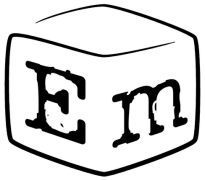


OS:DAY

ОС Embox: решение для встроенных систем

OS Day, 11 декабря 2014

Предпосылки

- Отказоустойчивый вычислитель
  - Статически задаваемые требования
- Разработка аппаратуры в ПЛИС
  - Низкие аппаратные требования
- Студенческая и научная деятельность
  - Открытый проект

Развитие проекта

- Робототехника
  - Lego Mindstorms
- Заказ на ОС РВ
  - x86, ARM, MIPS, Microblaze, PPC, SPARC
  - FAT, ext2/3/4, jffs2, nfs, cifs, ...
  - http, ftp, ssh, bootp, ntp, ...
  - Qt, java, boost, STL, ...

Сферы применения

- Системы с заранее заданными функциональными требованиями
- Системы с высокими требованиями по безопасности и надежности
- АСУ-ТП
- Телекоммуникационное оборудование
- Специализированная техника
- Встроенные системы

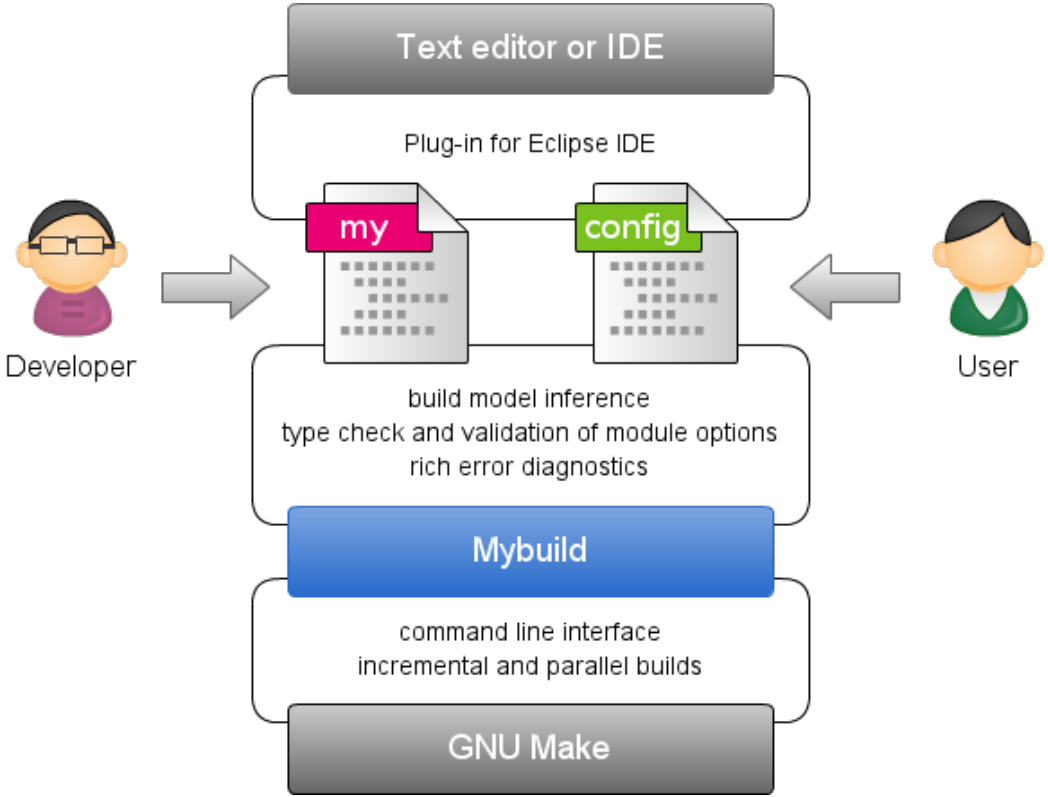
Свойства

- Удобная конфигурируемость
- Отсутствие избыточности кода
- Низкие аппаратные требования
- Кроссплатформенность
- Развитый сетевой стек
- POSIX-совместимость
- Простота отладки

Конфигурируемость

- Собственный DSL язык описания модулей
- Автоматическое разрешение зависимостей
- Статическая сборка образа
- Возможность задания параметров модулей и образа в целом

