Lessons Learned
&
Ideal Architecture

Honeywell

This presentation is not subject to Export Control

Requirements for a Successful Exploration Program

Honeywell

Exceed the reliability of the Space Shuttle, as a minimum

Level 1 Requirement

No OTS solution exceeds the reliability of the Space Shuttle

- Low cost
- Open architecture
- Expandable and/or re-configurable in the future

Level 2 Requirement

No existing space avionics solution is low cost, open, and expandable

This presentation is not subject to Export Control

What Parts of an Avionics System Can Be Open?

Honeywell

- System Databus & Backplane
- Processor
- I/O types
- Software environment (OS, tools)
- Mechanical

This presentation is not subject to Export Control

Why Openness?

Honeywell

- Avoid constraint of having to go to one supplier
- Risk avoidance of a single source
- Reduced development costs
  - OTS
  - Common use
  - Multiple suppliers – competition lowers price
- Reduced maintenance costs
  - Tools & licenses; knowledge base; training
- Reduced upgrade costs
  - Obsolete parts
  - Change of mission
  - Technology migration
  - Integration costs

Definitions...

Honeywell

- **Openness has been defined in a variety of ways:**
  - **Standards**
    - Documented standard
    - Widely used standards or interfaces
    - Plug-and-play (standard HW /SW interfaces)
    - Non-proprietary interfaces
  - **Commercially Available (multiple sources)**
    - Commercially available end items
    - Commercially available software development tools
  - **Long life availability**
  - **Open source code**
  - **Third-party compatibility**

Commercially Available End Items - Definition

Honeywell

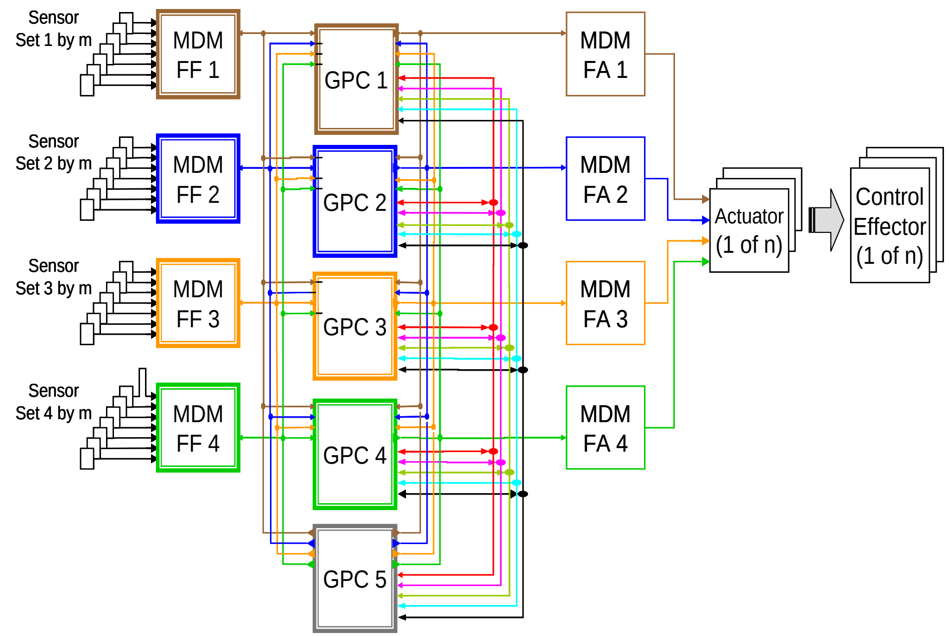
- **Definition**
  - **System architecture consists of readily-available similar parts, accessories, or enhancements provided by industry standard distributors**

- **Trade-off**
  - **Does maximum use of COTS components, lower the life cycle costs?**
    - ⇑ Minimizes the NRE
    - ⇑ Avoids a custom design/solution
    - ⇑ Faster development cycles
    - ⇔ Reduced support costs
    - ⇑ Standardization across large programs.
    - ⇓ Parts obsolescence is an industry concern

Shuttle Lessons Learned

Honeywell

- **Need for assured data consistency throughout the redundancy management system.**
  - **Shuttle uses a very complex method**
    - Four lock step synchronized computers
    - Redundant & monitored Terminal Units data flow
    - If any data does not match, the majority of the computers votes a “fault”
    - No autonomous deactivation of the computer
- **This scheme very complex compared to later architectures such as PAVE PACE used on ISS**
  - Expensive to build, code, and architect
  - Requires excess computational power
  - Additional redundant data communication to solve the Byzantine problem
  - High cost of ownership



A much simpler and strait-forward redundancy management system for future architectures is necessary to be cost effective

This presentation is not subject to Export Control

Lessons Learned from Shuttle

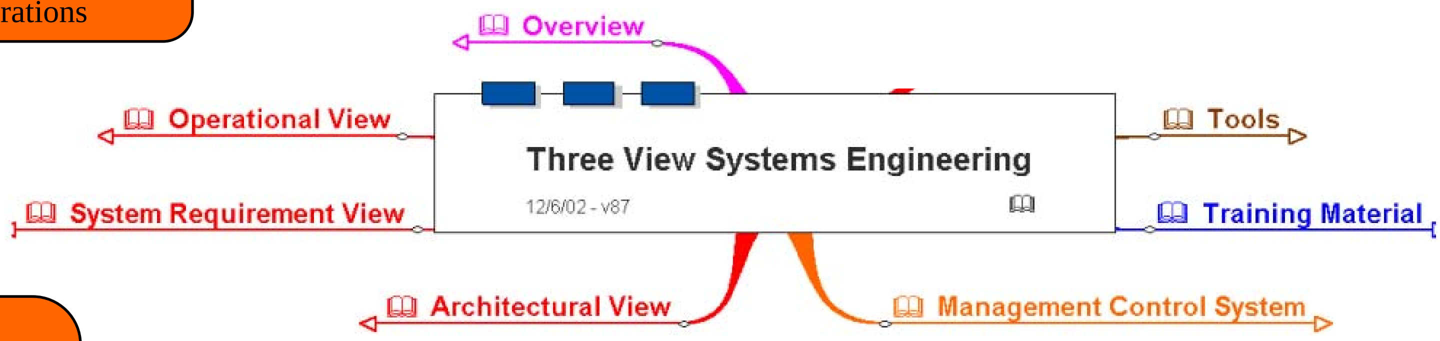
Honeywell

- **Shuttle avionics**
  - **300 major electronic components throughout the vehicle**
  - **Connected by more than 300 miles of electrical wiring.**
  - **Many different architectures, techniques, and electronics concepts.**
  - **Only two standard interfaces:**
    - Multiplex Interface Adapter (MIA)
    - Pyrotechnic Initiator Controller (PIC)
- **Undocumented/lost/forgotten requirements**
  - **Cannot be gained by review of existing documents.**
    - Honeywell in a unique position - primary supplier for Space Shuttle and ISS
    - Prime contractors retain a similar level of historical knowledge.
  - **Foremost undocumented requirement**
    - Predictable response to failures within the system.
    - Historic processor based systems have contained unpredictable failure modes
    - Today's processor "features" are legacy fixes for unpredictable behavior.
    - Watch dog timers, memory error correction, and Triple Modular Redundant (TMR) processors

Architectural Framework Requirements

Honeywell

OPERATIONAL VIEW
- What the system needs to do
- Concept of Operations



SYSTEM VIEW
- How the system does the job
- Flow Diagrams
- Behavior Models
- Operational Interface

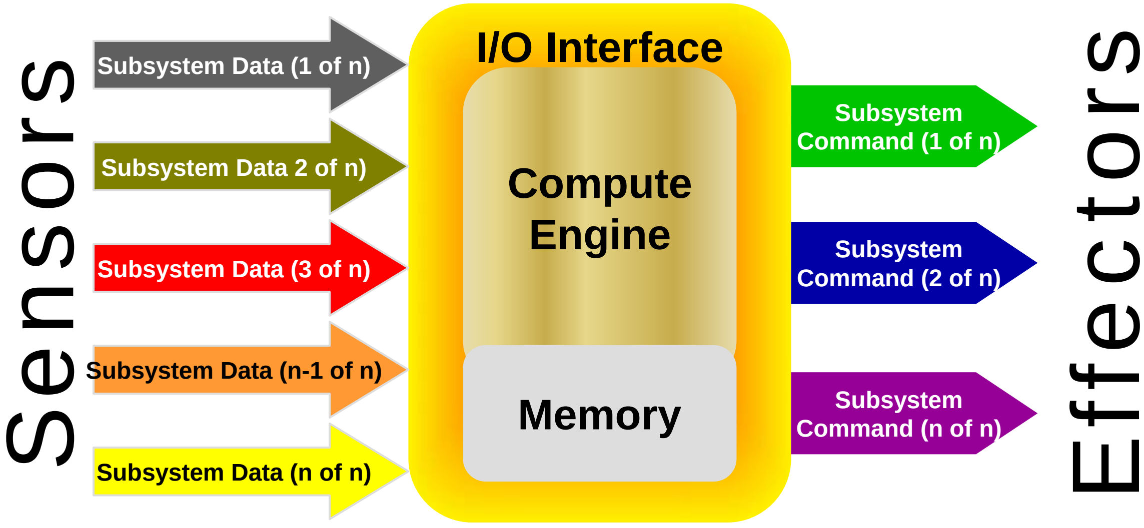
ARCHITECTURAL VIEW
- The physical implementation
- Block Diagrams
- ICDs
- Test Requirements

