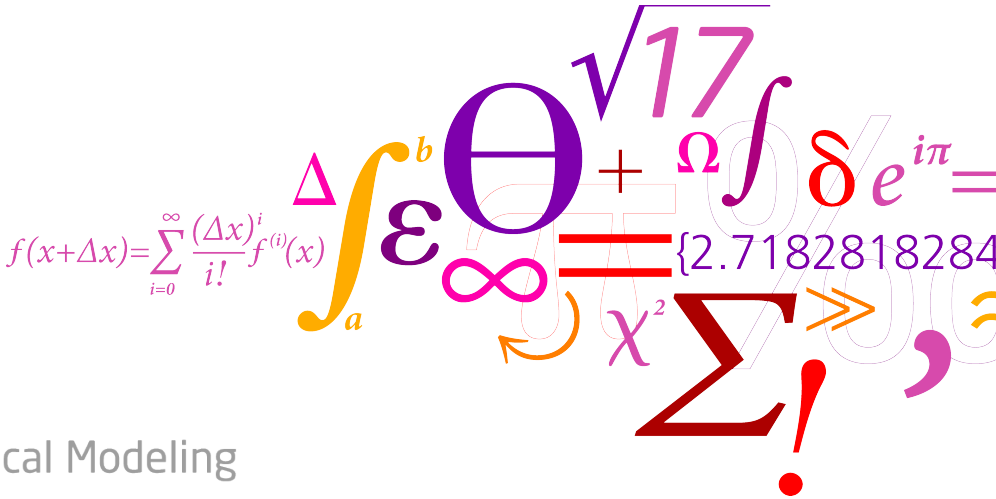
A Desktop 3D Printer in Safety-Critical Java

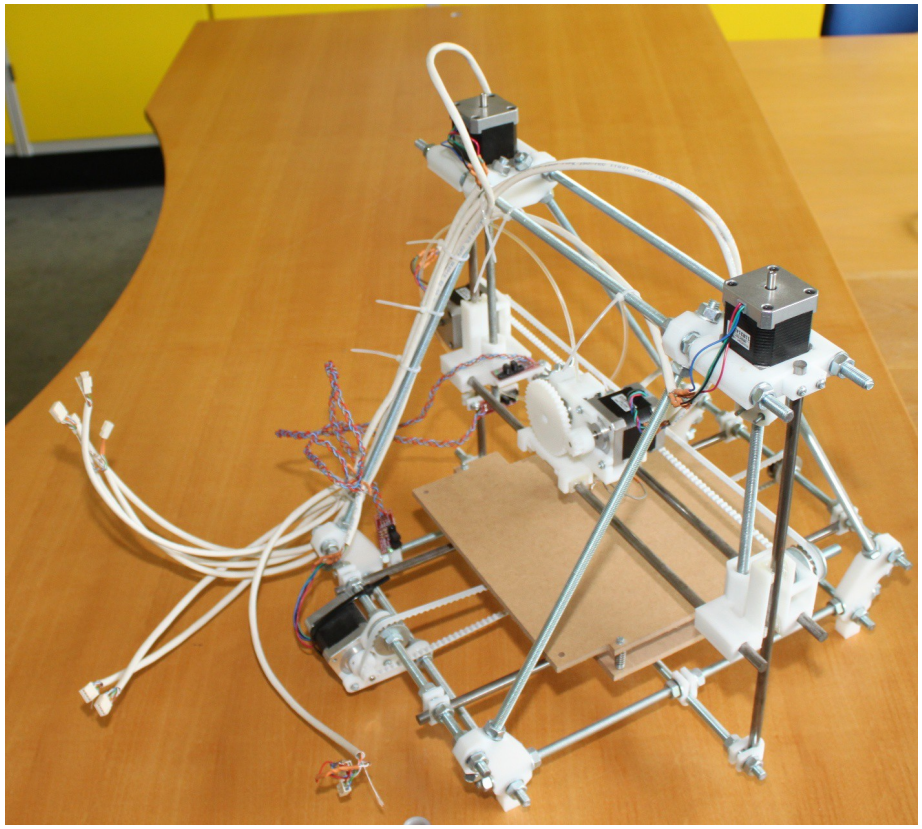
Tórur Biskopstø Strøm
Martin Schoeberl



DTU Informatics
Department of Informatics and Mathematical Modeling

RepRap

- 3D plastic printer
- Melts and extrudes plastic whilst moving in 3 dimensions
- Movement and extrusion is done according to G-codes
- G-codes are simple instructions generated from 3D drawings
- G-codes are often sent by a host computer to a controller



