

# Применение ОСРВ Embox для создания доверенного ПЛК

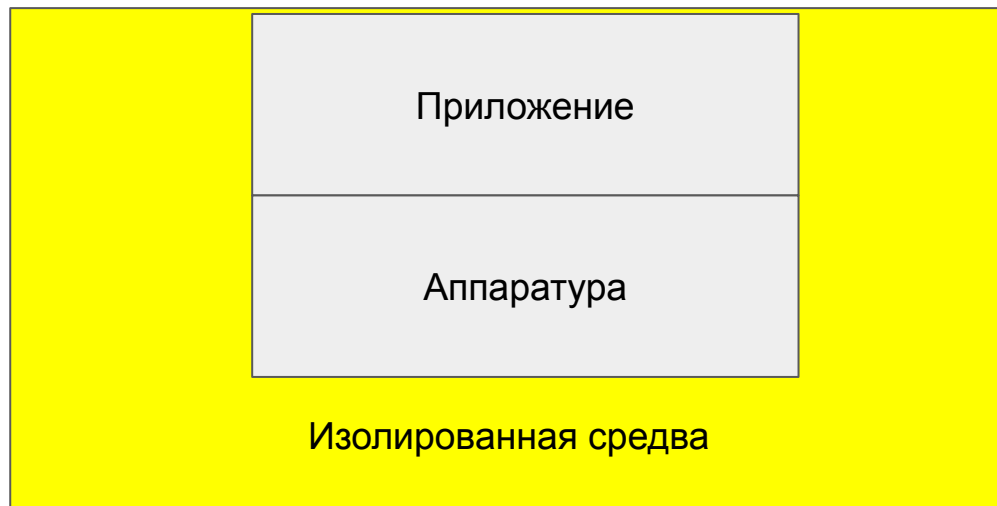
Антон Бондарев

OS:Day, 19 июня 2025

# Разработка встроенных систем

Высказывание одного разработчика АСУ-ТП:

“Когда я говорю о системе (АСУ-ТП) в ней вообще не должно быть заметно ОС, она только мешает”



# Среда исполнения



# Разработка встроенных систем

“Когда я говорю о системе (АСУ-ТП) в ней вообще не должно быть заметно ОС, она только мешает“

**ОС должна обеспечивать запуск и работу  
необходимого прикладного ПО**

# Embox

**Embox** — свободная операционная система реального времени (RTOS), разрабатываемая для встроенных систем.

Основная идея использование ПО **Линукс** в более безопасном и детерминированном, менее ресурсоемком и энергопотребляющем окружении

