ComputerScience

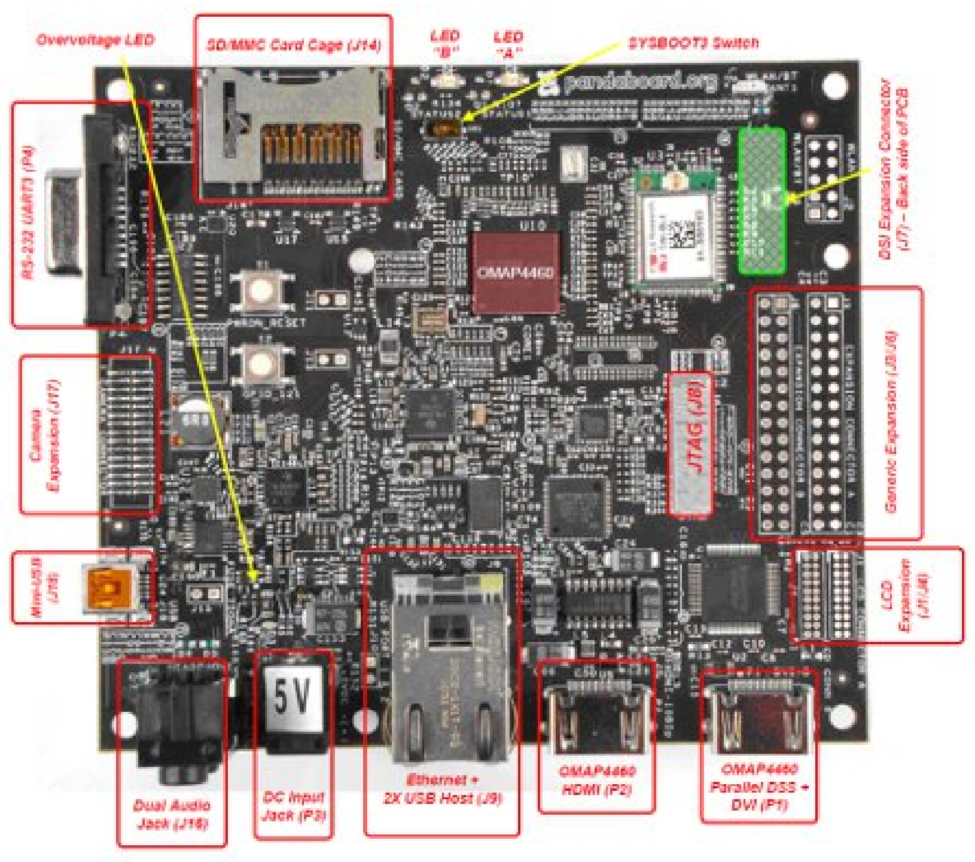
Boston University

Applying Language-based Static Verification in an ARM Operating System

Matthew Danish   Hongwei Xi   Richard West

Platform

- PandaBoard-ES OMAP4460



- Hobbyist platform based on TI OMAP4 system-on-chip, Cortex-A9 superscalar, dual-core ARM processor
- Relatively cheap, easy to obtain

