José Gabadinho :: Control Systems Specialist :: Paul Scherrer Institut

Lua IOC Support

AEK Controls Talk 08.Apr.2019

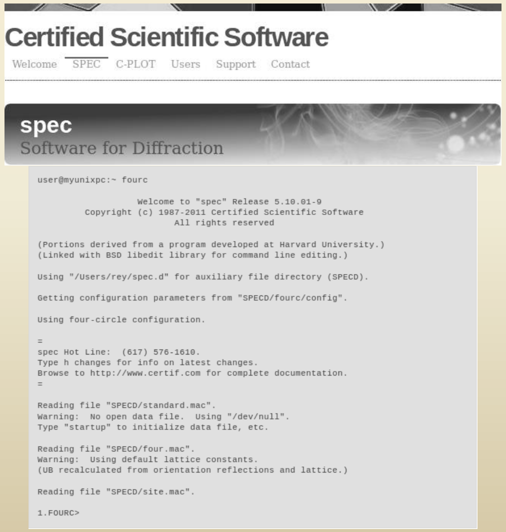
PAUL SCHERRER INSTITUT

ESRF vs. PSI/SLS

Before…

ESRF

The European Synchrotron



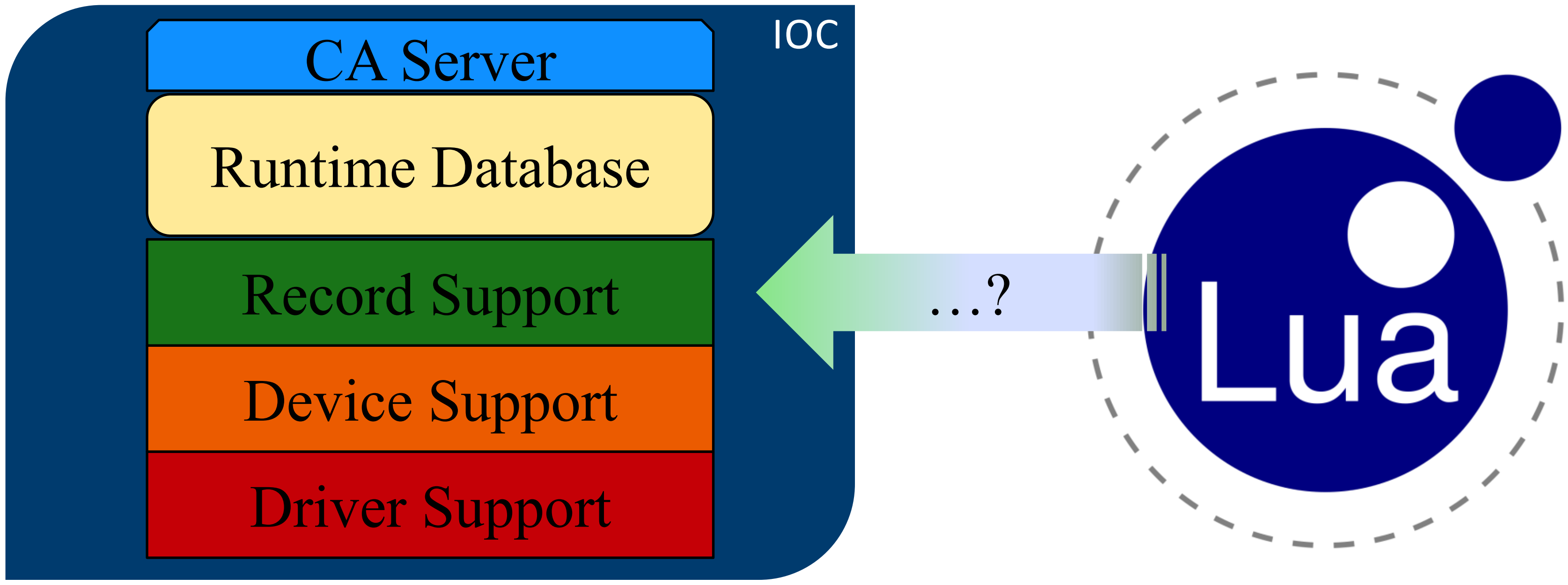
```
user@myunixpc:~ fourc

                   Welcome to "spec" Release 5.10.01-9
         Copyright (c) 1987-2011 Certified Scientific Software
                          All rights reserved

(Portions derived from a program developed at Harvard University.)
(Linked with BSD libedit library for command line editing.)

Using "/Users/rey/spec.d" for auxiliary file directory (SPECD).

Getting configuration parameters from "SPECD/fourc/config".

Using four-circle configuration.

=
spec Hot Line:  (617) 576-1610.
Type h changes for info on latest changes.
Browse to http://www.certif.com for complete documentation.
=

Reading file "SPECD/standard.mac".
Warning:  No open data file.  Using "/dev/null".
Type "startup" to initialize data file, etc.

Reading file "SPECD/four.mac".
Warning:  Using default lattice constants.
(UB recalculated from orientation reflections and lattice.)

Reading file "SPECD/site.mac".

1.FOURC>
```

Now!

