# Implementing MATLAB Communications Models on the HyperX 100 Core Processor

2012 Wireless Innovation Forum European Conference on Communications Technologies and Software Defined Radio
(SDR'12 – WInnComm – Europe)

WIRELESS INNOVATION FORUM™

27 - 29 June • Brussels, Belgium

**John Irza**

***Solutions Architect***

***Coherent Logix, Inc.***

***2012-06-29***

Coherent Logix™

Enabling Low Power Software Defined Systems

*Photo Courtesy NASA*

# Agenda

- Introduction
- Is MATLAB “Deployable”?
- Algorithms to Hardware
- Deployable Targets
- HyperX Processor
- HyperX Development Tools
- Conclusions

# Company Profile

**Coherent Logix designs and produces low-power, high performance, C-programmable processors (HyperX™) and RF chipsets (*rf*X™) for the embedded systems market – enabling low-power, real-time software defined systems.**

Portland, OR
San Jose, CA
Austin, TX
Boston, MA
Tokyo, Japan
Atlanta, GA

Wireless

Image / Video

Mil / Aero

High-Rel / Rad-Tol

Locations

Applications

# Is MATLAB “Deployable”?

**To the algorithm developer:**

- “Deployable” = deployable to HW accelerators
- GPUs, server farms, super-computer facilities, etc.

**To the product developer:**

- “Deployable” = deployable to end-products
- GPPs, DSPs, FPGAs, ASSPs, ASICs

**MATLAB can generate “C”**

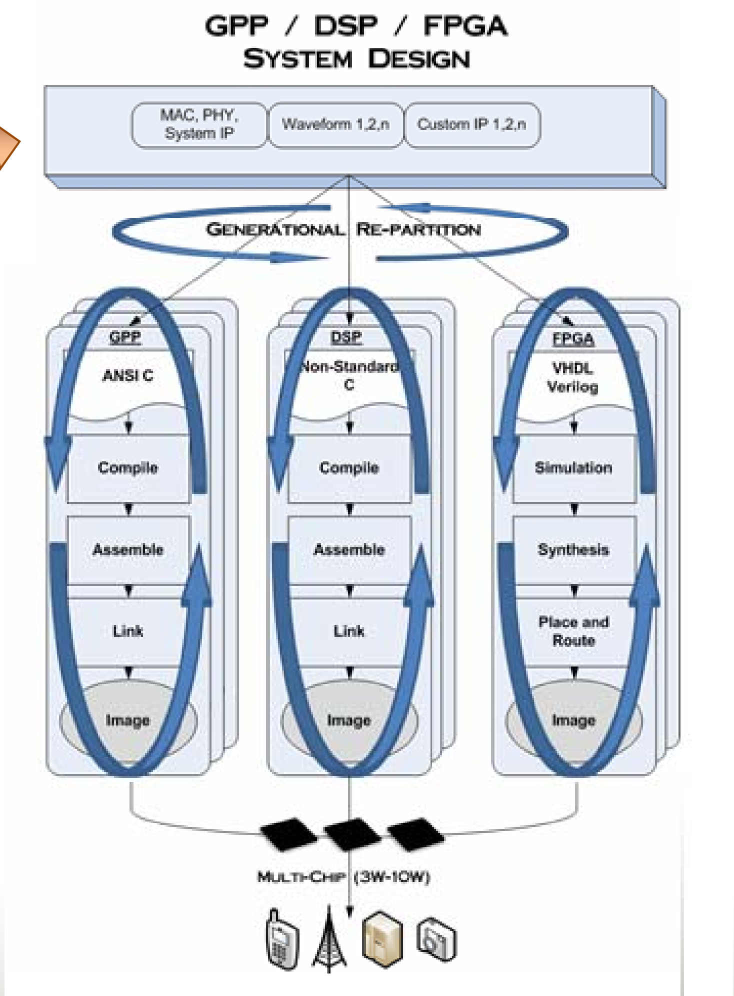
- *MATLAB Coder*: MATLAB → C
- *Simulink Coder*: Simulink & MATLAB Function Blocks → C

**MATLAB has multi-threaded capabilities**

- Limited threading but slowly increasing
- Closed solution, tied to OS/CPU

# Algorithms to Hardware

- **The tortured paths: design & verification**
- **Golden models written in MATLAB**
- **Tossed "over the wall"**
- **Re-written in C**
- **Re-written in HDL**
- **Re-writing code breaks the verification flow**
- **Broken flow results in:**
  - Longer product development time (loss time to market)
  - Longer simulation time (less time to explore algorithms & architectures)
  - Lost competitive advantage



