CPSC 875

John D. McGregor

ABS module – a work in progress

Purpose

- This is a unit that blends software architecture information with telematics information regarding anti-lock braking.
- Several resources are provided:
  - An AADL model
  - Results of using associated tools is provided
  - Information about ABS systems

Background information

- The material in this url is fundamental background
- http://teachersites.schoolworld.com/webpages/MTurner/files/studyguide%20legacy%20brakes.pdf
- ABS is a separate system from the main braking system. It begins operation when the wheels are observed to be slipping or about to lock up.

Context

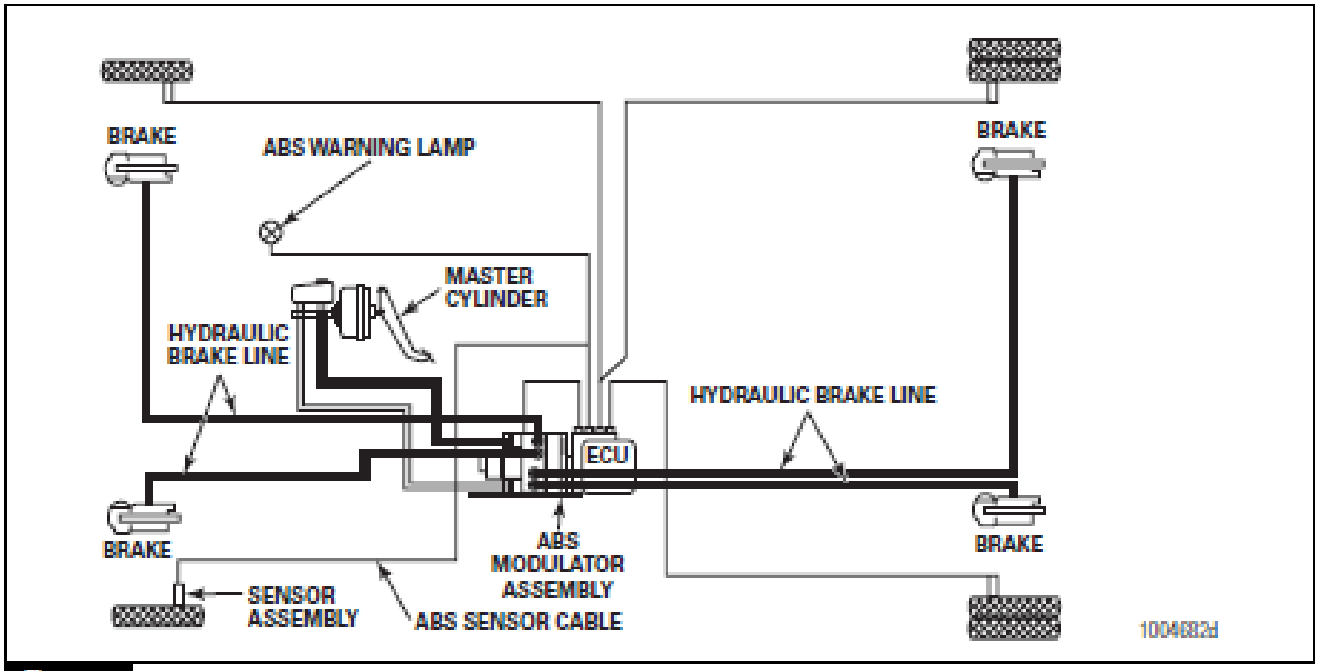


Figure 1.2

Much of the information in this module was taken from The Meritor WABCO E-Version Hydraulic ABS System

Mechanic's view

- Fault information is shown on a special screen in a diagnostic computer when it is attached to the vehicle.

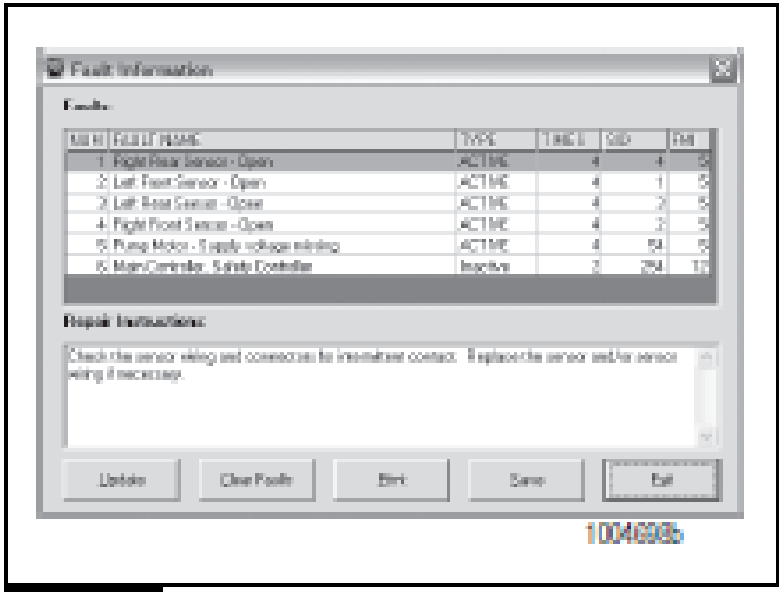


Figure 2.6

ABS

- An anti-locking brake system is one that monitors the speed of each wheel and senses when one or more of the wheels is about to lock up. The system releases pressure on that wheel to avoid the lock up but reapplies the pressure as soon as a threshold speed is reached.
- A typical system is composed of a controller, a pump, valves to release the pressure, and sensors to determine wheel speed.

ECU Control Loop

SENSORS

- Throttle position
- Intake air temperature
- Manifold air pressure
- Mass air flow (MAF)
- Fuel pressure
- In-cylinder pressure
- Coolant temperature
- Crankshaft position
- Camshaft position
- Engine speed
- Engine knocking
- Exhaust gas oxygen

PROCESS CONTROL LOOPS

Engine Control Unit (ECU)

ACTUATORS

- Fuel injection
- Idle speed control
- Ignition timing
- Multispark timing
- Dwell angle
- Valve timing (VVT)
- Camless valve actuation
- Exhaust gas recirc. (EGR)
- Turbo boost
- Transmission control

