PocketBeagle Walk Through
PocketBeagle walk-through

Friendly to novices and experts alike, the Beagle experience tracks mainline u-boot, Linux and Debian development, while augmenting it to enable development to start as quickly as possible. Attendees will get started interacting with the hardware via the command-line, shell scripts, Python and JavaScript. Attendees will be walked through the configuration details for the boot configuration, pin multiplexing, USB networking and other helper scripts they should get to know. Support and development processes within the BeagleBoard.org community will be covered. Exercises will pave the way for the other workshops to dive into their topic without needing to backtrack excessively on PocketBeagle-specific details.
Author and license

• Author
  - Jason Kridner
    Co-founder BeagleBoard.org, Texas Instruments Sitara apps
    https://beagleboard.org/about

• License
  - Creative Commons Attribution – Share Alike 4.0
    https://creativecommons.org/licenses/by-sa/4.0/
Outline

• BeagleBoard.org, PocketBeagle and BaconBits

• Developer experience
  – Command-line and shell script
  – JavaScript and Python
  – C/C++
  – C on PRUs

• Project examples

• Labs
BeagleBoard.org’s objectives

- **Education**
  - Design and use of open source SW/HW
  - Embedded computing

- **Collaboration**
  - Physical computing
  - Robotics
  - Industrial/machine controls
Inspiration from early PCs

http://www.sandywalsh.com/2012_07_01_archive.html
Inspiration from early PCs

- How do people learn about embedded computers with so much ground to cover?

http://www.sandywalsh.com/2012_07_01_archive.html
Inspiration from early PCs

• How do people learn about embedded computers with so much ground to cover?

  • Linux keeps history
  • Affordable -> hackable
  • Open from boot
  • High-level languages
  • Motivate with hardware

http://www.sandywalsh.com/2012_07_01_archive.html
Vision

• Creating with electronics should be as easy as creating a web page
Vision

• Creating with electronics should be as easy as creating a web page

• Appliances are better than applications
Vision

- Creating with electronics should be as easy as creating a web page
- Appliances are better than applications
- Open source software and hardware enable
  - Collaboration on the problem
  - Ability to understand and improve the fundamentals
Reality

- Boot-to-browser feels too limiting → booting to Debian distro
- Collaborative programming still complex → collaborate at the kernel
- Many possible development environments
  - command-line/ssh, Cloud9 IDE, node-red, pureData, SuperCollider, LabView, Matlab, Eclipse, Visual Studio, Scratch, Blockly
- Domain specific approaches
  - Machinekit/LinuxCNC, PLC, many IoT toolkits
  - Many rapid sensor approaches: capes, mikroBus, Grove/Grove Zero, PMOD
  - Many rapid build approaches: LEGO, printing/milling, Makeblock, Vex, various other aluminum kits
Approach

• Don’t try to boil the ocean
  − We seek to engage the open source community

• Help where we can
  − Blue supports Grove cables
  − PocketBeagle supports mikroBus click pinout
  − Many “BeagleBoard Compatible” devices targeting specific application areas
Board history

Fanless open computer (BeagleBoard) $249

Mint tin sized with industrial peripherals (BeagleBone) $69

Application focused BeagleBones $79

Smalls mint tin sized with super-flexible design - PocketBeagle $25
PocketBeagle objectives

- Get simple
  - 4-layer PCB done in both Kicad and EAGLE
  - Every expansion header pin has a useful predefined mode
- Get flexible
  - USB to holes, no on-board pin consumption, no header soldered
  - Support for 2 mikroBus Click boards (over 300 already exist)
- Get small
  - Stick with mint-tin survival-kit theme, but go to “smalls” (35mm x 55mm)
- Get low cost
  - System-in-package approach has can lower build costs
  - Launched/sustainable at $25
PocketBeagle key features

- **Processing**
  - 1-GHz ARM Cortex-A8 processor
  - 2x200-MHz programmable real-time units (PRUs)
  - ARM Cortex-M3 microcontroller for power and security
  - SGX530 graphics processor (OpenGLES)

- **Memory**
  - 512-MB DDR3
  - 4-KB I2C EEPROM

- **Interfaces**
  - USB 2.0 OTG
  - microSD
  - 72 expansion header pins
    - 8 analog inputs (6@1.8V, 2@3.3V)
    - 44 digital I/Os (18 enabled)
    - 3 UARTs (2 enabled)
    - 2 I2C ports
    - 2 SPI ports
    - 2 quadrature encoders accessible
    - 2 CAN bus controllers accessible
    - USB, power/reset buttons, battery/DC
PocketBeagle top