Coherent Logix™
Enabling Low Power Software Defined Systems™

# Writing Code for Deployable Targets

## GPPs, DSPs

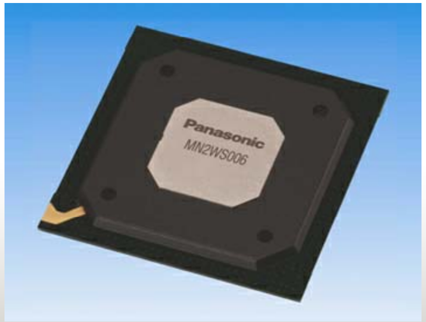
- Moderate development times
- Moderate speed operation

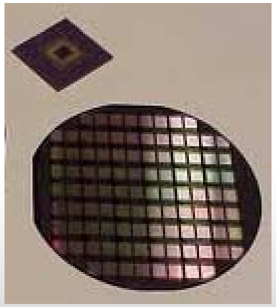
## FPGAs, GPUs

- Moderate→long development times
- High to very-high speed operation

## ASSPs, ASICs:

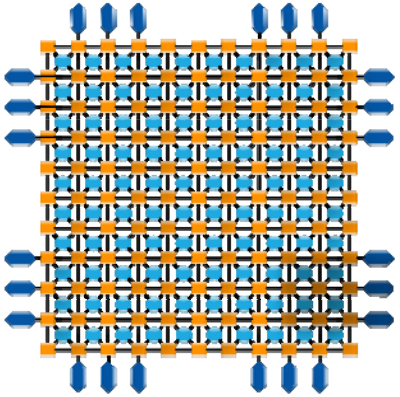
- Long development times
- High to very-high speed operation



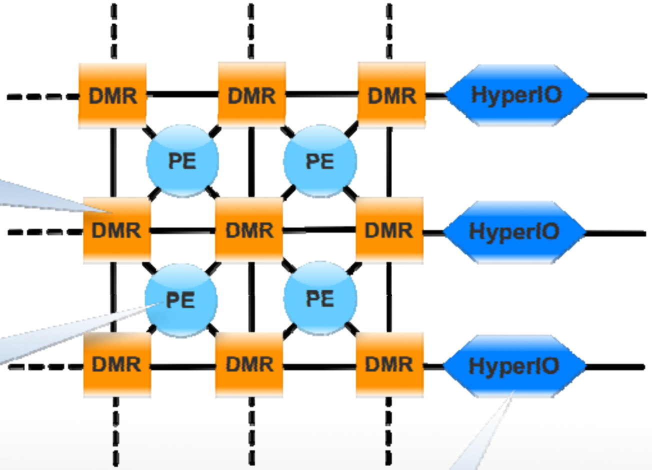


# HyperX Processor

## hx3100 Architecture



- 10 x 10 matrix of DSP cores (PEs)
- Floating-point and fixed-point
- 11 x 11 memory-network fabric (DMRs)
- "C" programmable
- Very low power (2.5W total)
- 24 High speed I/O peripheral ports



**121 Data Memory Routers**
8 kB data memory per DMR
8 independent DMA Engines
986 kB total configurable on chip memory
Real time algorithm topology switching
High Speed neighbor communications
High speed cross-chip communications
Autonomous Data Movement

**100 Processor Elements**
4 kB program memory per PE
500 MHz variable clock
8, 16, n x 16 integer
32 bit floating point
50,000 MIPS, 50 GMACS 16 bit
32 bit Floating Point, 25 GFLOPs