http://www.onsemi.com/pub_link/Collateral/TND393-D.PDF

Controller pump

Valve in each line sensor

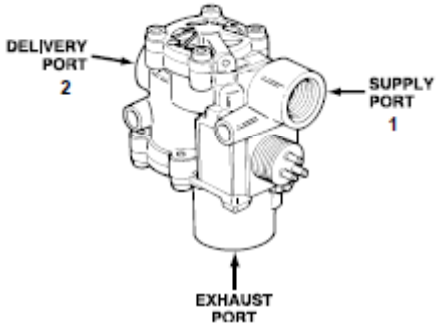
Valve in each line sensor

Valve in each line sensor

Valve in each line sensor

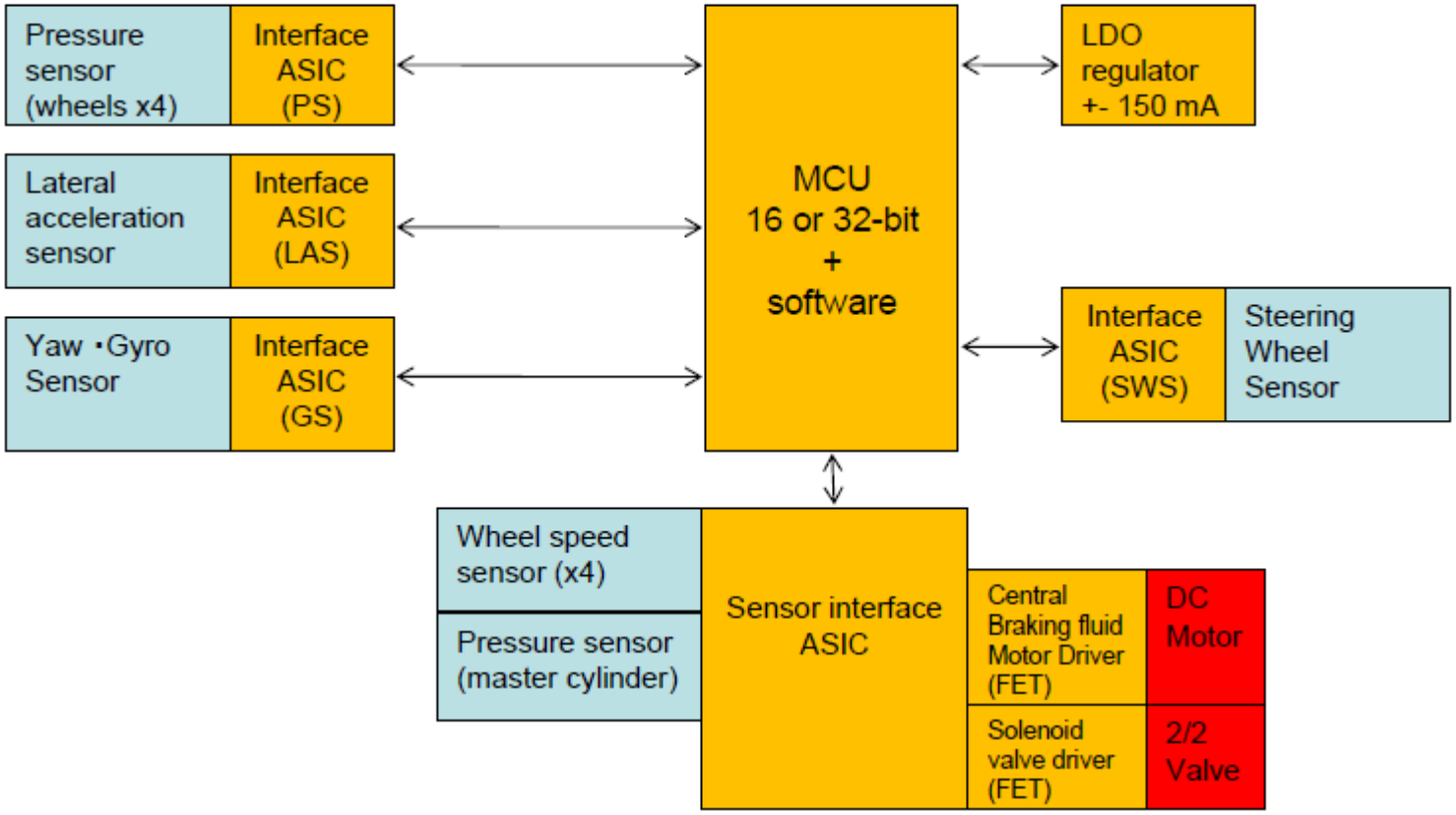
- For each wheel there is one sensor

and one valve



There is one controller and one pump for the complete system.

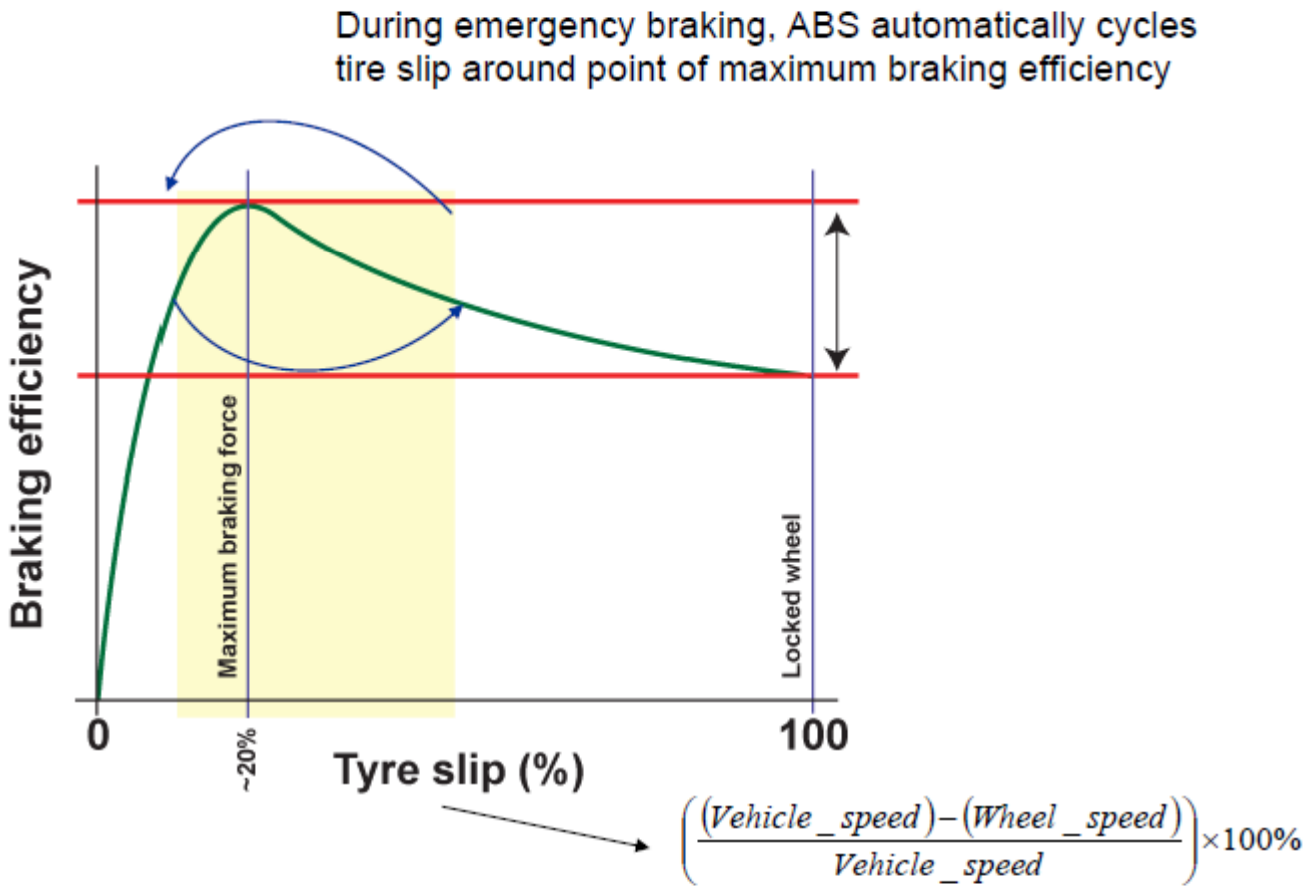
architecture



Speed sensors

- There are several different manufacturers of ABS systems.
- There are also several different technologies for sensing the speed at which the wheel is turning. Most, if not all of them, have a sensor that transmits an AC current to the controller. The frequency of that current is characteristic of the speed at which the wheel is turining.

