TECHNISCHE UNIVERSITÄT WIEN

TTTech

Dependable Computer Systems

Part 1: Dependable systems and incidents

Contents

- Dependability Problem Statement
- Examples of dependable systems and incidents
- The Therac-25 accidents
- Unintended Acceleration Incidents
- Reasons for low dependability
- Concept of coupling and interactive complexity

Dependability Problem Statement

Our society depends on a broad variety of computer controlled systems where failures are critical and may have severe consequences on property, environment, or even human life.

Aims of this lectures

- to understand the attributes and concepts of dependability,
- to understand reasons for low dependability and
- gain knowledge on how to build dependable computer systems

Which dependable systems are you aware of?

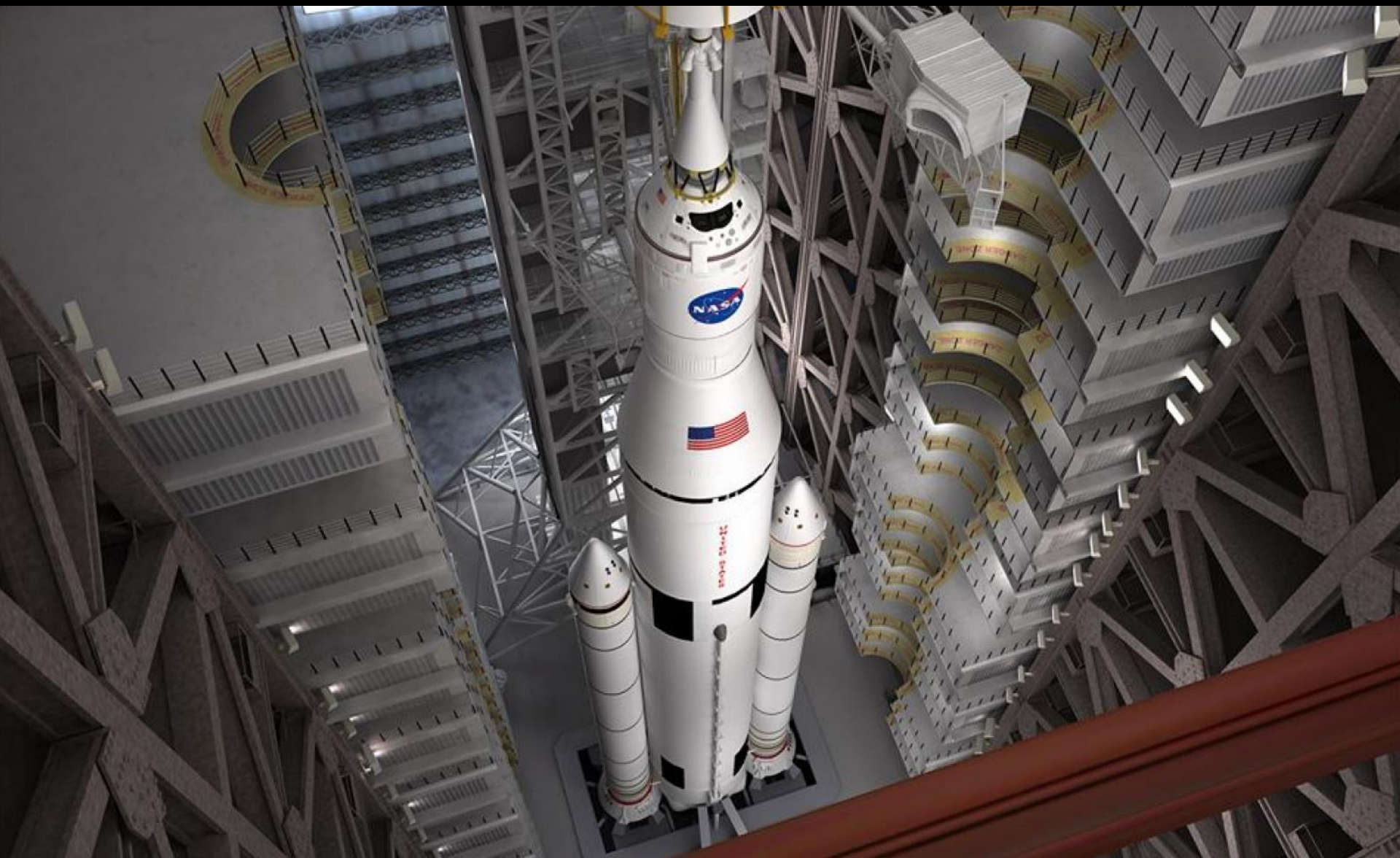
