

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

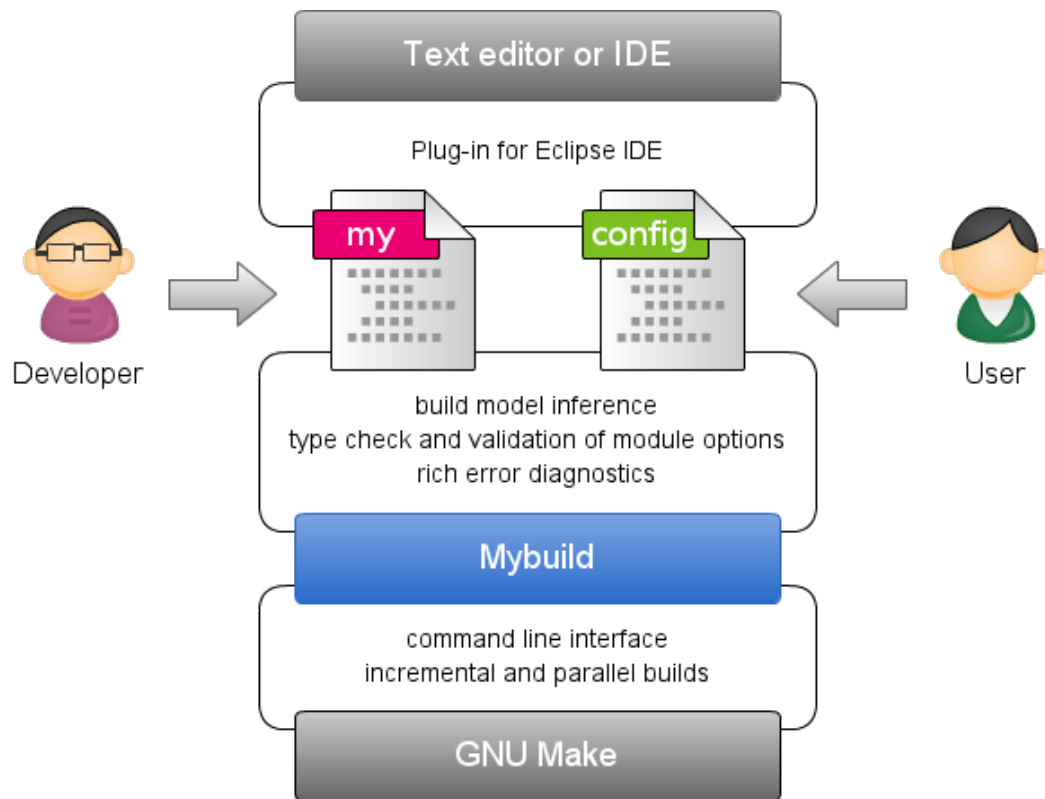
Свойства

- 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

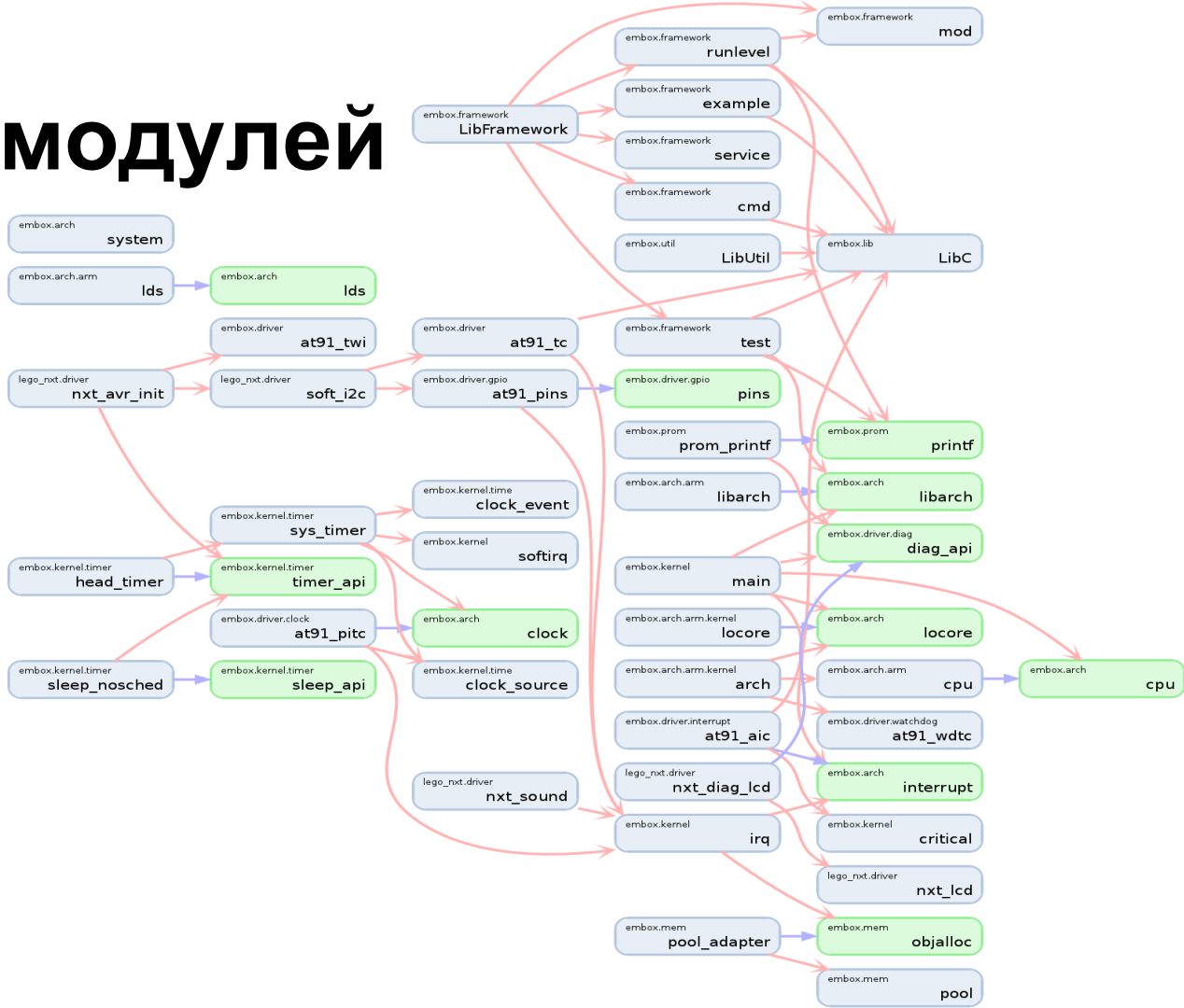