# Embox

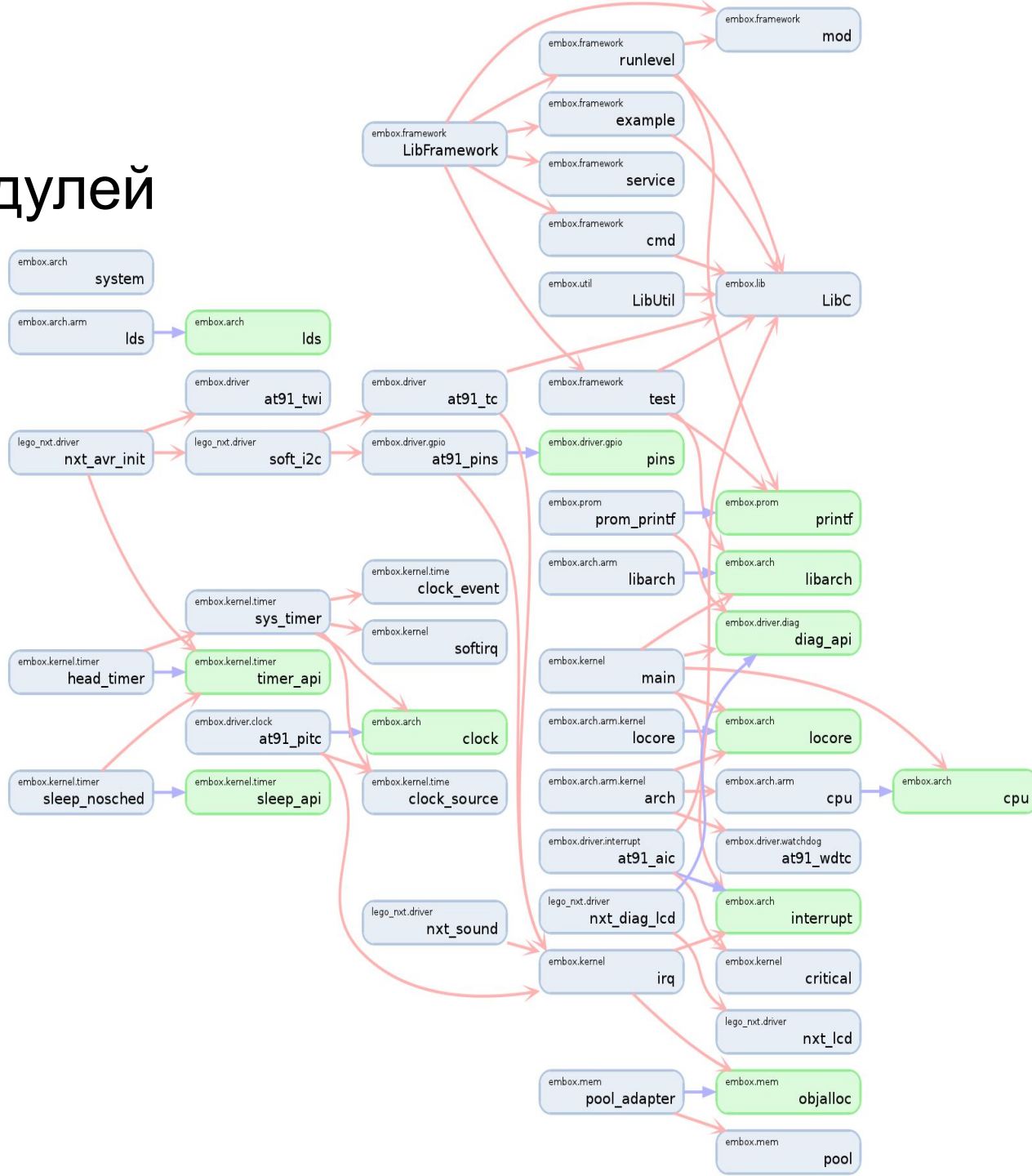
Операционная система под конкретную задачу, основанная на **специальном DSL языке** и использующая:

- Статическую информацию о задачах устройства
- Статическую конфигурацию системы
- Статический анализ зависимостей
- Статическую проверку параметров системы