Boeing 787

NASA Orion

Audi A8

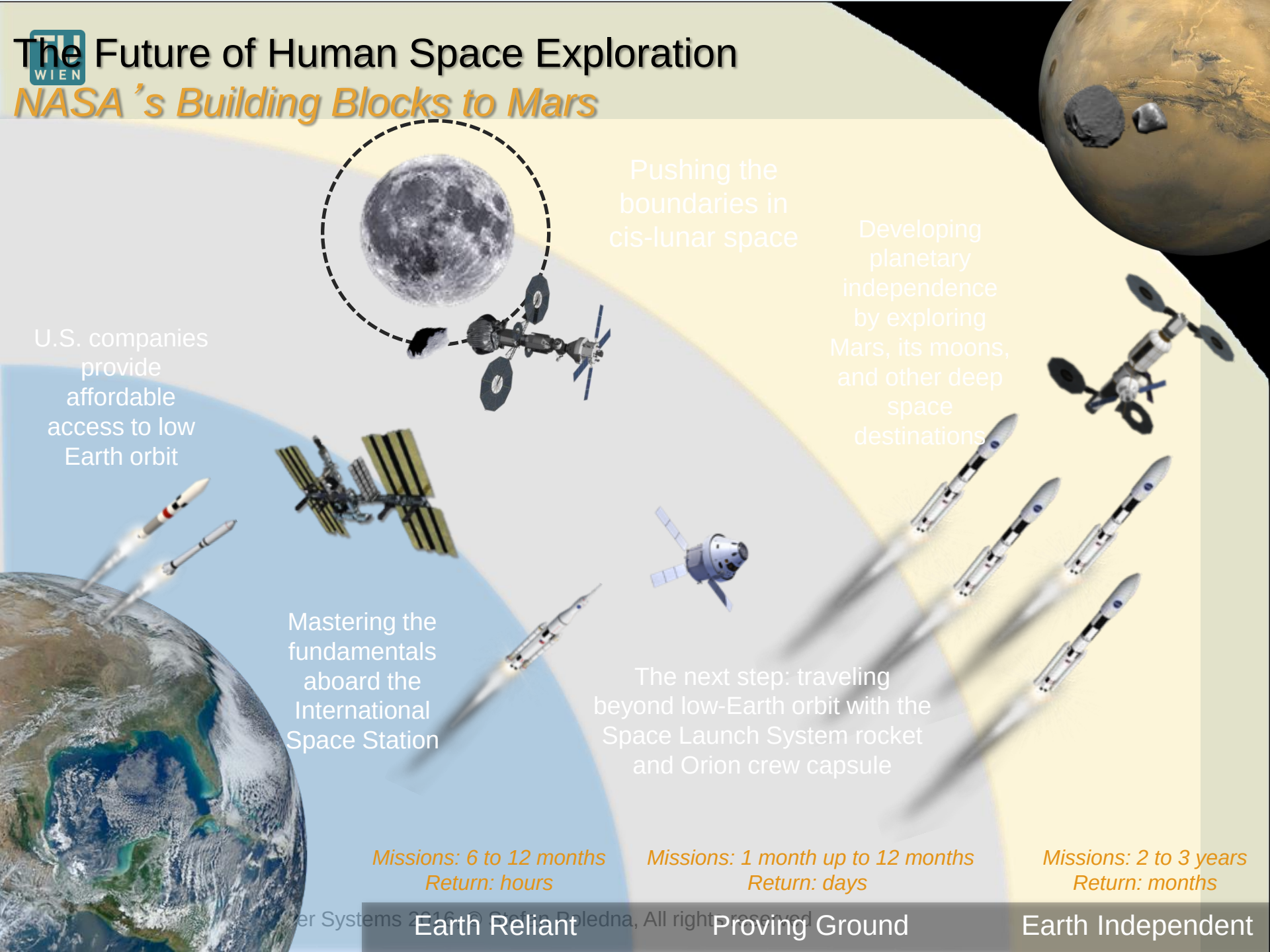
Airbus A380

America’s New Rocket: Space Launch System

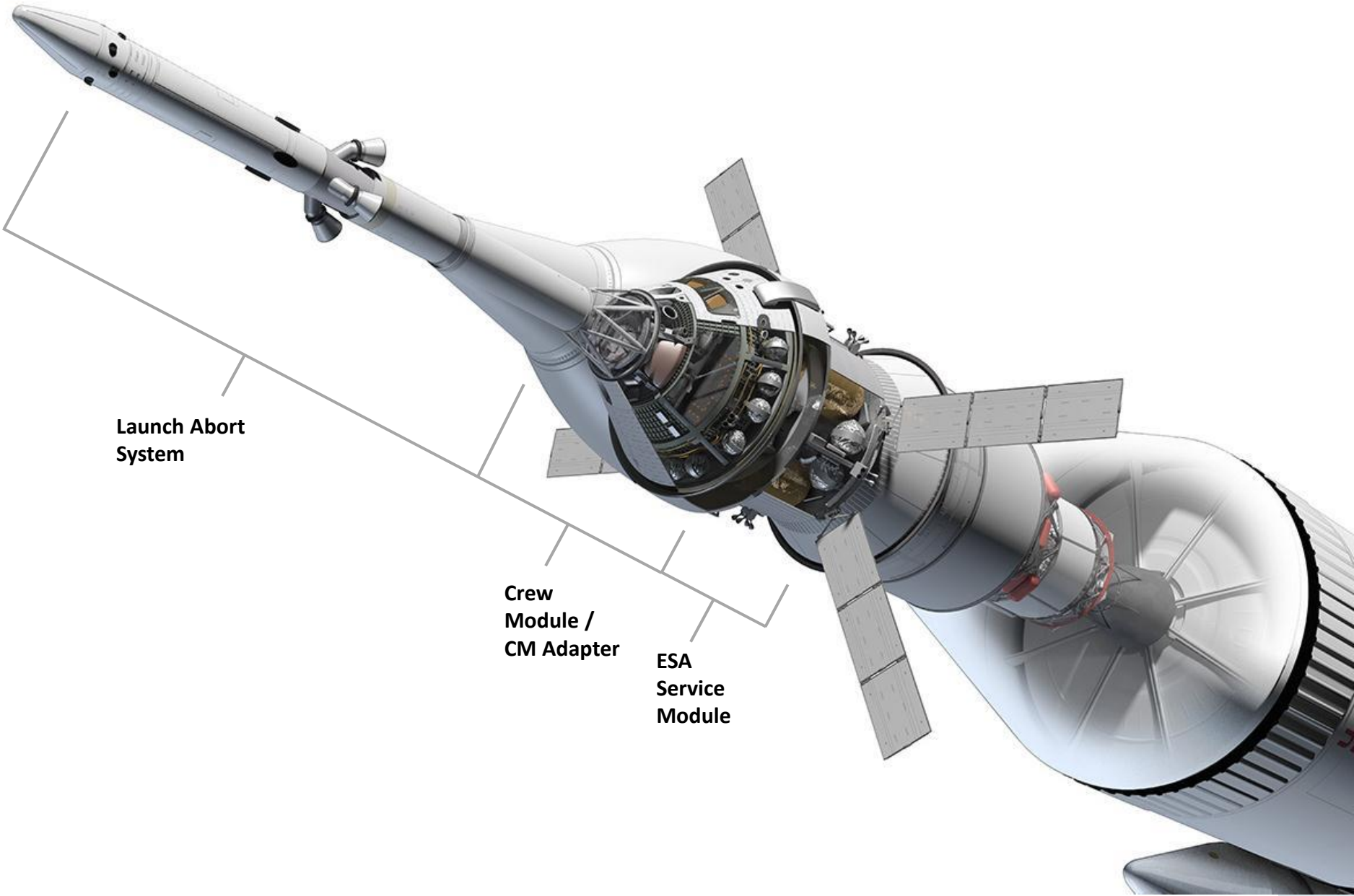


The Future of Human Space Exploration

NASA's Building Blocks to Mars

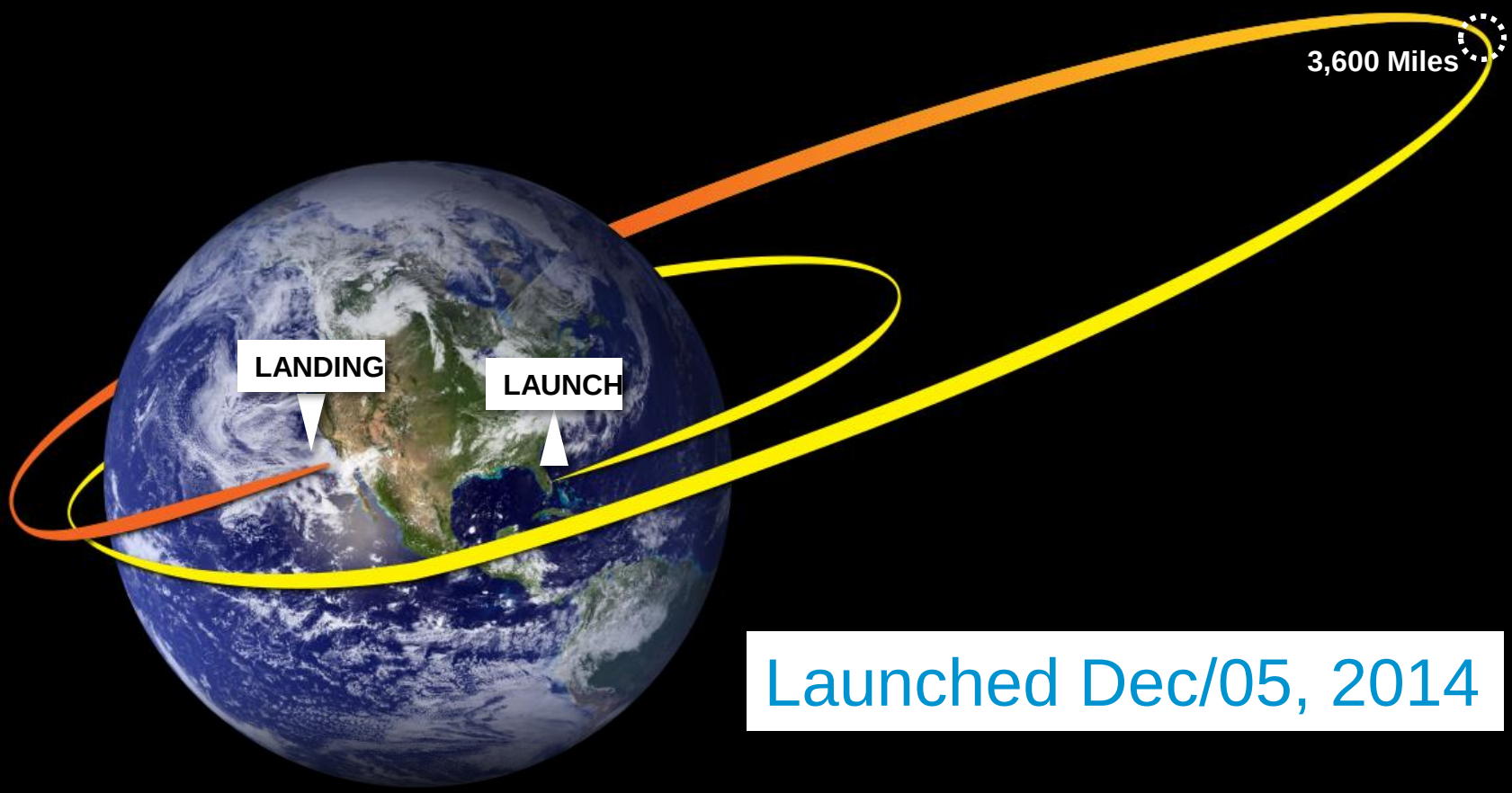


The Orion Spacecraft



This year NASA will fly a spacecraft built for humans farther than any has traveled in over 40 years.

