A UAV Test and Development Environment Based on Dynamic System Reconfiguration

Osamah A. Rawashdeh, Garrett D. Chandler, and James E. Lumpp, Jr.

Department of Electrical and Computer Engineering
University of Kentucky
Lexington, KY
Outline

• Motivation/Background

• Design Framework

• Runtime Behavior

• UAV Test and Development Environment
UAV Research at UK

• BIG BLUE: Baseline Inflatable-wing Glider, Balloon- Launched Unmanned Experiment.
• Ongoing project at UK to developing a test bed for Mars airplane technology.
• BIG BLUE is funded by NASA and KSGC
• ~ 40 students involved per year.
Framework

• Software is developed in a modular fashion.

• *Software modules* can have several implementations with different resource requirements and output qualities.

• Dependencies among modules are captured in *dependency graphs* (DGs).

• Modules are scheduled on an interconnected set of processing resources.
Framework (cont.)

• Fault detection:
  – By application code
  – Heartbeat messages
  – OS detected violations

• A *system manager* tracks status of hardware and software resources.

• Fault handling: system is dynamically reconfigured by deploying a new mapping of software modules to hardware resources.
System Architecture

– System Manager
  • Tracks status of resources
  • Finds and deploys configurations

– Processing Elements
  • Host I/O hardware
  • Real-time OS schedules modules

– Communication Bus
  • CAN 2.0 standard
  • Control messages
  • Data transfer

– Sensors and Actuators
Dependency Graphs

- DGs show the flow of information from sensors to actuators.

- DG nodes:
  - Software modules
    Executable code schedulable on a processing element.
  - Data variables
    Inputs and outputs of software modules.
    State variables are local to a software module.
  - I/O devices
    Interface to the environment.
Data Requirements

- **Dependency symbols:**
  - “k-out-of-n” gates: \( n > 0, 0 \leq k \leq n \).
  - “XOR”: only one input required.
  - “DEMUX”: for fanning out.
  - “AND”: all input required.

- Quality values are associated with variables.
Example Graph

- GPS
  - GPS driver
  - GPS altitude
  - High quality altitude estimator
  - Altitude (Quality1)
  - Release cmd

- Abs. pressure sensor
  - Sensor driver
  - Absolute pressure
  - Low quality altitude estimator
  - Altitude (Quality2)
  - 0/2

- Chute logic
- Chute release
- Chute driver
- Chute Release Solenoid
Runtime Behavior

- **Local management tasks:**
  - Scheduler
  - Network Interface
  - Module Loader

- **Module I/O data passed through mailboxes.**

- **Data routing is transparent to Modules.**
Current Research

• Expand bus via wireless link to the ground:
  – Rapid prototyping
  – Minimize risk to hardware
  – Flexible Reconfiguration

• Applying the framework to the design of BIG BLUE IV