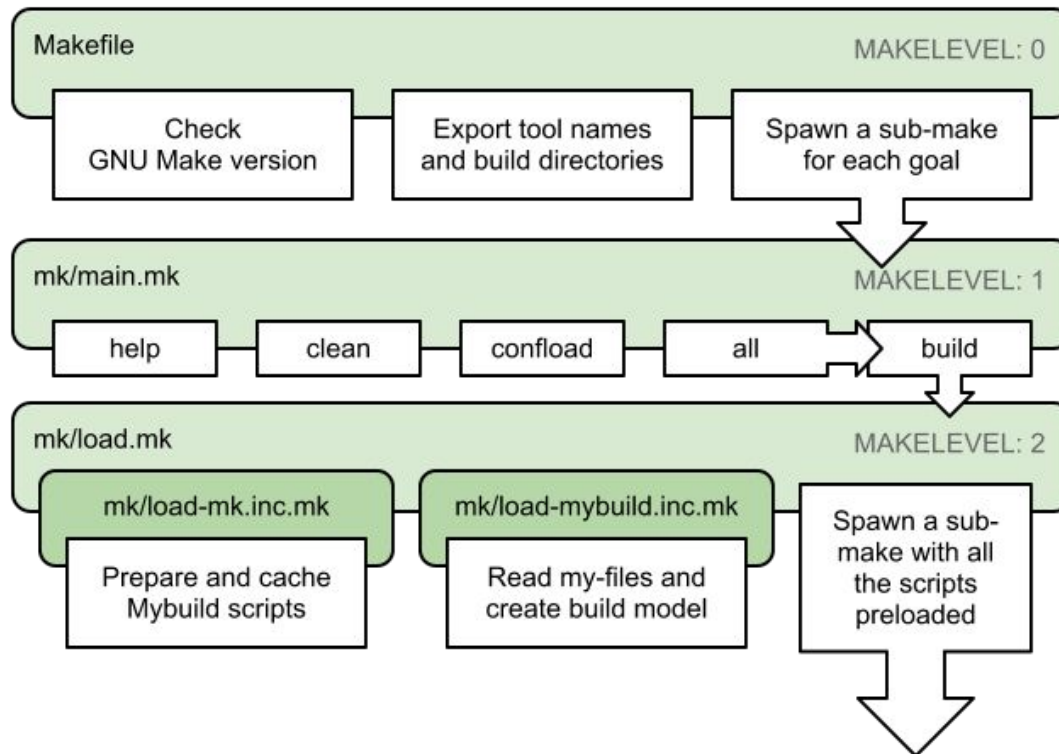
**Embox build system:**

KBuild + OpenEmbedded + DevTree

# Граф модулей



# Mybuild-процесс





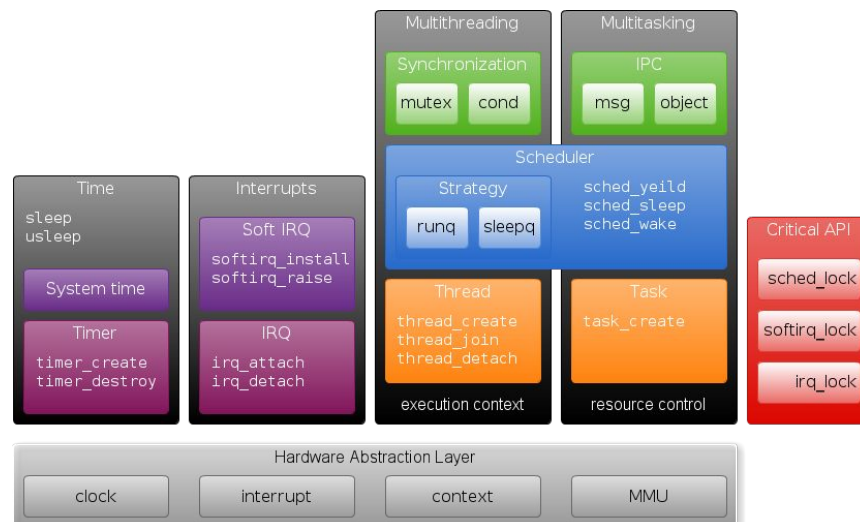
# Embox

Свойства полученной системы:

- Отсутствует избыточная функциональность
  - Экономия вычислительных ресурсов
  - Экономия энергопотребления
- Упрощение сертификации системы
- Невозможность запуска вредоносных приложений

# Embox

- Поддерживает: x86, ARM, RISC-V, MIPS, E2k, ...
- Ядро с вытесняющей многозадачностью
- Имеет файловую, сетевые, другие подсистемы
- Имеет набор привычных команд
- POSIX интерфейс



# Среда исполнения



# Embox: Hello world

Потребуются:

1. Исходный код
2. Описание модуля (Mybuild)
3. Описание системы (mods.conf)

# Embox: Hello world (source)

## Обычный C файл

```
#include <stdio.h>
```

```
int main(int argc, char **argv) {
```

```
    printf("Hello, world!\n");
```

```
}
```

# Embox: Hello world (Mybuild)

```
package embox.cmd
```

```
@AutoCmd
```

```
@Cmd(name = "hello_world", help="First Embox  
application")
```

```
module hello_world {
```

```
    source "hello_world.c"
```

```
}
```

# Embox: Hello World (mods.conf)

....

```
include embox.kernel.task.resource.idesc_table(idesc_table_size=32)
```

```
include embox.kernel.task.resource.sig_table(sig_table_size=20)
```

```
include embox.kernel.task.resource.env(env_per_task=4,env_str_len=64)
```

```
include embox.kernel.thread.thread_local_none
```

```
include embox.kernel.thread.thread_cancel_disable
```

```
include embox.kernel.thread.signal.siginfoq(siginfo_pool_sz=8)
```

```
@Runlevel(2) include embox.kernel.sched.strategy.priority_based
```

...

```
include embox.cmd.hello_world
```

....