- User Objectives Pull System Design
- Project Focus On Primary SE Products
- Readily Understood Requirements
- Stabilized Process For Future Improvement
- Requirements Organized For Rapid Change

Requirements: The Start of a System

Honeywell

- One unit to host autonomous activity, health management, and housekeeping functions
- Commercial interfaces to reduce life cycle costs
- Hardware completely de-coupled from the hosted software reduces Cost
- Expandable and/or re-configurable in the future



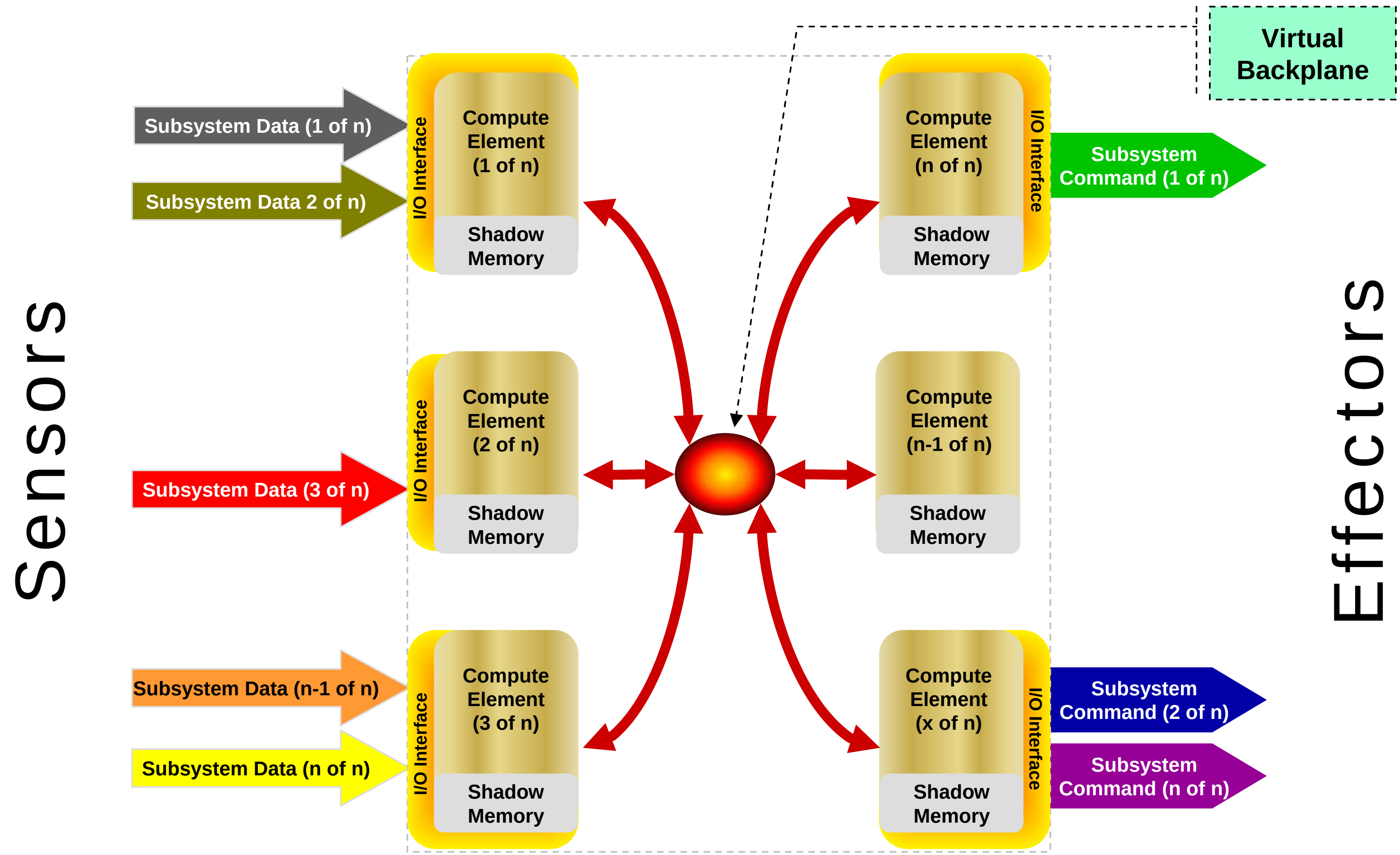
- Uses open architecture concepts to allow flexibility within the implementation
- Allows third party participation either in the development of the avionics hardware or at a future time
- Low cost system throughout the lifecycle
  - Low development cost
  - Low integration cost
  - Low future cost of ownership

