darpa.mil