PSCDRV
an EPICS driver toolkit for FPGA designers

Michael Davidsaver
NSLS2 BNL

The Problem

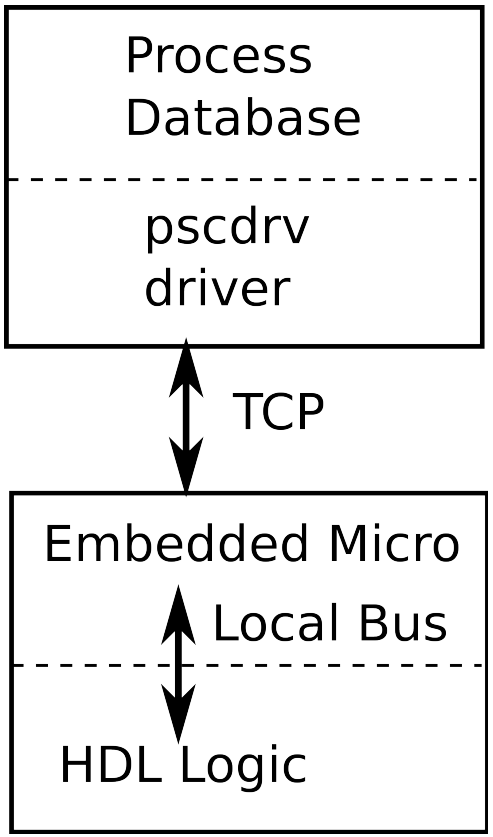
- PGA designers focus on PGA design...
  - Controls integration is second thought
- Ideally PGA designer works with programmer
  - Reality: Never enough engineers
- NSLS2 PGA developments
  - (ramping) Power Supply Controller
  - electron Beam Position Monitor
  - Cell Controller (fast orbit control network node)
    - Active Interlock

The Result

- Different EPICS driver for each application
  - asynRecord + aSub records
  - modified modbus driver
- Problems
  - Reliability and error handling
    - TCP connection management
      - Restart IOC+reset HW
  - Performance
    - Single duplex (request/response)
    - Under powered MAC (Xilinx Spartan 5)

How to Improve?

- Parts of a PGA system
  - Logic (HDL), embedded micro (C), and IOC
- IOC In FPGA?
  - One IOC per device
    - lots of files, lots of sockets
  - Consumes FPGA resources
    - Designs expand to fill available space



How to Improve? (2)

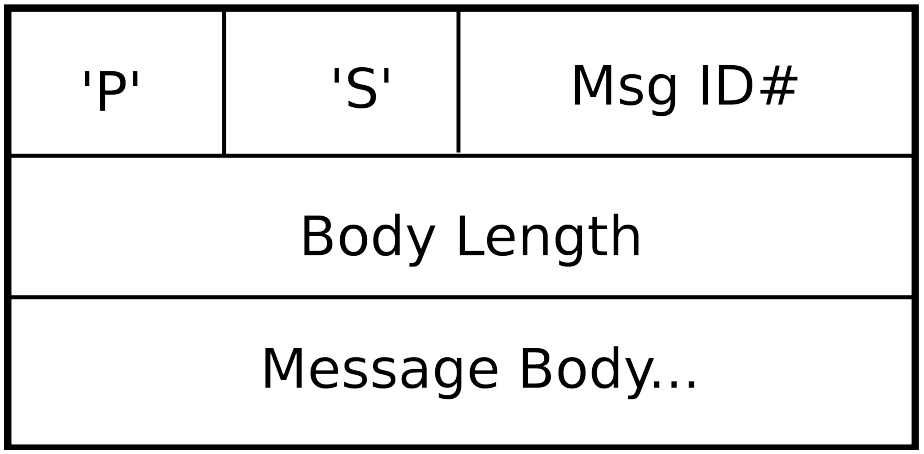
- PGA designers don't like C
  - Tried to use streamdevice
- Need to be fast and handle array data
- Make PGA designer self-sufficient

pscdrv overview

- An EPICS driver which is a TCP client
- PSC = Portable Streaming Controller
- Speaks a custom and semi-configurable protocol
  - **Not** request/response
  - Sync. settings from server (device)
- Values are (un)packed from binary messages into PDB records.
  - scalar/array values and HW timestamps

