Embedded Elixir

Elixir Taiwan Meetup
July 25, 2016

Jake Morrison <jake@cogini.com>

What is Embedded Programming?

Systems that interact with the physical world

Resource constrained systems

Machines controlled by software

Robots

Appliances

Systems that interact with the physical world

- Micro-controllers
  - 8-bit, e.g. 8051, PIC, Atmel (Arduino)
  - 16-bit, e.g. ARM
- Digital I/O
- Analog I/O
- Pulse Width Modulation
- Sensors
  - Temperature
  - Accelerometer
  - GPS

IoT

Data collection + networking

We are the winners of the "cell phone wars"

Raspberry PI (Broadcom)

Beaglebone (TI)

C.H.I.P. (Microtek)

Embedded Projects

GPS tracker + controller

VoIP IP-PBX

Logo inserter for satellite television

GPS tracker + controller

4 MB RAM / 2 MB ROM / no MMU :-(

GPS / GPRS / GPIO

Embedded Linux (uCLinux)

Over the air updates

Over the air configuration

C/C++ initially, later Lua

Robots

VoIP IP-PBX

Appliance

Embedded Linux (Ubuntu, OpenEmbedded)

Xen VMs

- Firewall
- Configuration
- Application

Python

Lots of resources, challenge is configuration and management

Logo inserter for satellite television

Embedded Linux (Ubuntu)

Proprietary drivers for SDI card

C++ for image manipulation

Erlang for supervision and configuration

Erlang was designed for this!

Telephone switch

Interfacing with switch hardware

"Soft" real time

SunOS, 32 MB of RAM

VxWorks RTOS

Erlang Features

Functional programming

- Outputs depend only on inputs
- No side effects
- Pattern matching: reject invalid input
- Crash dumps with state of whole system

Erlang Features

Supervision trees

Good behavior when hitting resource limits

Concurrency: isolate one request from another

Distributed programming: Reliability requires more than one computer

OTP standardizes behaviors, e.g. supervisor, client server, event handling

Erlang Features

Tracing live systems without big performance impact

Ability to see state of running system, e.g. observer

Built-in in-memory database, replicated across nodes

Inter-process Communication: NIF

Embed C in Erlang VM

High performance but dangerous

Good for things like crypto

Inter-process Communication: Port

VM supervises external process

Erlang code sends messages to port, which talks to external process

Communication over stdin/stdout

Lower performance but full isolation

Serialization overhead

Inter-process Communication: Erlport / Snake

Interop between Erlang and Python or Ruby

Pool of worker processes to handle jobs

Data structure conversion

http://erlport.org/

https://github.com/arthurcolle/elixir-snake

Inter-process communication: Erlang protocol libraries

Turn your code into an Erlang node

Protocol libraries for for C, Java and .NET

Reasonably good performance, still serialization overhead

Inter-process communication: Standard protocols

HTTP

AMQP

ZeroMQ

...

Over the Air Updates

Quadcopter In-flight Firmware Upgrade

https://www.youtube.com/watch?v=96UzSHyp0F8

Building embedded systems

- Erlang Releases
  - Combine VM and libraries used by the app
  - Handle hot code updates
- Watchdog
  - Erlang VM has its own supervisor
  - Start VM from /etc/inittab and you are done

Nerves

http://nerves-project.org/

https://hexdocs.pm/nerves/getting-started.html

Linux Kernel + Erlang VM + goodies

Erlang VM as init / PID 1

Nerves Modules

Configure network interfaces

Connect to WiFi networks

Use serial ports

Drive LEDs

Interface with input events /dev/input/event

Over-the-network firmware management

Simple Service Discovery Protocol (SSDP) Client and Server

Nerves Howto: Install Nerves

```
mix archive.install https://github.com/nerves-project/archives/raw/master/nerves_bootstrap.ez
```