ABS Principle



http://www.onsemi.com/pub_link/Collateral/TND393-D.PDF

Automotive Protection

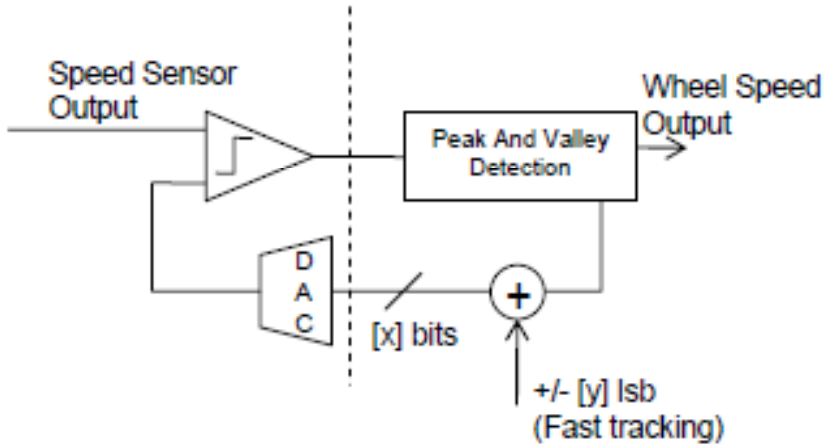
- Overvoltage and reverse battery (OVRB) protections
- Electrostatic discharge (HBM, MM, CDM…)
- Automotive transients:
  - AEC Q100 automotive standards
  - ISO 7637 pulses
    - Load dump
    - Schaffner pulses
  - Other local standards
- Output shorted to battery or ground
- Current sensing and limiting
- Over-temperature protection

less common in sensor interface

http://www.onsemi.com/pub_link/Collateral/TND393-D.PDF

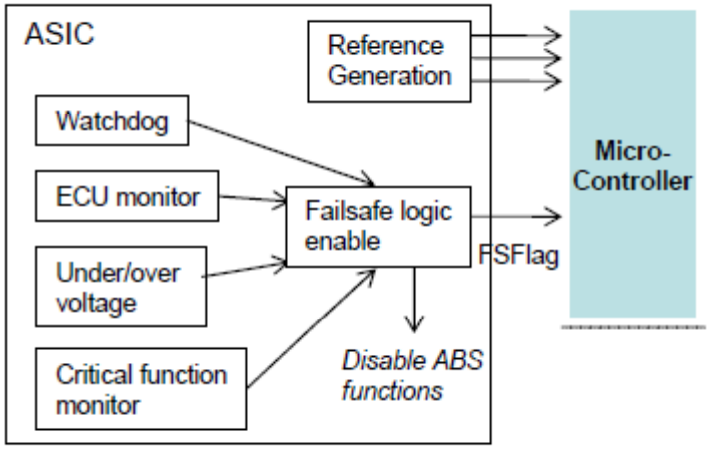
Speed sensor

- A control feedback loop is used to measure speed. The frequency of the current generated by brushes in the wheel encodes the speed.



Fault sensing

- A number of sensors are included in the ABS controller. Any one of them can generate a failure event.



Sensors

- General list of sensors

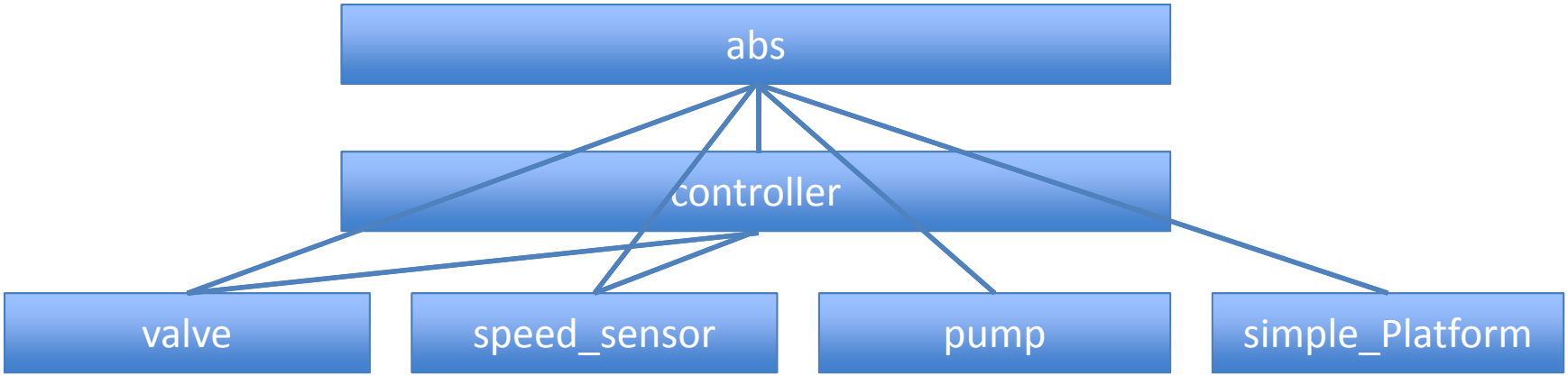
Physical quantity	Sensor	Electrical signal
Pressure	piezoresistive bridge	voltage
Air flow	thermistor	resistance
Angle / position	magnetic	inductance, resistance
Acceleration	MEMS capacitors	capacitance
Orientation (gyro)	MEMS tuning fork	charge
O_2 concentration	electrochemical	voltage
Urea concentration	thermistor	resistance
Distance	ultrasonic	voltage
Light	photodiode	current

Architecture model

- Software, in the form of firmware perhaps, runs in the controller to make decisions about which brakes to release and which to tighten to give maximum braking without the wheels locking up and the vehicle skidding.
- This module contains an evolving AADL model of a generic ABS system.

Layered dependencies

- Shows dependencies among packages
- It is a legal layered design



Design structure matrix (DSM)

	valve	pump	simple_Platform	speed_sensor	controller	abs
valve	.					
pump		.				
Simple_Platform			.			
Speed_sensor				.		
controller	x			x		
abs	x	x	x	x	x	.

This table shows dependencies in such a way as to see every thing below the diagonal. This signifies no circular dependencies.