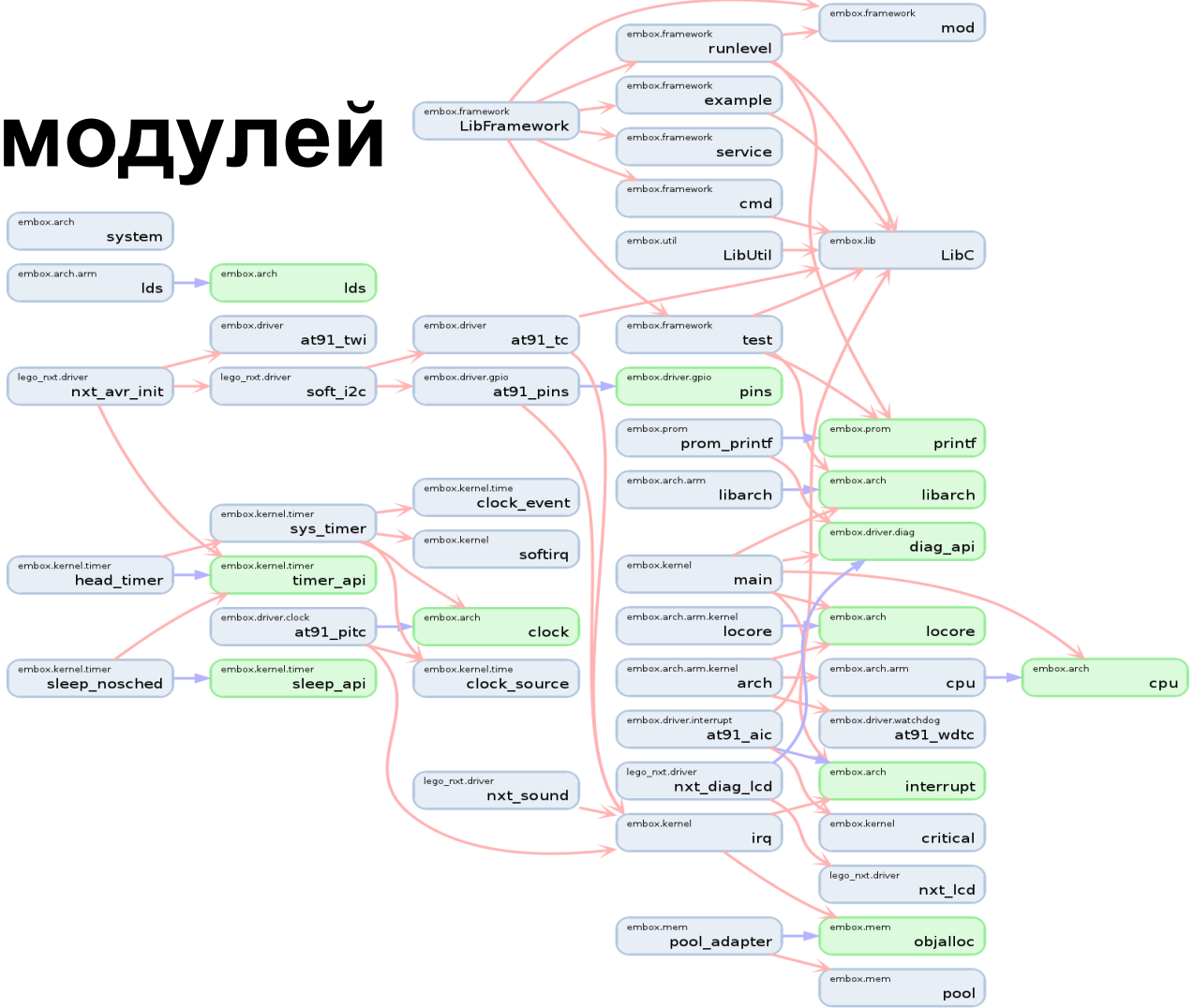
Процесс сборки



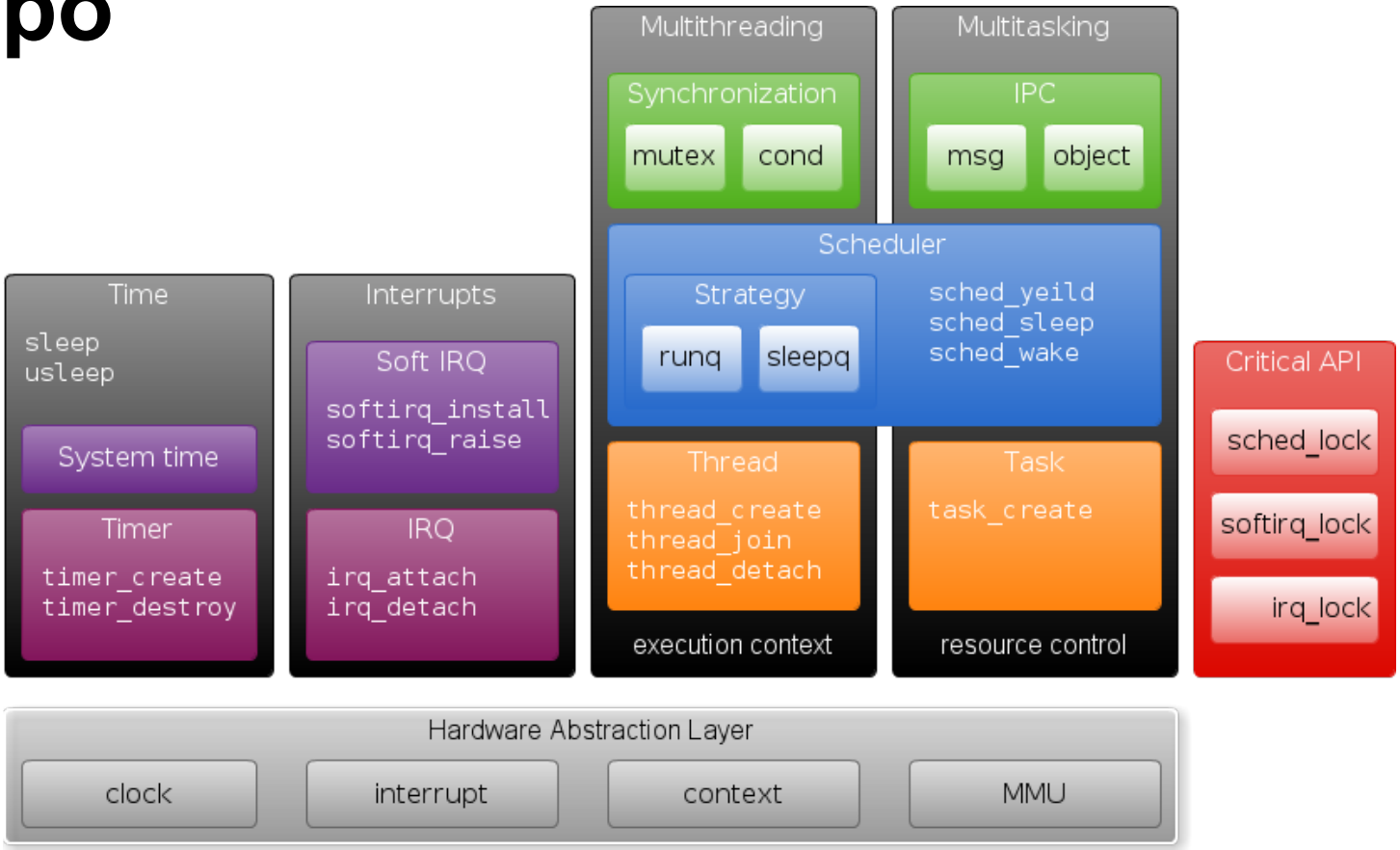
API для модулей

- Позволяют выбирать различные реализации интерфейса
  - Метод планирования
  - Метод выделения памяти
  - Метод управления задачами
- Удобно указывать зависимости
- Позволяют снизить уровень сложности конструкций **`#if`** / **`#ifdef`**

Граф модулей



Ядро


Multithreading
Synchronization
mutex
cond
Multitasking
IPC
msg
object
Scheduler
Strategy
runq
sleepq
sched_yeild
sched_sleep
sched_wake
Time
sleep
usleep
System time
Timer
timer_create
timer_destroy
Interrupts
Soft IRQ
softirq_install
softirq_raise
IRQ
irq_attach
irq_detach
Thread
thread_create
thread_join
thread_detach
Task
task_create
execution context
resource control
Critical API
sched_lock
softirq_lock
irq_lock
Hardware Abstraction Layer
clock
interrupt
context
MMU

Ядро

- Вытесняющая многозадачность
- Приоритеты потоков
- Защита от инверсии приоритетов
- Различные стратегии планирования
- Различные примитивы синхронизации
- Работа со службой времени
- ...

Слой аппаратных абстракций

- Быстрый перенос на новую платформу и архитектуру
- Поддержка различных процессорных архитектур
- Низкие требования по памяти
  (от 16KB ROM, 4KB RAM)
- Возможность работы на контроллерах (без MMU)
- Возможность использовать BSP от производителей