2 Orbits | 20,000 MPH entry | 3,600 Mile Apogee | 28.6 Deg Inclination



Launched Dec/05, 2014

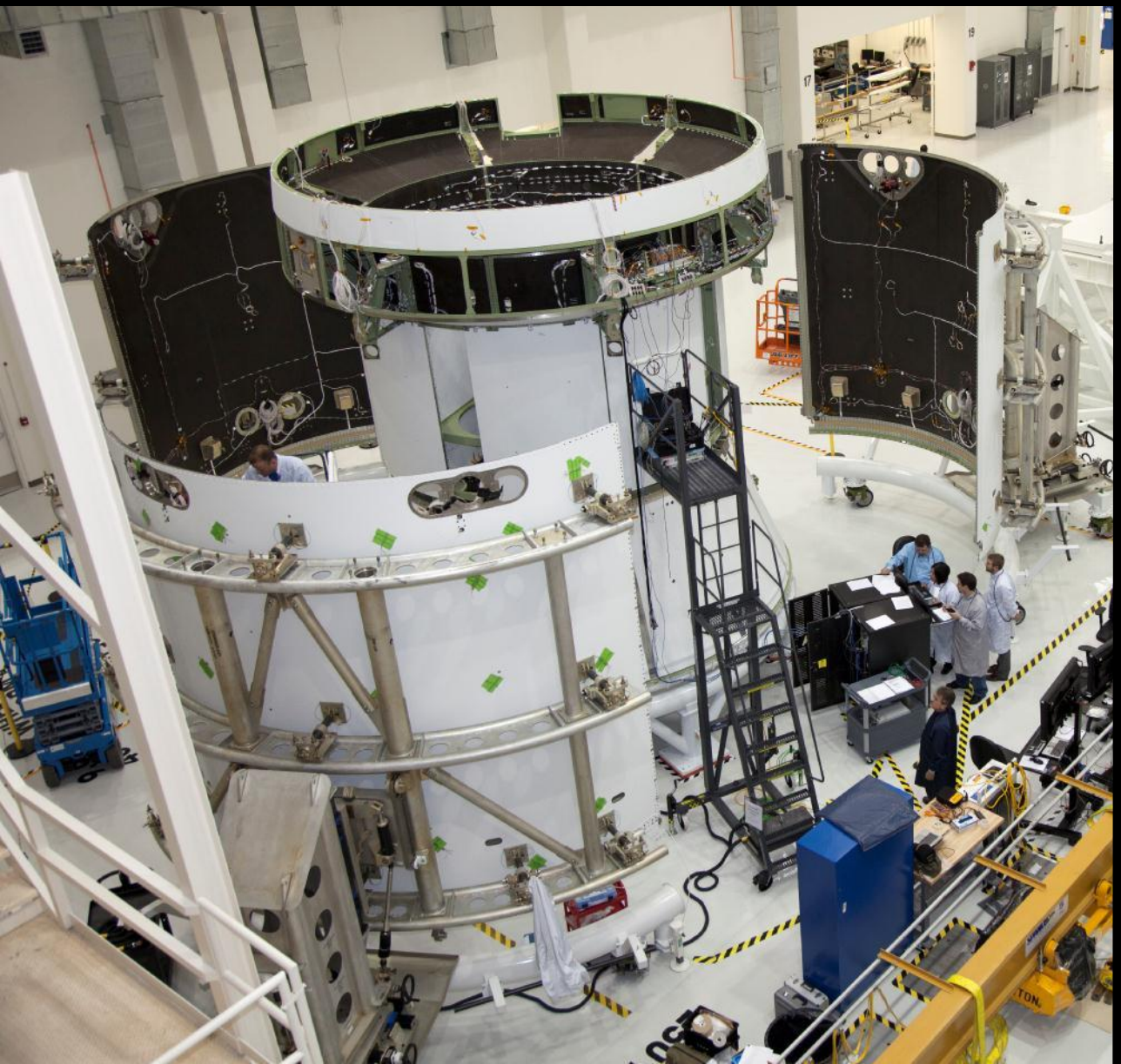
EFT-1 WILL EXERCISE 10 TOP LOSS OF CREW RISKS