SLS

EPICS

Embedding Lua into an EPICS IOC



Asynchronous Processing

Problem:

- Long or blocking process code hijacks EPICS scan thread

EPICS standard solution:

- Within record processing code: use PACT field to decide between starting and finishing of long/blocking code
- At the end of long/blocking code: must programmatically re-process the record

L.I.S. execution:

- Return a function+parameters
  - Executed by a callback thread
  - Re-processing is handled by L.I.S.
- Shortcut to PACT: *rec.pact(<set>)*
- Function can return extra parameters back to record processing function

Asynchronous Processing

Problem:

- Long or blocking process code hijacks EPICS scan thread

EPICS standard solution:

- Within record processing code: use PACT field to decide between starting and finishing of long/blocking code
- At the end of long/blocking code: must programmatically re-process the record

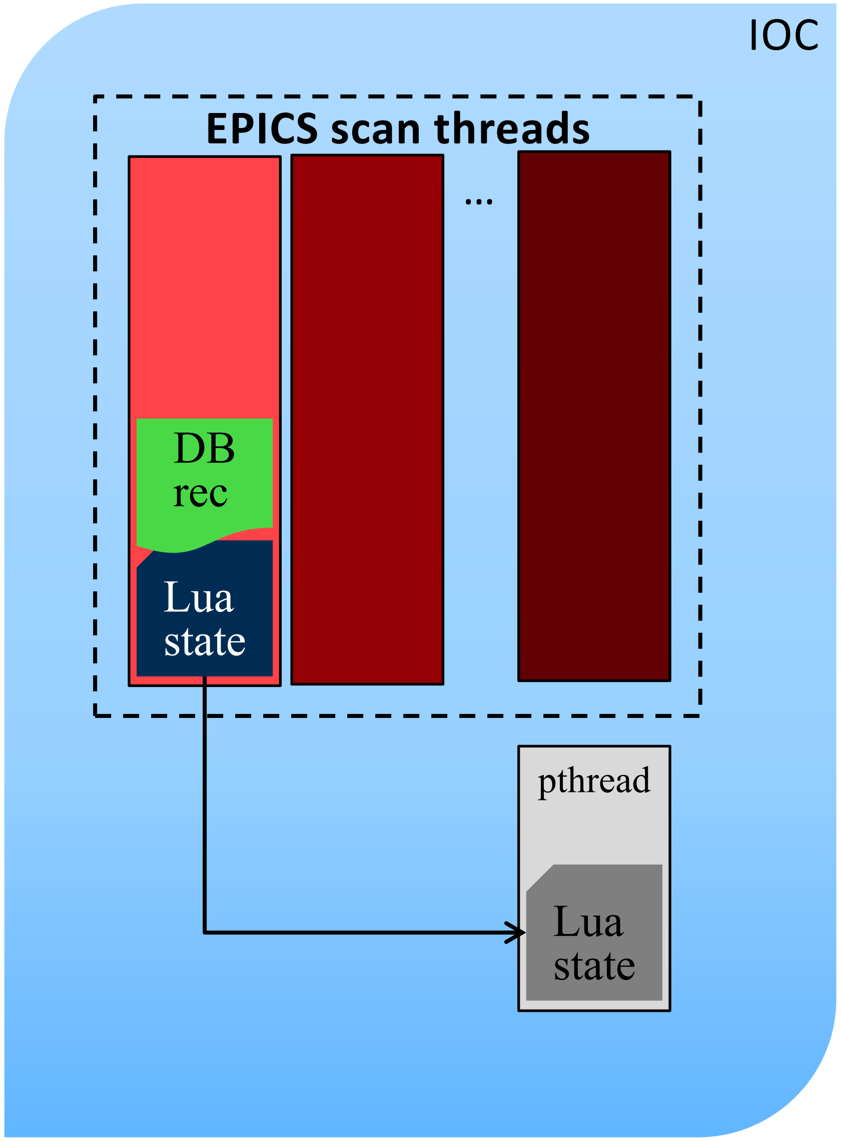
L.I.S. execution:

