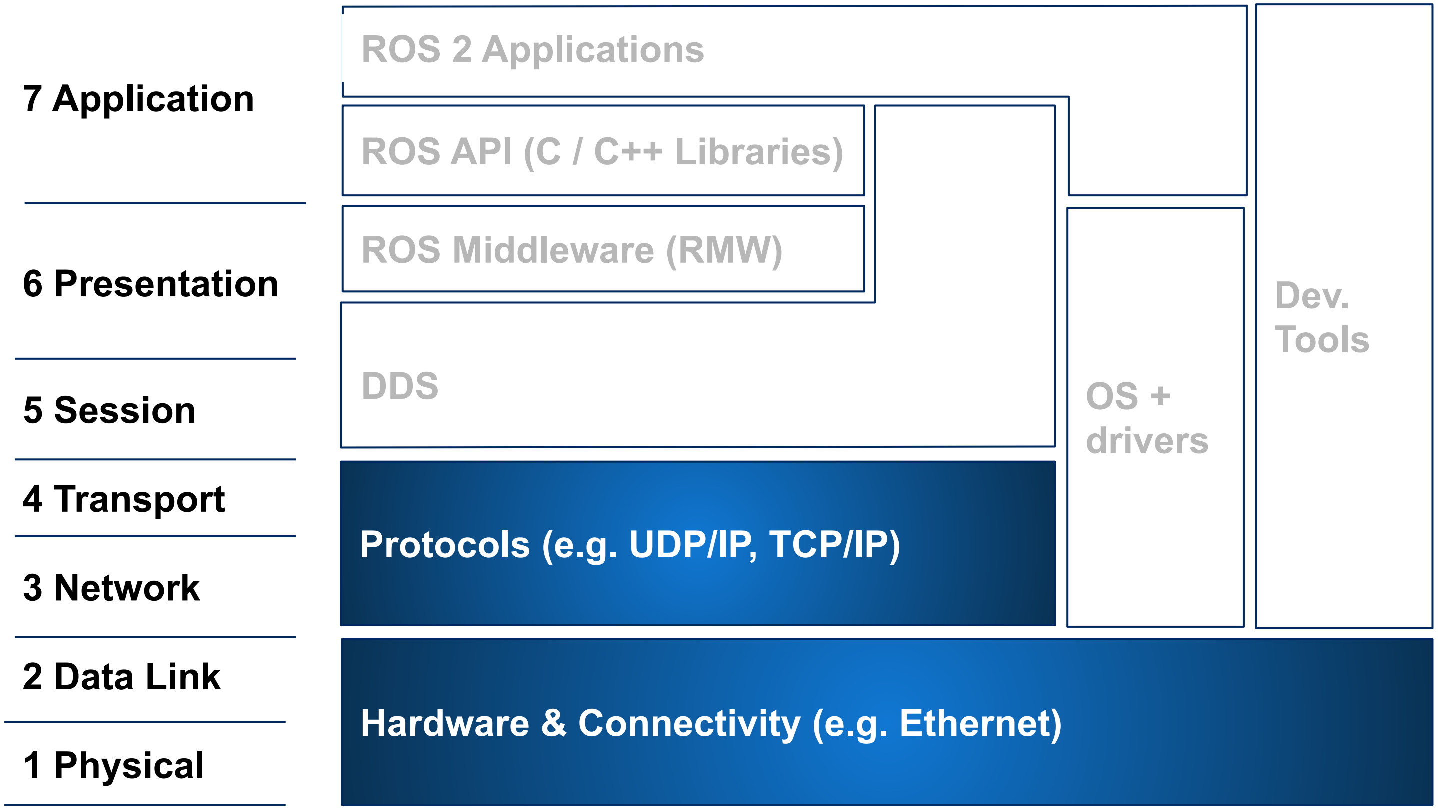
Real Time Hardware

David Crawley
Ubiquity Robotics

Real time needs to be real time across the stack - we are going to talk about OSI 1-4 in this talk



Definition of Real time

The control system must provide the control responses or actions to the stimulus or requests within specific times.

Definition of Real time

The control system **must** provide the control responses or actions to the stimulus or requests within **specific times**.

Let’s look at the data transmission problem

Should the robot slow down / stop?

If d ~ stopping distance then alter velocity / stop

Stopping distance = deceleration distance + response distance

Response distance = speed X (sense time + decision time + **data transmission time**)

Two peripherals want to talk to each other how long will it take for the data to arrive?