Процесс сборки



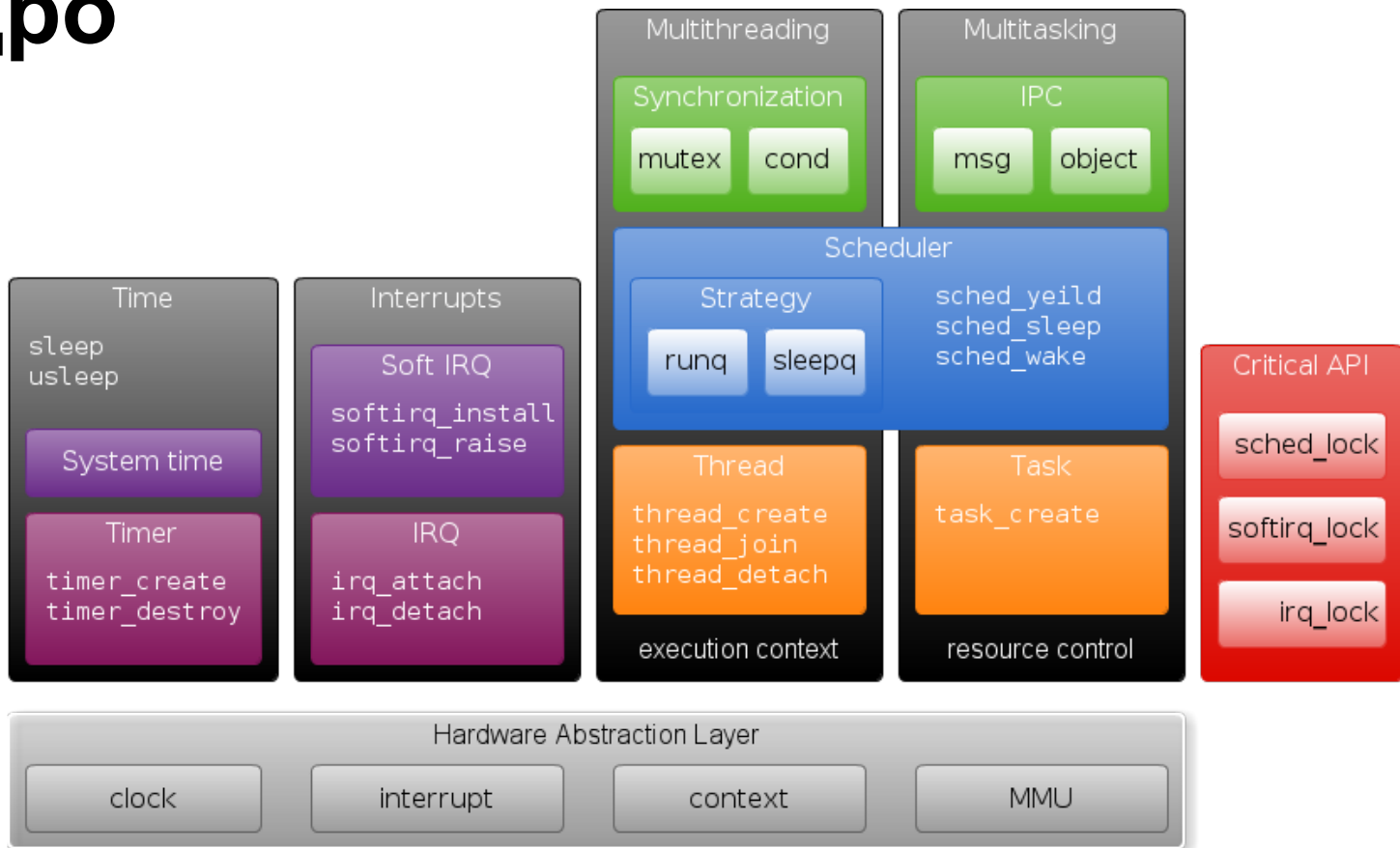
API для модулей

- 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



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