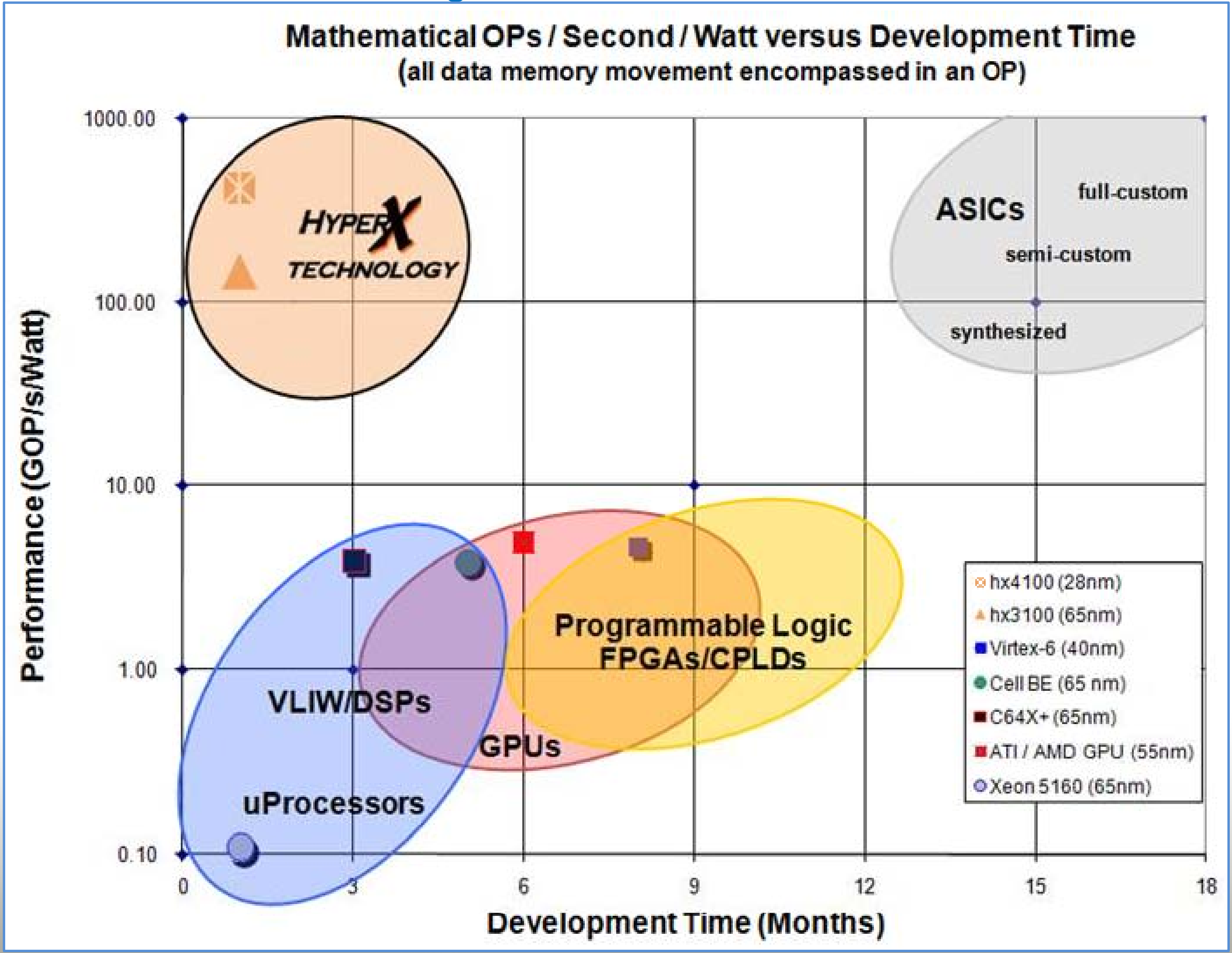
**High Speed I/O Routers**
16 HyperIO Channels, 16 bit wide (24 ch)[1]
DDR2: 4 channels, 32 Gbps (8 ch, 64 Gbps)[1]
LVDS: 8 channels, 64 Gbps (12 ch, 96 Gbps)[1]
CMOS: 6 channels, 12 Gbps (12 ch, 24 Gbps)[1]
104 Gbps simultaneous (168 Gbps)[1]
Access to 64 Gbps off-chip memory

[1] Available on die with full bond-out
[2] Fundamental mathematical operation, including all data movement

Coherent Logix™
Enabling Low Power Software Defined Systems™

# What is HyperX Technology? *Low Power Processing, Redefined.*

## Low Power ♦ High Performance ♦ Software Defined ♦ Scalable



**Comparison**

| | |
|---|---|
| Intel GPP: | 10,000 pJ/op |
| ARM: | 500-1000 pJ/op |
| DSP, FPGA: | 200-350 pJ/op |
| GPU: | 3000-200 pJ/op |
| ASICs: | 8-20 pJ/op |
| **hx3100B:** | **~8 pJ/op** |