Octavo Systems OSD3358-SM

- Power LED
- Clock
- USB 2.0 OTG
- Power
- User LEDs
- P2 Header
- Boot Config
- microSD Card Slot
- P1 Header

PocketBeagle Rev A2
PocketBeagle bottom

P2 Header

P1 Header
mikroBus Click

- Analog - AN
- Reset - RST
- SPI Chip Select - CS
- SPI Clock - SCK
- SPI Master Input Slave Output - MISO
- SPI Master Output Slave Input - MOSI
- VCC-3.3V power - +3.3V
- Reference Ground - GND

- PWM - PWM output
- INT - Hardware Interrupt
- RX - UART Receive
- TX - UART Transmit
- SCL - I2C Clock
- SDA - I2C Data
- +5V - VCC-5V power
- GND - Reference Ground
Connecting mikroBus Clicks
BaconBits objectives

- Designed specifically for e-ale training
- Inspired by “Bacon Cape” by Dave Anders
  - Designed for similar purpose on BeagleBone
- Provides target for common embedded interfaces
  - SPI, I2C, GPIO, PWM, ADC, USB, serial
- Avoided buying several modules
BaconBits features

- USB-to-Serial micro B
- USB Host A with power
- Power and Reset buttons
- GPIO push button
- ADC potentiometer thumbwheel
- PWM tri-color LED
- SPI 2-digit 7-segment display
- I2C accelerometer
BaconBits GPIO button

![Schematic diagram of GPIO button with connections to +3.3V, 4.7K resistor, SW1 BTN, and GND at P2_33.]

USER BUTTON
BaconBits ADC thumbwheel

Diagram showing connections:
- RV1 3352T-1-502LF
- ADC_PWR
- ANALOG_IN
- ADC_GND
- P1_18
- P1_19 (AIN0)
- P1_17

Diagram text: THUMBWHEEL
BaconBits RGB LED
BaconBits RGB LED
BaconBits SPI 7-segment display

EMULATED 7 SEGMENTS
PocketBeagle TechLab Cape

- Designed to be compatible with BaconBits
- beagleboard.org/techlab
Developer experience

• Customized Debian images – bbb.io/latest
• Self-hosted tools for ARMs and PRU
• Libraries for various high-level languages
• Scripts for common tasks
• Sources for bootloader, device tree, etc.
• Servers for network-based development
Single cable development

- Power, network, develop
- You can add a network and power many other ways
Some work in progress

- Add proxy for various services (in Buster IoT images today)
- Integrate common web-based WiFi provisioning
  - SeeedStudio BeagleBone Green Wireless ships with ‘wifidog’ → we will unify approach
- Cross-platform distro installer app
  - See USB NETCONSOLE presentation
- Support for Grove modules and mikroBus clicks
  - Focus on device-tree overlays and kernel patches
- Integration alignment with complete domain solutions
  - Intelligent Agent Replicape/Revolve, Bela Mini, BeagleLogic, etc.
- Improved and integrated PRU examples
- Move to distro friendly approaches for customizations
Download image
Write image to microSD with Etcher
Insert microSD and boot
Connect to the USB network
Open the IDE
USB gadgets

- Linux name for device/slave drivers
  - i.e., when not host
- USB devices have “classes”
  - Mass storage
  - Camera
  - Audio
  - Printer
  - “HID” or human-interface device like mouse and keyboard
  - Communications
USB gadgets

- Default image USB gadgets
  - Virtual mass storage
    - Serves you up README.htm
  - Virtual serial
    - Provides access to console after kernel boot
  - Virtual network
    - Enables access to ssh and web servers
Boot summary

- ROM
- sysboot pins
- uSD @128k
- SPL
- uSD @384k
- UBoot
- /boot/uEnv.txt
- Device tree
- overlay
- initramfs
- Linux kernel
- overlay
- rootfs
- /dev/mmcblk0p1
- systemd
- service
In Linux, everything becomes a file

• Much to learn
  - I'm used to microcontrollers: just give me the datasheet with register definitions and set me free!

• Training on boot & device drivers useful
  - Often geared more at system bring-up
  - What about the everyday user?
  - Where is that abstraction benefit?