LIDAR sensor

Computer

transmission time* = bits in packet / baud rate + propagation time

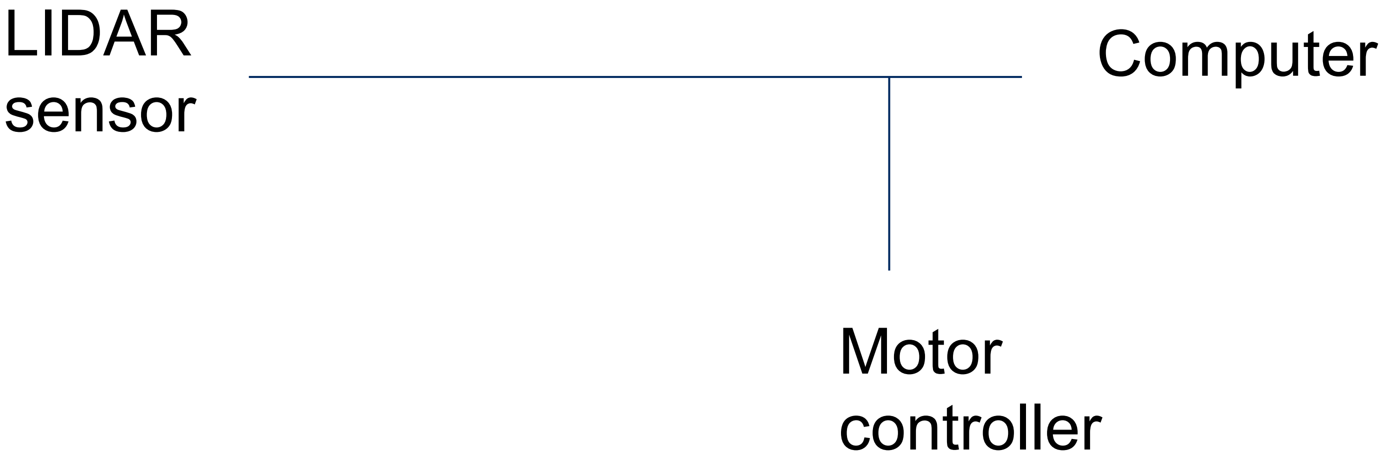
Specified Specified Specified

*Ignoring processing times on each end - e.g. device setup and hold times, processing etc. usually small, frequently deterministic

Definition of Real time

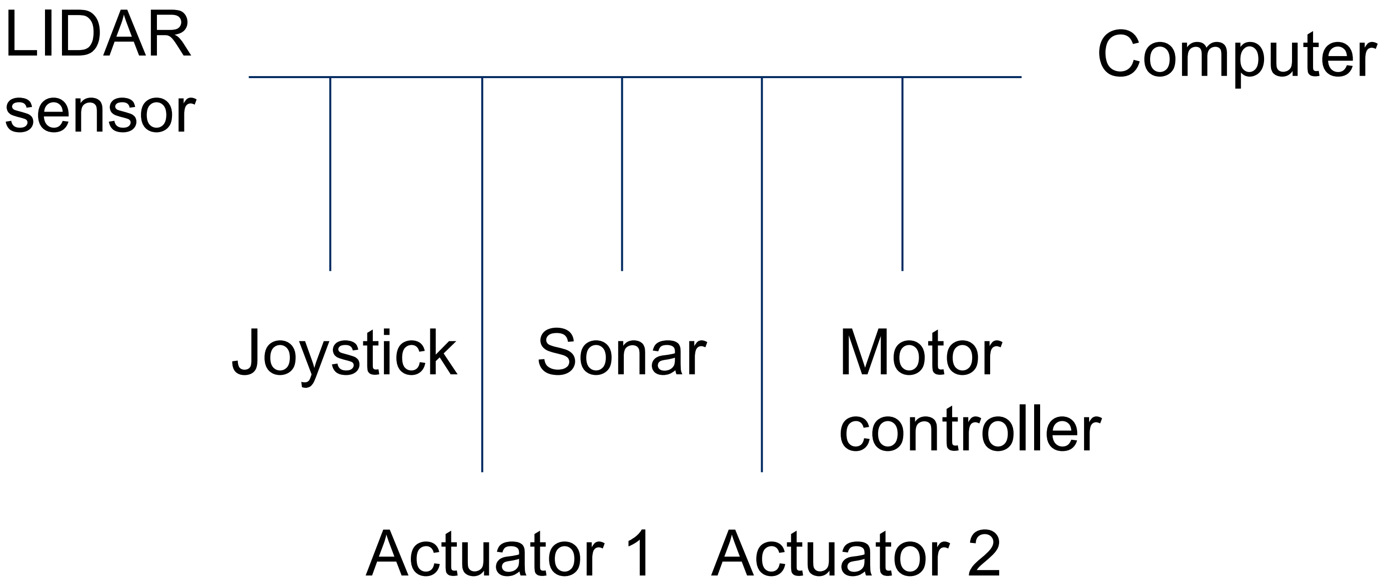
The control system **must** provide the control responses or actions to the stimulus or requests within **specific times**.

