Challenges in the Development of Virtual Controls for Class 1E Safety Applications

Presented by: Jeff Engelberger
Business Development Manager
DRS-Consolidated Controls, Inc.

ANS - Connecticut Section
May 10, 2012

Release Statement: Information included herein has been determined not to contain technical data and is categorized as EAR99 under the Export Administration Regulations (EAR). This information may be released to non-sanctioned parties/end-users without an export license provided that no technical elaboration is discussed or disclosed.
Introduction – Discrete Controls

• Traditional nuclear power plants use discrete controls in the main control room
  – Pushbuttons
  – Indicators
  – Meters
  – Manual / Auto (M/A) Stations, etc.

• Use of these devices results in:
  – Significant amount of control panel space
  – Significant amount of associated wiring
  – Fixed control room configuration
Introduction – Virtual Controls

• An alternative to discrete controls are computer based virtual controls

• Virtual controls consist of Graphical User Interfaces (GUIs) on touch screen video display units (VDU)

• Advantages include:
  – Reduction in panel wiring
  – Reduction in control panel space
  – Increased understanding of plant configuration and status
  – Increased information available to the operator at a centralized location
  – Reconfigurable to match the context of the current control application
Introduction – Virtual Controls

- Sample Process Control Screen
Introduction – Virtual Controls

- Sample Diagnostic Screen

![Sample Diagnostic Screen](image-url)
Challenges – Virtual Controls

• Several challenges are encountered when developing 1E Safety Related virtual controls for a nuclear power plant

• Most commercially available software does not meet the NRC or IEEE guidelines for safety critical software
  – How do you Verify and Validate commercial software?
  – What design processes were used during the development?
  – Microsoft Windows®?

• How do you manage customer expectations?
  – “It looks like Windows® but doesn’t act like Windows®!”
Key Software Challenges

• Several key software areas need to be addressed when developing the virtual controls
  – Basic Input Output System (BIOS)
  – Operating System (OS)
  – Graphics Libraries
  – Touch Screen Drivers
BIOS – Challenges

• The BIOS provides access to all basic hardware devices on the host CPU
• Commercial vendors of BIOS software are very protective of their intellectual property
• This makes verification nearly impossible to perform
• Additionally, since the BIOS provides for access to all possible hardware devices on the CPU, unused code must be removed
  – Several BIOS vendors contacted refused to even consider removal of BIOS components
BIOS – Solutions

• Two options to consider:
  – Purchase / license commercially available BIOS source code
    • Remove unnecessary hardware device code
    • Document the BIOS code
    • Perform Verification and Validation on the BIOS code
  – Custom design BIOS for host CPU providing only the required functionality
    • Write required hardware device code
    • Document the custom BIOS code
    • Perform Verification and Validation on the BIOS code

• The chosen alternative was to develop our own BIOS
• Development of a custom BIOS allows for complete design control using a design process compliant with all of the applicable specifications
  – 10 CFR50 Appendix B
  – ASME NQA-1, Part II Subpart 2.7
  – IEEE 7-4.3.2, IEEE 1012, IEEE 1028,
  – US NRC Regulatory Guide 1.152, USNRC RG 1.168
• Drivers for hardware devices not required in the final product are NOT included in the BIOS
• Comprehensive Verification and Validation was performed as all software was developed in house
Operating System – Challenges

• The Operating System (OS) is the software platform upon which ALL applications run

• Two approaches to investigate:
  – Purchase / license commercially available OS source code
    • Remove unnecessary OS code
    • Document the OS code
    • Perform Verification and Validation on the OS code
  – Custom design OS for host CPU providing only the required functionality
    • Write required OS code
    • Document the custom OS code
    • Perform Verification and Validation on the OS code
Operating System – Challenges

- There are no commercially available operating systems that meet the NRC requirements for safety applications
- OS’s like Windows®, Linux® or VxWorks® are not viable options
- Even if source code were available, the sheer size of the source code makes it virtually impossible to adequately perform verification and validation
- All code that is not required to perform safety functions would need to be removed
  - If the BIOS developers could not do it, the OS developers would have a much more daunting task
- Additionally, commercial operating systems typically utilize coding techniques that are difficult if not impossible to verify adequately
  - Interrupts, multi-tasking, recursion are just a few
After considering the challenges and available options, the chosen alternative was to develop our own Operating System.