Crew Module

Functional Testing Underway; On Track for May Delivery

Service Module

Assembly Complete – Ready for Integration



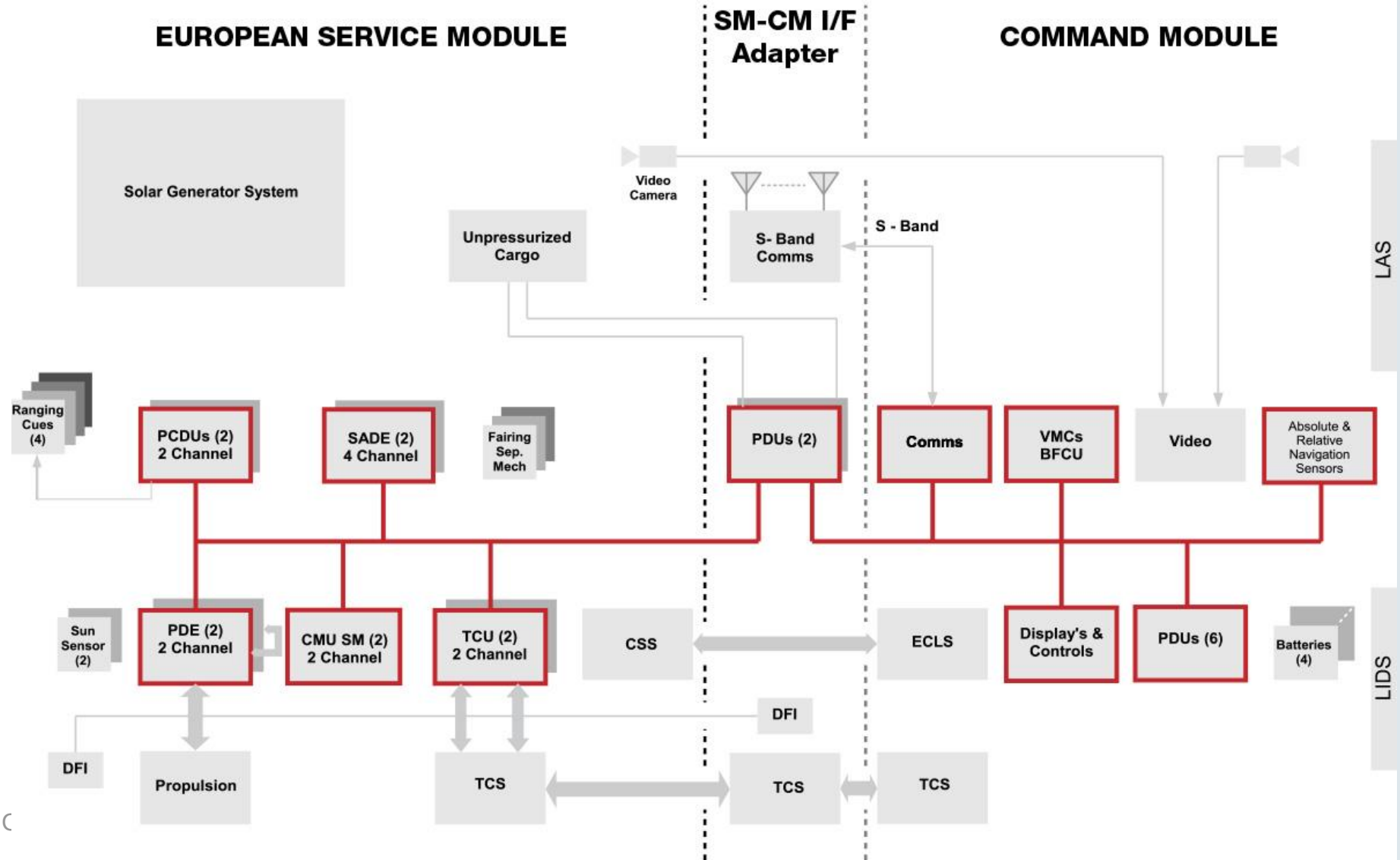
Launch Abort System

Assembly Complete – Ready for Integration

Time Triggered Gigabit Ethernet

The Backbone of Orion's State of the Art, High Reliability Avionics System

48 Network end points | 3 planes of connectivity for every device



Examples of dependable systems and incidents

“Fly-by-wire”

- pilot commands are transmitted as electrical commands
- a flight control system (FCS computer) is used
- the pilot flies the FCS and the FCS flies the plane
- military planes require FCS to get artificial stability
- for civilian use the advantages are:
  - weight savings
  - enhanced control qualities
  - enhanced safety

Fly-by-Wire Incidents

The SAAB JAS Gripen:

- 1989: Crash after sixth test flight due to exceeded stability margins at critical frequency, software was updated
- 1993: Crash on a display flight over the Water Festival in Stockholm,
  again due to pilot commands the plane became instable
- the cycle time of the Gripen FCS is 200 *ms*
- the probability of instability was estimated by the engineers as “sufficiently low”

The Airbus A320:

- 4 hull losses (plane crashes)
- all crashes are attributed to a mixture of pilot and computer or interface failures

A332, en-route, Atlantic Ocean, 2009

- 1 June 2009
- Airbus A330-200 being operated by Air France on a scheduled passenger flight from Rio de Janeiro to Paris CDG as AF447
- exited controlled flight and crashed into the sea with the loss of the aircraft and all 228 occupants
- loss of control followed an inappropriate response by the flight crew to a transient loss of airspeed indications in the cruise which resulted from the vulnerability of the pitot heads to ice crystal icing.

http://www.skybrary.aero/index.php/A332,_en-route,_Atlantic_Ocean,_2009

Patriot vs. Scud

During gulf war a Scud missile broke through the Patriot anti-missile defense barrier and hit American forces killing 28 people and injuring 98.

A software problem

- time is represented as an 32 *bit* integer and converted to 24 *bit* real number
- with the advent of time this conversion loses accuracy
- tracking of enemy missiles becomes therefore faulty
- the software problem was already known, and the update was delivered the next day

Critical Infrastructure Incidents

Bank of America financial system:

- development during 4 years costs $20 millions
- $60 millions in overtime expenses
- $1.5 billion in lost business
- system was abandoned after nearly one year in service

Airport of Denver, Colorado

- one of the largest airports worldwide
- intelligent luggage transportation system with 4000 “Telecars”, 35 *km* rails, controlled by a network of 100 computers with 5000 sensors, 400 radio antennas, and 56 barcode readers
- due to software problems about one year delay which costs 1.1 million $ per day

The “Bug”

Harsh environment:

- The “bug”: On a Mark II in 1945 a moth came between relay contacts
- train cars were changed form external to disc brakes, trains vanished from display
- near a broadcast transmission tower it was possible to "hear rock and roll on the toaster"
- an overripe tomato hung over an answering machine, dripping tomato juice into the machine which caused repeated call to the emergency line
- pigeons may deposit a "white dielectric substance" in an antenna horn

Examples may seem funny but:

- system are designed to endure within a given operational conditions
- it is **very hard** to anticipate the operational conditions correctly
- illustrates difficulties of *good* system design

Which other (recent) incidents are you aware of?