PSC Container Protocol

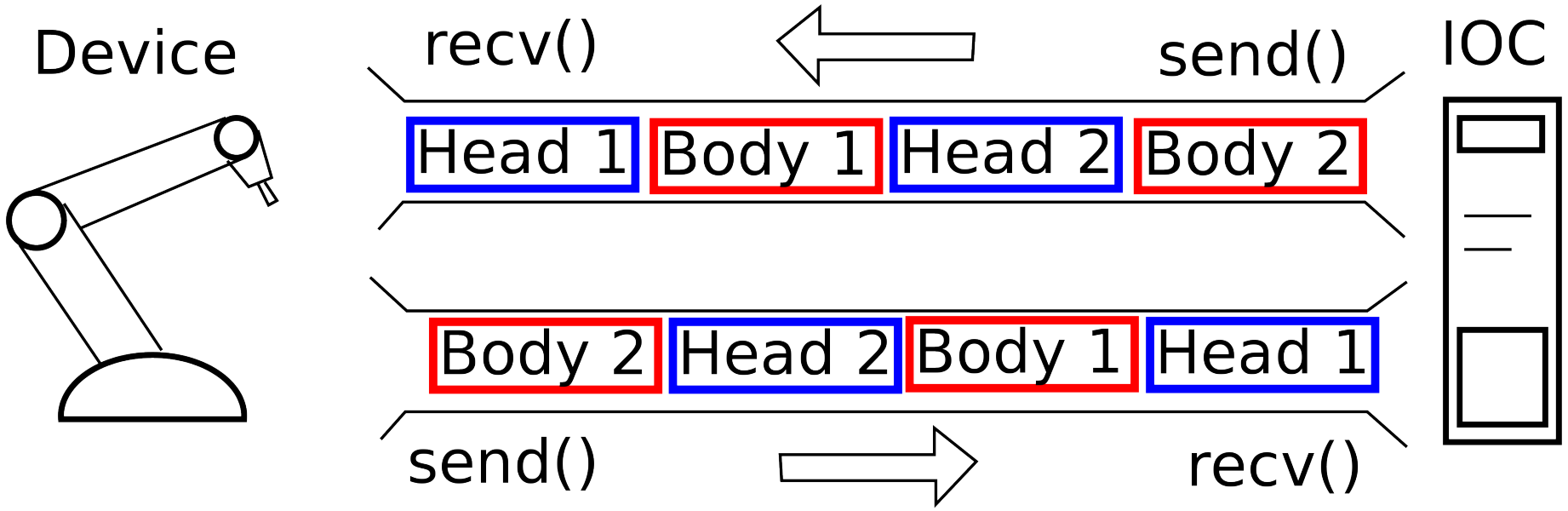
- TCP stream is a series of binary messages
  - Fixed 8 byte header w/ variable length body



- Message body is determined by configuration

Streaming

Device to IOC stream and IOC to Device stream are independent.



Not request response

IOC setup

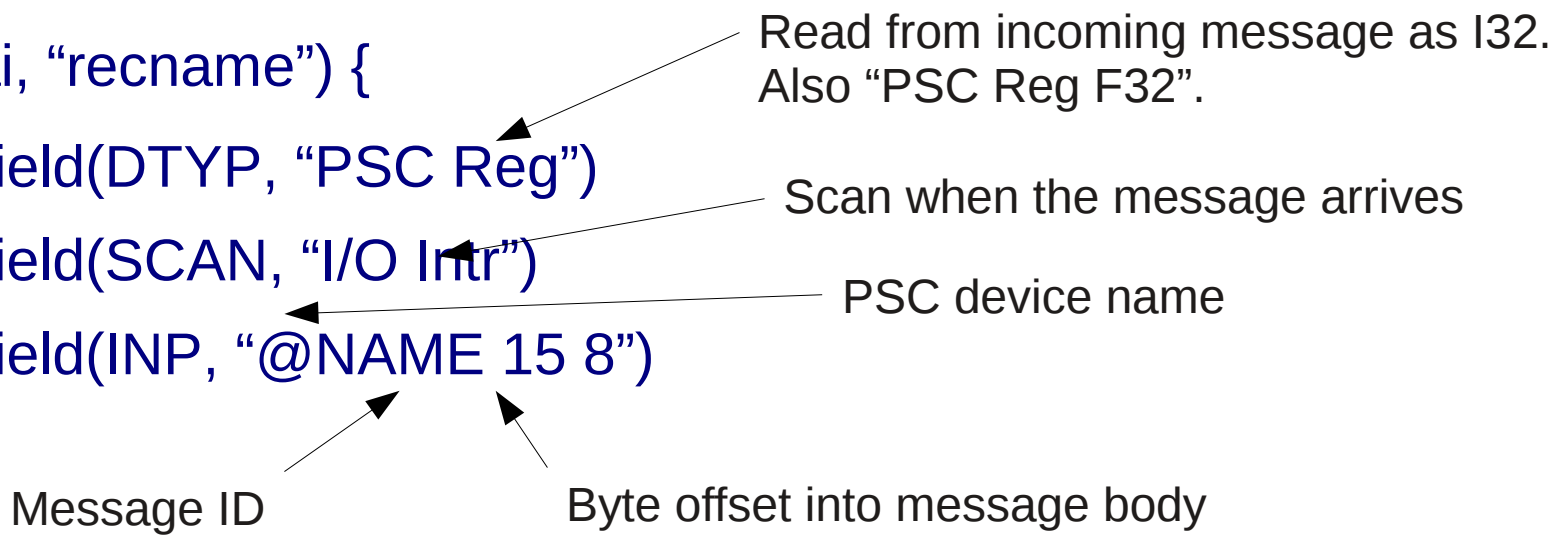
- In IOC start script

  createPSC(“NAME”, “10.0.0.1”, 4321, 1)

  setPSCSendBlockSize(“NAME”,20,32)