Simple Operating Loop structure was used:
- Ensures a deterministic (worst case) response time to all functions
- Multi-tasking and recursion were not employed in the code
- Limited the use of interrupts to communications, Only interrupts from the touch screen controller are serviced
- Loosely coupled modular design
- No unused code, only the required functionality is included

Comprehensive Verification and Validation was performed as all software was developed in house.
Graphics Libraries – Challenges

• Graphics libraries provide the basic components used to draw the text and symbols required to implement virtual control application screens

• These include:
  – Lines
  – Circles
  – Boxes
  – Fills
  – Text
  – Etc.
Graphics Libraries – Challenges

• Two approaches to consider:
  – Purchase / license commercially available Graphics Library source code
    • Remove unnecessary Graphics code
    • Document the Graphics code as required
    • Perform Verification and Validation on the Graphics code
  – Custom design Graphics Library providing only the required functionality
    • Write required Graphics code
    • Document the custom Graphics code
    • Perform Verification and Validation on the Graphics code
  – Custom design of robust and efficient graphics libraries is extremely time consuming and costly
Graphics Libraries – Solutions

- A well written, well documented commercial library had been identified and used on other critical DRS projects
- The decision was made to purchase an additional license for this software and modify the library as required for the Class 1E safety related application
- The Graphics library software was provided to test engineering for review for compliance with all applicable NRC and IEEE specifications
  - Overly complex code was modified to simplify the design
  - Unused code was removed
  - Changes were made to defective software
  - Any changes were provided back to the manufacturer for inclusion in future revisions
Touch Screen Drivers – Challenges

• Touch screen drivers are required to interface the specific touch screen hardware to the 1E custom OS
• Most touch screen interface manufacturers will not provide device driver source
• Touch screen interface manufacturers typically do not provide a generic touch screen driver that can be ported to a custom OS
• Since there is no commercially available touch screen interface driver software available, a custom developed solution is required
Touch Screen Drivers – Solutions

• The custom device driver software provides data buffering in accordance with IEEE 7-4.3.2
• This ensures that any hardware failure of the touch screen will not inhibit the system from performing its safety functions
• The touch screen interface is RS232 serial having a single point of entry into the safety application
  – With this single point of entry, it is possible to limit the impact of a failure of the touch screen driver
  – Data buffering and data validation have been implemented
• Software was designed to require multiple touches in differing locations to initiate an action
• This “command and confirm” sequence ensures that no action is initiated due to operator error or hardware malfunction
Video Display Unit (VDU)

- Currently installed in a plant in Asia undergoing construction and start-up testing
- Safety related qualified flat panel display
  - V&V’d software
  - Seismic, Environmental, EMI qualified
- Touch screen interface
  - No keyboard or mouse
- Communications interface with the PLμS 32™ 1E safety control system
- Soft control interface to I&C systems
- System diagnostics
Video Display Unit (VDU)

- The VDU includes a Display Controller and a Flat Panel Display Assembly
  - Display Controller – Intel Pentium based Single Board Computer with redundant communications interfaces in a Compact-PCI chassis
  - Flat Panel Touch Screen Display Assembly
- The VDU software provides soft controls of the equipment via a flat panel display / touch-screen
- The VDU provides the interface between a human and the PLµS 32™ Control System
- The VDU incorporates human factors requirements based on NUREG 0700
Conclusion

Through the use of several methodologies:

– Robust development processes
– Custom software
– Customized off-the-shelf source code
– Buffering software
– Common sense (KISS)

A VDU / touch screen based virtual control system can be developed and approved for use in 1E safety rated applications
References

- ASME NQA-1, Part II, Subpart 2.7 Quality Assurance Requirements for Computer Software for Nuclear Facility Applications.
- ANSI/IEEE 1028 Software Reviews and Audits
Contact Information

Thank You for Your Attention!

Credit to the authors:
Paul Stankiewicz, Principal Engineer
PStankiewicz@drs-ds.com
David Kulp, Program Manager
DKulp@drs-ds.com
Joseph Fowler, Systems Engineering Manager
JFowler@drs-ds.com

Questions?