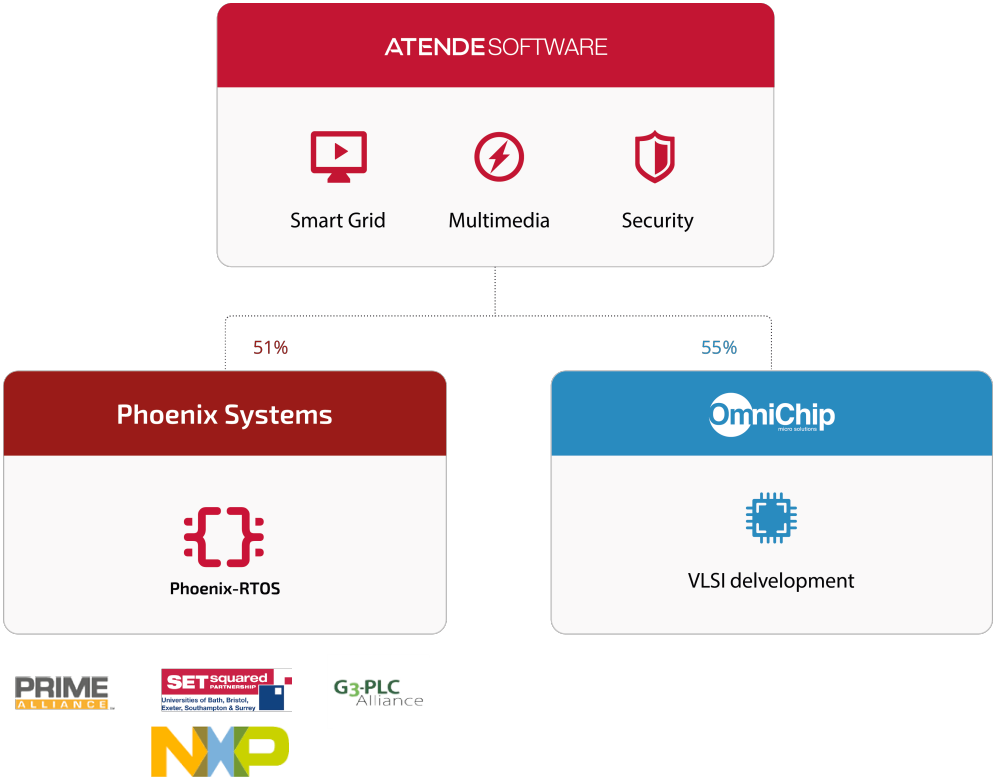
Phoenix Systems

Phoenix-RTOS – the next generation operating system for Internet of Things

Phoenix Systems



Why Phoenix-RTOS?

Computing power and resources of commodity microcontrollers reach level adequate for complex applications

Due to processing power most of functionalities can be implemented in software

Software defined solutions dramatically simplify the process of implementation of new IoT devices

Complex software inevitably calls for operating system capable to create its operation environment

Market analysis

There is a great demand for an efficient operating system for IoT that simplifies the embedded software development process

Manufacturers finally understand the need to use the operating system as a device basis

Popular operating systems for PCs are not compliant with IoT devices (too big hardware requirements, no real-time, low modularity and scalability)

Operating systems for IoT - the competition (Contiki, WindRiver Rocket, Zephyr, FreeRTOS):

- too simple application environment,
- limitations in memory size, number of threads and processes,
- no device drivers,
- no IoT communication stacks.

Phoenix-RTOS - introduction

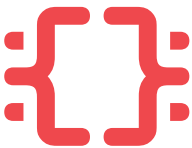
Phoenix-RTOS (1)

State-of-the-art advanced microkernel-based real-time operating system for IoT

Foundation for software defined solutions

Support for next generation microcontrollers (ARM Cortex-M7, RISC-V)

Extreme scalability and modularity



Phoenix-RTOS (2)

Hardware architectures

IA32, ARMv7, ARM,RISC-V, eSi-RISC

Object-based memory management

support for MMU and non-MMU architectures

memory size from 32 KB to 2^{64} KB

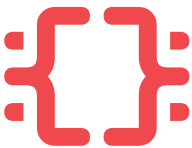
Communication stacks

TCP/IP

PLC: Phoenix-PRIME, Phoenix-G3 (Cenelec, FCC)

wireless: 802.15.4, 802.15.4g, Wireless M-Bus

Optimized standard C library (libphoenix) with POSIX extensions



Phoenix-RTOS – timeline

Phoenix (1999) - prototype

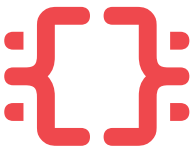
- IA32
- dedicated standard library and simple tools