- Return a function+parameters
  - Executed by a callback thread
  - Re-processing is handled by L.I.S.
- Shortcut to PACT: *rec.pact(<set>)*
- Function can return extra parameters back to record processing function

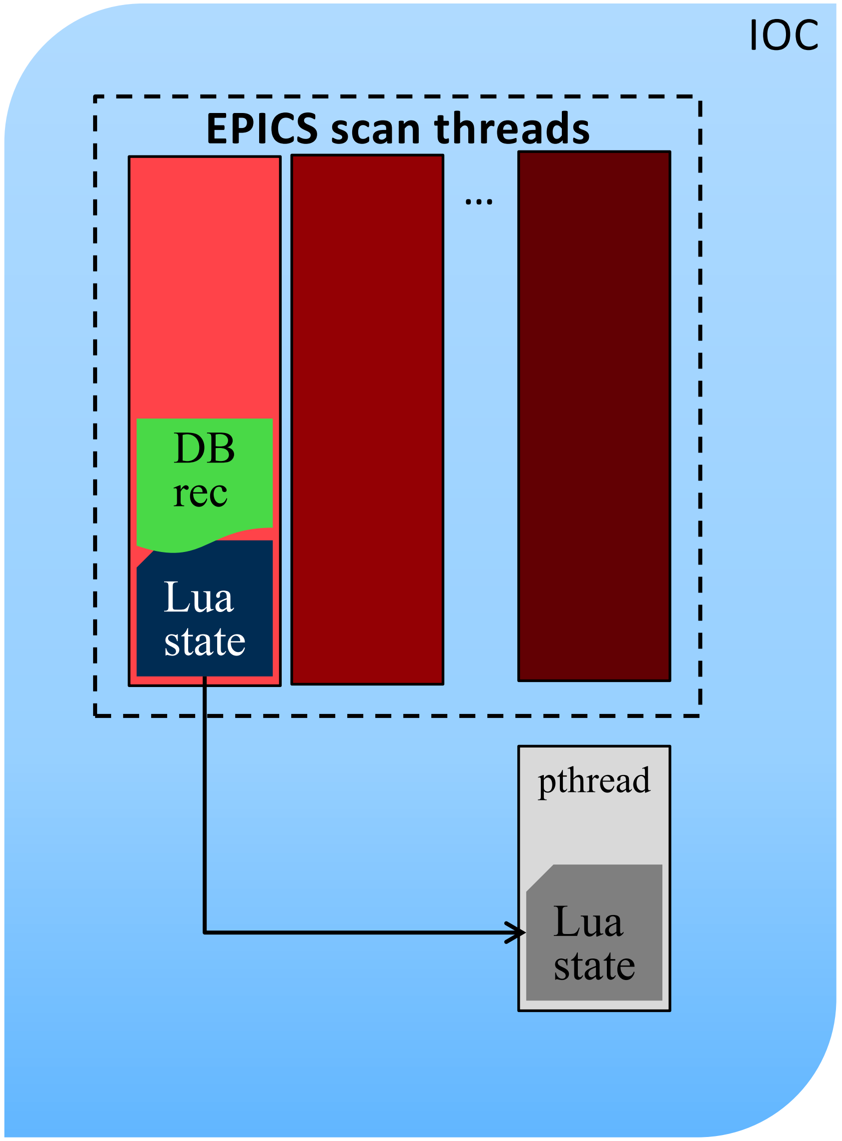
```
function my_proc(rec, proc_inp)
  if rec.pact() == 0 then
    rec.pact(1)
    return 0, long_proc, ”ping”
  end
  if proc_inp ~= nil then
    -- Do something fast
  end
  rec.pact(0)
end

function long_proc(rec, cb_inp)
  -- Do something slow
  return “pong”
end
```