Reading a Scalar

```
record(ai, “recname”) {
        field(DTYP, “PSC Reg”)
        field(SCAN, “I/O Intr”)
        field(INP, “@NAME 15 8”)
}
```



- When a message with ID #15 arrives.
- Extract 4 bytes starting at offset 8
- Interpret as a 32-bit signed MSB integer

Other operations

- (Un)pack many scalar values from a message
  - A block of registers which are all read/written together
  - 32-bit integer, 32 and 64-bit IEEE floating point
- Send single scalar values with an address
  - Address is 4 byte sub-header
  - IOC to device for settings
  - device to IOC to re-sync.

info(“SYNC”,”SAME”)

Other operations (2)

info(“TimeFromBlock”,”12”)

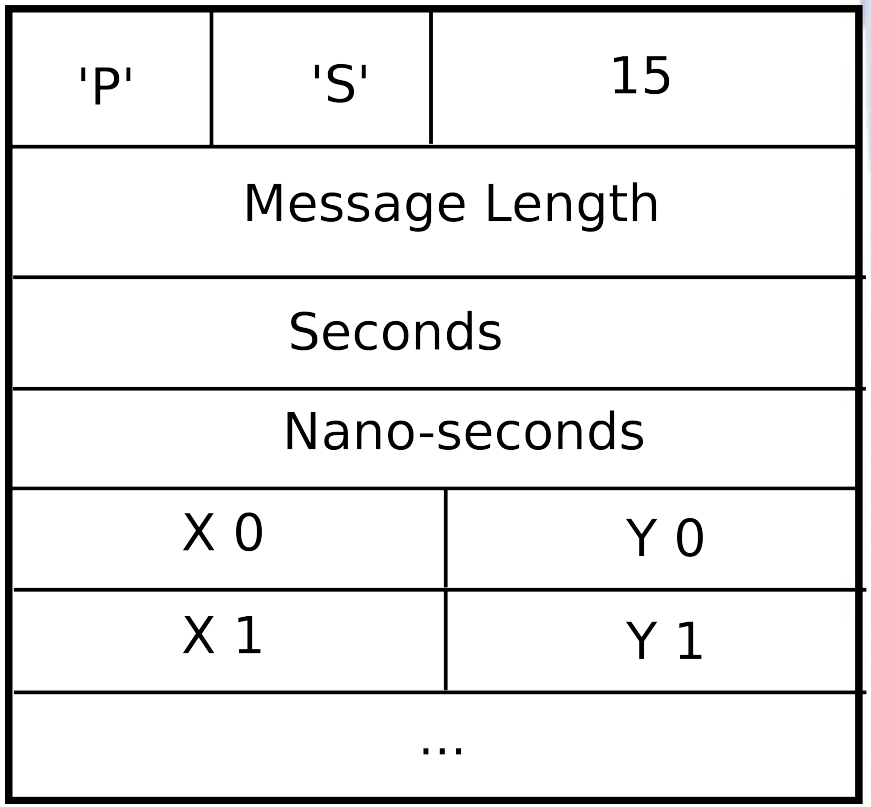
- Extract record timestamp from message
  - 2x 32-bit integers sec+ns (posix epoch)
- (Un)pack array data
  - Variable length
  - Contiguous or interleaved
  - Integer: 8, 16, 32 Float: 32, 64

field(INP, “@NAME 15 8 8”)

# of bytes between array elements including element size.

Array Example

```
record(waveform, “wf:X”) {
        field(“DTYP”,”PSC Block I16 In”)
        field(“SCAN”,”I/O Intr”)
        field(“FTVL”,”DOUBLE”)
        field(“NELM”,”1024”)
        field(“INP”,”NAME 15 8 4”)
        info(“TimeFromBlock”,”0”)
}
```



End

- Semi-generic TCP protocol and EPICS driver
- Intended to enable PGA designers to build fast and reliable IOCs.
- Future work
  - targetApp reference implementation of a PSC server

http://mdavidsaver.github.io/pscdrv/

Teasers

- carchivetools – Archive clients
  - https://github.com/epicsdeb/carchivetools
- pyDevSup - device support in python
  - http://mdavidsaver.github.io/pyDevSup/
- alarmmailer – email alarm aggregation
- cashark – wireshark dissector for CA
- ioclogserv2 – log server w/ rotation and filter
- cahtml – CA aware django templates