# Среда исполнения





# Embox: Mybuild (extbuild)

- Сборка ПО Линукс с их оригинальной сборкой
- Аналог
  - `./configure; make; make install;`

## Embox: Mybuild (extbuild)

```
@Build(stage=2,script="$$(EXTERNAL_MAKE)")
```

```
@Cmd(name = "nano",
```

```
...
```

```
"")
```

```
module nano {
```

```
    source "^BUILD/extbld/^MOD_PATH/install/nano.o"
```

```
    @NoRuntime depends embox.compat.posix.regex
```

```
    depends embox.compat.posix.LibCurses
```

```
}
```

# Embox: Mybuild (extbuild)

```
PKG_NAME := nano
```

```
PKG_VER := 2.2.6
```

```
PKG_SOURCES :=
```

```
http://www.nano-editor.org/dist/v2.2/$(PKG_NAME)-$(PKG_VER).tar.gz
```

```
...
```

```
$(CONFIGURE) :
```

```
export EMBOX_GCC_LINK=full; \
```

```
cd $(PKG_SOURCE_DIR) && ( \
```

```
./configure --host=$(AUTOCONF_TARGET_TRIPLET) \
```

```
--target=$(AUTOCONF_TARGET_TRIPLET) \
```

```
...
```

```
)
```

# Embox: Mybuild (extbuild)

**`$ (BUILD) :`**

```
cd $(PKG_SOURCE_DIR) && ( \  
  
  $ (MAKE) MAKEFLAGS='$(EMBOX_IMPORTED_MAKEFLAGS)'; \  
  
)  
  
touch $@
```

**`$ (INSTALL) :`**

```
cp $(PKG_SOURCE_DIR)/src/nano $(PKG_INSTALL_DIR)/nano.o  
  
touch $@
```

# Окружение разработчика

Для сборки требуется:

- Cross-compiler
  - Embox использует 'gcc' or 'llvm'
- Стандартная библиотека языка C
  - Собственная конфигурируемая библиотека
- Дополнительные библиотеки
  - Линкуются все модули (библиотеки) описанные в `mods.conf` или подтягиваемые по зависимостям

# Embox: пути к дополнительным библиотекам

Для добавление путей экспортируемых библиотекой используется аннотация **@BuildArtefactPath**, а при использовании аннотация **@BuildDepends**

# Среда исполнения



# Embox: Additional library

```
@BuildDepends(gcc_build)
```

```
@BuildArtifactPath(cppflags_before="-I$(abspath  
$(EXTERNAL_BUILD_DIR))/third_party/gcc/gcc_build/install/_target/include/c+  
+/_gcc_version -I$(abspath  
$(EXTERNAL_BUILD_DIR))/third_party/gcc/gcc_build/install/_target/include/c+  
+/_gcc_version/_target")
```

```
static module libstdcxx extends embox.lib.libstdcxx {
```

```
  @AddPrefix("^BUILD/extbld/third_party/gcc/gcc_build/install/libs")
```

```
  source "libstdc++.a"
```

```
  @NoRuntime depends gcc_build
```

```
}
```



# Embox: Additional library (BuildDepends)

```
@AutoCmd
```

```
@Cmd(name="stl_demo_sort1",  
      help="example for stl algo sort",  
      man="")
```

```
@BuildDepends(embox.lib.libstdcxx)
```

```
@Build(stage=2,script="true")
```

```
module stl_demo_sort1 {  
    source "sort1.cpp"
```

```
@NoRuntime depends embox.lib.libstdcxx
```

```
}
```