**VLIW/DSP Architectural Barriers to power:**

- Multilevel cache
- Buses
- Data Path Efficiency

**FPGA' Architectural Barriers to power:**

- Programmable Wire
- Fine grained parallel
- Forced distributed memory
- Timing closure

Coherent Logix™ Enabling Low Power Software Defined Systems™

# The hx3100B Processor Performance

## Performance Throughput

- @ 500 MHz
- 50,000 MIPS
- 50 16-bit GMACS
- 100 8-bit GMACS
- 25 GFLOPS (32-bit)

## DIE IO Bandwidth:

- 96 Gbps of LVDS IO
- 24 Gbps of CMOS IO
- 64 Gbps of DDR2 IO

## General Purpose Package IO Bandwidth:

- 64 Gbps of LVDS IO
- 12 Gbps of CMOS IO
- 32 Gbps of DDR2 IO

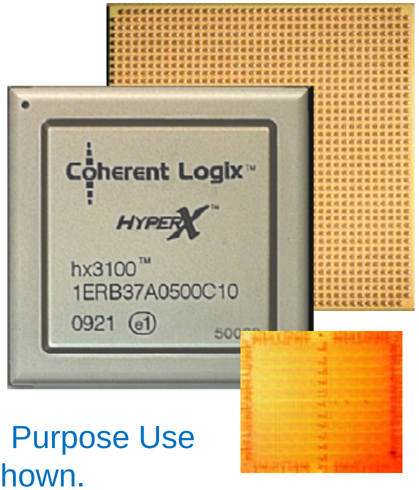
***Scalability***

***Performance***
***Power***
***Programmability™***

***25 mW to 2.5 W***
***(algorithm dependent)***

## Performance Efficiency

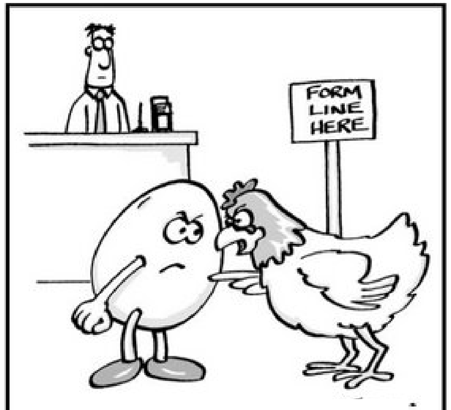
- @ 500 MHz
- 64 16-bit GMAC/s/W
- 128 8-bit GMAC/s/W
- 32 GFLOP/s/W (32-bit)
- > 2.4 TOP/s/W (16-bit RISC equivalent)



(*) General Purpose Use Package Shown.
Die Intended To Be Packaged To Application, e.g. below.

Coherent Logix™
Enabling Low Power Software Defined Systems™

# Development Tools



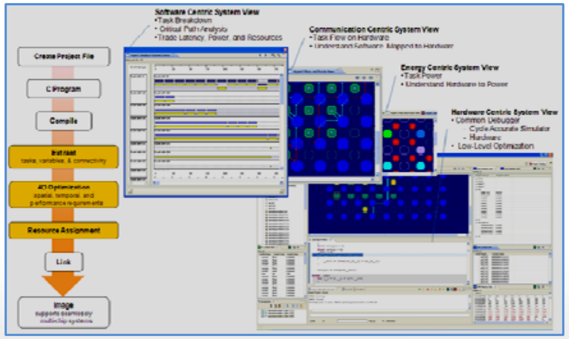
No, you back off. I was here before you.”