Phoenix-RTOS 2 (2004)

- IA32, ARM, eSI-RISC
- monolithic kernel (200K LoC), (300-600)K binary image
- Phoenix-PRIME
- newlib, busybox, libmeter, libcosem, UN*X applications

Phoenix-RTOS 3 (2017)

- IA32, ARM, ARMv7, RISC-V, eSI-RISC (multicore)
- microkernel (20K LoC), (30-100)K binary image
- Phoenix-PRIME, Phoenix-G3, Phoenix-802.15.4, Phoenix-WMBUS
- new memory management system (objects, MMU, non-MMU)
- libphoenix, libmeter, libcosem
- busybox, UN*X applications



Phoenix-RTOS

Phoenix-RTOS – applications

Smart Home

Military

Healthcare

Home electronics

Wearable technology

Smart Grid

Smart cities

Enterprise automation

Robotics

Automotive

Phoenix-RTOS – programmer perspective (1)

Advanced application environment maintaining small operating system size (100 KB code+ 20 KB data)

Real-time capabilities (low latency and jitter)

Modular and scalable architecture (microkernel based)

Software defined communication stacks (e.g. PLC PRIME, G3-PLC) constituting operating system`s modules

Support for battery devices keeping the synchronous application interface

Application frameworks simplifying development of IoT applications (Smart Grid)

Phoenix-RTOS

Phoenix-RTOS – programmer perspective (2)

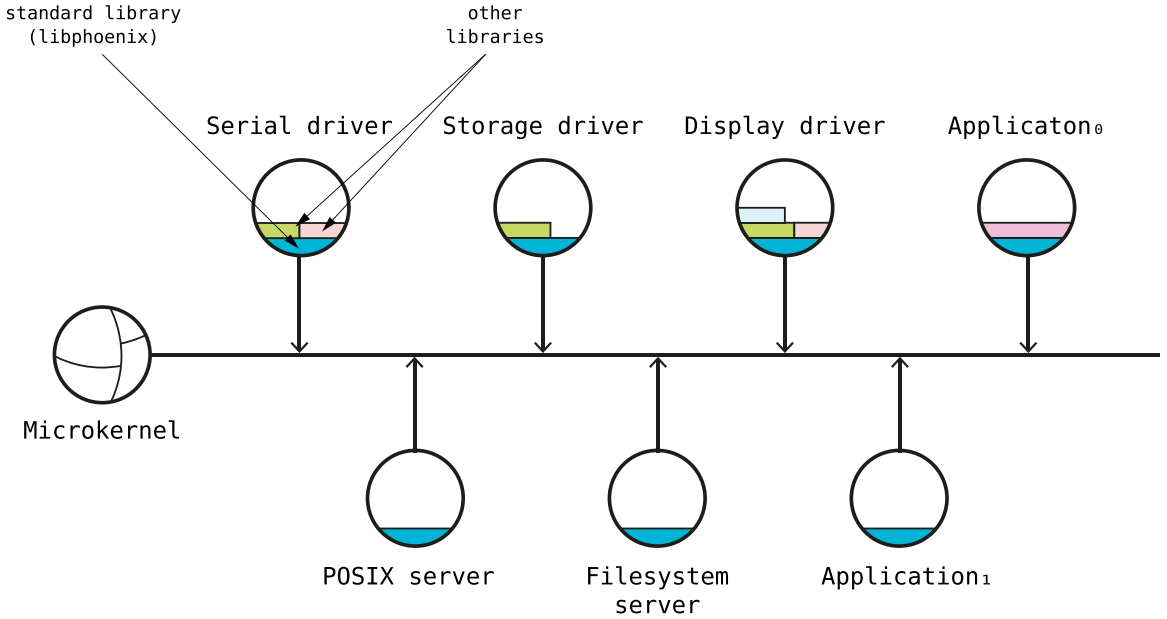
Support for numerous devices (drivers)