Top level system that shows a complete configuration

```
system implementation abs.four
        subcomponents
          --system configuration for a 4 wheel vehicle
          c1: process controller::abs_controller.generic;
          s1: system speed_sensor::speed_sensor.generic;
          s2: system speed_sensor::speed_sensor.generic;
          s3: system speed_sensor::speed_sensor.generic;
          s4: system speed_sensor::speed_sensor.generic;
          v1: system valve::abs_valve.generic;
          v2: system valve::abs_valve.generic;
          v3: system valve::abs_valve.generic;
          v4: system valve::abs_valve.generic;
          p1: system pump::pump.generic;
          --internal computing details for the controller to support analyses
          RT_1GHz: processor simple_Platform::Real_Time.one_GHz;
          Standard_Marine_Bus: bus simple_Platform::Marine.Standard;
          Stand_Memory: memory simple_Platform::RAM.Standard;
        connections
          conn1: port s1.speed->c1.wheel1speed;
          conn2: port s2.speed->c1.wheel2speed;
          conn3: port s3.speed->c1.wheel3speed;
          conn4: port s4.speed->c1.wheel4speed;
          BAC2: bus access Standard_Marine_Bus<->RT_1GHz.BA1;
          BAC5: bus access Standard_Marine_Bus<->Stand_Memory.BA1;
        properties
          Allowed_Memory_Binding=>(reference (Stand_Memory)) applies to c1;
end abs.four;
```

State machine for a valve

system implementation abs_valve.generic

modes

opened: **initial mode;**

blocking: **mode;**

releasing: **mode;**

opened-[block]->blocking;

blocking-[release]->releasing;

releasing-[open]->opened;

end abs_valve.generic;

The valve has three states: opened, blocking, and releasing. The implementation should have methods for moving into each of these states.

Binding

- Deployment_Properties::Actual_Memory_Binding => (**reference (Stand_Memory)) applies to c1;**
- Deployment_Properties::Actual_Connection_Binding => (**reference(Standard_CAN_Bus)) applies to conn1,conn2,conn3,conn4;**
- Deployment_Properties::Actual_Processor_Binding => (**reference (RT_1GHz)) applies to c1.selfTestThread,c1.operateThread;**

Thread properties

thread implementation operateThread.abs

calls operateSubProgram:{call_server: subprogram operate;};

properties

Dispatch_Protocol =>Periodic;

Compute_Execution_Time => 1 ms .. 2 ms;

Period =>50 ms;

end operateThread.abs;

While the ABS is actively braking this thread is dispatched periodically to check and release the brake if needed.

Thread properties

thread implementation selfTest.abs

calls selfTest:{call_server: subprogram doSelfTest;};

properties

Dispatch_Protocol =>Sporadic;

Compute_Execution_Time => 1 ms .. 2 ms;

end selfTest.abs;

This thread is dispatched sporadically when the ignition is turned on. The self test takes 1 to 2 ms to complete.

A subprogram named “doSelfTest” executes the self test actions.

Subprogram calls

thread implementation operateThread.abs

calls operateSubProgram:{call_server: subprogram operate;};

connections

c1: **parameter ws1 -> call_server.a1;**

c2: **parameter ws2 -> call_server.a2;**

c3: **parameter ws3 -> call_server.a3;**

c4: **parameter ws4 -> call_server.a4;**

c5:**parameter call_server.r1-> vs1;**

c6:**parameter call_server.r2-> vs2;**

c7:**parameter call_server.r3-> vs3;**

c8:**parameter call_server.r4-> vs4;**

properties

Thread_Properties::Dispatch_Protocol =>Periodic;

Timing_Properties::Compute_Execution_Time => 1 ms .. 2 ms;

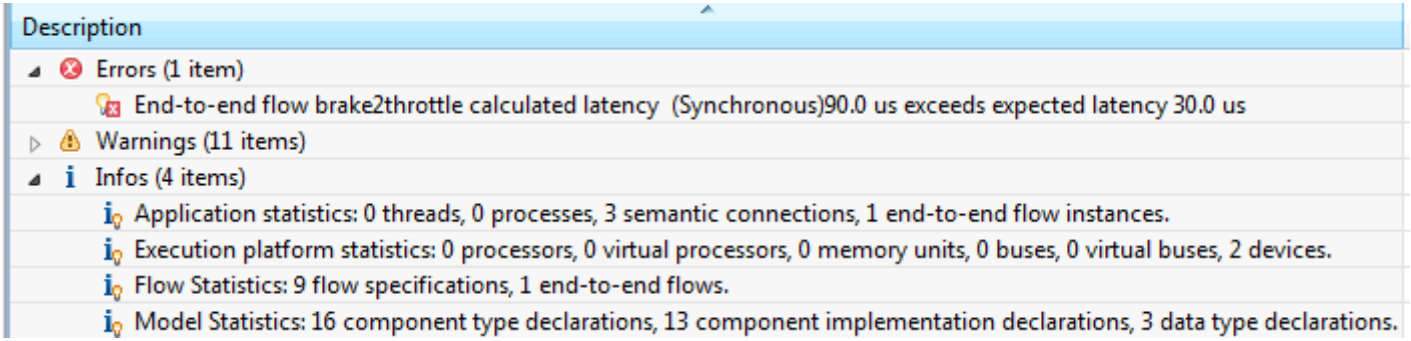
Timing_Properties::Period =>50 ms;

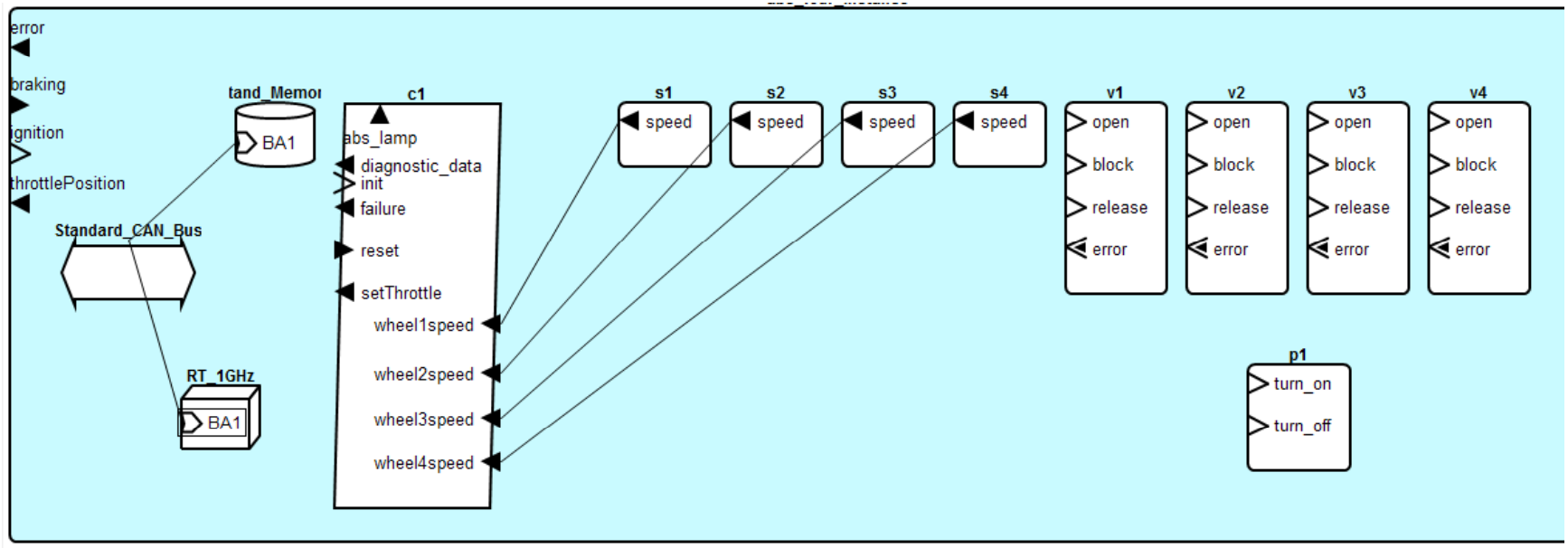
end operateThread.abs;