Project Goals

- Use a RepRap 3D desktop printer as a safety-critical use-case
- Create a SCJ level 1 implementation of the printer on top of JOP
- Evaluate the SCJ specification based on the use-case

RepRap as a Use-Case

- Not a real safety-critical system, however it is still useful
- Real-time requirements:
  - Read temperature
  - Maintain temperature
  - Move stepper motors at fixed speeds
  - Read end-stops
- Too high temperature can destroy hardware

Safety-Critical Java (SCJ)

- Specification based on the Real-Time Specification for Java (RTSJ)
- Subset of Java (and RTSJ)
- Aims to bring Java to safety-critical systems that need certification
- A notable difference with Java is the absence of a garbage collector:
  - Objects created in scopes
  - When a scope is left all objects created within are deallocated
- PeriodicEventHandlers (PEH) are periodic, Thread-like components
- Level 1 allows parallel PEHs

Safety-Critical Java (SCJ)

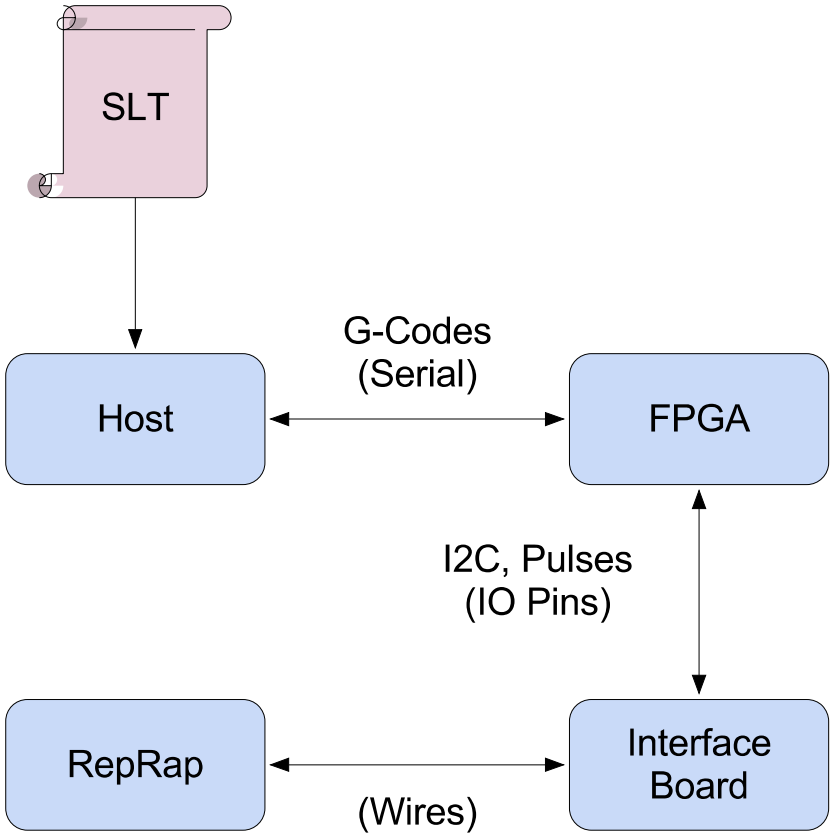
- A lack of safety-critical use-cases implemented in SCJ means a lack of evaluations:
  - Is SCJ useful for safety-critical applications?
  - Is the SCJ specification complete?
  - Is SCJ accessible for Java, and other, programmers?