Support for numerous filesystems

Software libraries for Smart Grid (libmeter, libcosem)

Popular tools for software development (GNU CC)

Phoenix-RTOS – microkernel architecture

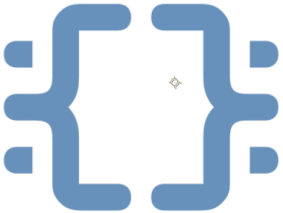


Phoenix-RTOS – versions

Native (ANSI C + ext.) – 80KB

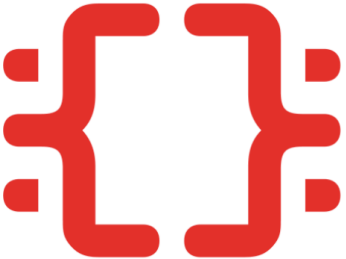
smart meters
low-power devices

POSIX - 200KB



data concentrators
devices with advanced functionality

ARINC 653



drones
avionics

(under development)

First implementations

Phoenix-RTOS 2 – first implementations

Andra DCU

Phoenix-PRIME board

Phoenix-RTOS – appliances (1)

Andra comander amiDC-3 (Poland)

PRIME DCU and balancing meter

PRIME Alliance certified BN

Energa-Operator network implementation

Quantity: 30K

Implemented 10K, 20K in production

Phoenix-RTOS – appliances (2)

Saiman SDM 1, SDM 3 (Kazakchstan)

Software defined smart energy meter (PLC PRIME), class 0.5

USB interface

Implementation in Almaty power grid network

Quantity: 200K

Short manufacturing series tests in progress (150 meters)

Phoenix-RTOS – appliances (3)

Phoenix-RTOS – appliances (4)

Phoenix-RTOS – appliances (5)

Apator-Metrix iSMART1 (Italy)

Software defined smart gas meter (certified)

W-Mbus, ZigBee interfaces

Quantity: 2M

Targeted also for other markets due to high modularity

Phoenix-RTOS – appliances (6)

Incotex DCU (Russia)

DCU and balancing meter (PLC PRIME, G3-PLC)

Fulfilling requirements for Russian market

Quantity: 60K

Targeted also for other markets in Asia and Middle East

NXP iMX.RT – next generation microcontroller

State-of-the-art microcontroller designed in 40nm architecture, ARM Cortex-M7

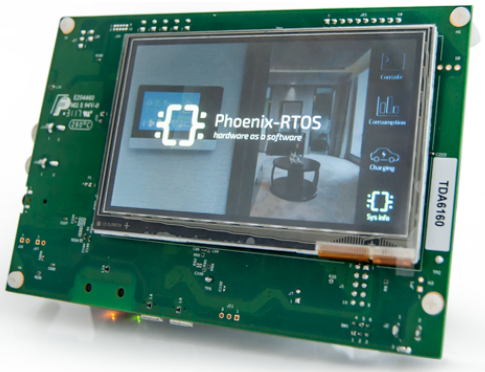
600 MHz, 512 KB memory, advanced peripheral devices

Various operating modes, up to 180 mW power consumption

Est. price 2.6 USD

Ideal for software defined solutions

Phoenix-RTOS promoted as the operating system, premiered during EUW'17 in Amsterdam



Phoenix-RTOS microkernel

Hardware Abstraction Layer

Kernel initialization

Definition of basic types

Definition of `syspage_t` structure

Spinlocks

Kernel console

String functions (`memcpy`, `memset`)

Low-level interrupts and exceptions handling

MMU and MPU handlings

Clock support (for scheduling)

Context definition and switching

Memory management (1)

