Basic Components of a Spacecraft Computer

- System definition
- Computer architecture
- Components
- Data coding
- Fault tolerance and reliability
- Hardware and software testing

Hardware, Software, and Documentation

A Typical Space/Ground Information System

Copyright 2008 by Robert Stengel. All rights reserved. For educational use only.
http://www.princeton.edu/~stengel/MAE342.html
Defining the System

- Identify the spacecraft bus and payload operational modes
- Allocate top-level requirements for the computer system
- Define sub-system interfaces
- Specify baseline computer system
  - Define computer system’s operational modes and states
  - Functionally partition and allocate computational requirements to
    - spacecraft sub-systems, hardware, or software
    - ground station
  - Analyze data flow
  - Evaluate candidate architectures
  - Select basic architecture
  - Develop baseline system configuration
- Do we need a new computing system, or can we use an old system that is already certified?

Requirements Definition

- What must the system do?
- Why must it be done?
- How do we achieve the design goal?
- What are the alternatives?
- What sub-systems perform specified functions?
- Are all functions technically feasible?
- How can the system be tested to show that it satisfies requirements?

Computer System State Diagram

- System states must be consistent with allocated requirements and with spacecraft’s and ground station’s concepts of operation (“conops”)
Computer System Functional Partitioning

- Group functions based on
  - Similarity
  - Complexity
  - Processing type
  - Urgency
  - Timing and throughput
  - External interface
  - Data storage requirement
  - Need for human participation
  - Flight safety
- Space/ground tradeoffs
  - Autonomy
  - Time criticality
  - Downlink bandwidth required/available
  - Uplink bandwidth required/available
- Hardware/software tradeoffs
  - Special-purpose hardware
  - Algorithmic complexity

Computer Architecture

- Central processor
  - Point-to-point interfaces between central processor and devices
  - Dedicated wiring and software
- Bus
  - Processors and devices communicate via a bus
  - Protocol software for transmission control
  - Standard interfaces
- Ring
  - Established arbitration (e.g., token-passing) for bus control
- Instruction set
  - Assembly language
  - Higher-order language

Computer Resource Estimation

- Defining processing tasks
  - Software requirements specification
  - Interface requirements specification
  - Principal classes
    - Control systems
    - System management
    - Mission data management
    - Operating system
      - Utilities
      - Built-in test
- Estimating software size and throughput
  - Processor instruction sets
  - Processor clock speeds
  - Historical data for similar processing tasks
  - Preliminary coding of example tasks

Development Phase Issues

- Hardware selection
  - Performance, cost, availability, vendor competition
- Developmental environment
  - Software languages, tools for coding, compiling, and testing
  - Host/target machines
- Development costs
  - Mission life cycle
- Development tools and methodologies
  - Specification and analysis aids
  - Design aids
  - Traceability analysis
  - Documentation aids
Computer System Integration and Test

Apollo GNC Software Testing and Verification

- Major areas of testing
  - Computational accuracy
  - Proper logical sequences
- Testing program
  - Comprehensive test plans
  - Specific initial conditions and operating sequences
  - Performance of tests
  - Comparison with prior simulations, evaluation, and re-testing
- Levels of testing
  - 1: Specifications coded in higher-order language for non-flight hardware (e.g., PCs)
  - 2: Digital simulation of flight code
  - 3: Verification of complete programs or routines on laboratory flight hardware
  - 4: Verification of program compatibility in mission scenarios
  - 5: Repeat 3 and 4 with flight hardware to be used for actual mission
  - 6: Prediction of mission performance using non-flight computers and laboratory flight hardware

Apollo GNC Software Specification Control

- Guidance System Operations Plan (GSOP)
  - NASA-approved specifications document for mission software
  - Changes must be approved by NASA Software Control Board
- Change control procedures
  - Program Change Request (NASA) or Notice (MIT)
  - Anomaly reports
  - Program and operational notes
- Software control meetings
  - Biweekly internal meetings
  - Joint development plan meetings
  - First Article Configuration Inspection
  - Customer Acceptance Readiness Review
  - Flight Software Readiness Review

Apollo GNC Software Documentation and Mission Support

- Documentation generation and review
  - Functional description document: H/W-S/W interfaces, flowcharts of procedures
  - Computer listing of flight code
  - Independently generated program flowchart
  - Users’ Guide to AGC
- Mission support
  - Pre-flight briefings to the crew
  - Personnel in Mission Control and at MIT during mission
A Little AGC Digital Autopilot Code