The simplest architecture is a "flat" system

This presentation is not subject to Export Control

A Backbone Designed for both Flexibility and Availability

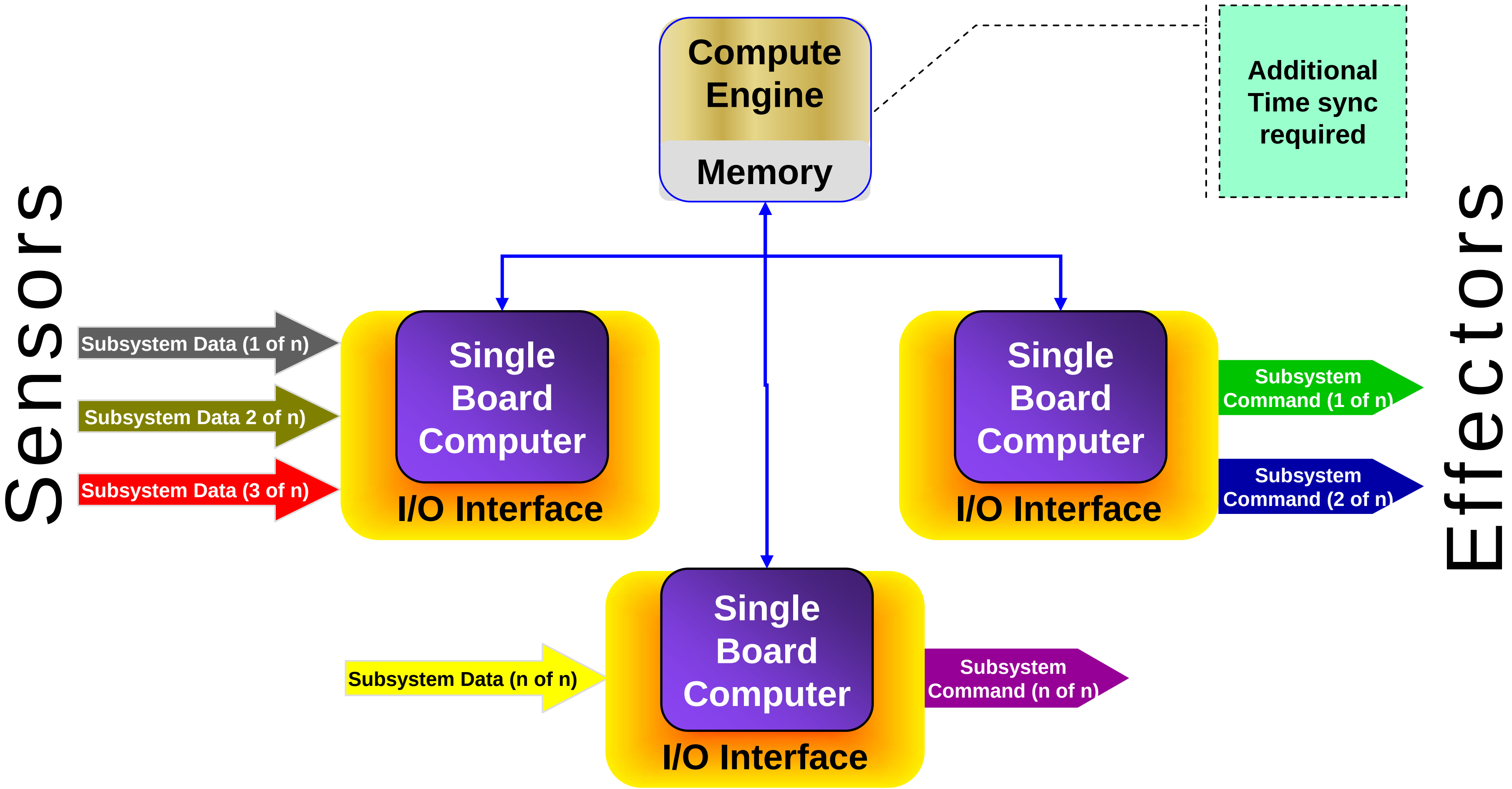
Honeywell



This presentation is not subject to Export Control

Distributed System Decomposition

Honeywell

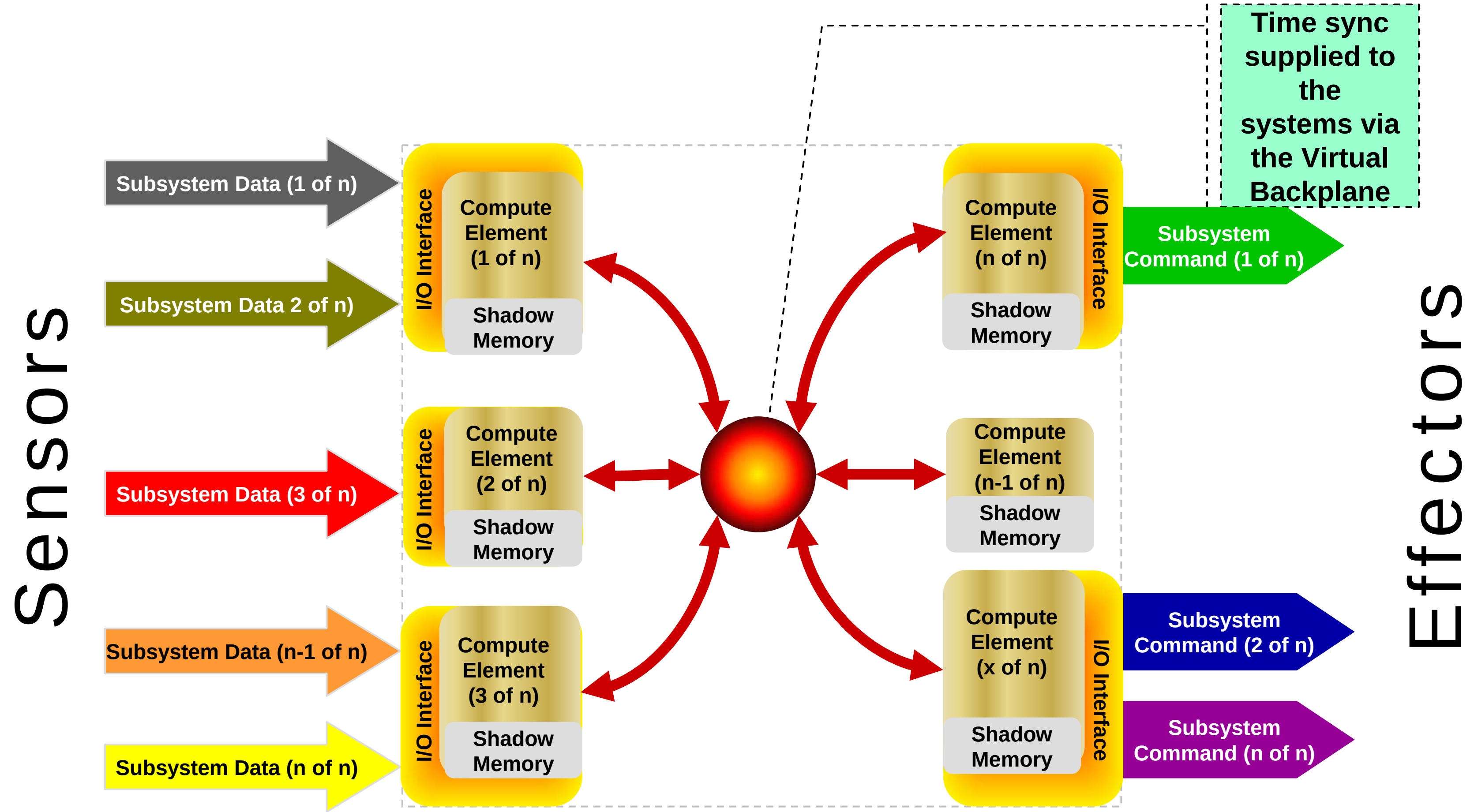


Additional system engineering and software effort is required to move data throughout the infrastructure.

This presentation is not subject to Export Control

Integrated System Decomposition

Honeywell

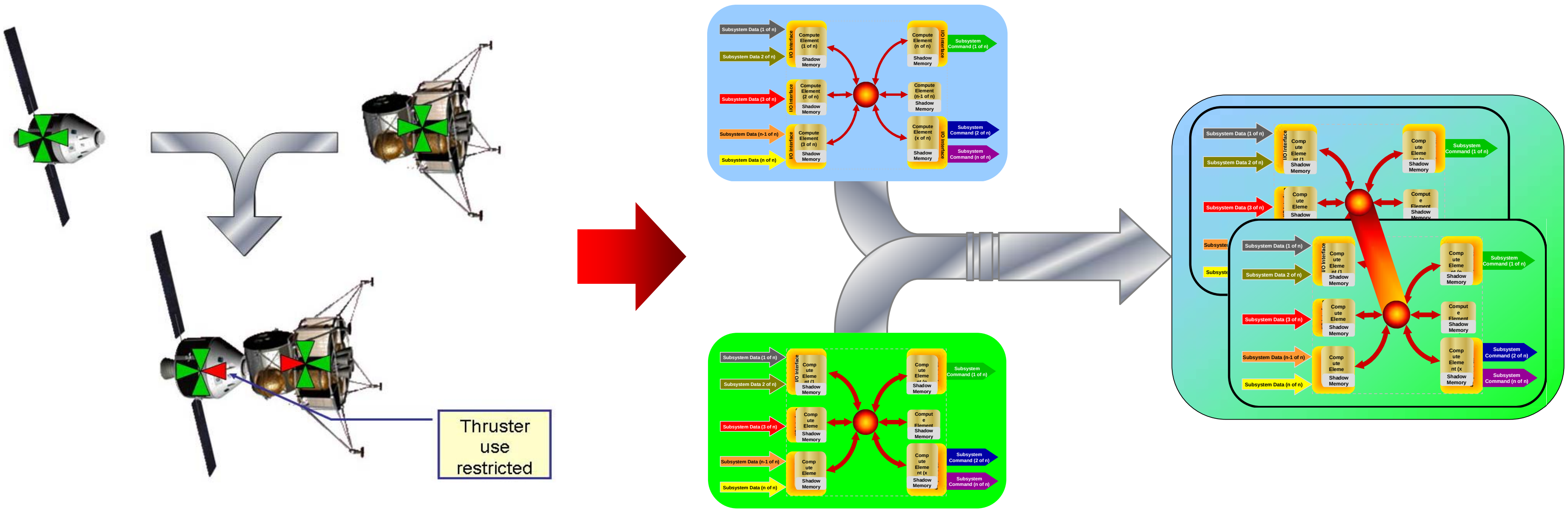


Data is moved through the system without additional software. Data is available throughout the system.

“System of Systems” Fusion

Honeywell

- Integrated Systems architecture supports reconfiguration
  - Individual systems are coupled through fusion of their individual Virtual Backplane™ elements
  - Several free-flying elements fuse into a new combined configuration



An updated system is achieved through predetermined, yet dynamic, re-configuration of individual element configuration tables

This presentation is not subject to Export Control

History of Aircraft Avionics

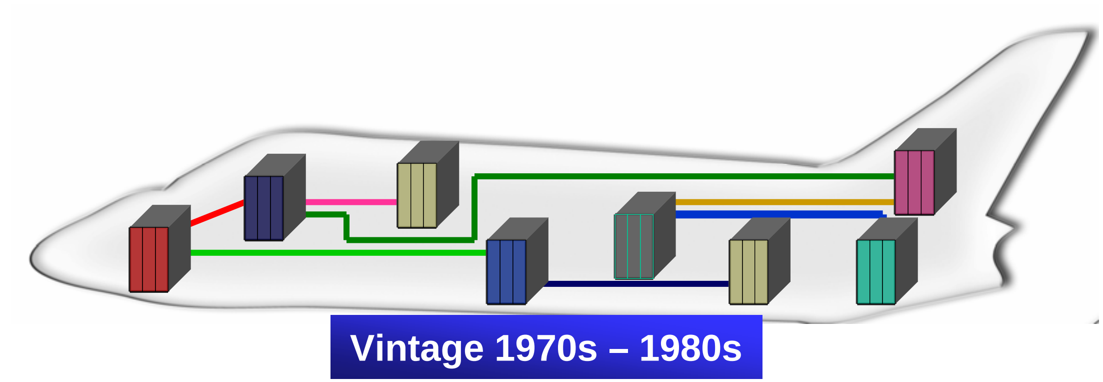
Honeywell

Centralized Computing System –

Original digital avionics controls were centralized with few fault containment zones

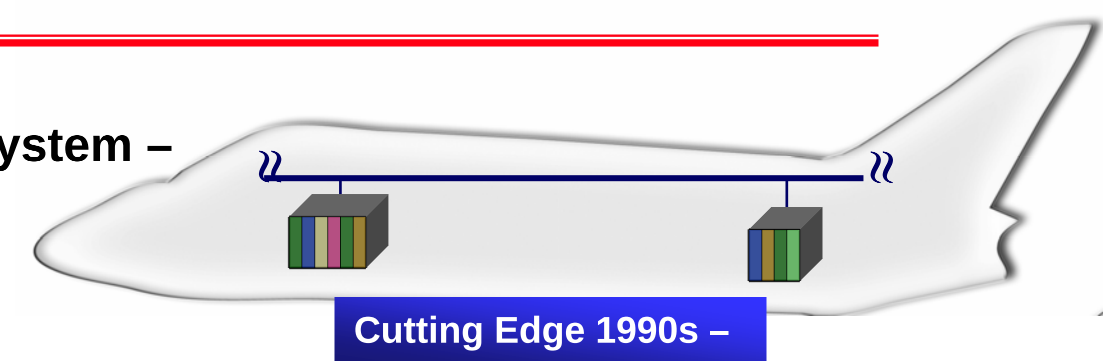
Federated Computing System –

Federated systems provided fault containment zones but increased interface complexity