Support for MMU and non-MMU architectures

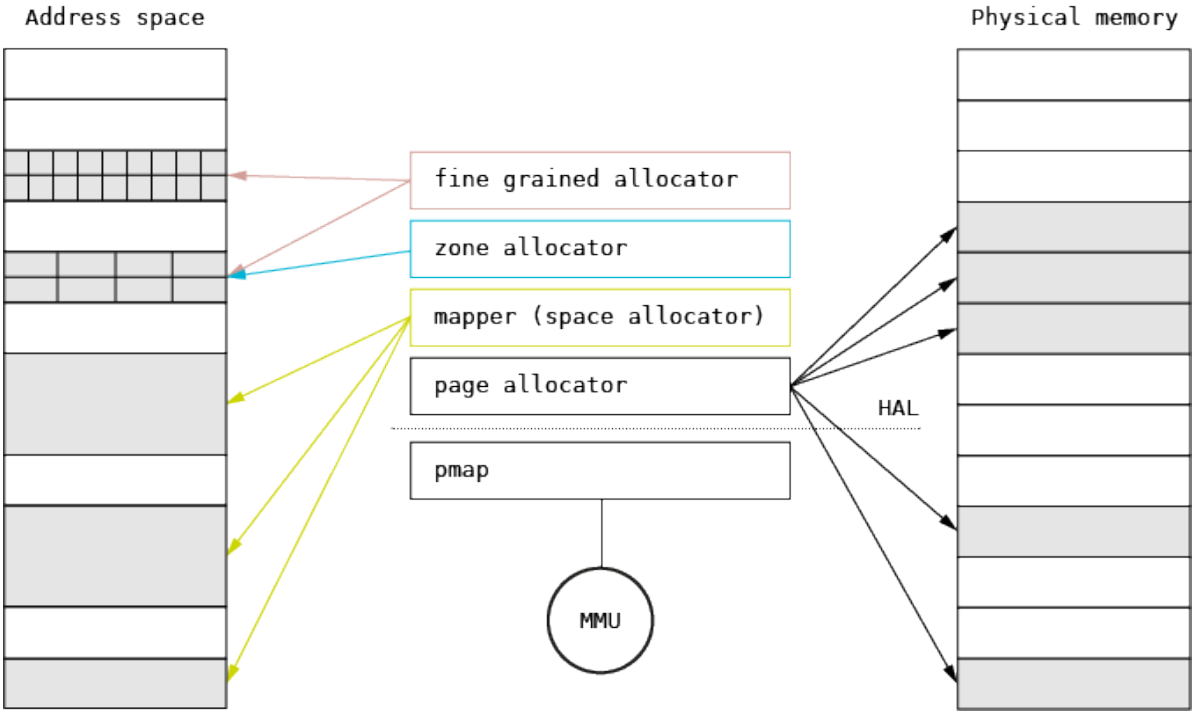
Memory virtualization

Memory protection

Many layers of memory allocation

Memory objects

Memory management (2)



Process and thread management (1)

Multicore support

Preemptive scheduling

Threads and processes (ELF)

Message passing

- data copying
- virtual memory entries passing

Phoenix-RTOS – process management (2)

Kernel level synchronization

- spinlocks
- locks with priority inheritance

User level synchronization

- mutexes
- conditional variables
- semaphores

Phoenix-RTOS – inter-process communication

Synchronous message passing

- `msgSend(),msgRecv(),msgRespond()`

Asynchronous message passing

- `msgPulse(),msgRecv()`

Naming services

- `register(), lookup()`

Based on virtual memory and memory sharing (data copying reduction)

Phoenix-RTOS user-space

Phoenix-RTOS – libraries

libphoenix

- standard C library & specific system functions
- available on github.com/phoenix-rtos/libphoenix
- POSIX mode

libmeter

- library for measurement of electric energy

libcosem

- DLMS/COSEM library

Device drivers

Available classes

adc	**multi (multi drivers)**	storage
display	net	tty
dma	pseudo	usb
esai	rtc	watchdog
gpio	security	..
i2c	spi	

Implemented as user level services (application servers handling interrupts)

Multi drivers support devices embedded into microcontrollers and optimize resource usage (e.g. on STM32 family)

Available on github.com/phoenix-rtos/phoenix-rtos-devices

Phoenix-RTOS – filesystems

Available file servers

- dummyfs - RAM and initial fileystems
- FAT - FAT12, FAT16, FAT32
- ext2 - GNU/Linux filesystem for block devices
- JFFS2 - filesystem for NOR and NAND
- meterfs - proprietary filesystem for Flash (minimal requirements)

Implemented as user level services (storage device drivers)

Available on github.com/phoenix-rtos/phoenix-rtos-filesystems

Communication stacks

Communication stacks

TCP/IP

open network stack
available on github.com/phoenix-rtos/phoenix-rtos-net/)