Java Optimized Processor (JOP)

- Hardware implementation of the Java Virtual Machine
- Time-predictable
- VHDL source files allow porting to different FPGAs
- Ports and hardware on the specific FPGA is added using SIMPCON and accessed using ”hardware” objects

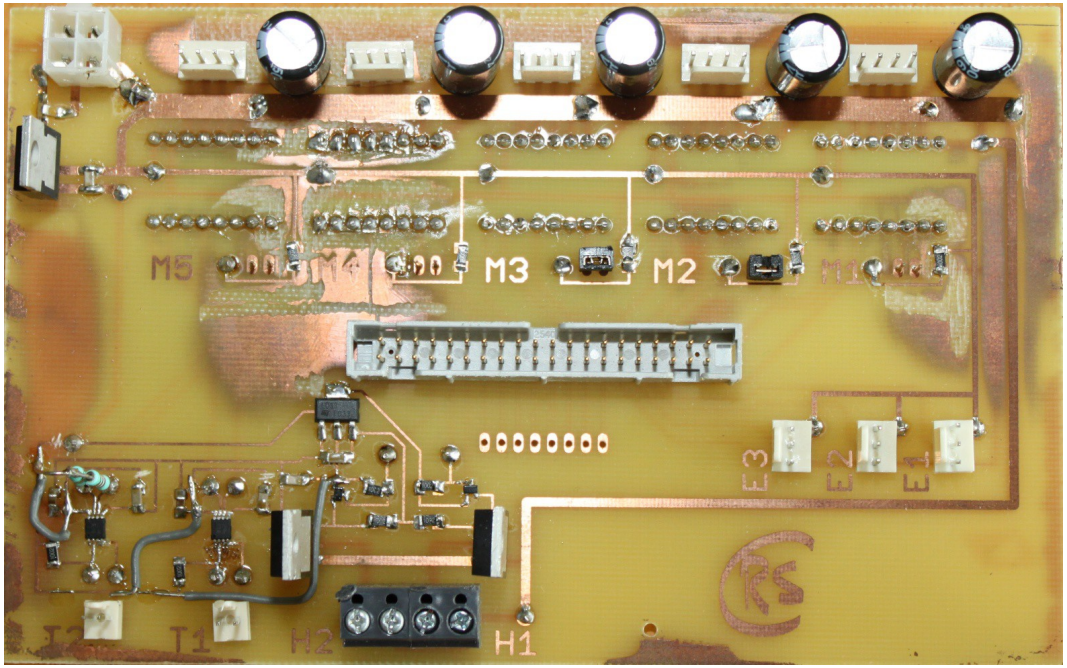
Implementation – Hardware Overview

- 2 hardware objects
- Serial data guaranteed to be processed with 115200 baud rate