# Среда исполнения

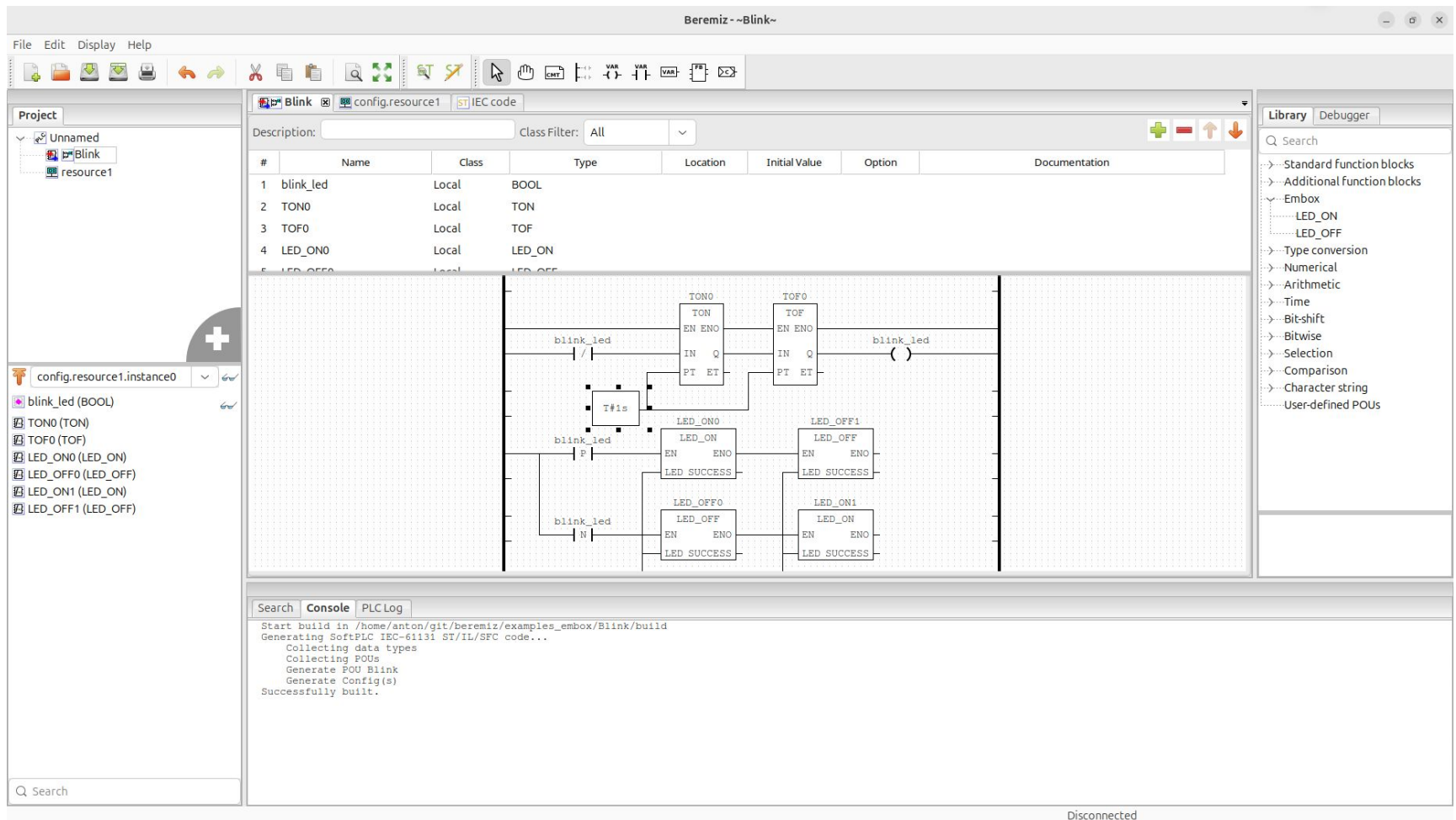


# Приложения АСУ-ТП

- В АСУ-ТП применяются (Программируемые Логические Контроллеры) ПЛК. Их разработка регламентирована стандартом МЭК 61131
- Раздел 3 (МЭК 61131-3) специфицирует языки программирования для ПЛК
- Открытые проекты для средства разработки для ПЛК
  - Beremiz (МЭК 61131-3)
  - 4Diac forte (IEC 61499)