ATS

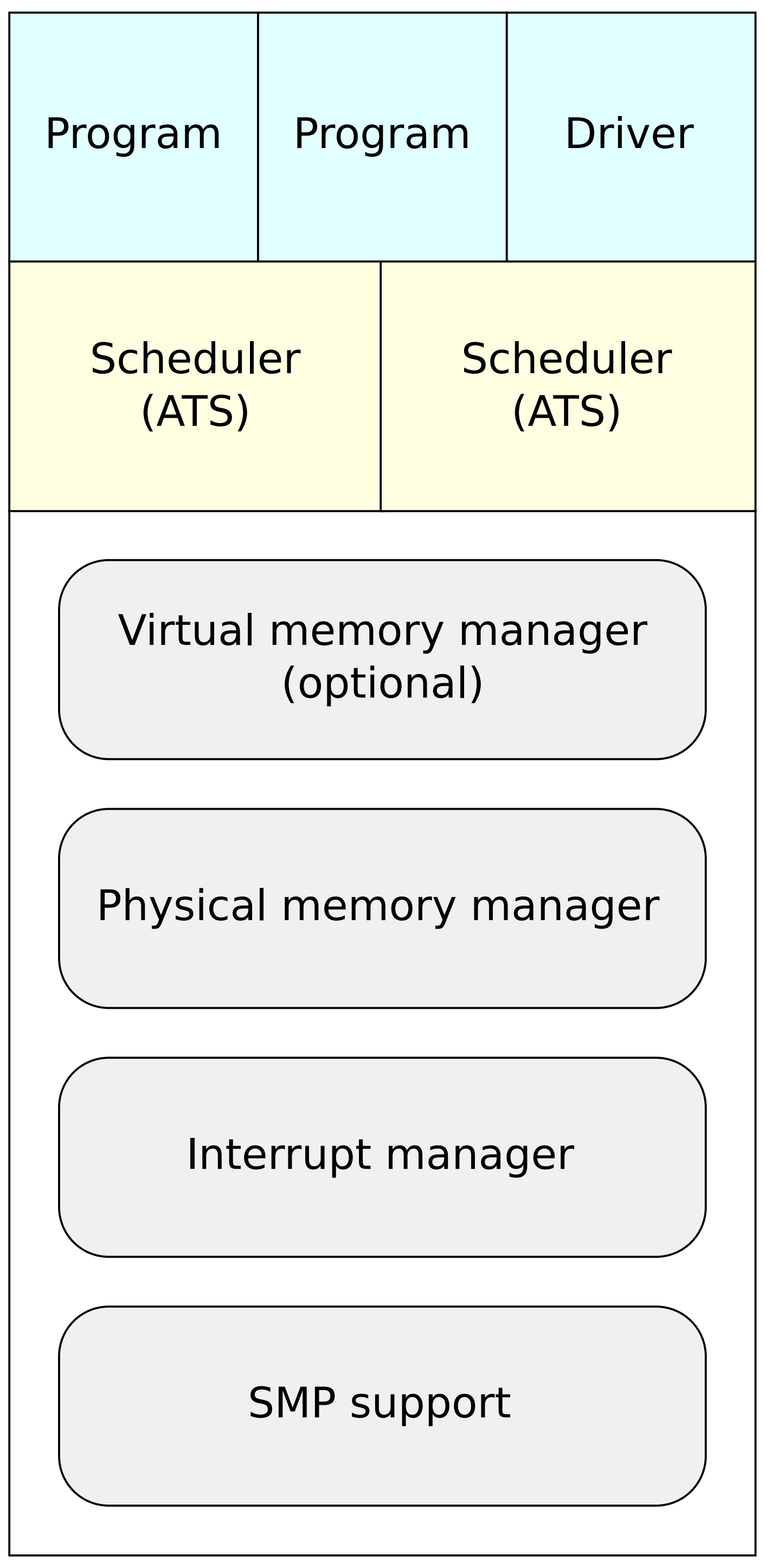
- Functional programming language, advanced type system
- Promotes “programming with theorem-proving”
- Supports pointer arithmetic and manual memory management
- Easy, natural integration with C
- Uses C-based low-level data representation
- Well-suited for low-level, systems programming
- `http://www.ats-lang.org/`

Functional programming

- High-level language
- Computation is the evaluation of mathematical functions
- Avoids state and mutable data
- Promotes safer, more compose-able software
- Encourages mathematical reasoning about programs: *referential transparency*

Terrier OS

- A simple, compact operating system for the PandaBoard
- Approximately 7000 lines of C and 500 lines of ATS
- Supports virtual memory, basic I/O, and multiple cores
- Aimed towards micro-kernel style, but also supports single-address space
- `http://github.com/mrd/terrier`


Program
Program
Driver
Scheduler (ATS)
Scheduler (ATS)
Virtual memory manager (optional)
Physical memory manager
Interrupt manager
SMP support

Basic Terrier OS Structure

Proofs and types

- Motivation: static verification for safety and reliability
- ATS has advanced features beyond functional programming
- Dependent types can specify logical propositions which are machine-checked
- Linear types provide a logic of reasoning about resource management
- The linear types are resources themselves that must be acquired and released to satisfy the type checker

Case study: RMS

- Rate-monotonic static priority
- Scheduler must track CPU usage of each process and set timer
- Budgets refreshed periodically
- Unoptimized `schedule` runtime only a few thousand cycles
- Example with dependent type representing future point in time:

```
val [now:int] now:  tick now = timer_32k_value()
...
val future:  [t:int | t > now] tick t
```

- “exists tick t such that $t > now$”
- Example with linear type which ensures timer is always set before scheduler may return:

```
fun pvttimer_set
(proof:  must_set_timer_v | t:  span):  void
```

- Scheduler must call this function to satisfy proof obligation
- Goal: verify RMS properties in high-level language with minimal run-time overhead

Future plans

- Rewrite more components of OS into ATS
- Refine and enhance types to statically verify useful properties
- Support more hardware and provide better software environment

Related work

- SPIN: Modula-3 type safety
- Singularity: MSIL type safety and Sing# invariants for IPC, single process space
- House/Famke: Haskell/Clean run-time environments ported to bare metal
- seL4: Haskell prototype translated to C with machine-checked refinement proof
- Cyclone: fat pointers, polymorphism, safer “C”
- CCured: checks pointer safety, inserts runtime checks if needed

E-mail addresses: `{md, hwxi, richwest}@cs.bu.edu`   **Website:** `http://cs-people.bu.edu/md`