## Which came first?

## Chicken vs. Egg

## Software vs. Silicon

### HyperX ISDE

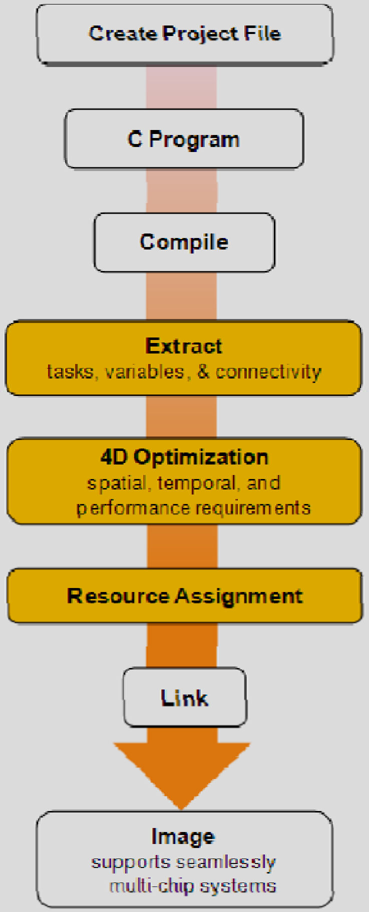


**vs.**

### HyperX Chip

**HyperX software tools were developed for many years before the first chip was fabbed.**

# HyperX Integrated System Development Environment

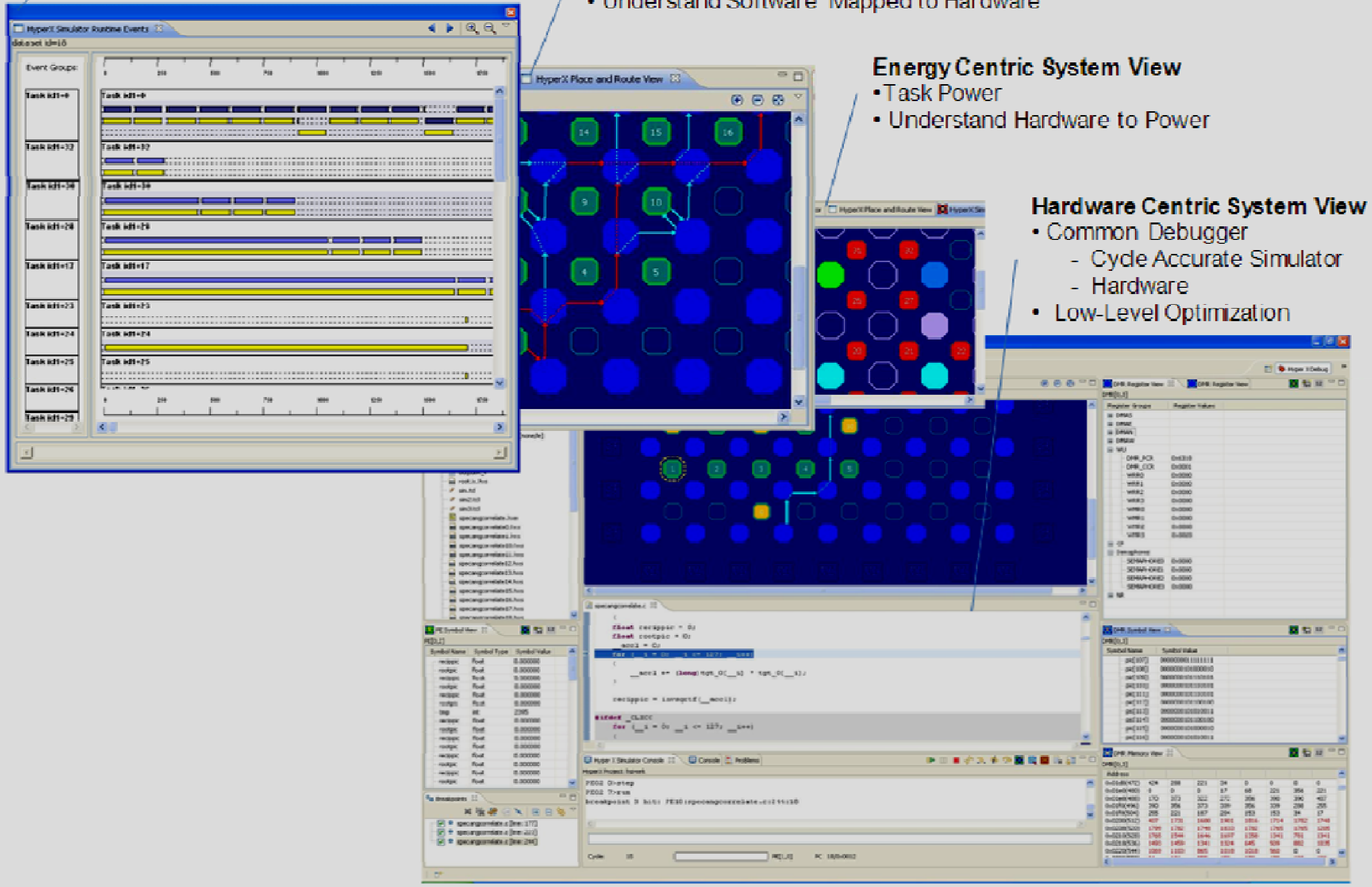


**Software Centric System View**
- Task Breakdown
- Critical Path Analysis
- Trade Latency, Power, and Resources

**Communication Centric System View**
- Task Flow on Hardware
- Understand Software Mapped to Hardware

**Energy Centric System View**
- Task Power
- Understand Hardware to Power

**Hardware Centric System View**
- Common Debugger
  - Cycle Accurate Simulator
  - Hardware
- Low-Level Optimization



Coherent Logix™
Enabling Low Power Software Defined Systems™

# HyperX ANSI-C Development Flow

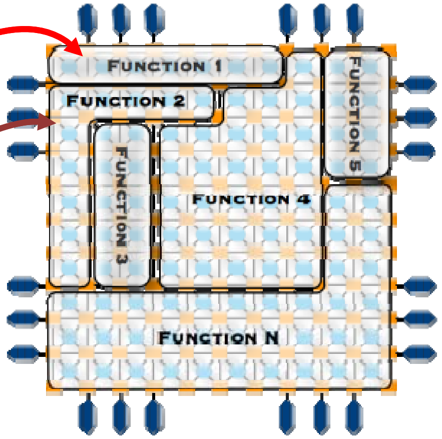