Phoenix-PRIME

PRIME 1.3.6
PRIME 1.4

Phoenix-G3

available in 2018

Phoenix- 802.15.4

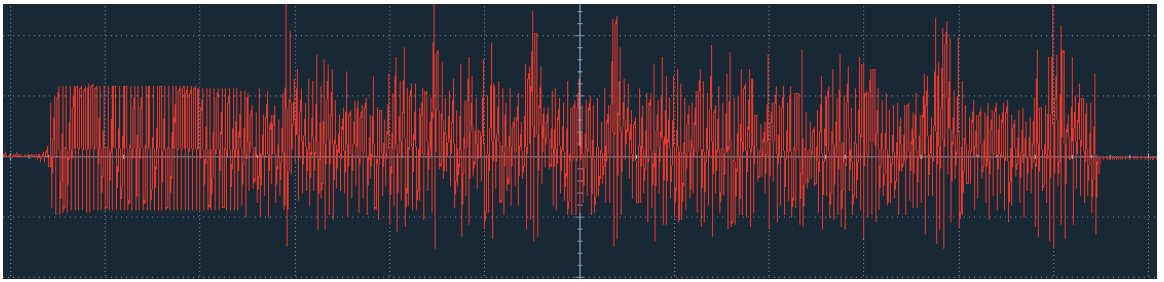
available in 2019
802.15.4, 802.15.4g
ZigBee and WiSun

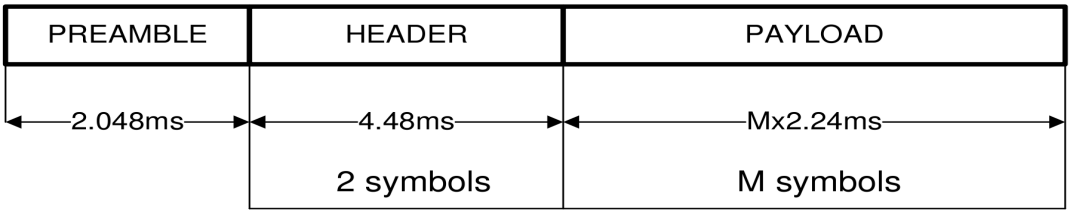
Phoenix-WMBUS

Software-defined PLC (e.g. PRIME)

Software defined PowerLine Communication is based on digital signal processing using in-CPU ADC/DAC converters and Phoenix-RTOS real-time capabilities

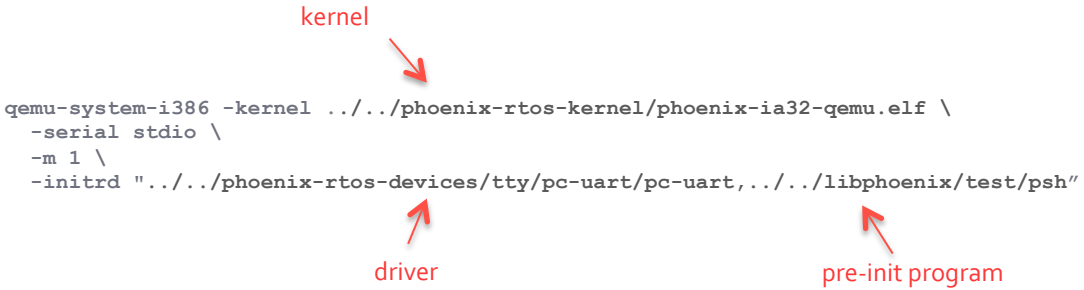
Sample PRIME signal (frame) is presented below





How to use?

Running on IA32 emulator




```
Phoenix-RTOS microkernel v. 2.71
hal: GenuineIntel Family 6 Model 6 Stepping 3 (4/4)
hal: +fpu+de+pse+tsc+msr+pae+apic+pge+cmov+pat
vm: Initializing page allocator (172+324)/960KB, page_t=16
vm: [32.][24K]SYPPCKKKP[5A][12.][5A][72.]B[80x][16B][1048256x][64B]
vm: Initializing memory mapper: (58*52) 3016
vm: Initializing kernel memory allocator: (64*48) 3072
vm: Initializing memory objects
proc: Initializing thread scheduler, priorities=8
syscalls: Initializing syscall table [39]
main: Starting syspage programs (2) and init
pc-uart: Initializing UART 16550 driver
pc-uart: Detected interface on 0x3f8 irq=4
(psh)%
```

www.phoenix-rtos.com