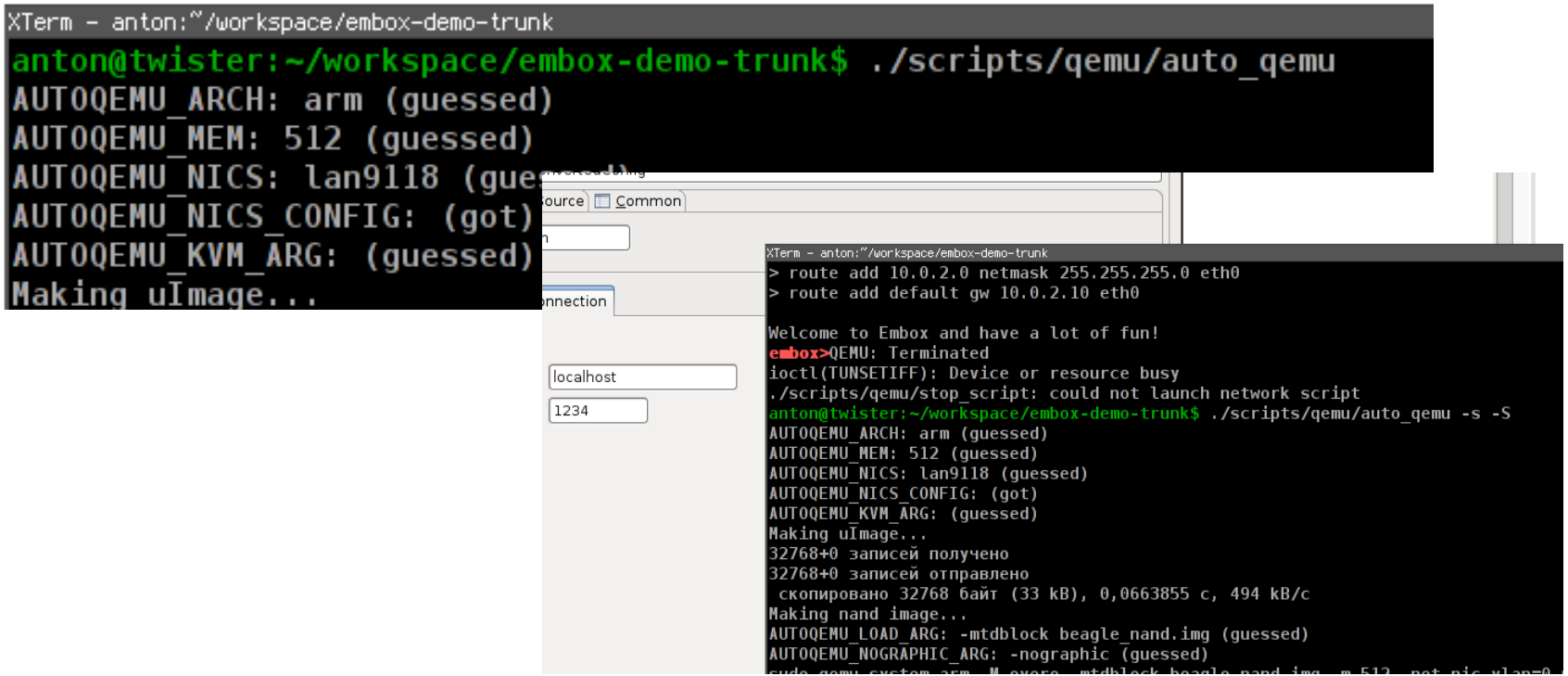
Безопасность

- Стандартные методы защиты данных
  - Firewall
  - Дискретный доступ
  - Мандатный доступ
  - Шифрование
- Защита от вредоносного кода на уровне статической сборки и анализа образа