Tools

	A	B	C	D	E	F	G	H	I	J	K	L
1	owner	flow	model ele	name	deadline (	sampling (	partition (	flow spec	additional	total (ms)	expected (ms)	
2												
3												
4	ETEF (Syn	brake2thr	Subcompo	b	0.0 us	0.0 us	0.0 us	0.0 us	0.0 us	0.0 us	30.0 us	
5	ETEF (Syn	brake2thr	Connectic	b.position	30.0 us	0.0 us	0.0 us	0.0 us	30.0 us	30.0 us	30.0 us	
6	ETEF (Syn	brake2thr	Subcompo	abs1:norm	0.0 us	0.0 us	0.0 us	30.0 us	60.0 us	60.0 us	30.0 us	
7	ETEF (Syn	brake2thr	Connectic	abs1.throt	30.0 us	0.0 us	0.0 us	0.0 us	90.0 us	90.0 us	30.0 us	
8	ETEF (Syn	brake2thr	Subcompo	t:throttle	0.0 us	0.0 us	0.0 us	0.0 us	90.0 us	90.0 us	30.0 us	

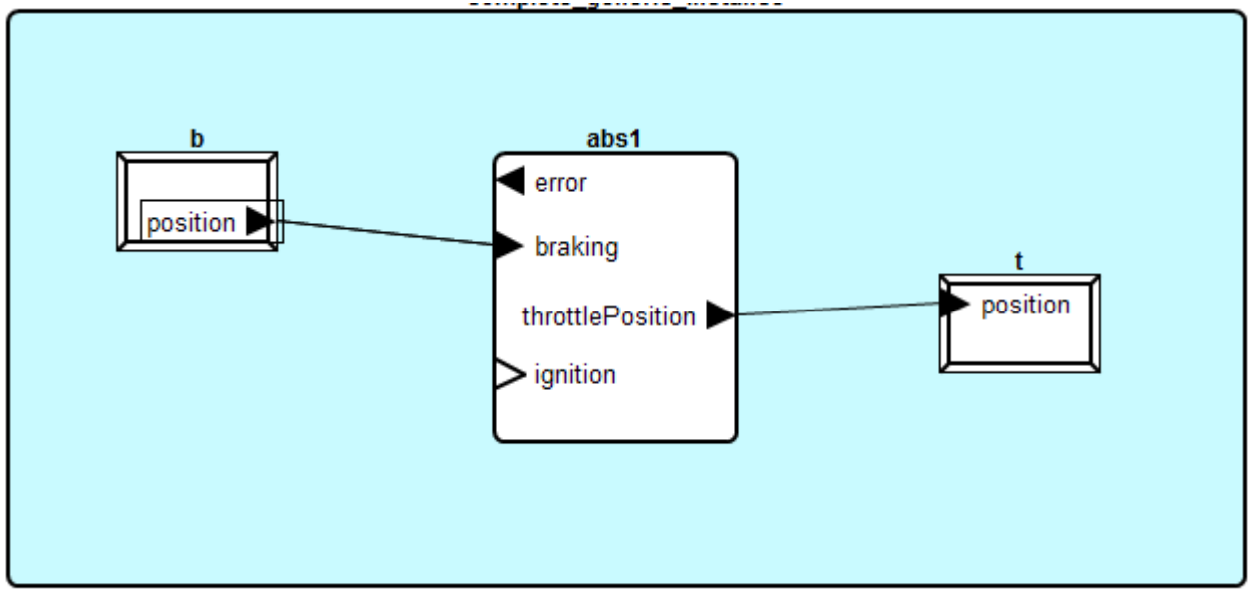
Description	Resource	Path	Location	Type
Errors (1 item)				
End-to-end flow brake2throttle calculated latency (Synchronous)90.0 us exceeds expected latency 30.0 us	complete_complete_generic_Instance.aaxl2	/telematicsImplem...	Unknown	Flow Latency Analysis Marker
Warnings (11 items)				
Infos (4 items)				
Application statistics: 0 threads, 0 processes, 3 semantic connections, 1 end-to-end flow instances.	complete_complete_generic_Instance.aaxl2	/telematicsImplem...	Unknown	Model Statistics Marker
Execution platform statistics: 0 processors, 0 virtual processors, 0 memory units, 0 buses, 0 virtual buses, 2 c	complete_complete_generic_Instance.aaxl2	/telematicsImplem...	Unknown	Model Statistics Marker
Flow Statistics: 9 flow specifications, 1 end-to-end flows.	complete_complete_generic_Instance.aaxl2	/telematicsImplem...	Unknown	Model Statistics Marker
Model Statistics: 16 component type declarations, 13 component implementation declarations, 3 data type	complete_complete_generic_Instance.aaxl2	/telematicsImplem...	Unknown	Model Statistics Marker