Spacecraft Computers

• Spacecraft computing hardware; analogous to Macs and PCs, but
  – Must be ultra-reliable
  – A few generations behind the state-of-the-art
• Memory
• Input/output
• Fault tolerance
• Special-purpose peripherals

Memory

• Read-only memory (ROM)
  – Non-volatile
  – Non-alterable
  – Store critical programs
  – EAROM, EEROM, EEPROM
• Flash memory (special EEPROM)
• Random-access memory
  – Volatile
• Special-purpose memory
  – Multi-port
  – Cache
  – Multiply-accumulate
• Disk
  – Magnetic
  – CD, DVD

Input/Output

• Ports
  – Data transfer between processor and bus
    – Serial I/O ports
    – Parallel I/O ports
    – I/O-mapped ports
    – Memory-mapped ports
• Direct memory access
  – Sub-systems access memory without going through the processor for large blocks of data or high data rate
• Multi-port memory
  – Simultaneous data access by two or more devices
• Interrupts
  – May be generated by a timer or an event, changing processor function
  – Synchronize activity of multiple processors
  – Context switching and storage
• Timers
• Bus interface
Special-Purpose Peripherals (Signal-Processing Hardware)

• Data acquisition

![Diagram of data acquisition process]

Some Flight Computer Variations

Fault Tolerance Requirements

• Failure at a single point should not cause failure of entire system
• It should be possible to isolate the effects of a single component failure
• It should be possible to contain individual failures to prevent failure propagation
• Reversionary modes should be available (“fail-safe” design)
  – backup software
  – backup hardware
Fault Tolerance

- Radiation hardness
- Single-event upsets
- CMOS latch-up
- Parity
- Error detection and correction
- Triple modular redundancy
- Multiple execution
- Fault roll-back
  - repeat the function if error is sensed
- Fault roll-forward
  - correct the error and move on
- Watchdog timers
  - detect unusual execution time for program function
  - force a restart if fault is detected
- Improper sequence detection
- Hardware vs. software errors

Radiation Hardness and Single-Event Upsets

- Radiation degrades semiconductor devices
- Ionization due to Gamma rays may trap charges in devices, altering their function
  - Can produce a single-event upset
- Random and age-related failures must be anticipated
  - Shielding
  - Radiation-hardened dielectrics
- Single-event upset (SEU)
  - Radiation flips a bit in data or instruction
- CMOS latch-up
  - Large transient current flow may destroy the device
  - Build in a circuit breaker that shuts off current before damage is done

Parity and Error Detection

- \( n \)-bit word = \((n - 1) \) bits of data plus a parity bit (e.g., ASCII 8-bit word for 7-bit code)
- Parity bit is computed (XOR gates) so that the number of “ones” in the word is even (or odd)
- Word is transmitted
- Error in one bit of the word is detected if the number of “ones” is not even (or odd)

A wants to transmit: 1001
A computes parity bit value: \( 1 \oplus 0 \oplus 0 \oplus 1 = 0 \)
A adds parity bit and sends: 10010
B receives: 10010
B computes overall parity: \( 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 = 0 \)
B reports correct transmission after observing expected even result.

- If error is detected, B requests re-transmission from A
- Error-correcting codes as in telemetry (convolution and block codes, memory refreshing, redundancy)

Apollo Guidance Computer

- Parallel processor
- 16-bit word length (14 bits + sign + parity)
- Memory cycle time: 11.7 \( \mu \)sec
- Add time: 23.4 \( \mu \)sec
- Multiply time: 46.8 \( \mu \)sec
- Divide time: 81.9 \( \mu \)sec
- Memory (ceramic magnetic cores)
  - 36,864 words (ROM)
  - 2,048 words (RAM)
- 34 normal instructions
- Identical computers in CSM and LM
- Different software (with many identical subroutines)
  - 70 lb
  - 55 w

- There were NO computer hardware failures during Apollo flights
Astronaut Interface With the AGC

- Computer Display Unit or Display/Keyboard

Sentence
- Subject and predicate
- Subject is implied
  - Astronaut, or
  - GNC system
- Sentence describes action to be taken employing or involving the object