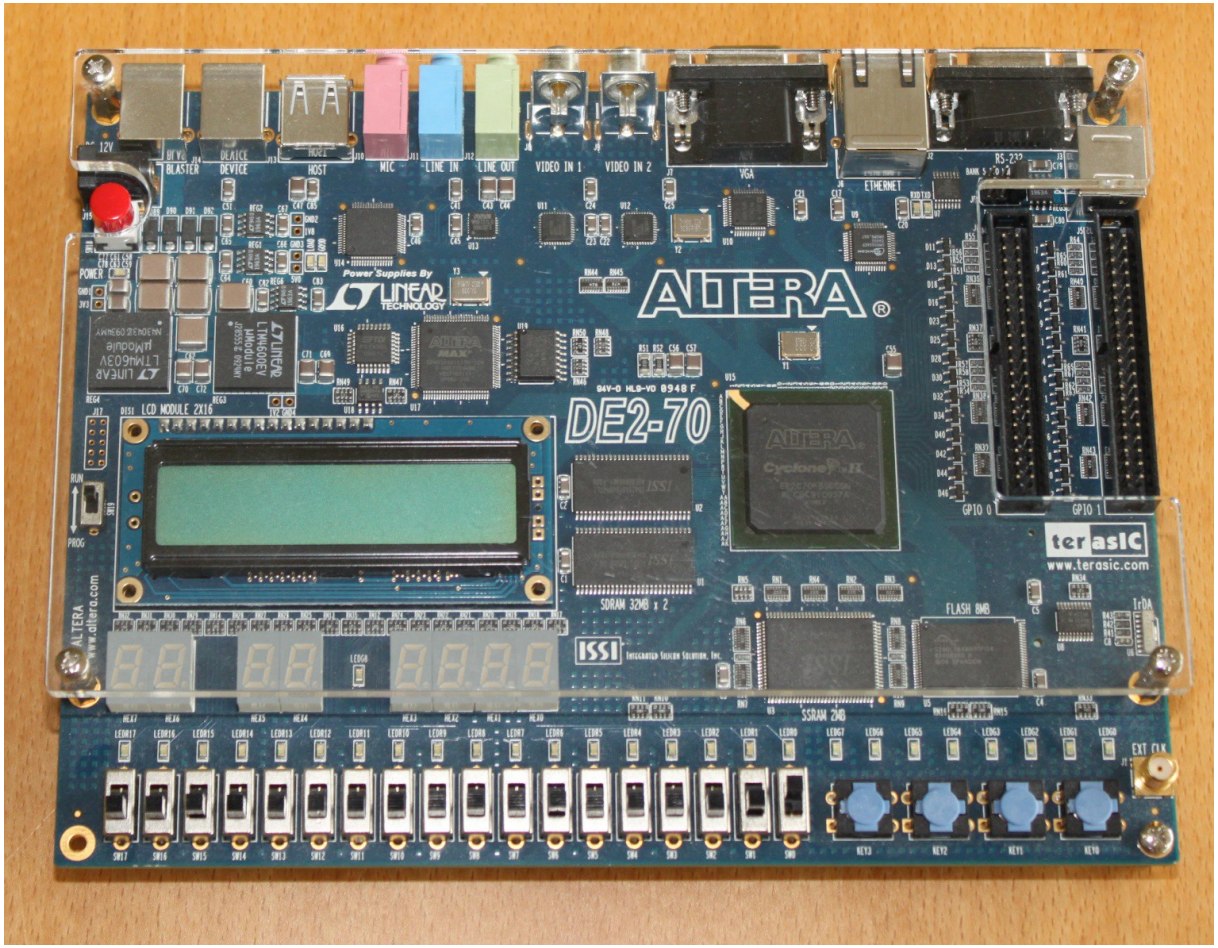


Implementation – Interface Board

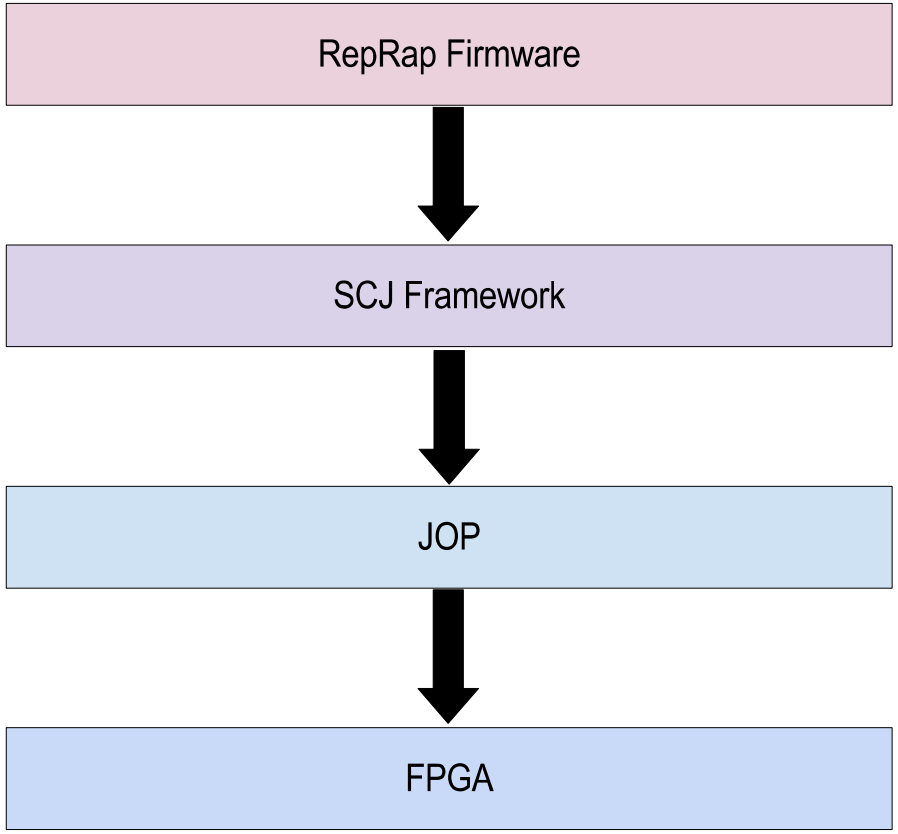
- Voltage level shifters
- Motor drivers
- Heater drivers