The Therac-25 accidents

The Therac-25 accidents

Therac-25 is a machine for radiation therapy (to treat cancer)

Between June 1985 and January 1987 (at least) six patients received severe overdoses:

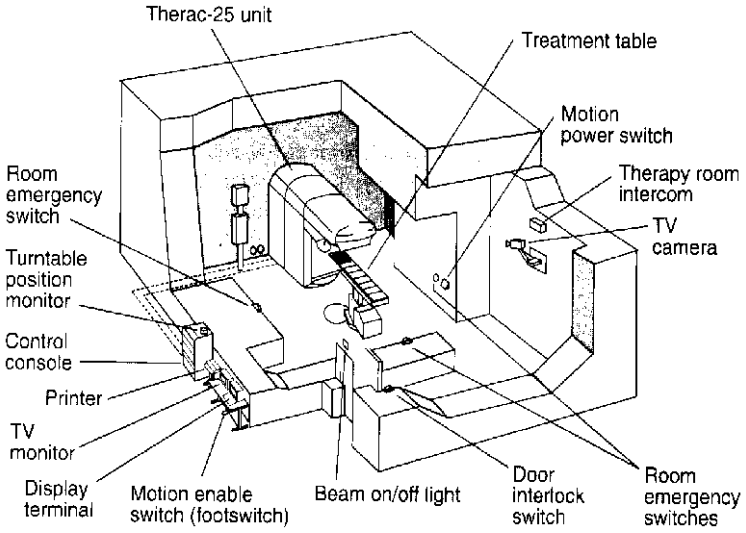
- two died shortly afterwards
- two might have died but died because of cancer
- the remaining two suffered of permanent disabilities

Functional principle

- “scanning magnets” are used to spread the beam and vary the beam energy
- Therac is a “dual-mode” machine
- electron beams are used for surface tumors
- X-ray for deep tumors

X-ray and Electron Mode

- a tungsten target and a “beam flattener” is moved in the path to the rotating turntable
- the target generates the X-rays but absorbs most of the beam energy
- the required energy has to be increased by a factor of 100, compared to electron mode



Typical Therac-25 facility

Major Event Time Line

1985

Jun

3rd: Marietta, Georgia, overdose.
Later in the month, Tim Still calls AECL and asks if overdose by Therac-25 is possible.

26th: Hamilton, Ontario, Canada, overdose; AECL notified and determines microswitch failure was the cause.

Jul

AECL makes changes to microswitch and notifies users of increased safety.
Independent consultant (for Hamilton Clinic) recommends potentiometer on turntable.

Sep

Georgia patient files suit against AECL and hospital.

Oct

8th: Letter from Canadian Radiation Protection Bureau to AECL asking for additional hardware interlocks and software changes.

Nov

Yakima, Washington, clinic overdose.

Dec

1986

Attorney for Hamilton clinic requests that potentiometer be installed on turntable.

Jan

31st: Letter to AECL from Yakima reporting overdose possibility.

24th: Letter from AECL to Yakima saying overdose was impossible and no other incidents had occurred.

Feb

Major Event Time Line (cont. 1986)

Mar
21st: Tyler, Texas, overdose. AECL notified; claims overdose impossible and no other accidents had occurred previously. AECL suggests hospital might have an electrical problem.

Apr
7th: Tyler machine put back in service after no electrical problem could be found.
11th: Second Tyler overdose. AECL again notified. Software problem found.
15th: AECL files accident report with FDA.

May
2nd: FDA declares Therac-25 defective. Asks for CAP and proper renotification of Therac-25 users.

Jun
13th: First version of CAP sent to FDA.
23rd: FDA responds and asks for more information.

Jul
First user group meeting.
26th: AECL sends FDA additional information.

Aug
30th: FDA requests more information.

Sep
12th: AECL submits revision of CAP.

Nov
Therac-20 users notified of a software bug.

Dec
11th: FDA requests further changes to CAP.
22nd: AECL submits second revision of CAP.

FDA =	US Food and Drug Administration
CAP =	Corrective Action Plan

Major Event Time Line (cont. 1987)

Jan
17th: Second overdose at Yakima.
26th: AECL sends FDA its revised test plan.

Feb
Hamilton clinic investigates first accident and concludes there was an overdose.
3rd: AECL announces changes to Therac-25.
10th: FDA sends notice of adverse findings to AECL declaring Therac-25 defective under US law and asking AECL to notify customers that it should not be used for routine therapy. Health Protection Branch of Canada does the same thing. This lasts until August 1987.

Mar
Second user group meeting.
5th: AECL sends third revision of CAP to FDA.

Apr
9th: FDA responds to CAP and asks for additional information.

May
1st: AECL sends fourth revision of CAP to FDA.
26th: FDA approves CAP subject to final testing and safety analysis.

Jun
5th: AECL sends final test plan and draft safety analysis to FDA.

Jul
Third user group meeting.
21st: Fifth (and final) revision of CAP sent to FDA.

1988

Jan
29th: Interim safety analysis report issued.

Nov
3rd: Final safety analysis report issued.

Lessons learned from Therac-25 accident:

- Accidents are seldom simple
- Accidents are often blamed to single source
- Management inadequacies, lack of following incident reports
- Overconfidence in software
- Involvement of management, technicians, users, and government
- Unrealistic risk assessment
- Less-than-acceptable software-engineering practices

Unintended Acceleration Incidents

Unintended Acceleration Examples

Sudden Acceleration Car Accidents Compilation.mp4

Toyota Unintended Acceleration Incident

2007/Sep: Toyota recall to fasten floor mats

2009/Aug: Toyota Lexus ES 350 sedan crash

- unintended acceleration reached 100 mph
- four passengers died, 911 emergency phone call during event
- crash was blamed on wrong floor mats causing pedal entrapment

2009/Oct: Extended floor mat recalls

2010/Jan: Sticky gas pedal recall

2010/Feb: US congressional investigation

2010/May: CBS News “Toyota Unintended Acceleration has killed 89”

2010-2011: NASA investigation of unintended acceleration

- conclusion: no electronic-based cause for unintended high-speed acceleration
- tight timeline and limited information

2012/Dec: Toyota settlement for $1.6 Billion USD

http://users.ece.cmu.edu/~koopman/pubs/koopman14_toyota_ua_slides.pdf

Toyota Unintended Acceleration Incident (cont.)