• Let's just walk a working system!
What is the baseline?

http://refspecs.linuxfoundation.org/lsb.shtml

• Every Linux system may be customized
  - This is the nature of open source
  - Stuff still needs to work together

• The Linux Standard Base
  - Umbrella for various Linux Foundation groups
  - A specification and a testkit
  - Documents typical libraries, functions and files expected to be found by the developer
debian@beaglebone:~$ sudo apt install -y lsb
debian@beaglebone:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Debian
Description: Debian GNU/Linux 9.5 (stretch)
Release: 9.5
Codename: stretch
Filesystem Hierarchy Standard

http://www.pathname.com/fhs/

- /tmp: temporary files
- /var: data that is changes at run-time
- /proc: "information from processes" (virtual)
- /sys: "system filesystem" (virtual)
- /dev: device files
- /media: mount point for removable media
- /lost+found: data without directory entry
- /mnt: mount point for temporary mounted file systems
- /opt: add-on application software packages
<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>UID</th>
<th>GID</th>
<th>Links</th>
<th>Permissions</th>
<th>Status</th>
<th>Type</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>fb</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>misc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sysvipc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>apm</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>filesystems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>modules</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>thread-self</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>asound</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mounts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>timer_list</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>buddyinfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>interrupts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mtd</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tty</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bus</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iomem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uptime</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>buddyinfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iomem</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>net</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uptime</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cmdline</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kallsyms</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pvr</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vmstat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>consoles</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>keys</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sched_debug</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>zoneinfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cpu</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>key-users</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>schedstat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cpuinfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kmsg</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>self</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>crypto</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kpagecgroup</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>slabinfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>devices</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kpagecount</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>softirqs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>device-tree</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>kpageflags</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>loadavg</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>swaps</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>driver</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>locks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sys</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>execdomains</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>meminfo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sysrq-trigger</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
debian@beaglebone:~$ cat /proc/cpuinfo
processor : 0
model name : ARMv7 Processor rev 2 (v7l)
BogoMIPS : 995.32
Features : half thumb fastmult vfp edsp thumbee neon vfpv3 tls vfpd32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x3
CPU part : 0xc08
CPU revision : 2
Hardware : Generic AM33XX (Flattened Device Tree)
Revision : 0000
Serial : 1741GPB42934
The file interface abstraction

• What can I do with files?
  - open, read, write, close, delete
  - What is an "ioctl"?
  - What is "mmap"?

• What is a virtual file system?
  - Looks like a file, but executes code in the kernel
  - Not really storing anything to media
  - A bit like a "ram disk"
Kernel.org documentation
http://www.kernel.org/doc/

• Documentation extracted from the Linux kernel and mirrored on the web where Google can find it:
  • Documentation - Text files in the kernel source tarball's Documentation subdirectory
  • htmldocs - Kernel Documentation maintained in docbook format (output of "make htmldocs")
  • Menuconfig - help text for each kernel configuration option (from kconfig source)
  • README various README files scattered around Linux kernel source
  • RFC - List of IETF RFCs referred to by kernel source files. Links to both the text of the RFC and the source files that refer to it
  • Output of kernel's "make help"

• Standards documents applicable to the Linux kernel
• Other web pages containing kernel documentation
• Translations to other languages
• Documentation on memory management
• Miscellaneous
Kernel Application Binary Interface

http://www.kernel.org/doc/Documentation/ABI/

• Low-level kernel interface from "userland"

• Status of interface
  • Stable
    • Encouraged to use freely
    • Guaranteed for at least two years
  • Testing
    • Mostly complete, but might change
    • Let developers know how you are using
    • Where you'll find most of the good stuff
  • Obsolete
    • Scheduled for removal
  • Removed
Kernel Application Binary Interface

Types of interfaces

- Syscalls
  - Trap interface with IDs
  - May be possible to have a direct entry

- SYSFS
  - Virtual file system
  - See also DEBUGFS and CONFIGFS

http://www.kernel.org/doc/Documentation/ABI/
Syscalls


- open/read/write/lseek/close/unlink
- ioctl
- mknod
- fork/select/poll/...
- mkdir/...
- mount/umount
- mmap
What is SYSFS?

- Virtual file system that exposes drivers to userspace
- `mount | grep sysfs`
  - `sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)`
- `/sys/devices` - driver hierarchy
- `/sys/bus` - symbolic links to bus owners
- `/sys/class` - common interfaces
- `/sys/block` - block interface
- How about some examples?
/sys/module