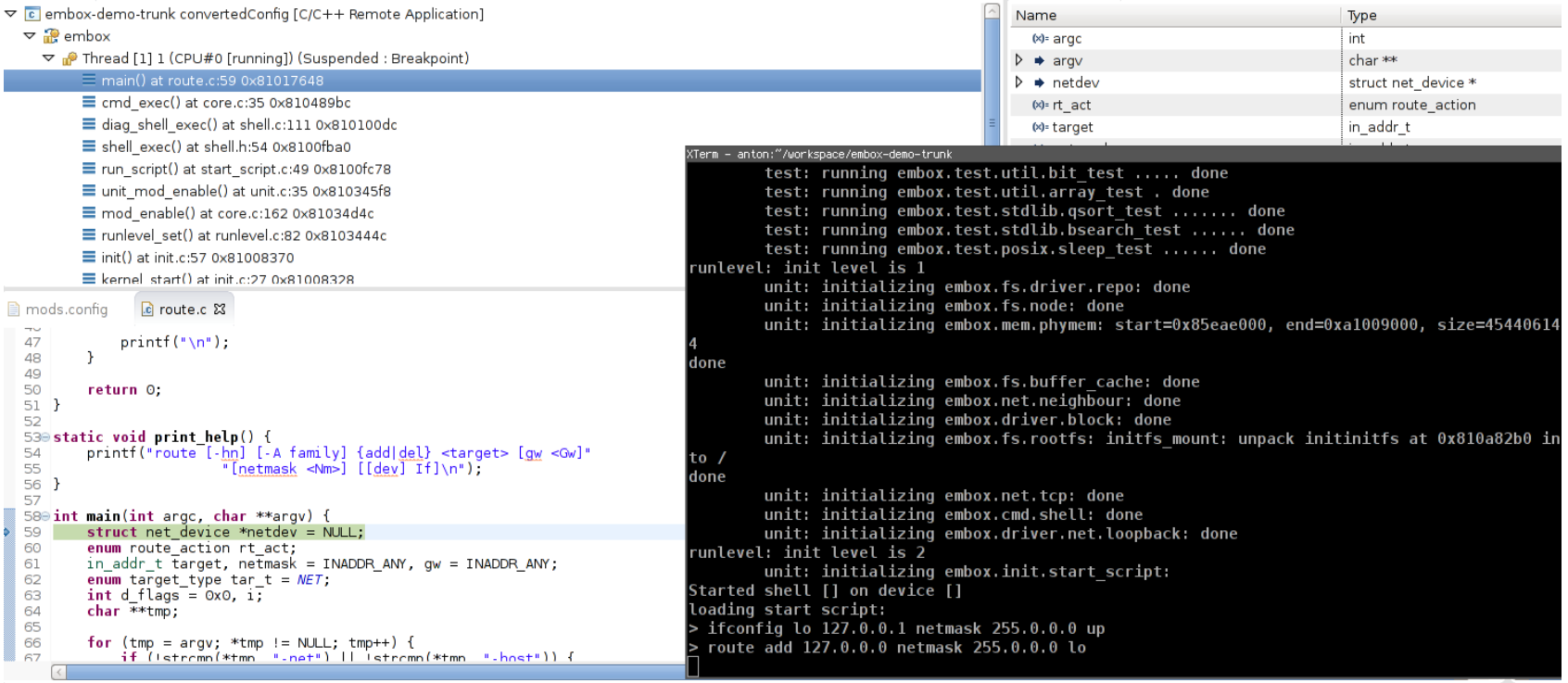
Средства разработки

- Используются распространенные и открытые средства разработки (gcc, gdb, Eclipse)
- Ядро написано на языке C (gnu99). Используется собственная стандартная библиотека
- Поддержка распространенных языков: C++, LISP, Lua, Python, Java ME, Tcl

Простой запуск и отладка



Настольная отладка



Unit-тесты

```
Embox kernel start
runlevel: init level is 0
        unit: initializing embox.driver.interrupt.irqmp: done
        unit: initializing embox.kernel.task.task_resource: done
        unit: initializing embox.kernel.task.task_table: done
        unit: initializing embox.driver.clock.gptimer: done
        unit: initializing embox.kernel.task.kernel_task: done
        unit: initializing embox.kernel.time.jiffies: done
        unit: initializing embox.kernel.thread.starter: done
        unit: initializing embox.mem.static_heap: done
        unit: initializing embox.security.seculog_file: done
        unit: initializing embox.security.smac: done
        unit: initializing embox.kernel.time.timer: done
        unit: initializing embox.kernel.time.kernel_time: done
        unit: initializing embox.net.dev: done
        test: running embox.test.math.math_test .. done
        test: running embox.test.recursion . done
        test: running embox.test.stdlib.qsort_test ....... done
        test: running embox.test.stdlib.bsearch_test ...... done
        test: running embox.test.util.array_test . done
        test: running embox.test.util.hashtable_test ... done
        test: running embox.test.util.tree_test ........... done
        test: running embox.test.util.slist_test ................. done
        test: running embox.test.util.list_test ..................................................... done
        test: running embox.test.util.bit_test ..... done
        test: running embox.test.hal.ipl_test . done
        test: running embox.test.framework.mod.member.ops_test . done
```

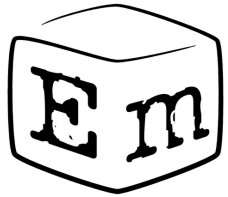
Профилирование

- Встроенная поддержка профилировщиков
  - семплирующий
  - инструментирующий
- Возможность профилирования системы в целом
- Встроенный механизм определения покрытия кода

Выводы

- ОС для встроенных систем с ограниченными ресурсами
- ОС для специализированных устройств с четко заданной функциональностью
- ОС для устройств, где надежность и безопасность доминируют

Контакты



Страница проекта

- **https://code.google.com/p/embox/**

Антон Бондарев

- **anton.bondarev2310@gmail.com**