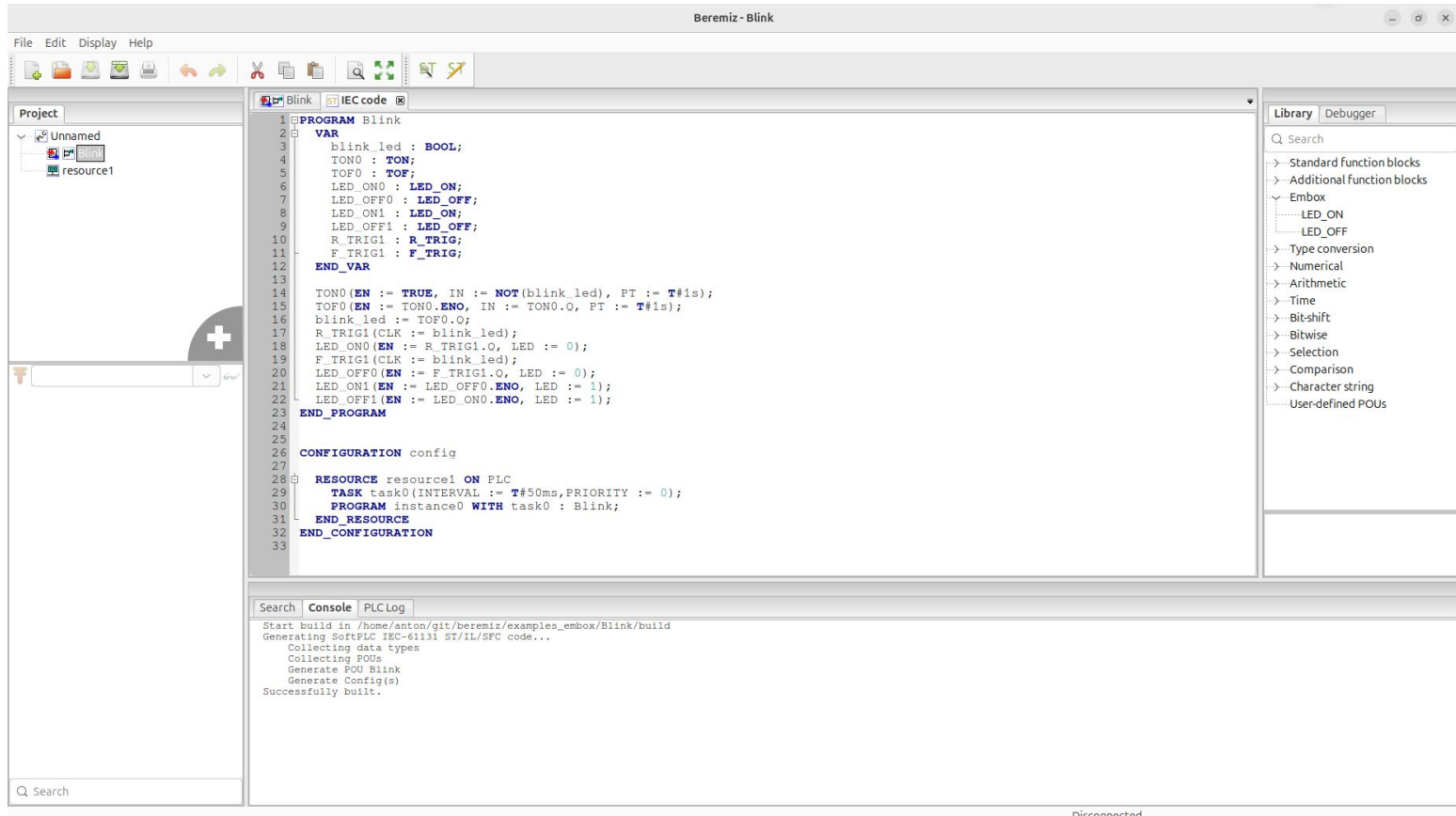
# Приложения АСУ-ТП

- Генерируются допустимые функциональные блоки



# Embox на K1921BГ015

- Генерируются ST
- Транслируется в Си доработанным matIEC

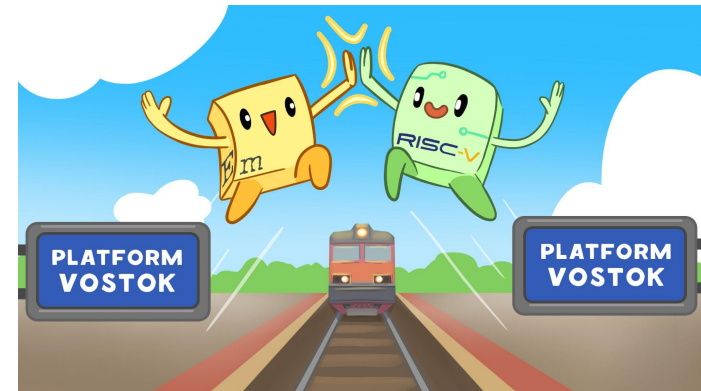


# Среда исполнения



# Embox на K1921ВГ015

- НИИЭТ (входит в ГК Элемент)
- RISC-V, ПЗУ 1mb, ОЗУ 256 + 64 кБ
- Разработали только драйвера
- Перенос реальных проектов



- <https://habr.com/ru/articles/881784/>

# Среда исполнения





# Заключение

- Создание статического изолированного окружения позволяет повысить безопасность и не потерять в производительности
- При разработке объектов КИИ должна учитываться безопасность системы в целом.
- Возможный способ повышения безопасности это использование принципа Secure By Design

# Ссылки

Страница проекта

<http://embox.github.io/>

Репозиторий проекта

<https://github.com/embox/>

Youtube канал

<https://www.youtube.com/@embox-rtos>

Телеграмм канал

[https://t.me/embox\\_chat](https://t.me/embox_chat)

# Contacts

Anton Bondarev

- [anton.bondarev2310@gmail.com](mailto:anton.bondarev2310@gmail.com)
- @anton\_bondarev\_embox
-