• /sys/module/MODULENAME
  - .../parameters: options you can provide
  - .../refcnt: number of times in use

debian@beaglebone:~$ ls /sys/module
8250          fb
apparmor       firmware_class
auth_rpcgss    fscrypto
block          module
bone_capemgr   hid
can            hid_logitech
cec            hid_logitech_hidpp
cfg80211       i2c_algo_bit
configfs       ima
cpufreq        iptable_filter
cpuidle        iptable_mangle
cryptomgr      iptable_nat
dns_resolver   ip_tables
drm            ipv6
drm_kms_helper ir_kbd_i2c
dvb_core       kernel
dynamic_debug  keyboard
EEPROM_93cx6    leds_pwm
ehci_hcd       libahci
etnaviv        libata
evdev          libcomposite

lockd          pruss          sysrq          usb_f_ecm
mma8452        pruss_intc     tcp_cubic      usb_f_mass_storage
mmcdnk         pruss_soc_bus   tda18271       usb_f_rndis
module         pvrsvkm        tda827x        usbhid
mt20xx         r8188eu        tda8290        usb_storage
netpoll
nf_conntrack
nf_conntrack_ipv4
nf_defrag_ipv4
nf_nat
nf_nat_ipv4
nf
nfs
nfs_layout_nfs41_files
nfsv4
omapdmr
omap_mailbox
onenand
overlay
pinctrl_mcp23s08
printk
pru_rproc
rc_core
rcupdate
rcutree
rfkill
rng_core
scsi_mod
tunable
snd
snd_pcm
snd_timer
spidev
spurious
srcutree
sunrpc
suspend
tearDown
usbhid
usbhid
usbhid
/sys/class/leds

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-class-led

- /sys/class/leds/LED
  - .../brightness: 0-max_brightness, >0 = on
  - .../max_brightness: default is 255
  - .../trigger: triggers available from kernel
  - .../inverted: invert on/off state
Must be explicitly exported to userspace and not claimed by kernel code

/sys/class/gpio
  - ../../../export: asks the kernel to export a GPIO to userspace
  - ../../../unexport: to return a GPIO to the kernel
  - ../../../gpioN: for each exported GPIO #N
    - ../../../value: always readable, writes fail for input GPIOs
    - ../../../direction: r/w as: in, out (low); write: high, low
    - ../../../edge: r/w as: none, falling, rising, both
  - ../../../gpiochipN: for each gpiochip; #N is its first GPIO
    - ../../../base: (r/o) same as N
    - ../../../label: (r/o) descriptive, not necessarily unique
    - ../../../ngpio: (r/o) number of GPIOs; numbered N to N + (ngpio - 1)
On-chip peripherals (OCP)

```
debian@beaglebone:~$ ls /sys/devices/platform/ocp
40300000.ocmcram  480c8000.mailbox  53100000.sham  ocp:P1_32_pinmux  ocp:P2_20_pinmux
44e07000.gpio     480ca000.spinlock  53500000.aes  ocp:P1_33_pinmux  ocp:P2_22_pinmux
44e09000.serial    481c0000.i2c       56000000.sgx  ocp:P1_34_pinmux  ocp:P2_24_pinmux
44e0b000.i2c      481a0000.spi       driver_override  ocp:P1_35_pinmux  ocp:P2_25_pinmux
44e0d000.tscadc    481a8000.serial    modalias  ocp:P1_36_pinmux  ocp:P2_27_pinmux
44e35000.wdt       481ac000.gpio     ocp:cape-universal  ocp:P2_01_pinmux  ocp:P2_28_pinmux
44e3e000.rtc       481ae000.gpio     ocp:14_wkup@44c0000  ocp:P2_02_pinmux  ocp:P2_29_pinmux
47400000.usb       481cc000.can      ocp:P1_02_pinmux  ocp:P2_03_pinmux  ocp:P2_30_pinmux
48022000.serial    481d0000.can      ocp:P1_04_pinmux  ocp:P2_04_pinmux  ocp:P2_31_pinmux
48024000.serial    48300000.epwmss    ocp:P1_06_pinmux  ocp:P2_05_pinmux  ocp:P2_32_pinmux
4802a000.i2c      48302000.epwmss    ocp:P1_08_pinmux  ocp:P2_06_pinmux  ocp:P2_33_pinmux
48030000.spi      48304000.epwmss    ocp:P1_10_pinmux  ocp:P2_07_pinmux  ocp:P2_34_pinmux
48042000.timer     48310000.rng      ocp:P1_12_pinmux  ocp:P2_08_pinmux  ocp:P2_35_pinmux
48044000.timer     49000000.edma      ocp:P1_20_pinmux  ocp:P2_09_pinmux  of_node
48046000.timer     49800000.tptc      ocp:P1_26_pinmux  ocp:P2_10_pinmux  power
48048000.timer     49900000.tptc      ocp:P1_28_pinmux  ocp:P2_11_pinmux  subsystem
4804a000.timer     4a326004.pruss-soc-bus  ocp:P1_29_pinmux  ocp:P2_17_pinmux  uevent
4804c000.gpio     4a326004.pruss-soc-bus  ocp:P1_30_pinmux  ocp:P2_18_pinmux
48060000.mmc      4c000000.emif      ocp:P1_31_pinmux  ocp:P2_19_pinmux
```
Reading events