Nerves Howto: Generate and compile just like any Elixir project

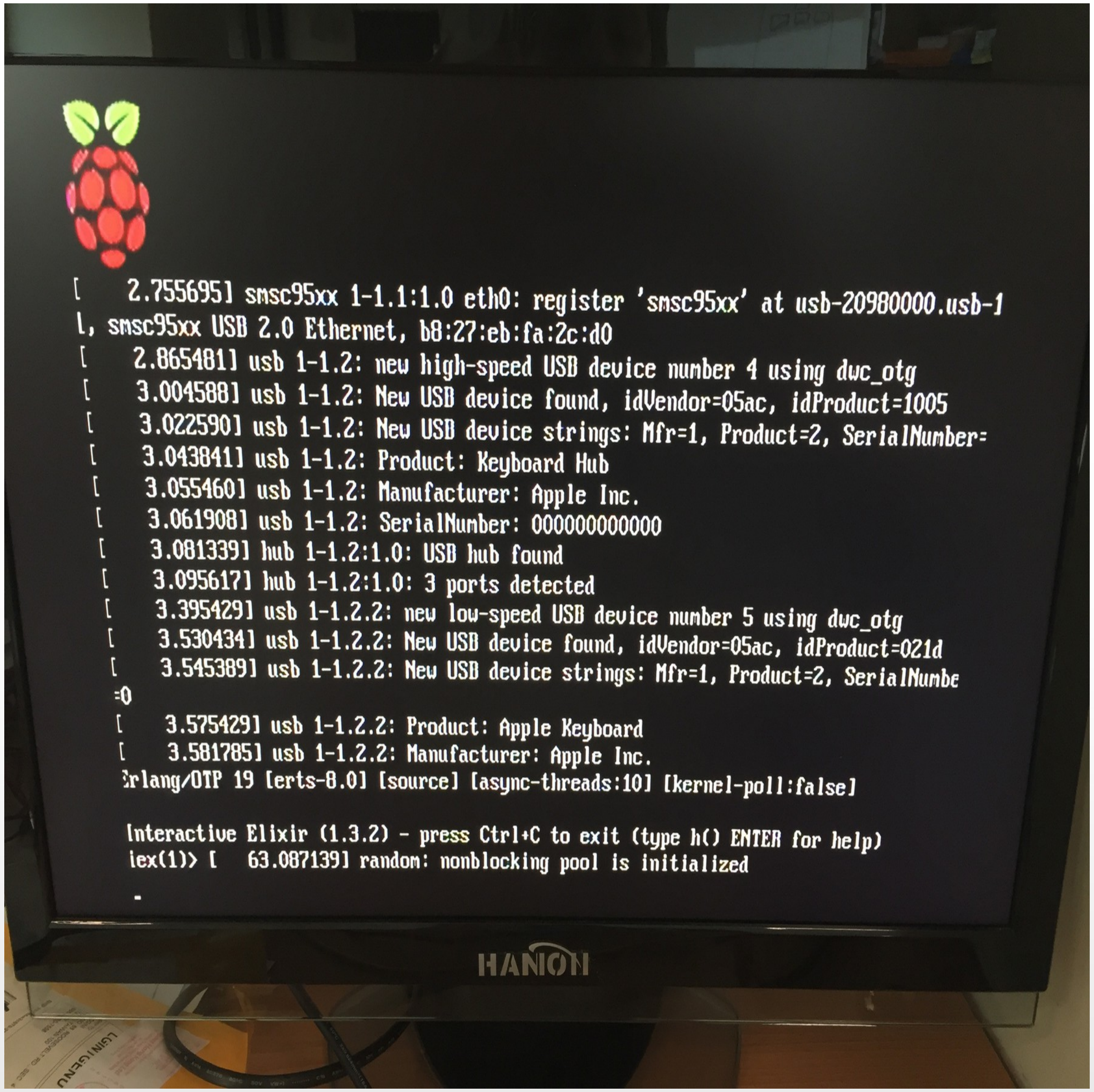
```
mix nerves.new hello_nerves --target rpi3
cd hello_nerves
mix deps.get
mix compile
```

Nerves Howto: Build your firmware and burn it to an SD card

mix firmware

mix firmware.burn

Nerves Howto



```
[    2.755695] smsc95xx 1-1.1:1.0 eth0: register 'smsc95xx' at usb-20980000.usb-1
1, smsc95xx USB 2.0 Ethernet, b8:27:eb:fa:2c:d0
[    2.865481] usb 1-1.2: new high-speed USB device number 4 using dwc_otg
[    3.004588] usb 1-1.2: New USB device found, idVendor=05ac, idProduct=1005
[    3.022590] usb 1-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=
[    3.043841] usb 1-1.2: Product: Keyboard Hub
[    3.055460] usb 1-1.2: Manufacturer: Apple Inc.
[    3.061908] usb 1-1.2: SerialNumber: 000000000000
[    3.081339] hub 1-1.2:1.0: USB hub found
[    3.095617] hub 1-1.2:1.0: 3 ports detected
[    3.395429] usb 1-1.2.2: new low-speed USB device number 5 using dwc_otg
[    3.530434] usb 1-1.2.2: New USB device found, idVendor=05ac, idProduct=021d
[    3.545389] usb 1-1.2.2: New USB device strings: Mfr=1, Product=2, SerialNumbe
=0
[    3.575429] usb 1-1.2.2: Product: Apple Keyboard
[    3.581785] usb 1-1.2.2: Manufacturer: Apple Inc.
Erlang/OTP 19 [erts-8.0] [source] [async-threads:10] [kernel-poll:false]

Interactive Elixir (1.3.2) - press Ctrl+C to exit (type h() ENTER for help)
iex(1)> [   63.087139] random: nonblocking pool is initialized
```

Questions?