2013/Oct: Bookout/Schwarz Trial

- 2007 crash of a 2005 Toyota Camry
- Dr. Koopman & Mr. Barr testified as software experts
- Testified about defective safety architecture and software defects

Jury awarded $3 million compensation

Key technical element of criticism is the Electronic Throttle Control System (ECTS)

http://users.ece.cmu.edu/~koopman/pubs/koopman14_toyota_ua_slides.pdf

Electronic Throttle Control System (ETCS)

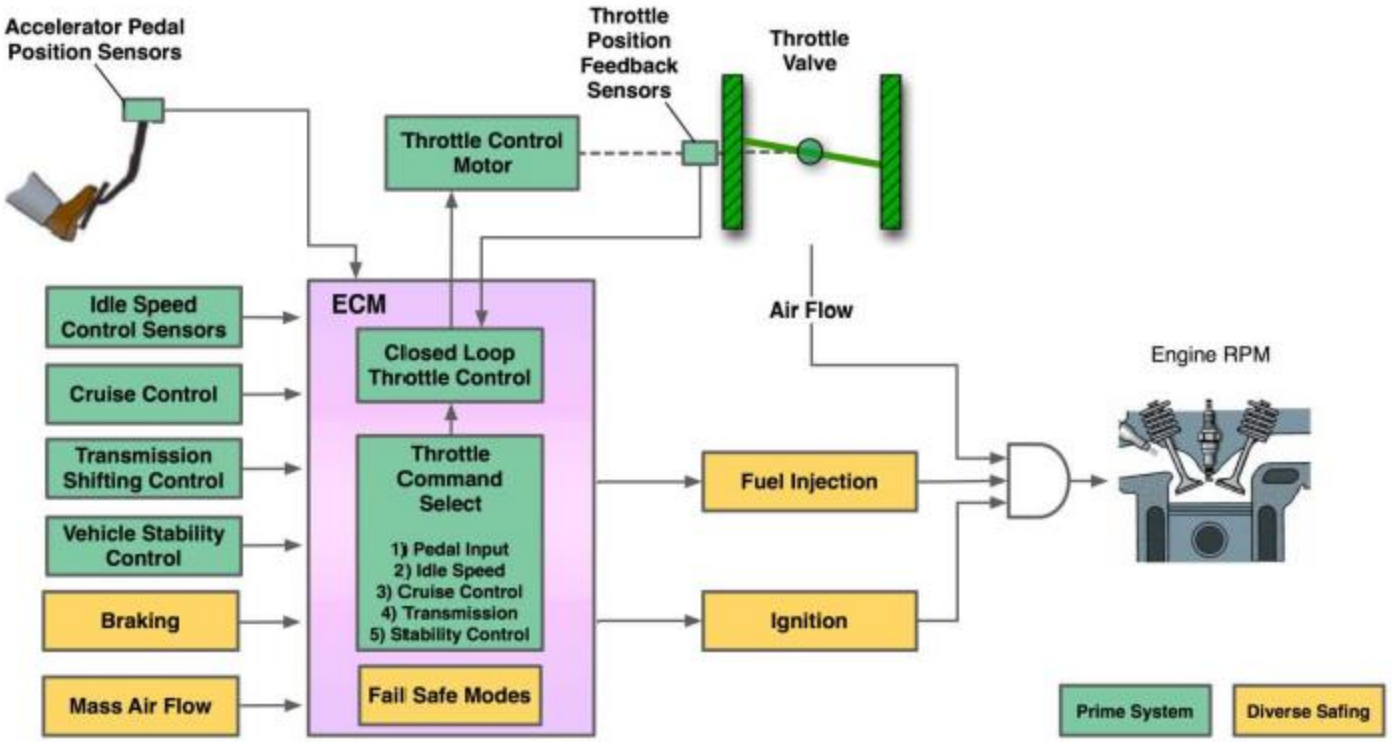


Figure 6.4-1. ETCS-i Major Functions

http://www.nhtsa.gov/staticfiles/nvs/pdf/NASA-UA_report.pdf

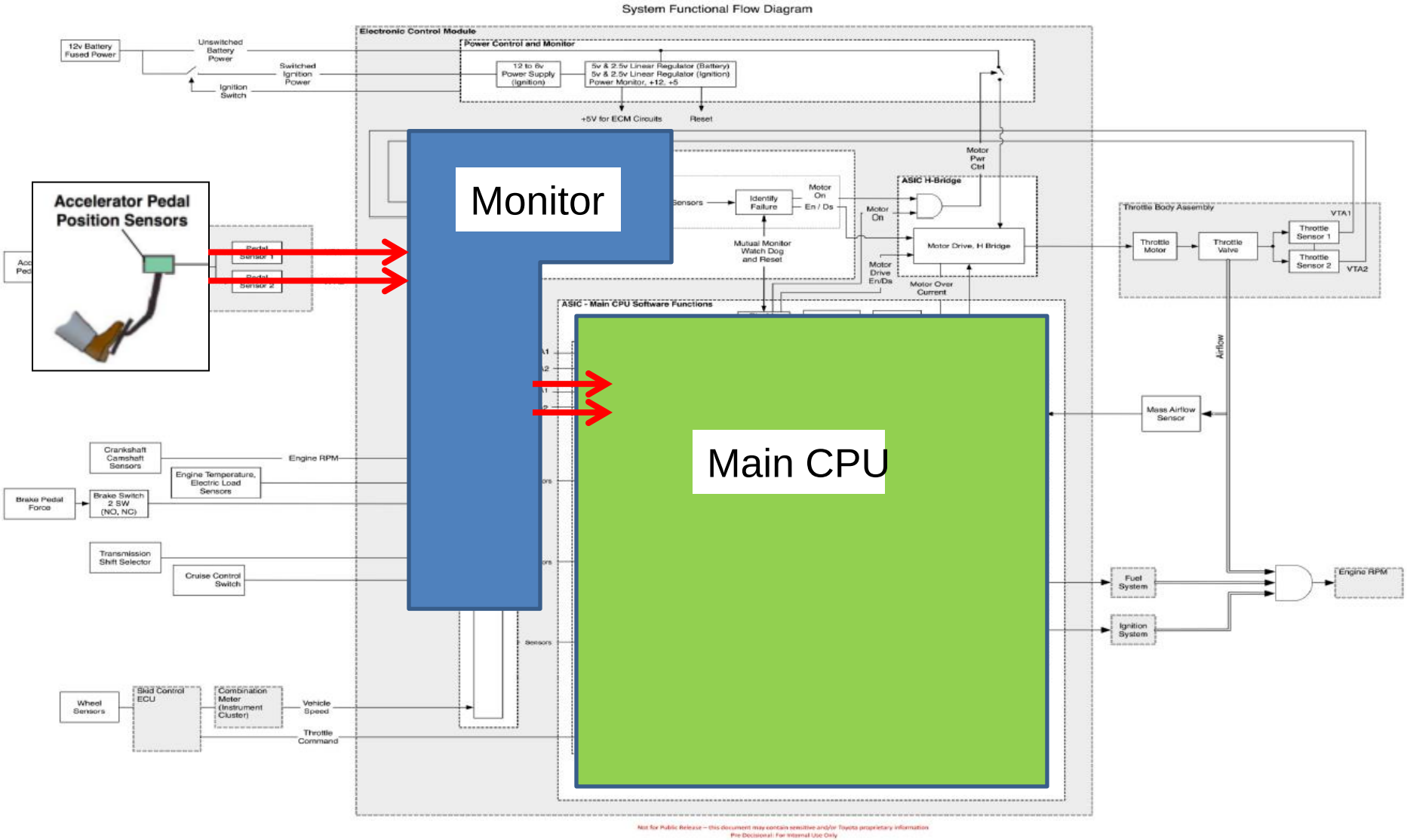


Figure 6.4.1-1. Overall System Functional Block Diagram

http://www.nhtsa.gov/staticfiles/nvs/pdf/NASA-UA_report.pdf

ETCS Criticism

Safety architecture

- Shortcomings in failsafes
- Shortcomings in the watchdog design
- Non-independent Fault-Containment Regions

Software Quality

- 256,600 Non-Commented Lines of C source
- 9,273 – 11,528 global variables (ideally 0 writable globals)
- Spagetti code, untestable functions according to McCabe cyclomatic complexity metric
- Use of recursion, no mitigation for stack overflow
- Concurrency issues

ETCS Criticism (cont)

Certification

- Critical SW is typically developed by following standardized processes, e.g., MISRA SW Guidlines
- Toyota does not claim to have followed MISRA
- Mike Barr's team found 80,000 violations of MISRA C

Reasons for low dependability

What would you think are reasons for low dependability?

Reasons for low dependability

- **Chips with everything:**
  Computers are increasingly used for all types of devices and services.
- **Interface design:**
  Complex systems must have a “friendly” interface that is easy to understand and must not confuse or mislead the user.
- **The “system” includes the operator:**
  The total system requires some functions to be carried out by the operator.
- **The “system” includes the documentation:**
  Operator failures may occur due to hard to understand or misleading documentation.
- **The “system” includes its operating procedures:**
  Just as the operator and the documentation are regarded as part of the system, so must the procedures for using it.

Reasons for low dependability (cont)

- **“System” failures are human failure:**