Original ANSI C Program

```
int buf1[8], buf2[8];

// function 1
void func1(void)
{
     ....
}

// fucntion 2
void func2(void)
{
     ...
}

int main(void)
{
     // Call function 1
     func1();

     // Call function 2
     func2();
} // main()
```

ANSI C Program with *MPI-API*
*that expresses system parallelism*

```
int buf1[8], buf2[8];

// function 1
void func1(void)
{
     ....
}

// fucntion 2
void func2(void)
{
     ...
}

int main(void)
{
     // Task 0
     if (MPX_RANK == 0){
       // Call function 1
       func1();
       // Send to Task 1
       MPX_Send(&buf1, 8, MPX_INT, 100);
     }
     // Task 1
     if (MPX_RANK == 1){
       // Receive from Task 0
       MPX_Recv(&buf2, 8, MPX_INT, 100);
       // Call function 2
       func2();
     }
} // main()
```

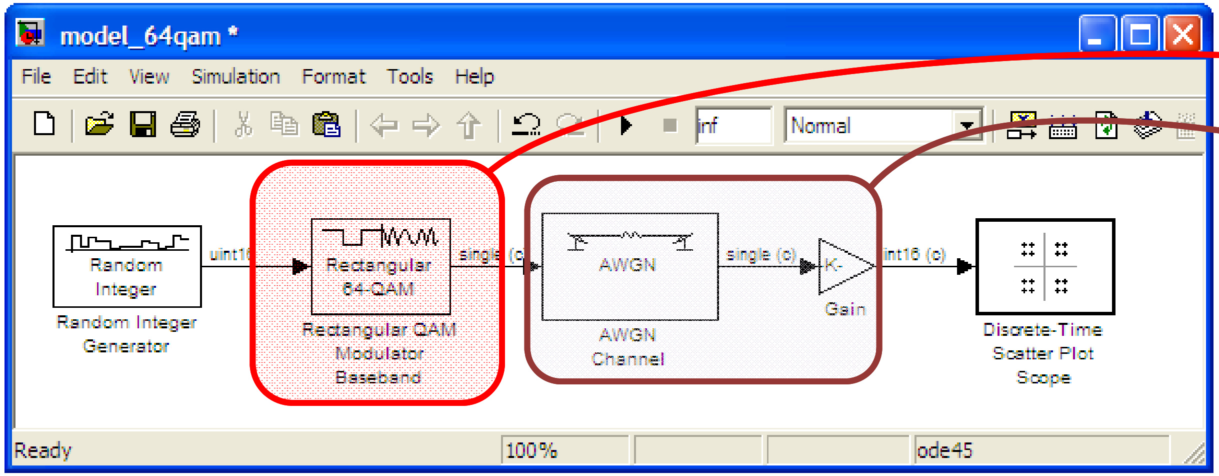
Deploy
Parallel Functions
across the 100 cores