Integrated Modular Computing System –

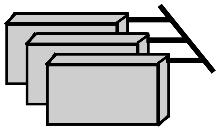
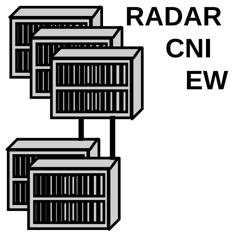
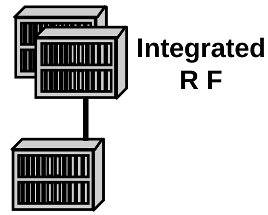
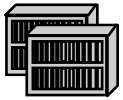
IMA provides the best of centralized and federated systems



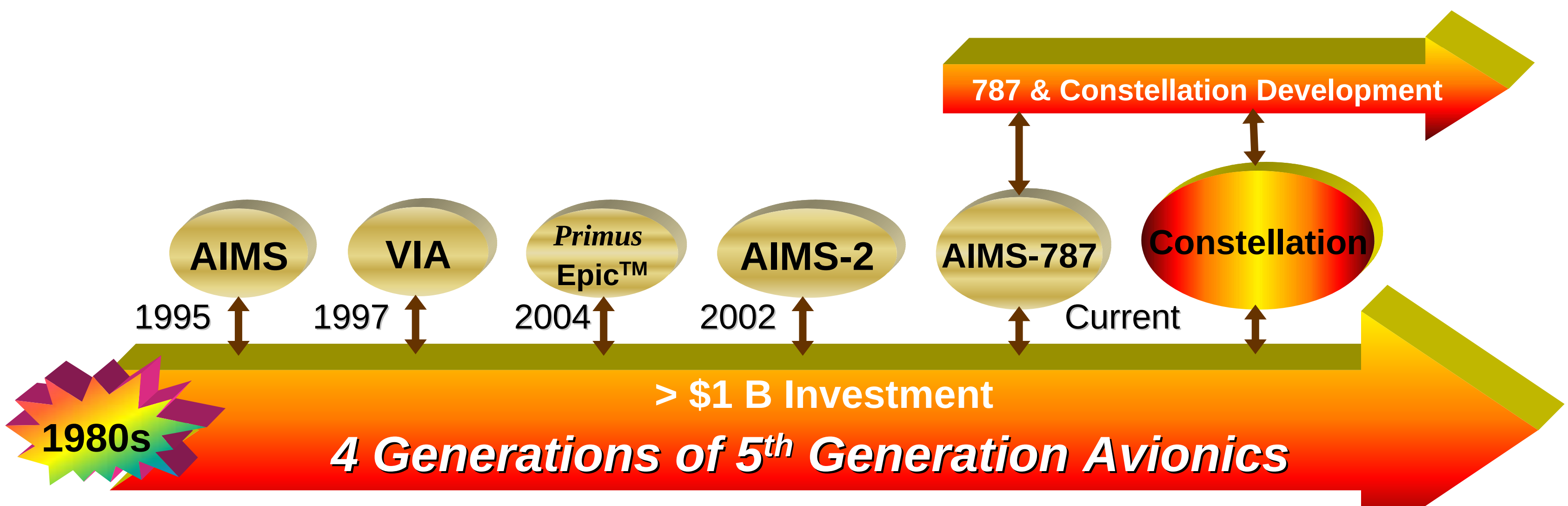
This presentation is not subject to Export Control

Integrated Modular Avionics History

Honeywell

2nd Generation DAIS	3rd Generation PAVE PILLAR	4th Generation PAVE PACE / ISS	5th Generation Low Cost Integrated Avionics
			
• Federated, LRU based systems	• Open architecture, common modules for RF • Technology drives core processing performance	• Integrated, LRM based system • Open architecture, common modules for core processing	• Technology drives system performance • Architecture drives schedule / risk / affordability

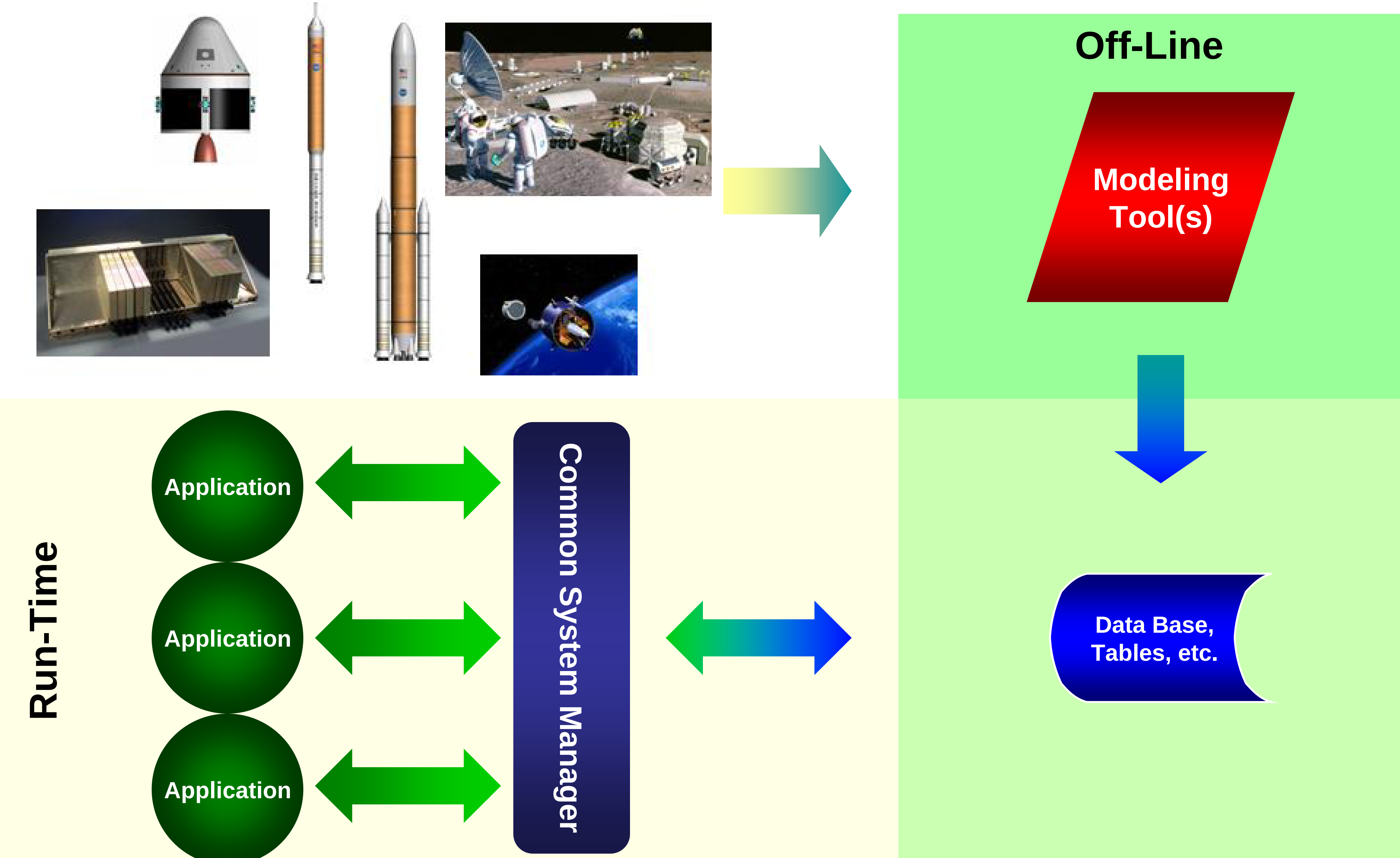
From a presentation by Ron Szkody on 29 May 1996 to the Integrated Sensor System (ISS) Open System Architecture (OSA) Joint Task Force (sponsored by United States Air Force Wright Laboratory /AAST-30