But things are never that simple

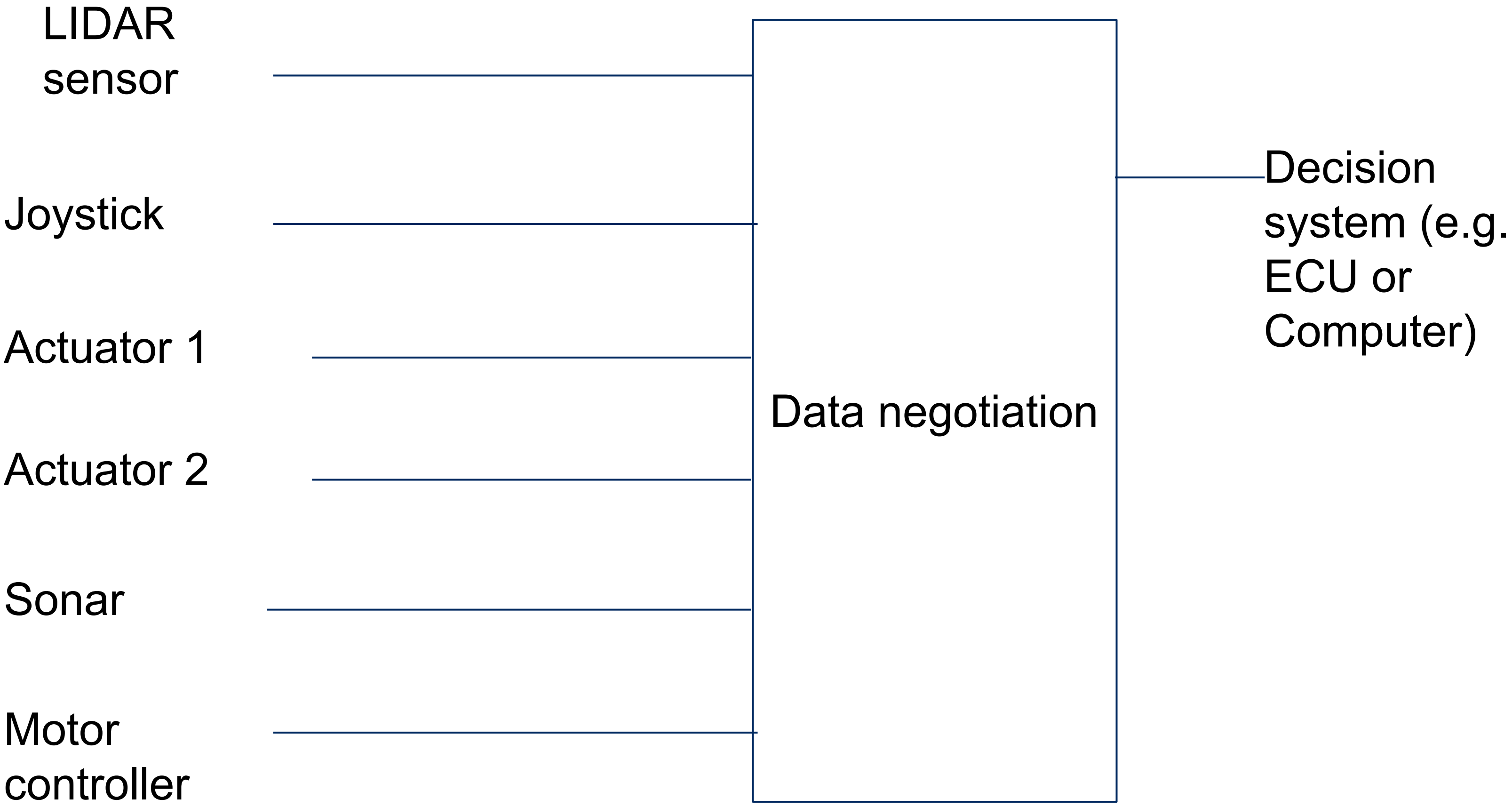


transmission time* = bits in packet / baud rate + propagation time
+ **wait time for shared resource**