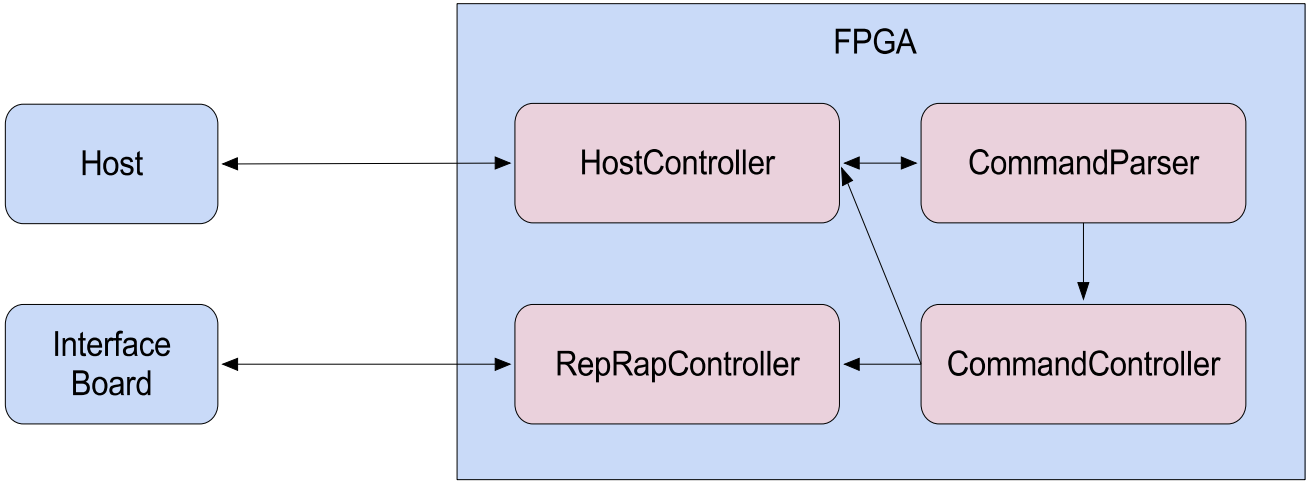
Implementation - FPGA



Implementation – Controller Layers



Implementation – PeriodicEventHandlers



Evaluation

- PEHs are similar to Java Threads:
  - Functionality distribution is similar
  - Automatically scheduled
  - Objects created during execution are automatically deallocated
- The scope size of a PEH is specified when creating it:
  - Object size is platform dependent
  - Tool to calculate the maximum potential size of a PEH is desirable – SizeEstimator is cumbersome

Evaluation

- If representing a schedulable task-set, PEHs are guaranteed to execute in the specified period
- Application and platform must be WCET analysable to guarantee schedulability
- WCET analysis performed on application showed following results:
  - Unbounded loops cannot be used
  - Busy blocks cannot be used, e.g. when reading input
  - Library code must be modified to support this, e.g. String.substring
- Schedulable task-set was possible by avoiding most library code
- Does not include task switching time

Evaluation

- Some programming difficulties arise in the absence of garbage collection:
  - Objects created in a scope cannot be referenced except in the current scope or nested scopes
  - Result generated in one PEH must be stored in higher scoped, shared objects to be accessible in another PEH
  - Very different from normal Java where Objects are freely referenced
  - This changes behaviour of library code, e.g. StringBuffer.toString

Evaluation

	SCJ firmware	Teacup
Firmware size (KB)	79	~32
Maximum steps per second	500 @ 60 MHz	17570 @ 20 MHz

- SCJ firmware size not optimized
- SCJ Maximum steps obtained from WCET analysis – “best” worst-case performance
- Performance not directly linked to SCJ specification – platform dependant
- However might still indicate that SCJ is not optimal for low-level tasks such as pulsing stepper motors

Conclusion

- PeriodicEventHandlers are similar to Threads
- Absence of garbage collection noticeably changes programming style
- Maximum PeriodicEventHandler memory consumption must be analysable
- Application and platform must be WCET analysable, which also alters programming style
- Slow stepping might indicate that safety-critical Java is not useful for low-level hardware controlling
- Possible to implement a RepRap as a level 1 SCJ application

Demonstration and Questions

DTU Informatics
Department of Informatics and Mathematical Modeling