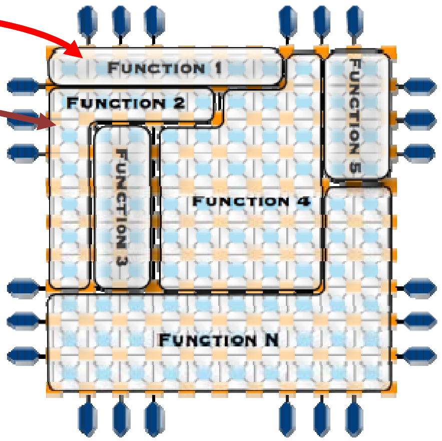


# HyperX Simulink Development Flow

Simulink Model
with specific blocks targeted
for HyperX implementation

Deploy
Specific blocks
across the 100 cores

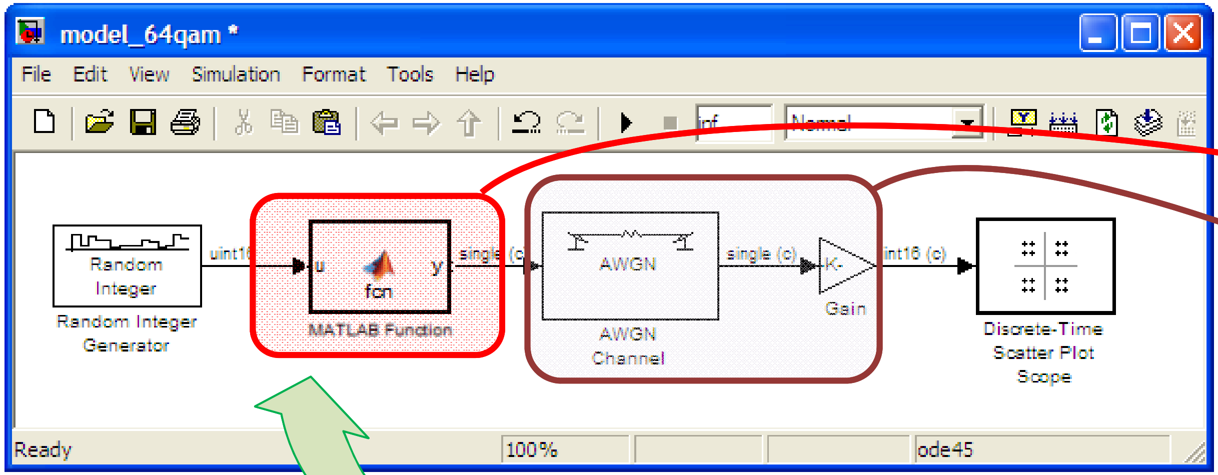


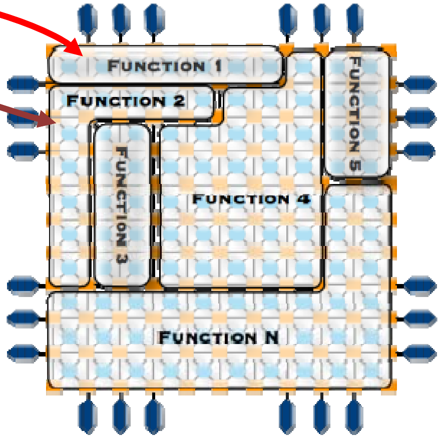


# HyperX MATLAB Development Flow (Today)

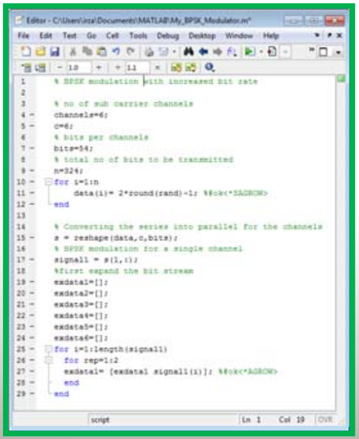
Simulink Model
with specific blocks targeted
for HyperX implementation

Deploy
Specific blocks
across the 100 cores





“The MATLAB Function block lets you compose a MATLAB language function in a Simulink model that generates embeddable code.”