This presentation is not subject to Export Control

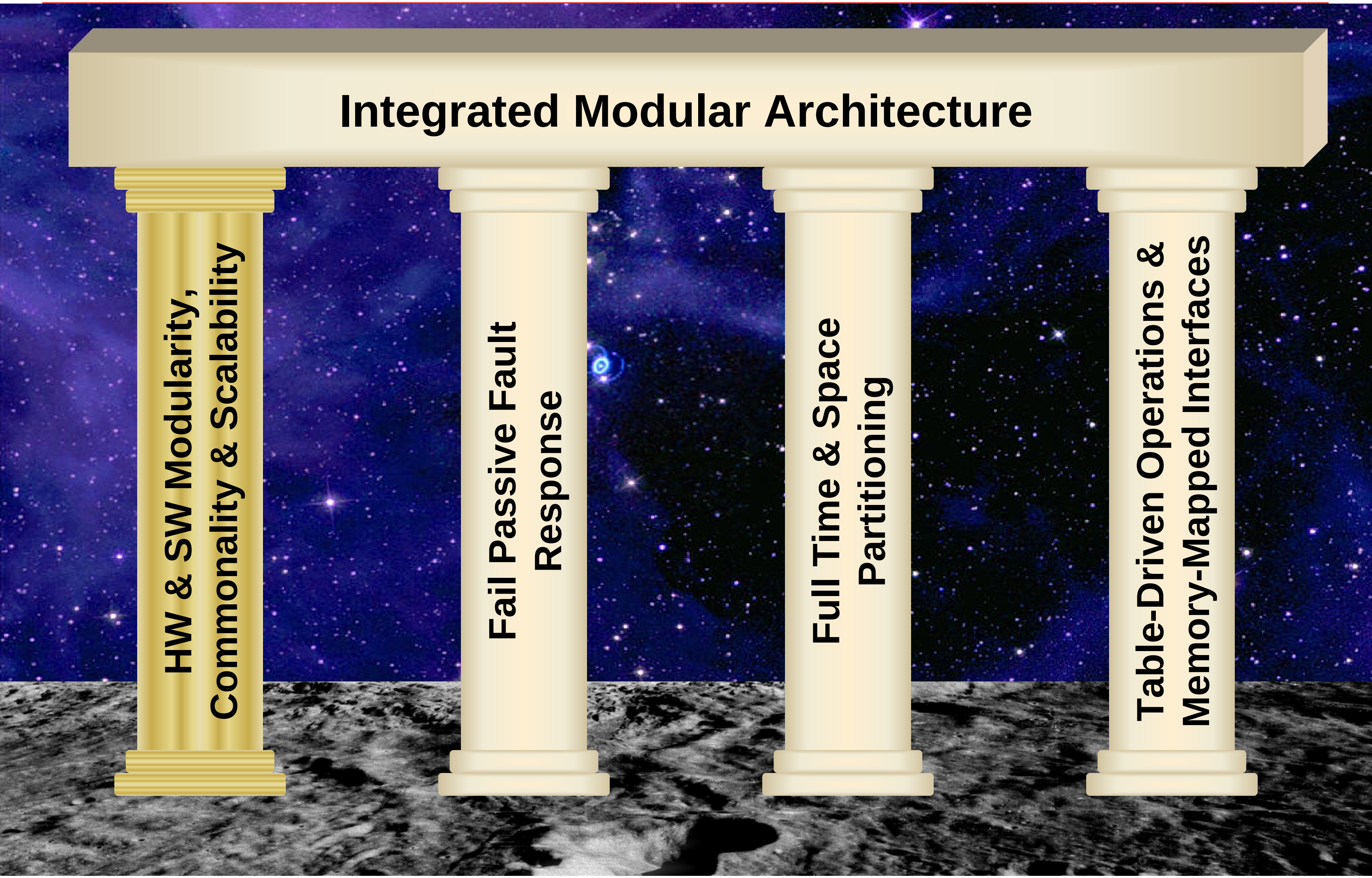
Minimizing Application Dependencies

Honeywell



This presentation is not subject to Export Control

Honeywell

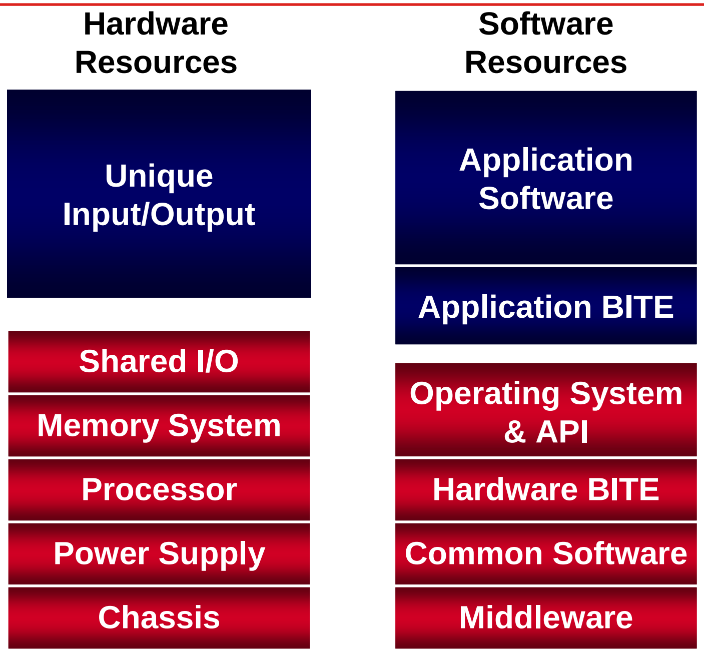

Integrated Modular Architecture
HW & SW Modularity, Commonality & Scalability
Fail Passive Fault Response
Full Time & Space Partitioning
Table-Driven Operations & Memory-Mapped Interfaces

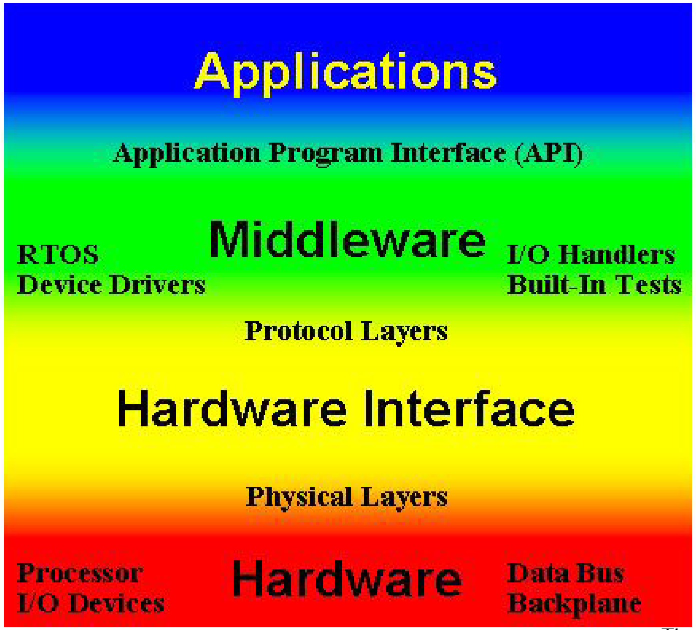
Modularity, Commonality, & Layers of Abstraction

Honeywell

Integrated Modularity Avionics benefits

- Reduced size, weight, power
- Reduced NRE cost and schedule
- Reduced parts, part types, and spares
- Reduced training and maintenance costs
- Reduced software development, certification, and modification costs





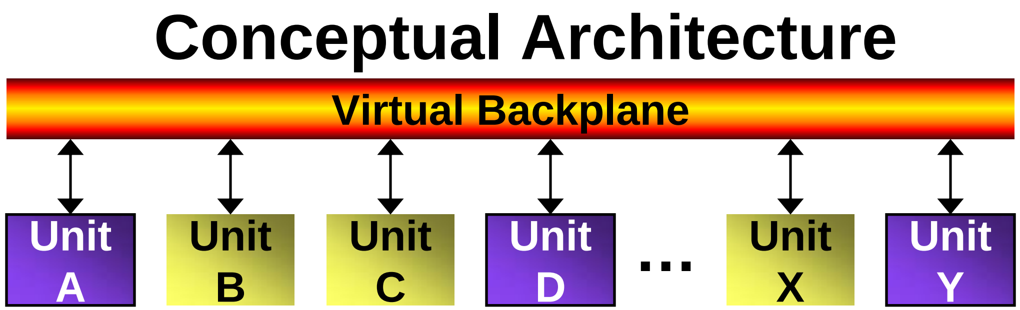
Layered approach benefits

- Layered approach protects applications from physical implementation
- System upgrades, modifications, and changes require minimal or no impact to applications
  - Operating system version or vendor
  - Processor, I/O types, data buses
  - Redundancy Levels