  Not only the operator, but other humans and ultimately the designer are causing system failures.
- **Complexity:**
  Problem inherent complexity—not solution induced complexity—is hard to handle.
- **System Structure:**
  Unsuitable system structures can lead to low dependability
- **Wrong assessment of peak load scenario:**
  Systems can only be designed to handle a priori known peak load scenarios.
- **Wrong assessment of fault hypothesis:**
  Systems can only be designed to handle a priori known fault hypothesis.

Reasons for low dependability (cont.)

- **Low dependability of components:**
  "A system is as strong as its weakest link"
- **Misunderstanding of application:**
  Customer and system manufacturer have different understandings of the services
- **Incomplete problem description:**
  Unintended system function due to incomplete problem description
- **Coupling and interactive complexity:**
  cf. next slide
- **Discontinuous behavior of computers:**
  cf. foil after slide
- **No system is fool-proof**

Concept of coupling and interactive complexity

The concept of coupling and interactive complexity is a model to explain what type of systems are potentially hazardous [Perrow 1984].

- **Tightly coupled systems:**
  In a tightly coupled system components affect one another automatically with great rapidity, so that errors propagate too quickly for a human operator to detect, contain and correct them.
- **Interactive complex systems:**
  In an interactive complex system components interact in many ways simultaneously, so that the behavior of the system (as a whole) is inherently difficult to understand.

Problem of discontinuous behavior or the Problem of Software

- discrete computers are symbol manipulating machines
- symbols are represented in binary form of 0´s and 1´s
- computers are finite state machines
- large state space (combinatorial explosion)
- mapping of actual state and input to new state
- in contrast to analogue systems there is no continuos trajectory
- discontinuous trajectories are intractable by simple mathematics
- is worse than chaotic behavior (of analog systems)
- continuous or analog systems have an infinite number of stable states while discrete systems have only a small (finite) number of stable states