Editor - C:\Users\ira\Documents\MATLAB\My_BPSK_Modulator.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
script
Ln 1 Col 19 OVR

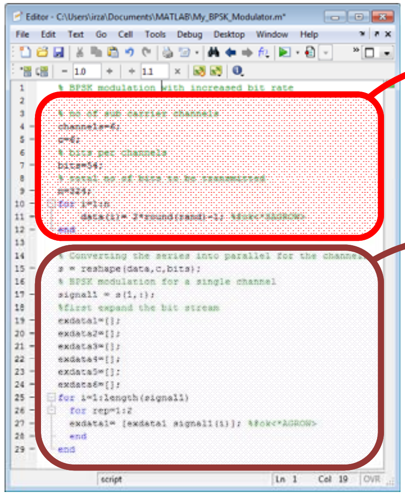
```
% BPSK modulation with increased bit rate

% no of sub carrier channels
channels=6;
c=6;
% bits per channels
bits=54;
% total no of bits to be transmitted
n=324;
for i=1:n
    data(i)= 2*round(rand)-1; %#ok<*SAGROW>
end

% Converting the series into parallel for the channels
s = reshape(data,c,bits);
% BPSK modulation for a single channel
signal1 = s(1,:);
%first expand the bit stream
exdata1=[];
exdata2=[];
exdata3=[];
exdata4=[];
exdata5=[];
exdata6=[];
for i=1:length(signal1)
  for rep=1:2
  exdata1= [exdata1 signal1(i)]; %#ok<*AGROW>
  end
end
```

MATLAB code

Coherent Logix™
Enabling Low Power Software Defined Systems™

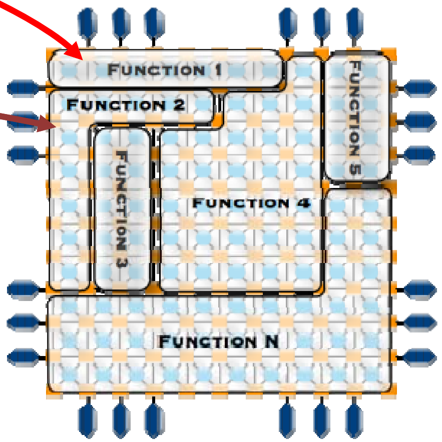
# HyperX MATLAB Development Flow (Future)

MATLAB Model
with specific functions targeted
for HyperX implementation


Editor - C:\Users\irza\Documents\MATLAB\My_BPSK_Modulator.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
script
Ln 1
Col 19
OVR

```
% BPSK modulation with increased bit rate

% no of sub carrier channels
channels=6;
c=6;
% bits per channels
bits=54;
% total no of bits to be transmitted
m=324;
for i=1:m
    data(i)= 2*round(rand)-1; %#ok<*AGROW>
end

% Converting the series into parallel for the channe
s = reshape(data,c,bits);
% BPSK modulation for a single channel
signal1 = s(1,:);
%first expand the bit stream
exdata1=[];
exdata2=[];
exdata3=[];
exdata4=[];
exdata5=[];
exdata6=[];
for i=1:length(signal1)
  for rep=1:2
  exdata1= [exdata1 signal1(i)]; %#ok<*AGROW>
  end
end
```

Deploy
Specific blocks
across the 100 cores



Q: How do we get there?

A: MathWorks now offers “MATLAB Coder”

→ No longer “forced” to use Simulink to generate C code from MATLAB

Coherent Logix™
Enabling Low Power Software Defined Systems™

# Conclusions

- **Is MATLAB "Deployable"?**
  - Yes, perhaps, but "C" is the *Lingua Franca*
  - YMMV ("Your mileage may vary")
- **Challenges of Common Deployable Targets**
  - GPPs, DSPs, FPGAs, GPUs: Do not solve the "system solution"
  - ASSPs, ASICs: Cost prohibitive, long design cycles
  - Software solution is written to a specific target
- **HyperX Processor**
  - HyperX software tools make using 100 cores practical
  - C and Simulink flows support traditional and MBDF designs
- **Deploying MATLAB on the HyperX**
  - Today: Use MATLAB Function Blocks in Simulink
  - Future: Use MATLAB system models directly (no Simulink needed)

# Thank you!

2012 Wireless Innovation Forum European Conference on Communications Technologies and Software Defined Radio (SDR'12 – WInnComm – Europe)

27 - 29 June • Brussels, Belgium

WIRELESS INNOVATION FORUM™

**John Irza**

**Solutions Architect**

**irza@coherentlogix.com**

**(781) 648-2144**

Coherent Logix™

Enabling Low Power Software Defined Systems

*Photo Courtesy NASA*