• TBD
Instantiating a device: .dts example

&i2c0 {
    pinctrl-names = "default";
pinctrl-0 = <&i2c0_pins>;

    status = "okay";
clock-frequency = <400000>;

tps: tps@24 {
    reg = <0x24>;
};

    baseboard_eeprom: baseboard_eeprom@50 {
        compatible = "at,24c256";
        reg = <0x50>;

        #address-cells = <1>;
        #size-cells = <1>;
        baseboard_data: baseboard_data@0 {
            reg = <0 0x100>;
        };
    };
};

Pin muxing configuration
(routing to external package pins)

Enabling this device, otherwise ignored
Node property: frequency

List of devices on i2c0

I2C bus identifier

From arch/arm/boot/dts/am335x-boneblue.dts
Pin multiplexing

- Modern SoCs have too many hardware blocks compared to physical pins exposed on the chip package.
- Therefore, pins have to be multiplexed.
- Pin configurations are defined in the Device Tree.
- Correct pin multiplexing is mandatory to make a device work from an electronic point of view.
Device Tree

- See kernel documentation for bindings
  - devicetree/bindings/eeprom/eeprom.txt
- Local copies enable you to extend on the fly
  - /opt/source/dtb-4.9-ti
  - /opt/source/bb.org-overlays
- Overlays loaded in u-boot, but also possible via kernel configfs
## DT pin definitions

```plaintext
&am33xx_pinmux {
    ...
    i2c0_pins: pinmux_i2c0_pins {
        pinctrl-single,pins = <
            AM33XX_IOPAD(0x988, PIN_INPUT_PULLUP | MUX_MODE0)       /* (C17) I2C0_SDA.I2C0_SDA */
            AM33XX_IOPAD(0x98c, PIN_INPUT_PULLUP | MUX_MODE0)       /* (C16) I2C0_SCL.I2C0_SCL */
        >;
    };
    ...
}
...  
&i2c0 {
    pinctrl-names = "default";
    pinctrl-0 = <&i2c0_pins>;

    status = "okay";
    clock-frequency = <400000>;
    ...
};
```

<table>
<thead>
<tr>
<th>Package pin</th>
<th>SoC signal name</th>
<th>Allows to select a given SoC signal</th>
<th>Package pin name</th>
<th>Configures the pin: input, output, drive strength, pull up/down...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

From arch/arm/boot/dts/am335x-boneblue.dts
DT: matching devices and drivers

Platform drivers are matched with platform devices that have the same compatible property.

```c
static const struct of_device_id omap_i2c_of_match[] = {
    {
        .compatible = "ti,omap4-i2c",
        .data = &omap4_pdata,
    },
    ...
};
...
static struct platform_driver omap_i2c_driver = {
    .probe          = omap_i2c_probe,
    .remove         = omap_i2c_remove,
    .driver         = {
        .name   = "omap_i2c",
        .pm     = OMAP_I2C_PM_OPS,
        .of_match_table = of_match_ptr(omap_i2c_of_match),
    },
};
```

From drivers/i2c/busses/i2c-omap.c
debian@beaglebone:~$ config-pin -i p1.36
Pin name: P1_36
Function if no cape loaded: pwm
Function if cape loaded: default gpio gpio_pu gpio_pd
gpio_input spi_sclk pwm pruout pruin
Function information: ehrpwm0a default gpio3_14 gpio3_14
gpio3_14 gpio3_14 spi1_sclk ehrpwm0a pru0_out0 pru0_in0
Kernel GPIO id: 110
PRU GPIO id: 142
debian@beaglebone:~$ config-pin -q p1.36
P1_36 Mode: default Direction: in Value: 0
debian@beaglebone:~$ config-pin p1.36 pruout
debian@beaglebone:~$ config-pin -q p1.36
P1_36 Mode: pruout
show-pins.pl

- perl /opt/scripts/device/bone/show-pins.pl -v
Enabling PRU

- 2 possible drivers: remoteproc or uio
- Enabled via device tree at boot
  - Different systems might have different defaults
mikroBus Click usage

- See bbb.io/pbmb
- Supported with device-tree overlays loaded in u-boot
Demonstrations

- BaconBits demo

-
Some current projects

- Bela Mini
- PocketPilot
Contributions and issues

• Cape/add-on support
  – https://github.com/beagleboard/bb.org-overlays

• Image deltas
  – https://github.com/beagleboard/image-builder

• In-system examples
  – https://github.com/beagleboard/bone101
Questions?
Thank you!