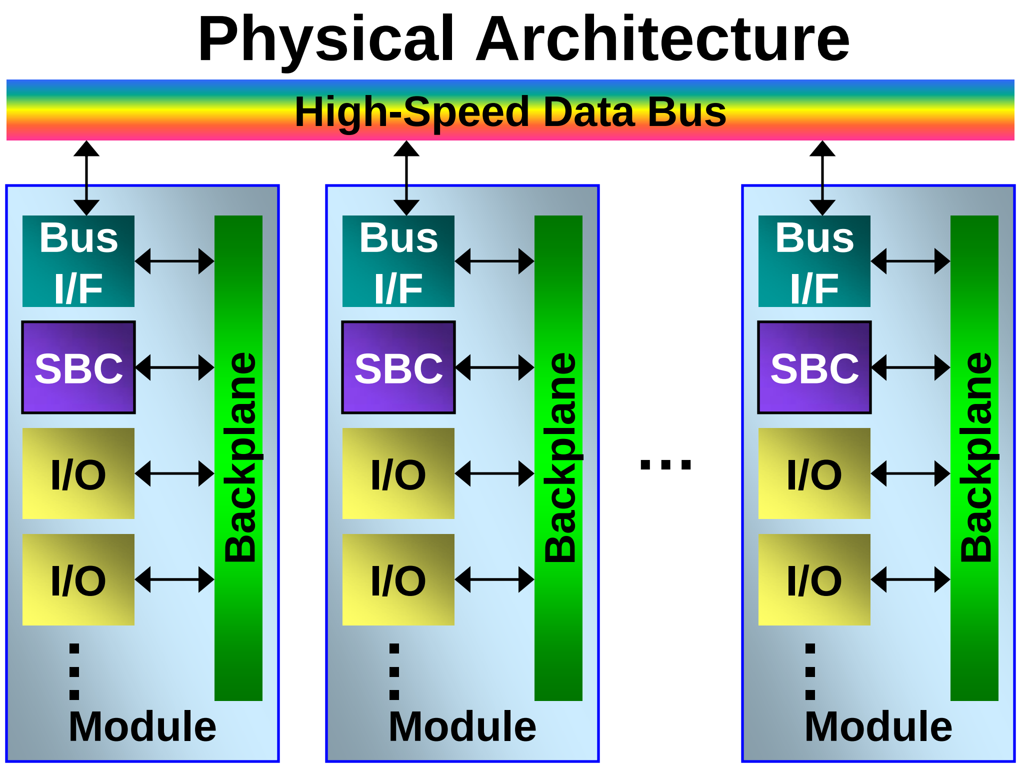
Virtual Backplane™ Concept

Honeywell

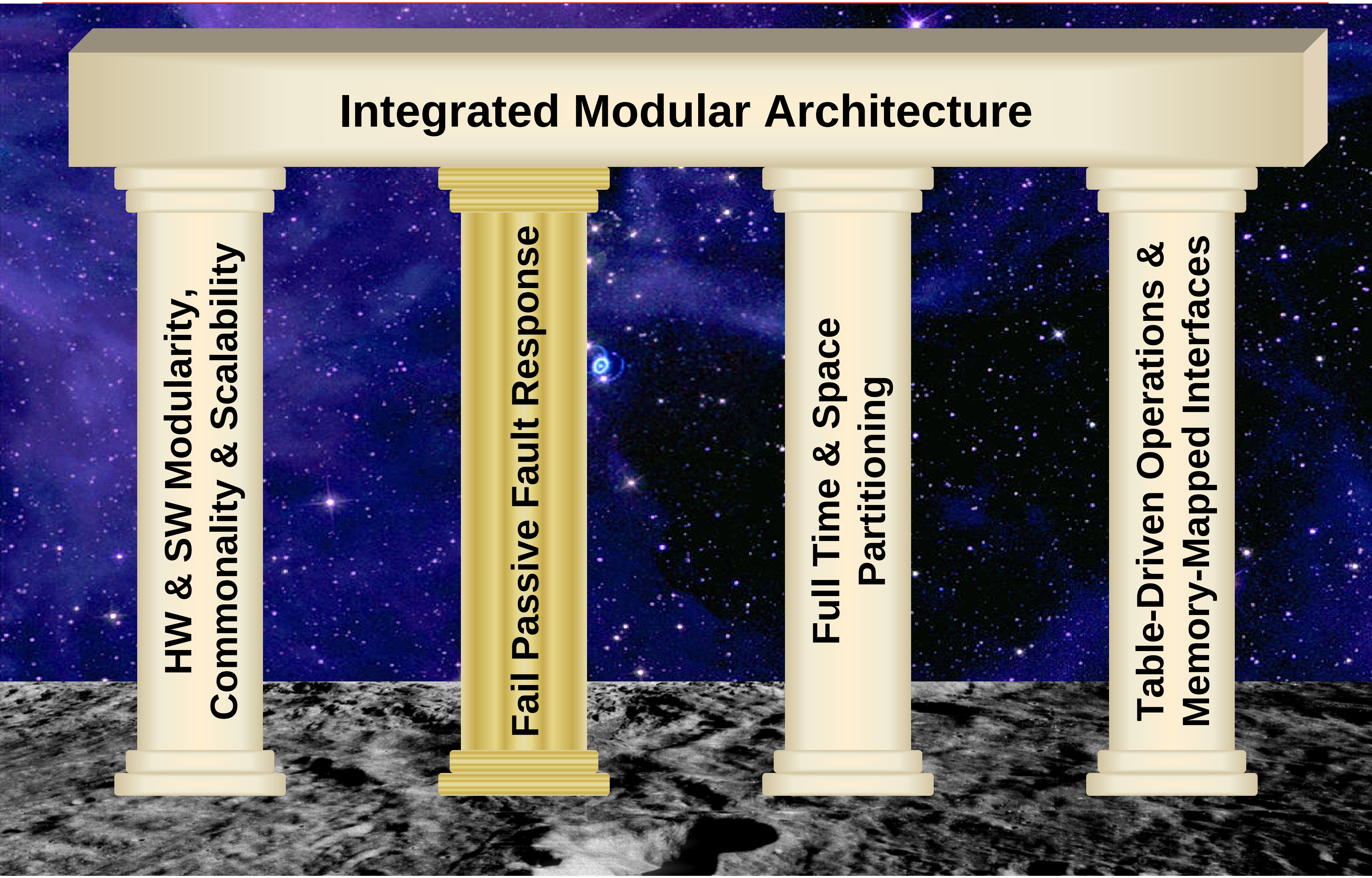
The system behaves as if all units are peers…



…regardless of the physical implementation

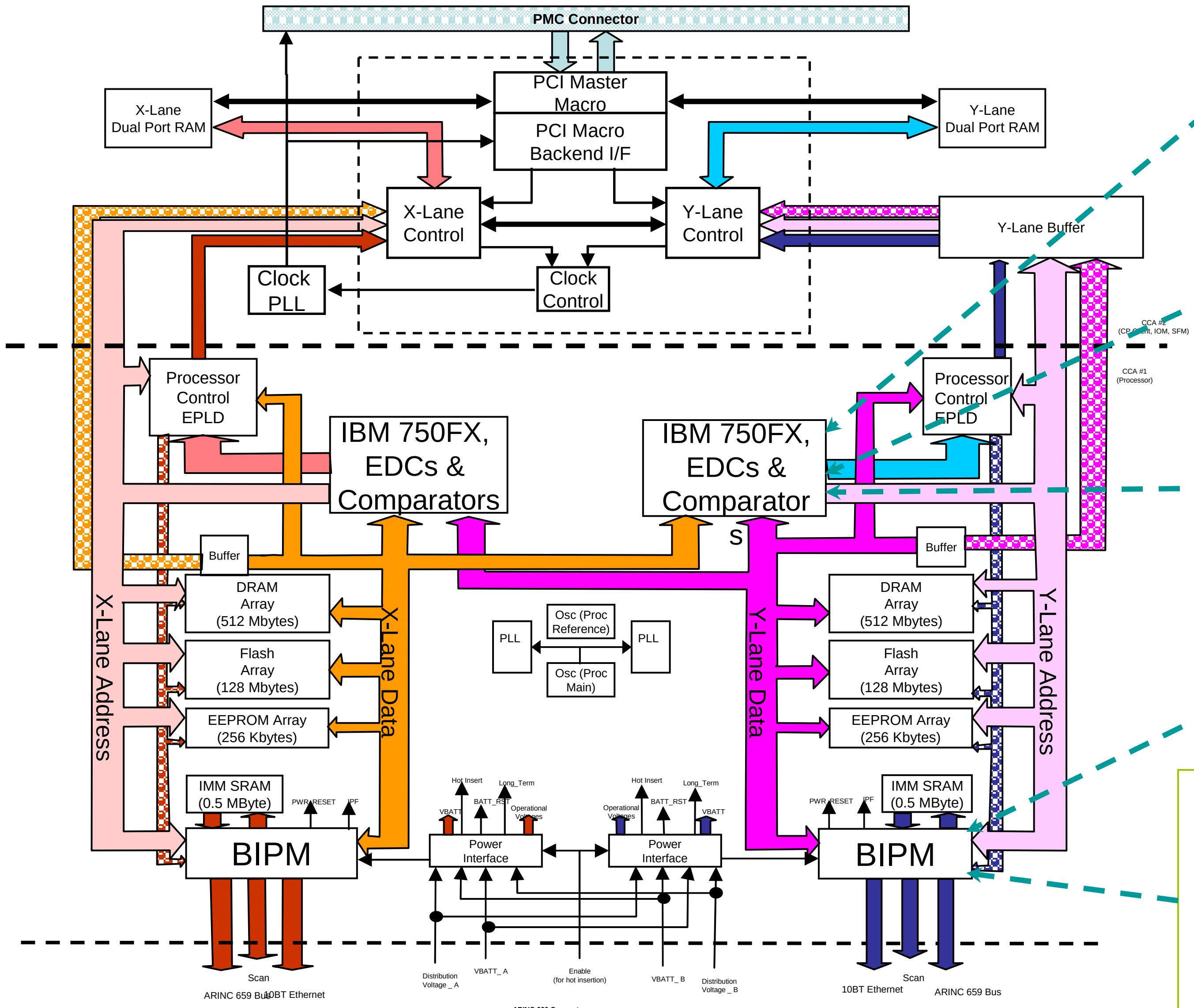


This presentation is not subject to Export Control


Integrated Modular Architecture
HW & SW Modularity, Commonality & Scalability
Fail Passive Fault Response
Full Time & Space Partitioning
Table-Driven Operations & Memory-Mapped Interfaces

One Approach - Lock-Step Processor

Honeywell



Error Immunity
- 100% detection & correction of single-bit errors
- 100% detection of double-bit errors

Space Partitioning (MMU)
- Write protection over IMM, SRAM, H/W Resources

Fault Containment (Dual Cross-Compare Monitors)
- Lockstep comparison
- MMU integrity validation

Data Integrity Mgt
- Freshness Monitoring for Partition Usage

Bus Error Containment
- Dual Lockstep transactions
- Total fault containment
- 100% detection of single-bit transmission errors
- Dual redundant transmission
- De-centralized control

