ZIO I/O framework for Linux
ZIO - The Ultimate I/O Framework

Federico Vaga
What is ZIO?

- an I/O framework for the Linux kernel
  - COMEDI: old buggy unsupported framework
  - IIO: modern over-engineered framework
- centralize all complexity within the framework
- fast driver development
ZIO Winning Points

- concurrent I/O on several channels
- very flexible abstraction
- optimization for large data blocks
- huge number of peripherals
- support for off-line analysis
- support for off-line generation of output data
Device types

- ZIO supports 3 kind of I/O: digital, analogue, and time
  - Input: collect data at a specific time or event
  - Output: drive waveforms at a specific time or event
  - TDC: returns the timestamp of an input pulse
  - DTC: outputs a pulse at a predefined time stamp
Block

- all data transfers happen by blocks
- it includes both data and meta-data (control)
- data within ZIO never travels without meta-data
ZIO I/O framework for Linux

ZIO Block

Meta-data

Meta-data (control)

- 512 byte fixed structure
- sequence number
- data information
  - number of samples
  - sample size, active bits and alignment
  - endianess
- source information
  - device name, cset and channel index
  - device configuration
- trigger information
  - trigger name
  - trigger configuration
  - trigger time-stamp (input or output)
ZIO timestamp

- suggestion: seconds, nano-seconds and custom
- each driver can do what it want
- user-space make sense of timestamp with meta-data

```c
struct zio_timestamp {
    uint64_t secs;
    uint64_t ticks;
    uint64_t bins;
};
```
Input Flow

user space

file operations

buffer

trigger

driver

device

read

retr_block

alloc_block

store_block

input_cset

data_done

raw transfer

free_block
Output Flow

user space  file operations  buffer  trigger  driver  device
write  alloc_block  store_block  retr_block  output_cset  raw transfer
  data_done  free_block
Current Status of ZIO

Implementation status:
- 4 different drivers (virt., GPIO, UART, ADC)
- 3 trigger types (IRQ, timer, transparent)
- 2 generic buffer type (kmalloc, vmalloc)

Future driver implementations (end April)
- fine delay
- 100MSPS ADC 4-channel