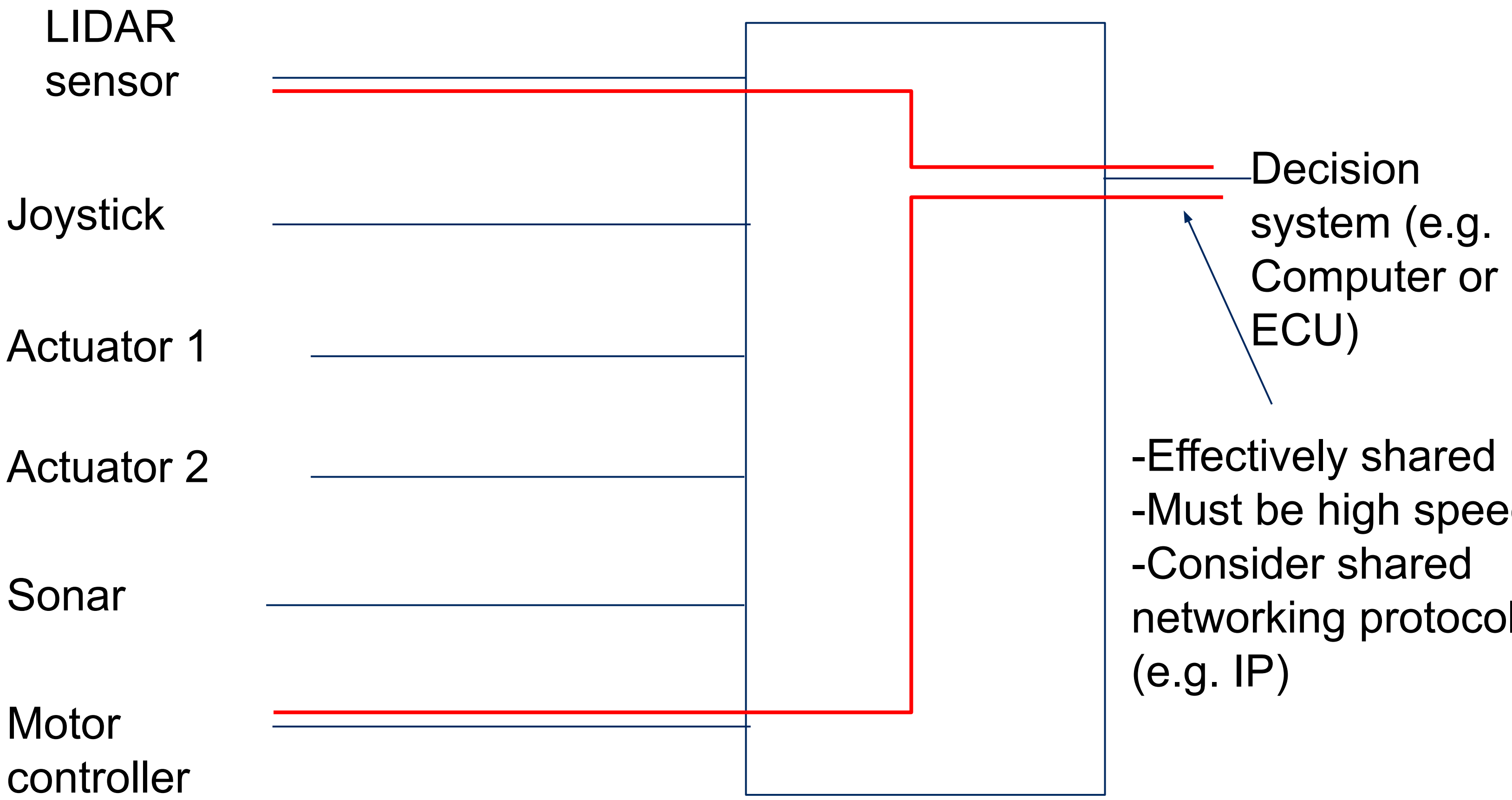
But things are rarely even that simple



OK - let's make it all point to point

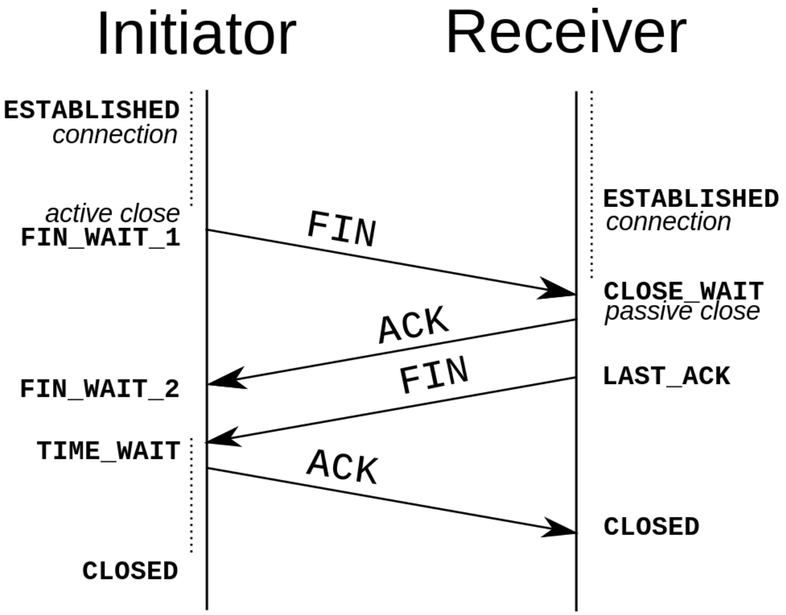


Parts of comms infrastructure effectively shared



-Effectively shared
-Must be high speed
-Consider shared networking protocols (e.g. IP)

How data communication works on ethernet / TCP/IP



Collisions possible

Handshaking prior to transmit

Re-transmit on error

Guaranteed packet arrival - timing not guaranteed

Cannot guarantee timing

Definition of Real time

The control system **must** provide the control responses or actions to the stimulus or requests ~~within **specific times**~~.

How data communication works on ethernet / UDP

Collisions possible

No handshaking

No re-transmit on error

Timing is definable in simple situations

Cannot guarantee error on failure

Definition of Real time

The control system ~~**must**~~ provide the control responses or actions to the stimulus or requests within **specific times**.

What about UDP with timing information in datagram for failure notification?

All nodes have precise time

All datagrams have time stamp on transmission

Define minimum frequency of datagram transmission

Receiving node generates error if it lacks recent datagram

Can guarantee data is recent or failure

Definition of Soft Real time

The control system will provide a response or generate a failure signal within **specific times**.

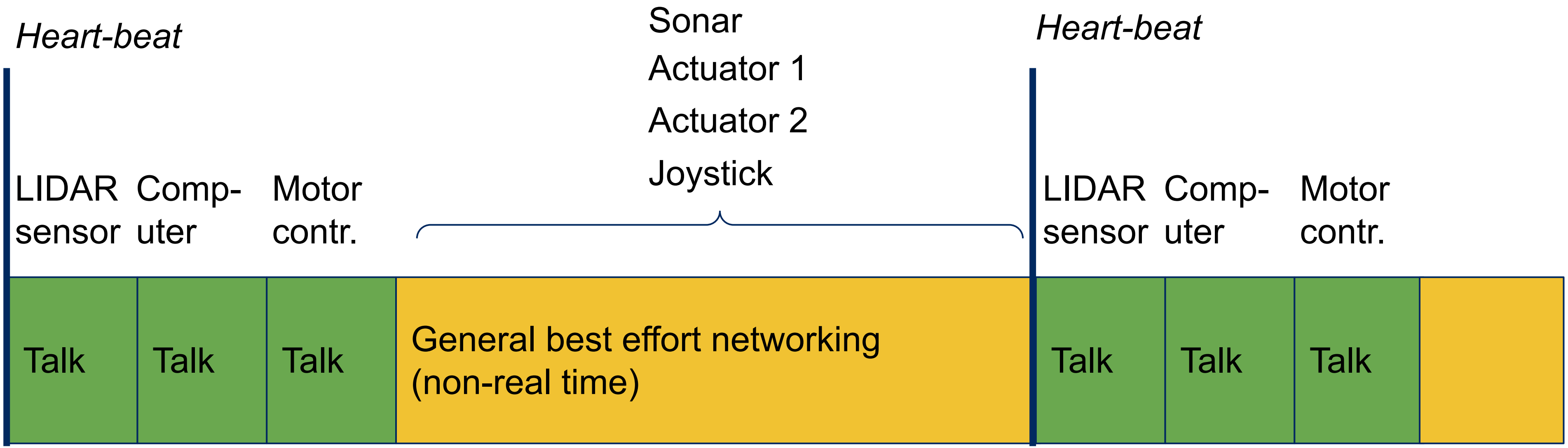
Failure rate equal to communication utilization*

An alternative - time slicing

1) Propagate precise timing to all time sensitive nodes

2) Schedule time slices when transmission time is reserved to a node

3) Switch packets from all prescribed talkers to all prescribed listeners in those time slices



Definition of Real time

The control system **<u>must</u>** provide the control responses or actions to the stimulus or requests within **<u>specific times</u>**.

OK so how do we implement it?