This presentation is not subject to Export Control

Fail Passive Fault Response

Honeywell

- **Lack of cross channel trust typically handled by shuttle-like voting schemes**
  - Extraneous cross-channel data link required
  - Complex voting algorithms
- **Alternative is fail passive architecture**
  - Faulty modules remove themselves from operation until corrected
  - Does not require a cross-channel voting mechanism
- **Several approaches to fail passive**
  - Lock-step processor (also provides 100% fault coverage)
  - Command-monitor processor pairs
  - Polynomial encoding

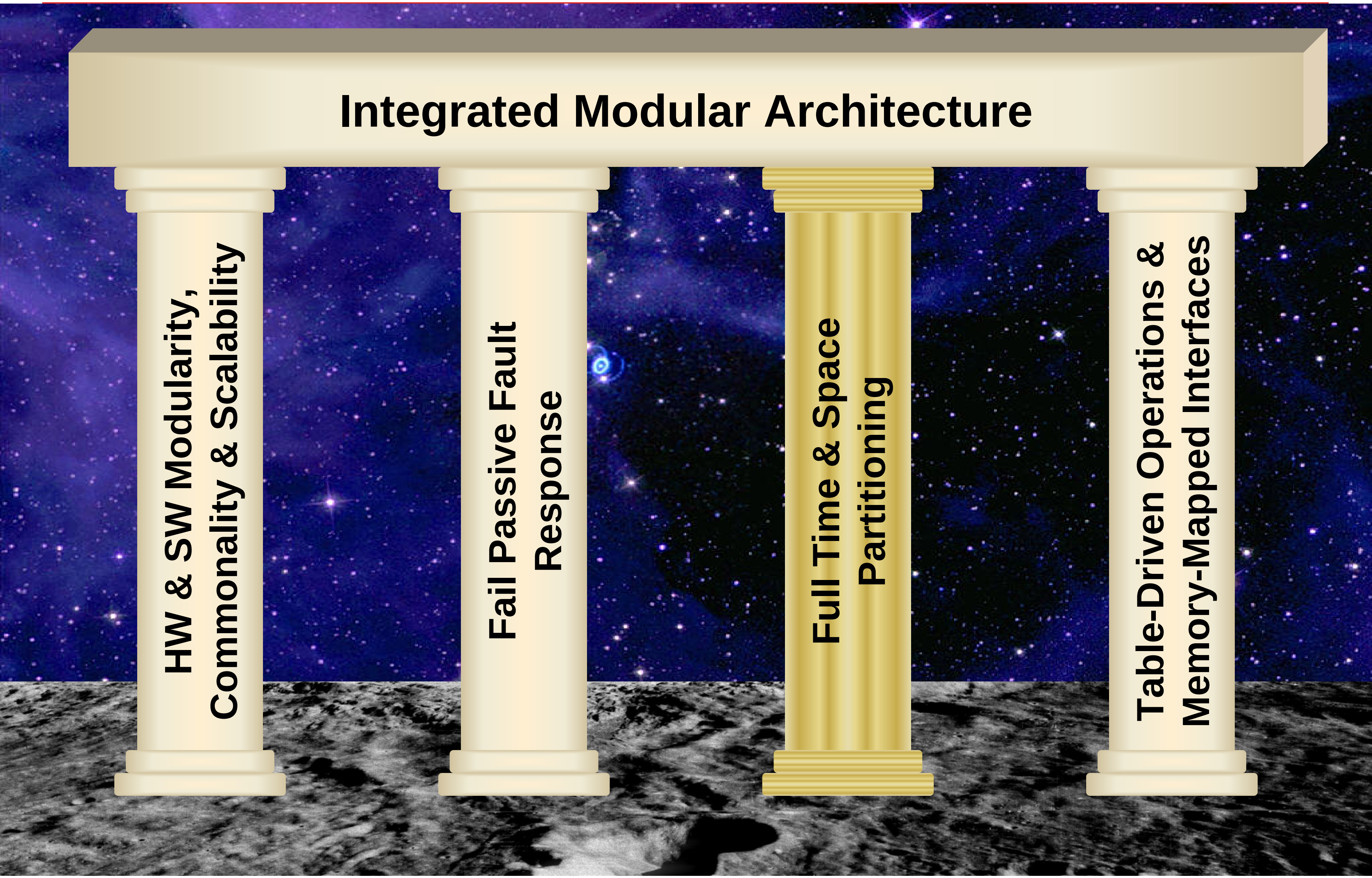
Required Fault Tolerance	Self Test Coverage	Cross Channel Trust	Number of Redundant Channels
0 Faults	N/A	N/A	≥1
1 Fault	100% <100% <100%	Truthful Truthful Lies*	≥2 ≥3 ≥4
2 Sequential Faults**	100% <100% <100%	Truthful Truthful Lies*	≥3 ≥4 ≥5
2 Simul-taneous Faults***	100% <100% <100%	Truthful Truthful Lies*	≥3 ≥5 ≥7

* Classic Byzantine Fault: number of required channels is established by a formal proof

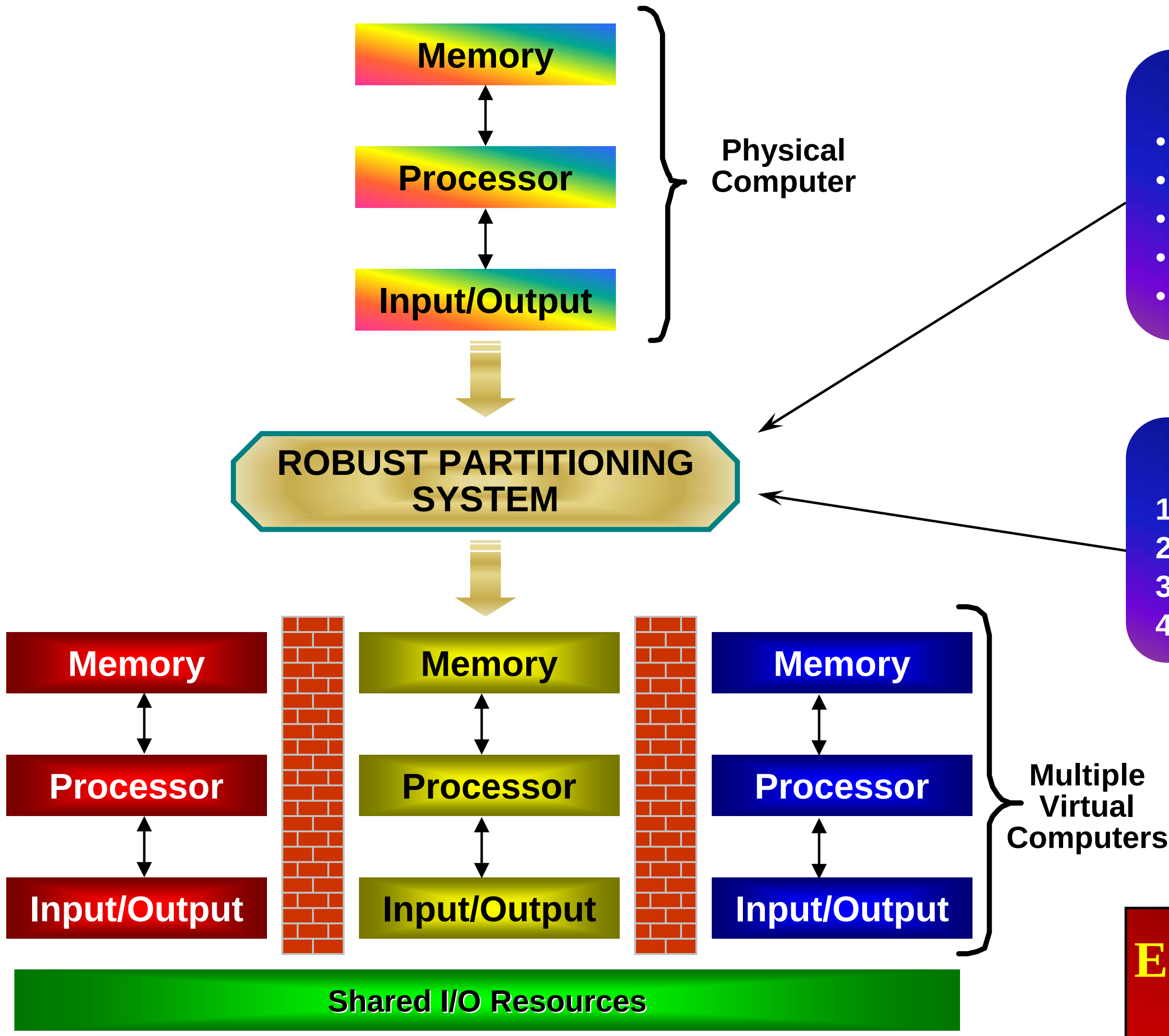
** 1st failure removed before second failure occurs

*** 1st failure not removed before second failure occurs

Fail silent architecture enables minimal number of required channels to compensate for fault conditions


Integrated Modular Architecture
HW & SW Modularity, Commonality & Scalability
Fail Passive Fault Response
Full Time & Space Partitioning
Table-Driven Operations & Memory-Mapped Interfaces

Time and Space Partitioning – a Hardware View



Key implementation concepts

- Memory Management Unit
- High Integrity Time Base
- Table Driven Communication
- Memory Mapped I/O
- High integrity Operating System