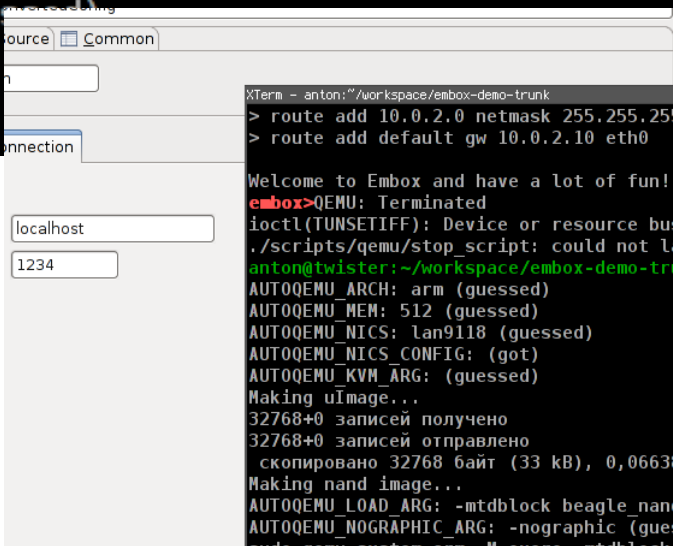
- Стандартные методы защиты данных
 - Firewall
 - Дискретный доступ
 - Мандатный доступ
 - Шифрование
- Защита от вредоносного кода на уровне статической сборки и анализа образа