Requires different hardware compared to standard ethernet to support separate queues of differing priority

Audio Video Bridge (AVB) Standards

IEEE 802.1BA-2011: Audio Video Bridging (AVB) Systems
IEEE 802.1AS-2011: Timing and Synchronization for Time-Sensitive Applications (gPTP)
IEEE 802.1Qav-2009: Forwarding and Queuing for Time-Sensitive Streams (FQTSS)
IEEE 802.1Qat-2010: Stream Reservation Protocol (SRP)
IEEE 1722-2011 Layer 2 Transport Protocol for Time Sensitive Applications (AV Transport Protocol, AVTP)
IEEE 1722.1-2013 Device Discovery, Enumeration, Connection Management and Control Protocol (AVDECC).

Time Sensitive Networking (TSN) Standards

IEEE 802.1Qcc Enhancements to SRP
IEEE 802.1Qch Cyclic Queuing and Forwarding (CQF)
IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP)
IEEE 802.1Qbv Enhancements to Traffic Scheduling: Time-Aware Shaper (TAS)
IEEE 802.1Qbv in more detail: Time slices and guard bands
IEEE 802.3br and 802.1Qbu Interspersing Express Traffic (IET) and Frame Preemption
IEEE 802.1Qcr Asynchronous Traffic Shaping
IEEE 802.1Qca Path Control and Reservation (PCR)
IEEE 802.1CB Frame Replication and Elimination for Reliability (FRER)
IEEE P802.1CS Link-Local Registration Protocol
IEEE P802.1Qdd Resource Allocation Protocol
IEEE P802.1ABdh Link Layer Discovery Protocol v2

What hardware exists

Description

Network Interface Cards (NIC)	- Any Intel i210 ethernet chipset card supports AVB - drivers not always out of the box - Macintosh with BCM57761 chipsets support "out of the box" - Cards available for ~$50
Switches	- Switches exist - but optimized for audio connectivity - Lack features that are desirable in robotics - Expensive
Chipsets	-Many NIC chipsets exist (e.g. Yukon 88E8059, Intel i210, BCM57761) - Several Switch chipsets exist (e.g. - Many automotive qualified -Driver support is variable

Hardware available

Drivers and support less than ideal

* Check to see if supported with: sudo avbdeviced --list-interfaces

What about my sonar / low cost sensor?

-Ethernet not ideal for all sensors (e.g. Sonar, other low cost sensors)

- Most robots have sensors with a variety of interfaces

- Some intelligence to run ROS / DDS desirable

Attaching a general microcontroller to switch with a variety of interfaces desirable

OK what about decision time

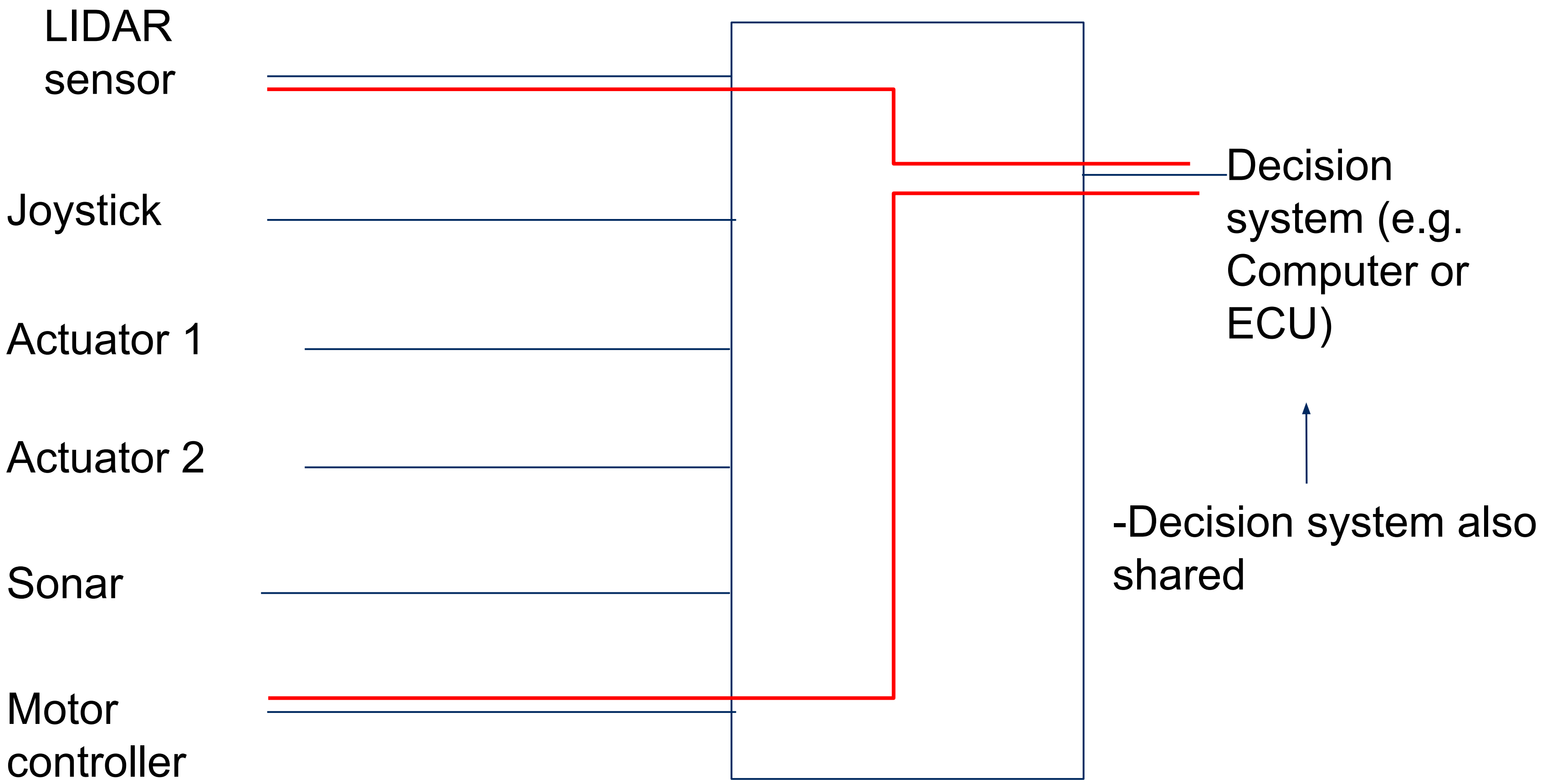
Should the robot slow down / stop?