Partitioning Domains

1. Memory Space
2. Computation Time
3. Input/Output Access
4. Backplane Access

Each partition operates as if it were hosted on a dedicated computer

This presentation is not subject to Export Control

Time and Space Partitioning – a Software View

Honeywell

BITE
- Continuous BITE
- Status Generation
- Maintenance I/F
- Non-Volatile Memory Mgmt
- Loader Entry Detection
- Fault Server

OS API

I/O
- 1394
- 1553
- Ethernet
- Analog I/O
- Discrete I/O
- RS-422

OS API

Application A

Critical Functions

OS API

Application B

Essential Functions

OS API

• • •

Application N

Non-Essential Functions

OS API

Partitioned Operating System
- OS API
- Memory Management
- Application Fault Response
- Time Management

Non-Resident Boot
- Hardware Abstraction Layer (HAL2)
- Non-Resident Boot Initialization
- Power-Up BITE (POST)
- Platform Fault Response
- Software Loader
- Platform Load Verification
- Module Load Verification
- Cabinet Initialization

Resident Boot
- Boot Initialization
- Hardware Abstraction Layer (HAL1)

Common Monitor
- System Monitor/Debug Interface

Tools Interface
- Partition Monitor/Debug Interface

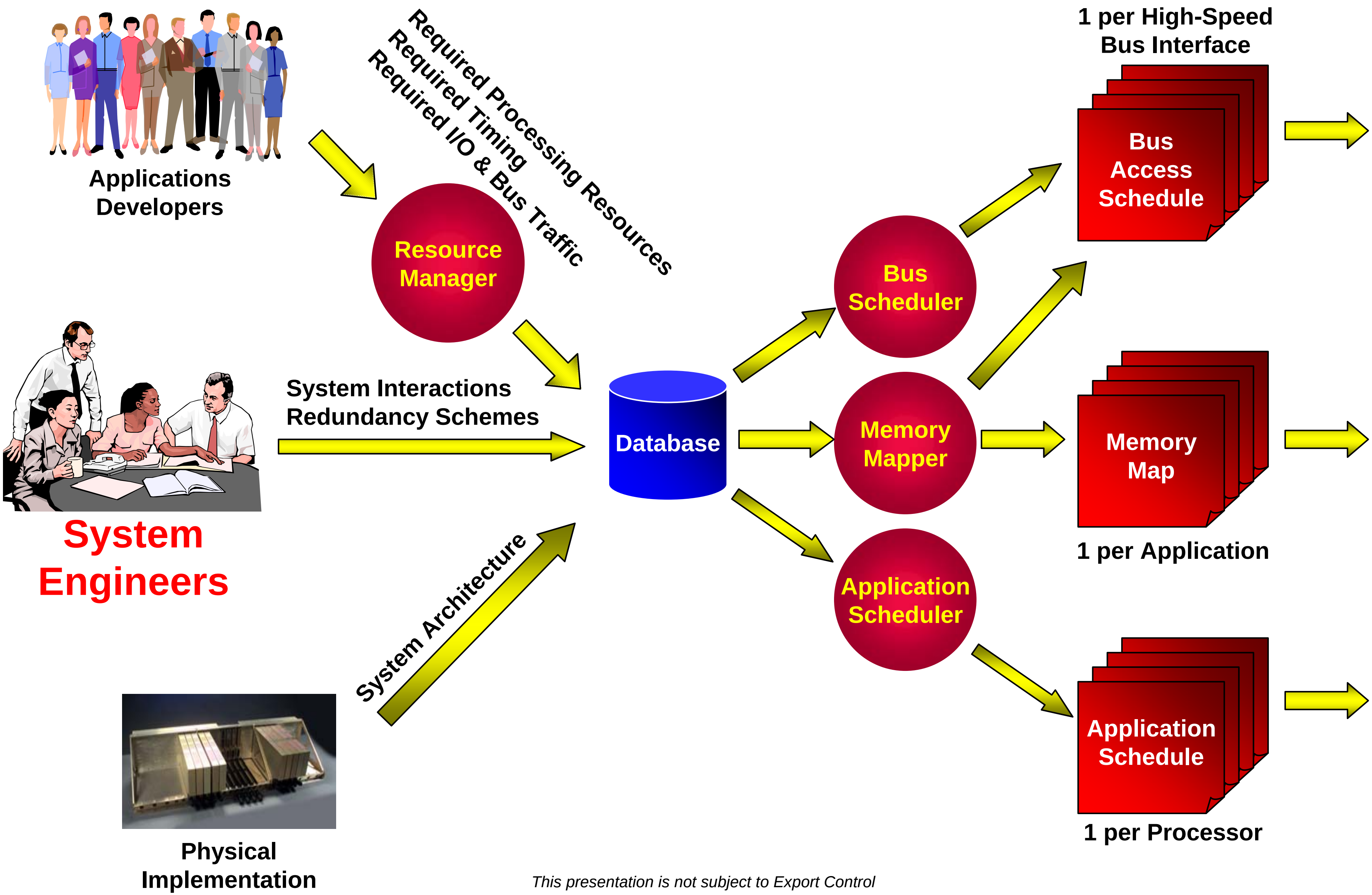
Honeywell

Integrated Modular Architecture

- HW & SW Modularity, Commonality & Scalability
- Fail Passive Fault Response
- Full Time & Space Partitioning
- Table-Driven Operations & Memory-Mapped Interfaces

Offline Table Generation Simplifies System Modifications

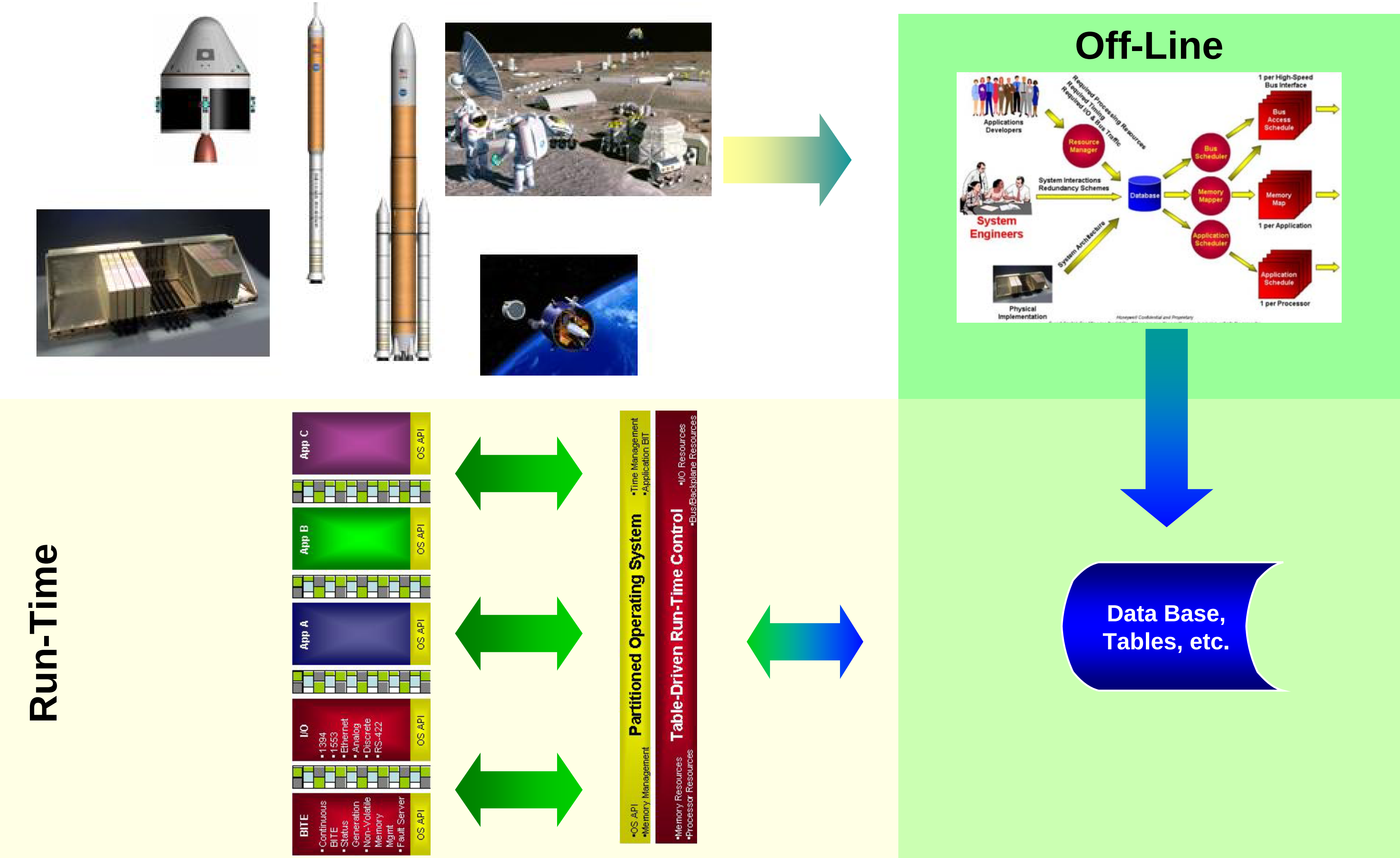
Honeywell



This presentation is not subject to Export Control

Designing for Modifiability – Computing Platforms

Honeywell



This presentation is not subject to Export Control