C1 – controller
S1 – s4 – speed sensors
V1 – v4 – valves
P1 – pump

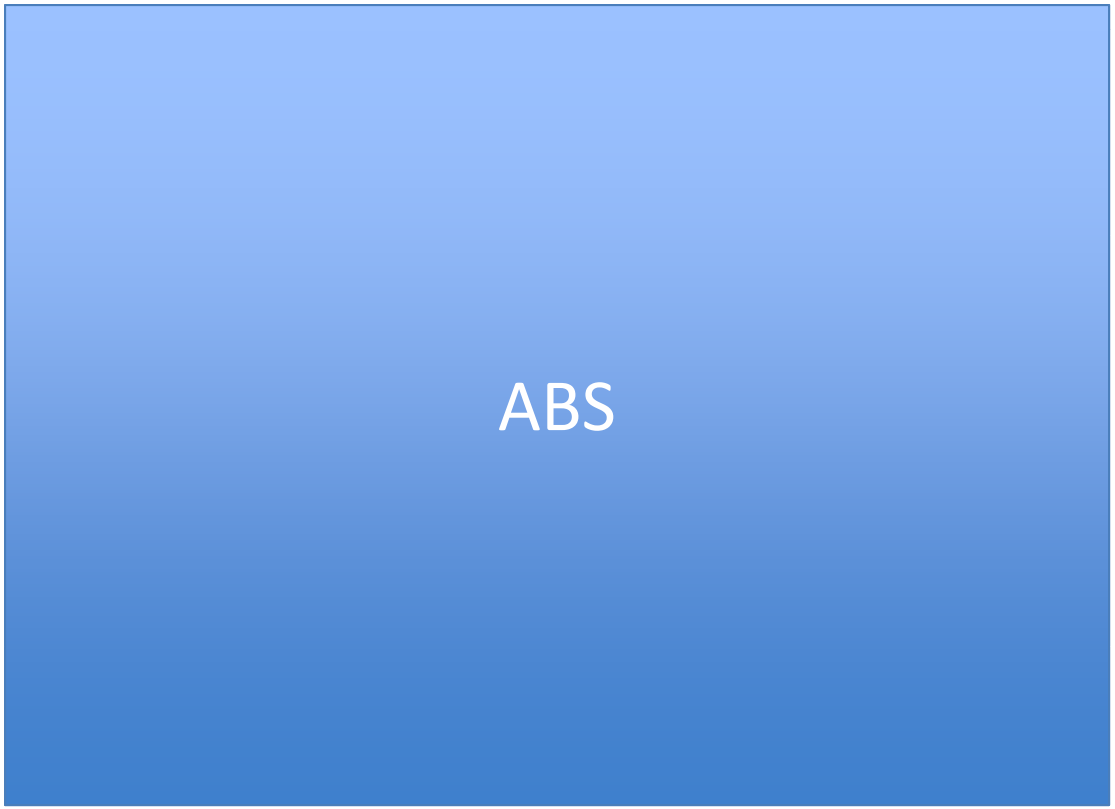
Context model for ABS system



b = brake
t = throttle
abs1 = anti-lock brake system

Architecture evolution

- We begin a new system with essentially a monolithic block.