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

```
XTerm - anton:~/workspace/embox-demo-trunk
anton@twister:~/workspace/embox-demo-trunk$ ./scripts/qemu/auto_qemu
AUTOQEMU_ARCH: arm (guessed)
AUTOQEMU_MEM: 512 (guessed)
AUTOQEMU_NICS: lan9118 (guessed)
AUTOQEMU_NICS_CONFIG: (got)
AUTOQEMU_KVM_ARG: (guessed)
Making uImage...
```



```
XTerm - anton:~/workspace/embox-demo-trunk
> route add 10.0.2.0 netmask 255.255.255.0 eth0
> route add default gw 10.0.2.10 eth0

Welcome to Embox and have a lot of fun!
embox>QEMU: Terminated
ioctl(TUNSETIFF): Device or resource busy
./scripts/qemu/stop_script: could not launch network script
anton@twister:~/workspace/embox-demo-trunk$ ./scripts/qemu/auto_qemu -s -S
AUTOQEMU_ARCH: arm (guessed)
AUTOQEMU_MEM: 512 (guessed)
AUTOQEMU_NICS: lan9118 (guessed)
AUTOQEMU_NICS_CONFIG: (got)
AUTOQEMU_KVM_ARG: (guessed)
Making uImage...
32768+0 записей получено
32768+0 записей отправлено
скопировано 32768 байт (33 kB), 0,0663855 с, 494 kB/c
Making nand image...
AUTOQEMU_LOAD_ARG: -mtdblock beagle_nand.img (guessed)
AUTOQEMU_NOGRAPHIC_ARG: -nographic (guessed)
sudo qemu-system-arm -M versa -mtdblock beagle_nand.img -m 512 -net nic -vlan=0
```

Настольная отладка

embox-demo-trunk convertedConfig [C/C++ Remote Application]

embox

Thread [1] 1 (CPU#0 [running]) (Suspended : Breakpoint)

```
main() at route.c:59 0x81017648
cmd_exec() at core.c:35 0x810489bc
diag_shell_exec() at shell.c:111 0x810100dc
shell_exec() at shell.h:54 0x8100fba0
run_script() at start_script.c:49 0x8100fc78
unit_mod_enable() at unit.c:35 0x810345f8
mod_enable() at core.c:162 0x81034d4c
runlevel_set() at runlevel.c:82 0x8103444c
init() at init.c:57 0x81008370
kernel_start() at init.c:27 0x81008328
```

mods.config

route.c

```
76     printf("\n");
47     }
48 }
49
50     return 0;
51 }
52
53 static void print_help() {
54     printf("route [-hn] [-A family] {add|del} <target> [gw <Gw>]
55           [netmask <Nm>] [[dev] if]\n");
56 }
57
58 int main(int argc, char **argv) {
59     struct net_device *netdev = NULL;
60     enum route_action rt_act;
61     in_addr_t target, netmask = INADDR_ANY, gw = INADDR_ANY;
62     enum target_type tar_t = NET;
63     int d_flags = 0x0, i;
64     char **tmp;
65
66     for (tmp = argv; *tmp != NULL; tmp++) {
67         if (!strcmp(*tmp, "-net") || !strcmp(*tmp, "-host")) {
```

Name	Type
argc	int
argv	char **
netdev	struct net_device *
rt_act	enum route_action
target	in_addr_t

```
XTerm - anton:"/workspace/embox-demo-trunk
test: running embox.test.util.bit_test ..... done
test: running embox.test.util.array_test . done
test: running embox.test.stdlib.qsort_test ..... done
test: running embox.test.stdlib.bsearch_test ..... done
test: running embox.test.posix.sleep_test ..... done
runlevel: init level is 1
unit: initializing embox.fs.driver.repo: done
unit: initializing embox.fs.node: done
unit: initializing embox.mem.phymem: start=0x85eae000, end=0xa1009000, size=45440614
4
done
unit: initializing embox.fs.buffer_cache: done
unit: initializing embox.net.neighbour: done
unit: initializing embox.driver.block: done
unit: initializing embox.fs.rootfs: initfs_mount: unpack initinitfs at 0x810a82b0 in
to /
done
unit: initializing embox.net.tcp: done
unit: initializing embox.cmd.shell: done
unit: initializing embox.driver.net.loopback: done
runlevel: init level is 2
unit: initializing embox.init.start_script:
Started shell [] on device []
loading start script:
> ifconfig lo 127.0.0.1 netmask 255.0.0.0 up
> route add 127.0.0.0 netmask 255.0.0.0 lo
```

Console Tasks Problems Executables Memory

embox-demo-trunk convertedConfig [C/C++ Remote Application] gdb

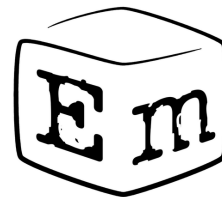
Профилирование

- Встроенная поддержка профилировщиков
 - семплирующий
 - инструментующий
- 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