If d ~ stopping distance then alter velocity / stop

Stopping distance = deceleration distance + response distance

Response distance = speed X (sense time + **decision time** + data transmission time)

What about the ECU / computer



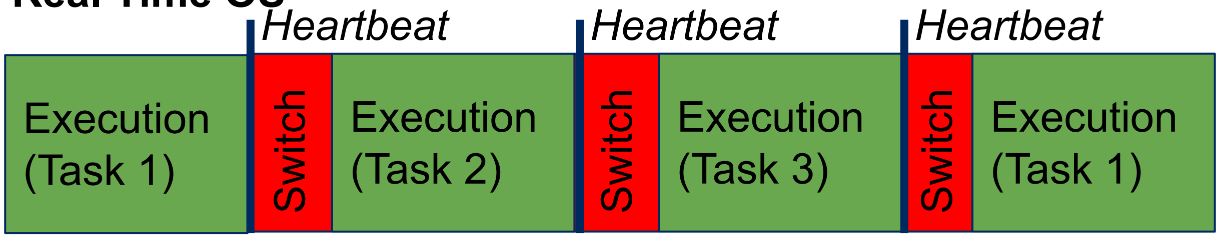
What about the micro-controller / ECU?

General Purpose OS

Execution time optimized to minimize switching

Time until resumption dependent on tasks - non-deterministic

Real Time OS



Within a priority level* execution time fixed

Time until task is executed again is deterministic

*Modern RTOS usually operate with a priority scheduler that runs the next highest priority task that is ready, when an event occurs that can change readiness the scheduler is run. Time slicing is still used within a given priority level

A Few Caveats

Several things limit true determinism even with an RTOS

Hard real time ECU solve this with hardware features

-Out of order (OOO) execution
-Branch Prediction
-Speculative Execution
→ Implement Heartbeat in hardware

-Cache hit / miss → Build in / use tightly coupled memory (TCM)

-Low level interrupts / exceptions → Build in hardware exception handling

Trade-off between performance and Real-time

<u>High Throughput</u>

-Changes in context / switching minimized
- Use resources when available (order less important)

<u>Real Time</u>

-Switching between processes maximized
-Use resources in correct order

However we aren’t just talking a single process - we have multiple nodes

ILLUSTRATIVE

Step 1 → Step 2 → Step 3 → Step 4 → Step 5

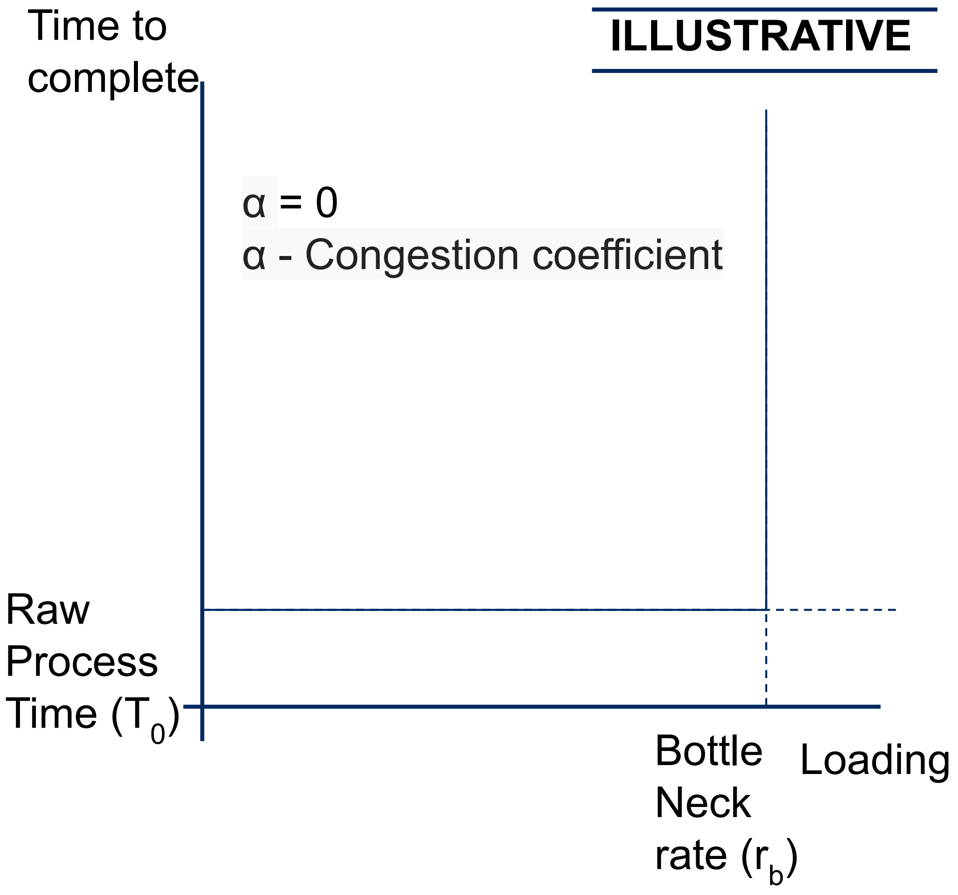
In a multi-node process where:

-Each step take identical time

-There are no shared resources or loop-backs

-The network is large

-All the steps are synchronized



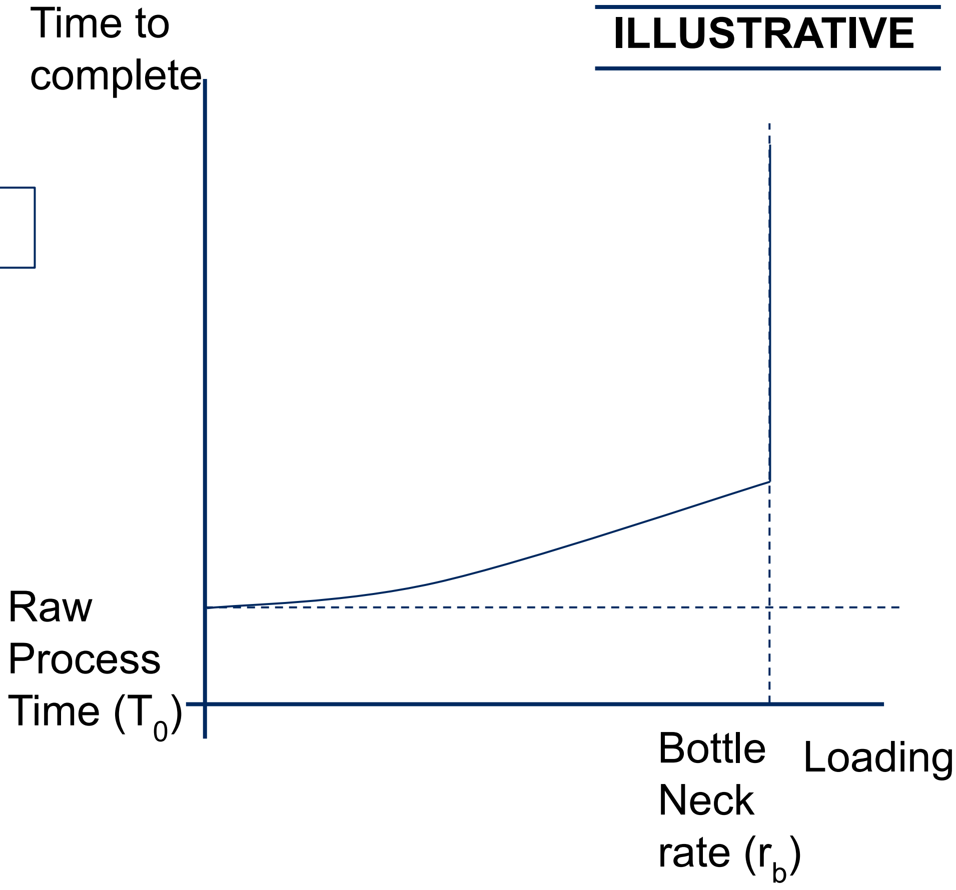
Multi-node process with different completion times

In a multi-node process where:

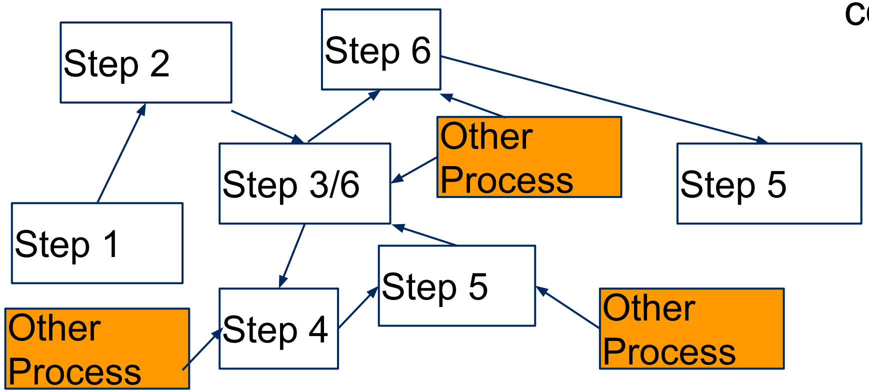
-Steps take different time

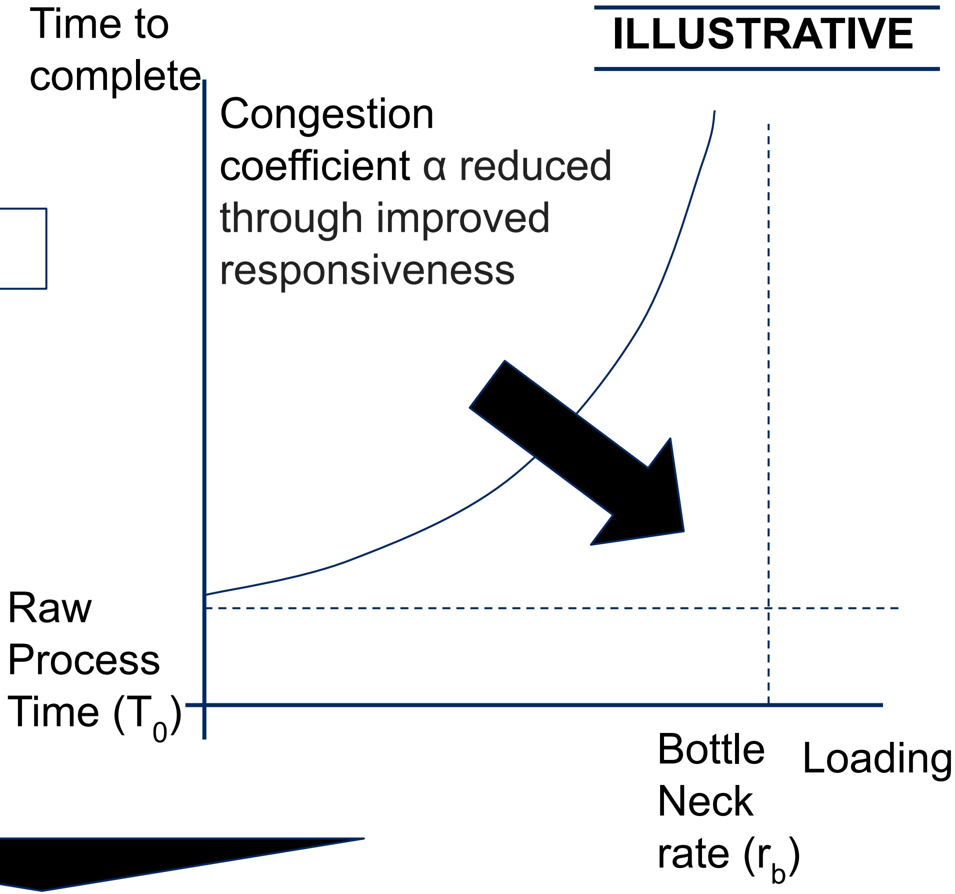
-There are no shared resources or loop-backs