Predicate
- Verb = Action
- Noun = Variable or Program

See http://apollo.spaceborn.dk/kskv-sim.html
And http://www.biblio.org/apollo/ for simulation

Verbs and Nouns in Apollo Guidance Computer Program

Verbs (Actions)
- Display
- Enter
- Monitor
- Write
- Terminate
- Start
- Change
- Align
- Lock
- Set
- Return
- Test
- Calculate
- Update

Selected Nouns (Variables)
- Checklist
- Self-test ON/OFF
- Star number
- Failure register code
- Event time
- Inertial velocity
- Altitude
- Latitude
- Miss distance
- Delta time of burn
- Velocity to be gained

Selected Programs (CM)
- AGC Idling
- Gyro Compassing
- LET Abort
- Landmark Tracking
- Ground Track Determination
- Return to Earth
- SPS Minimum Impulse
- CSM/IMU Align
- Final Phase
- First Abort Burn

Triple Modular Redundancy: Software

- Software implementation for serial data transmission
  - Each word is transmitted three times
  - Voting logic compares the three versions and chooses the version transmitted by two (or all three)
  - Serial data transfer rate is slowed by a factor of three

See http://apollo.spaceborn.dk/kskv-sim.html
And http://www.biblio.org/apollo/ for simulation

Triple Modular Redundancy: Hardware

- Parallel hardware implementation for fault tolerance
  - Each sensor, computer, or actuator is replicated three times
  - Multiple execution
  - Voting logic compares the three versions of each output and chooses the version
    - transmitted by two (or all three),
    - middle value, or
    - average value
  - Cost and maintenance implications

See http://apollo.spaceborn.dk/kskv-sim.html
And http://www.biblio.org/apollo/ for simulation
Reliability

- Probability of Success during Period of Operation

\[ R(t) : \text{Probability of success} \]
\[ P(t) : \text{Probability of failure} \]

\[ R(t) = 1 - P(t) \]

Reliability of a Single String

- Reliability of a string of components = product of individual reliabilities

\[ R_{1-n}(t) = R_1 R_2 \ldots R_n \]

Reliability Assessment

- Tools for reliability assessment: Testing
  - Levels of test: development, qualification, acceptance, function
  - Destructive physical analysis

- Tools for reliability assessment: Analysis
  - Statistical distributions
  - Statistics, regression, and inference
  - Fault trees and reliability prediction
  - Confidence level or interval

Reliability of Parallel (Redundant) Components

- Probability of failure of parallel components = product of individual probabilities

\[ P_{13}(t) = P_1(t)P_2(t)P_3(t) \]

\[ R(t) = 1 - P(t) \]

\[ R_{13}(t) = 1 - P_{13}(t) = 1 - P_1(t)P_2(t)P_3(t) \]
Reliability of a Switched Redundant System

- Reliability of the switch must be considered

$$R_{system}(t) = R_1(t) \left[ 1 - \left[ 1 - R_2(t) \right] \left[ 1 - R_3(t) R_2(t) \right] \right] R_3(t)$$

$$= R_1(t) \left[ 1 - P_2(t) P_{S2}(t) \right] R_3(t)$$

Reliability of Parallel Strings

$$P_{1n}(t) = P_1(t) P_2(t) K \ P_n(t)$$

$$R(t) = 1 - P(t)$$

$$R_{1n}(t) = 1 - P_{1n}(t) = 1 - P_1(t) P_2(t) K \ P_n(t)$$

Reliability of a String of Parallel Components

$$R_{system}(t) = \sum_{x=r}^{n} \binom{n}{x} R^x (1 - R)^{n-x}$$

- Binomial coefficient

$$\binom{n}{x} = \frac{n!}{x! (n-x)!}$$

Space Shuttle Quintuply Redundant Flight Control Computers

- Five identical IBM AP-101 computers
  - Magnetic core memory later upgraded to semiconductor memory
  - Primary system: 4 parallel computers with identical coding and complex redundancy management software
  - Backup system: 5th computer with independent coding of the same functions
  - Concern for generic software failures
  - HAL/S programming language
Space Shuttle Quintuply Redundant Flight Control Computers

Space Shuttle Avionics Integration Lab

IBM AP-101 Input/Output Processor and Central Processing Unit