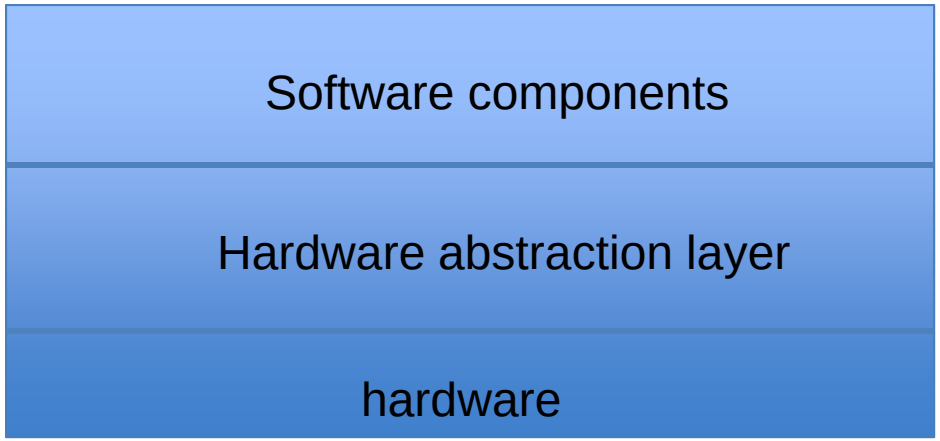


Alternatives

- Decompose into software based on hardware elements

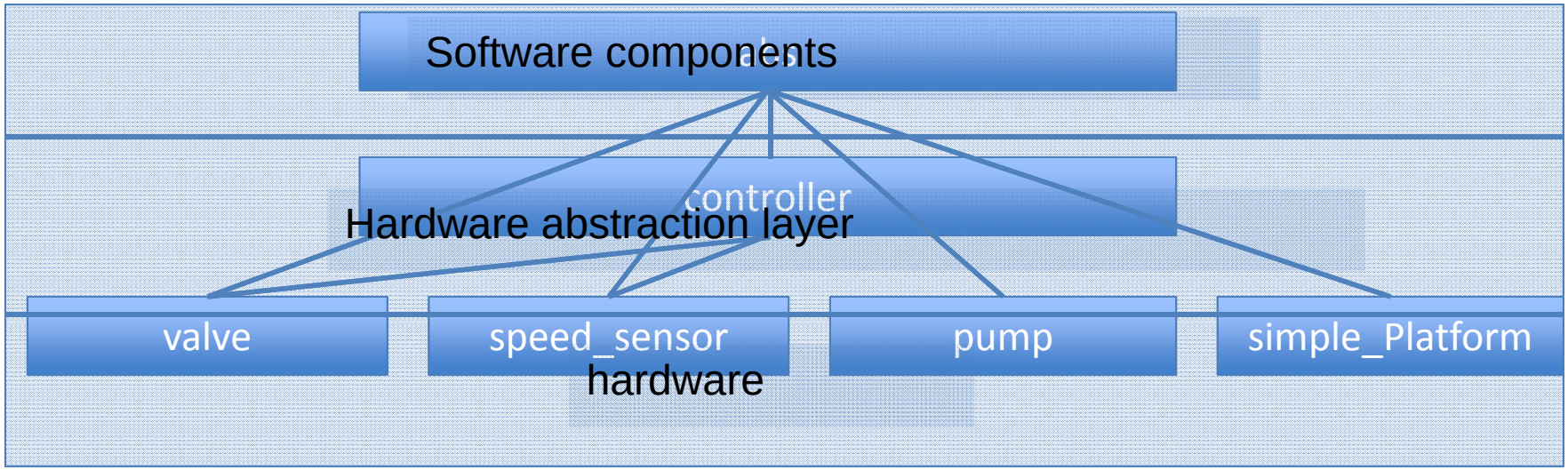
Both are more modular

- Or a canonical layered decomposition



Combination

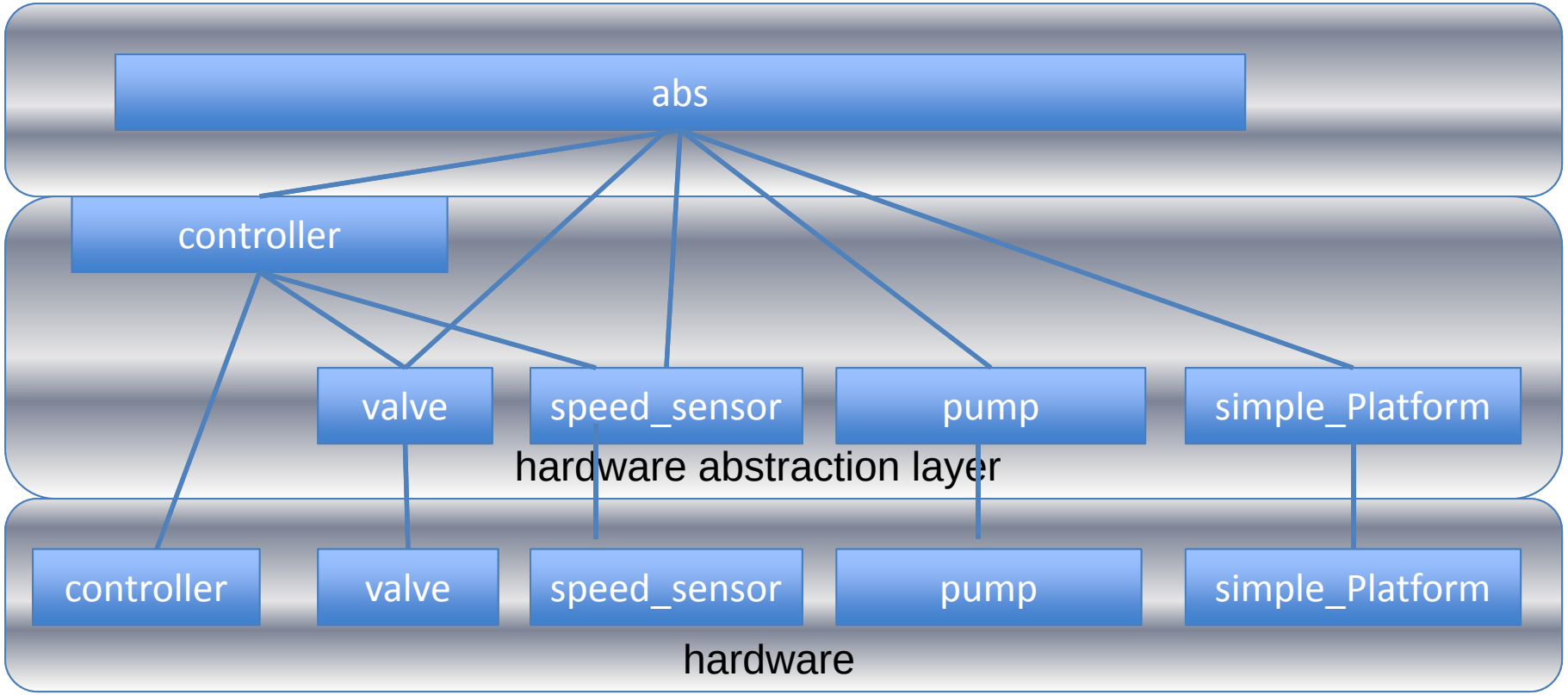
- The abs component is what is left of the monolith and represents the root of the aggregation hierarchy
- The other pieces represent the controller functional hierarchy