-The network is large



Complex multi-node





In a multi-node process where:
- Steps take different time
- There are shared resources and potentially loop-backs
- The network is large with many nodes
- Queuing happens via retry

Real time techniques may help performance in complex multi-node systems

What hardware do we need to do real time!

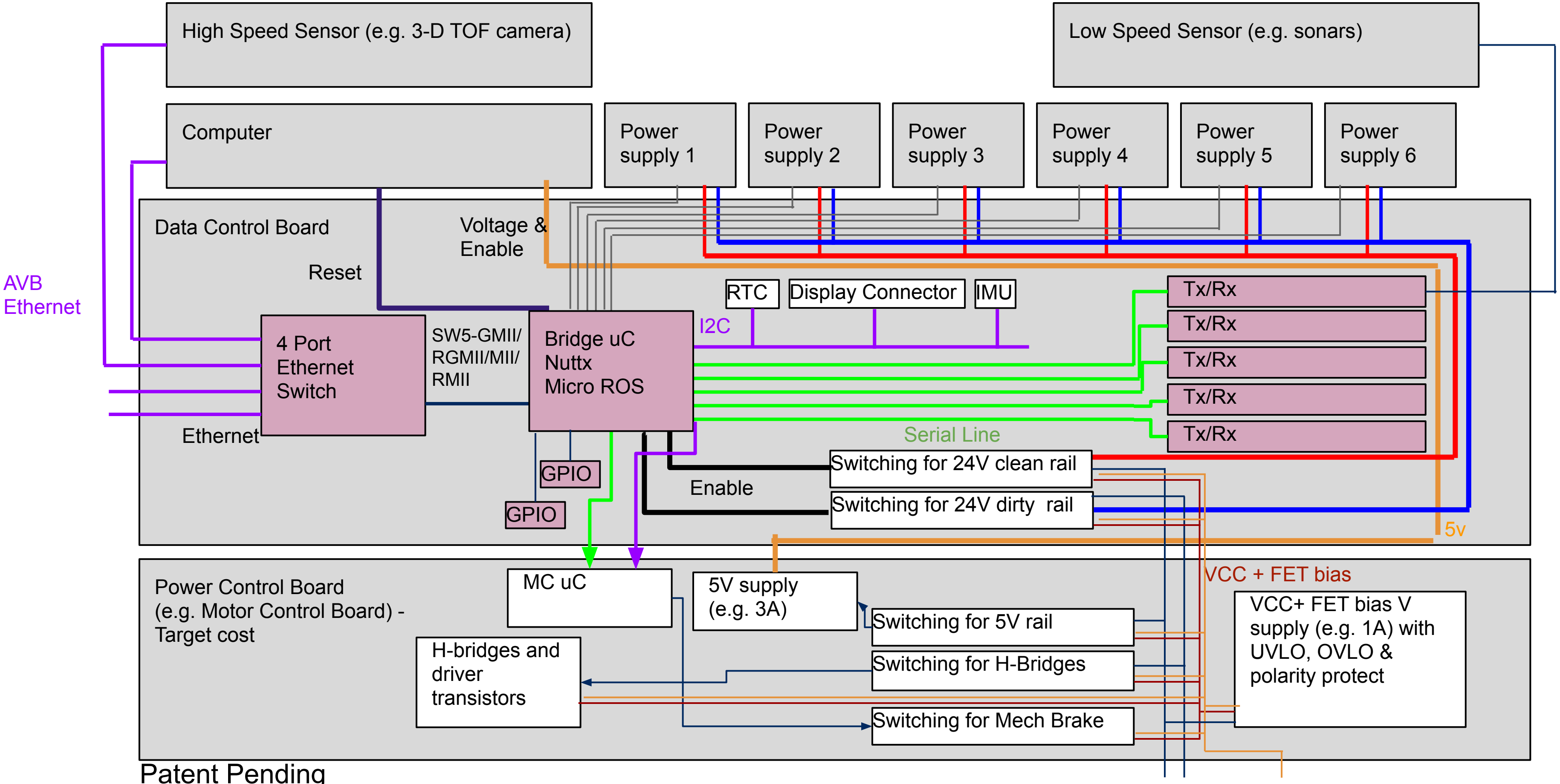
-Time sliced network fabric with bandwidth greater than all sensor output and decision output

- Low cost point to point communication fabric to handle simple sensors

- Processing unit with RTOS and real time support

- A bunch of other things not covered here!

Proposed Magni V6 Architecture



Patent Pending

Conclusion

Real time capability is not a standard feature of most communications hardware but is within reach due to relatively recent standards work

RTOS are an important part of real time ECU but there are several other hardware considerations

Ubiquity Robotics will be releasing hardware for real time for low cost robotics

Reach out for more info

David Crawley

Phone / Whatsapp +1 415 309 8966

dc@ubiquityrobotics.com

Appendix