Further decomposition

- Hardware and hardware abstraction

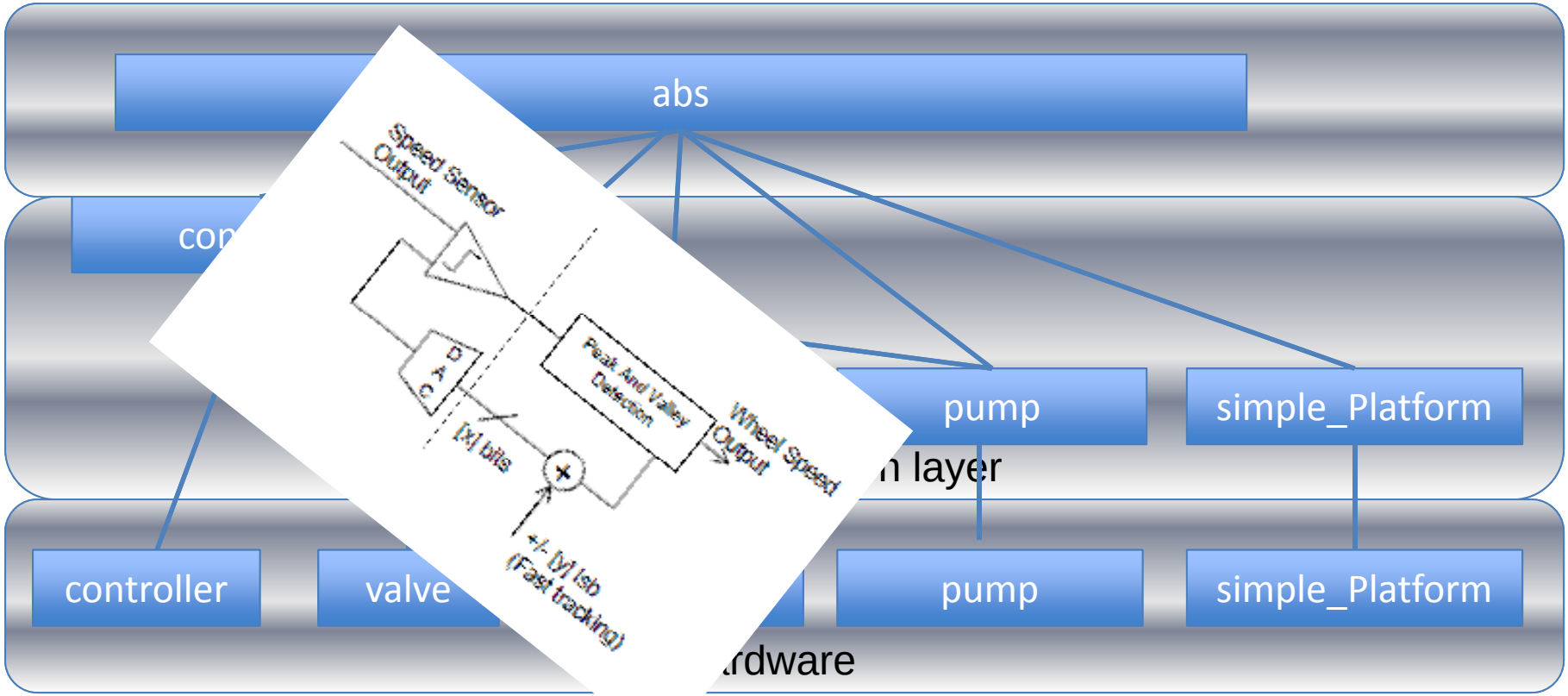
Abstraction increases portability



Control feedback loop

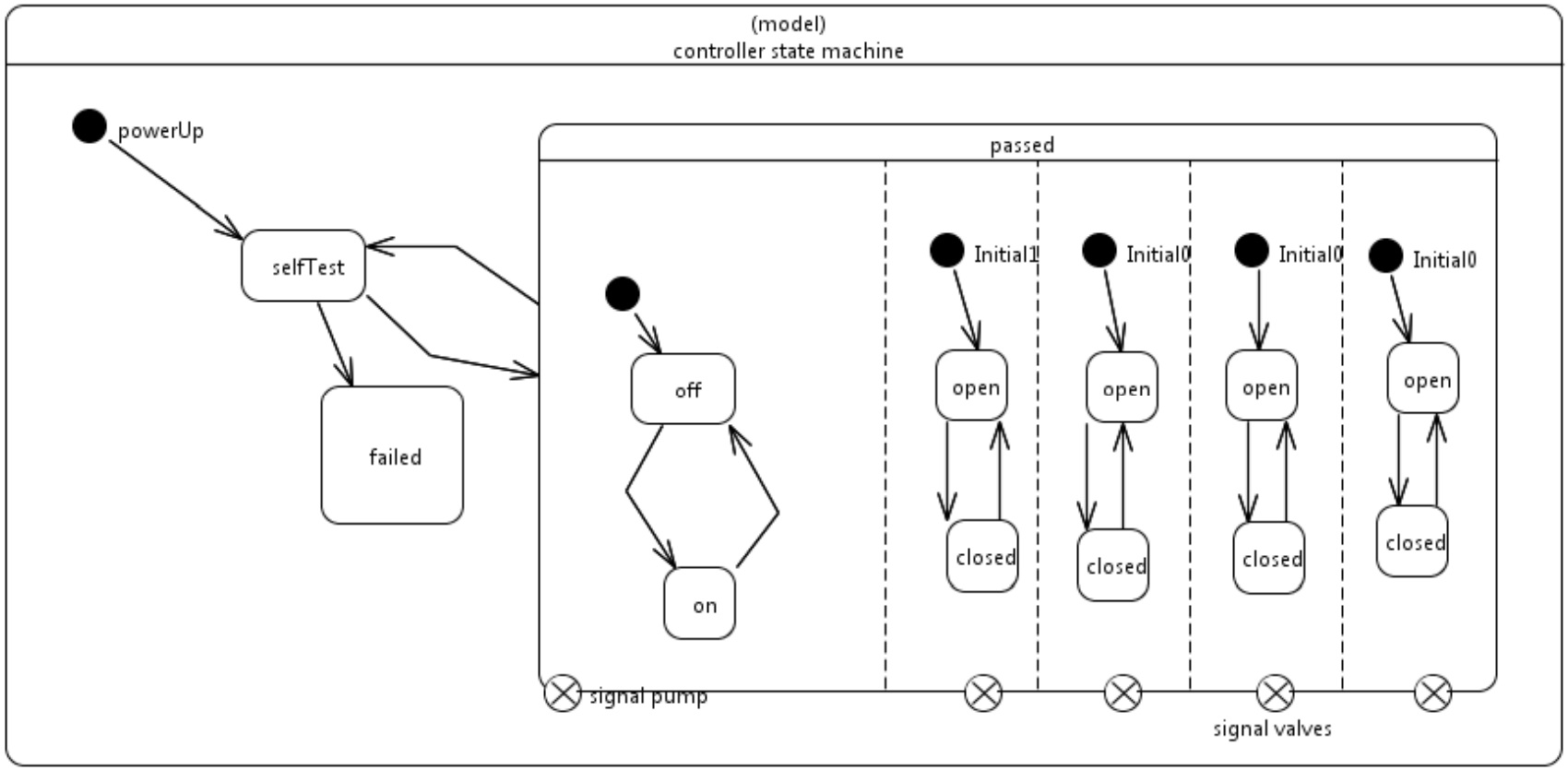
- All speeds are read, controller determines what to do, feedback to valves that changes the speed of 1 or more wheels

Simplicity decreases latency



Controller state machine

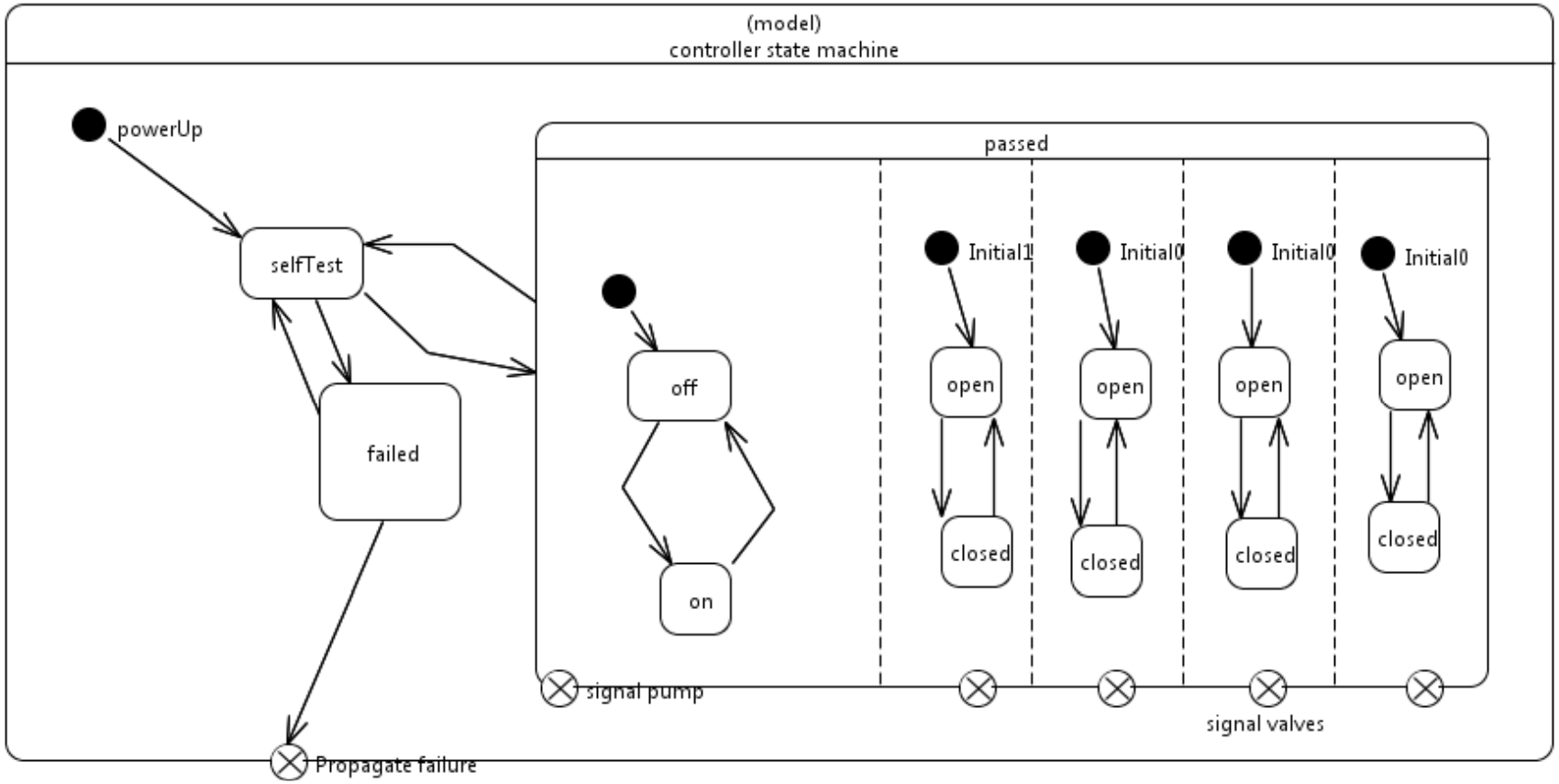
States encapsulate complexity



Threads

- The controller process is multi-threaded
- Synchronization is simple
  - two threads but only one active at a time
  - only one thread in the state machine at a time
- These are bound to the processor

With Error Path



Error handling

- In some cases a separate thread is used for error handling
- Definitely separate threads are used for watchdog timers, etc.

subprogram

```
subprogram implementation operate.generic
annex behavior_annex{**
    states
     s0: initial state;
     s1: return state;
    transitions
     normal: s0 -[]->s1{ovf:=false;};
     overflow: s0 -[]-> s1{ovf:=true;};
**};
end operate.generic;
```

references

These readings give some general and some specific information about ABS

- http://www.meritorwabco.com/MeritorWABCO_document/tp9738.pdf
- http://www.onsemi.com/pub_link/Collateral/TND393-D.PDF