PAUL SCHERRER INSTITUT PSI

Preemptive Worker Threads



Preemptive Worker Threads



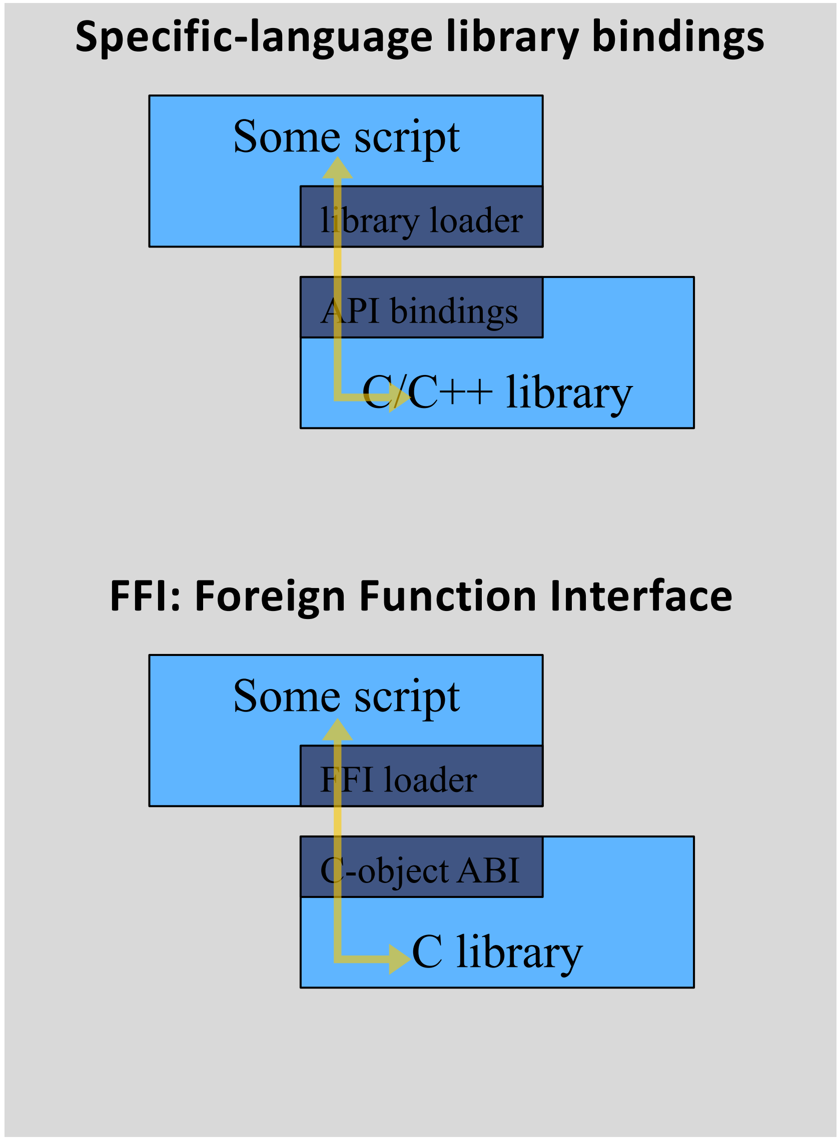
Record Lua state:

- Created by L.I.S.
- Scheduling handled by IOC
  - Executed in an EPICS scan (or callback) thread
- Interface to IOC in *luaiocsup* table

Worker thread Lua state:

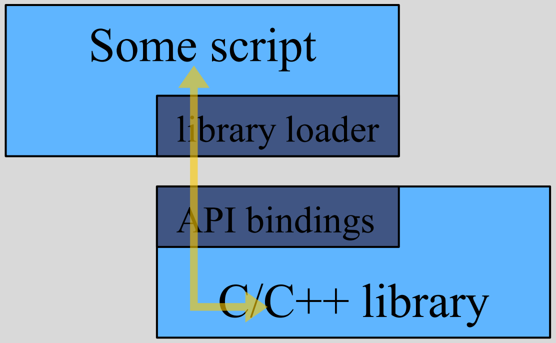
- Created by user's Lua code (suggested library: *lua-llthreads2*)
  - Requirement: allowing an initialization routine, for L.I.S. API bindings
- Scheduled preemptively by O.S.
- I/O Intr interface to IOC:
  - *luaiocsup.scanio_request(<ioscanid>, <params...>)*

API Bindings *vs.* Foreign Function Interface

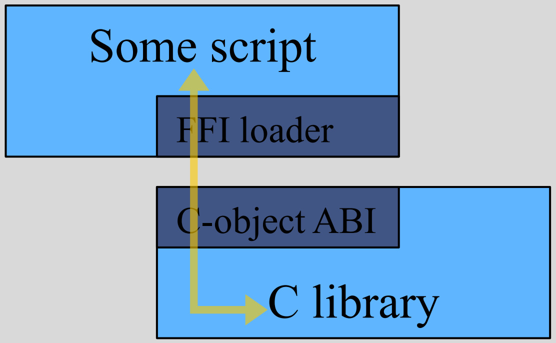


API Bindings *vs.* Foreign Function Interface

Specific-language library bindings



FFI: Foreign Function Interface



<u>GNU Scientific Library + Google-fu</u>

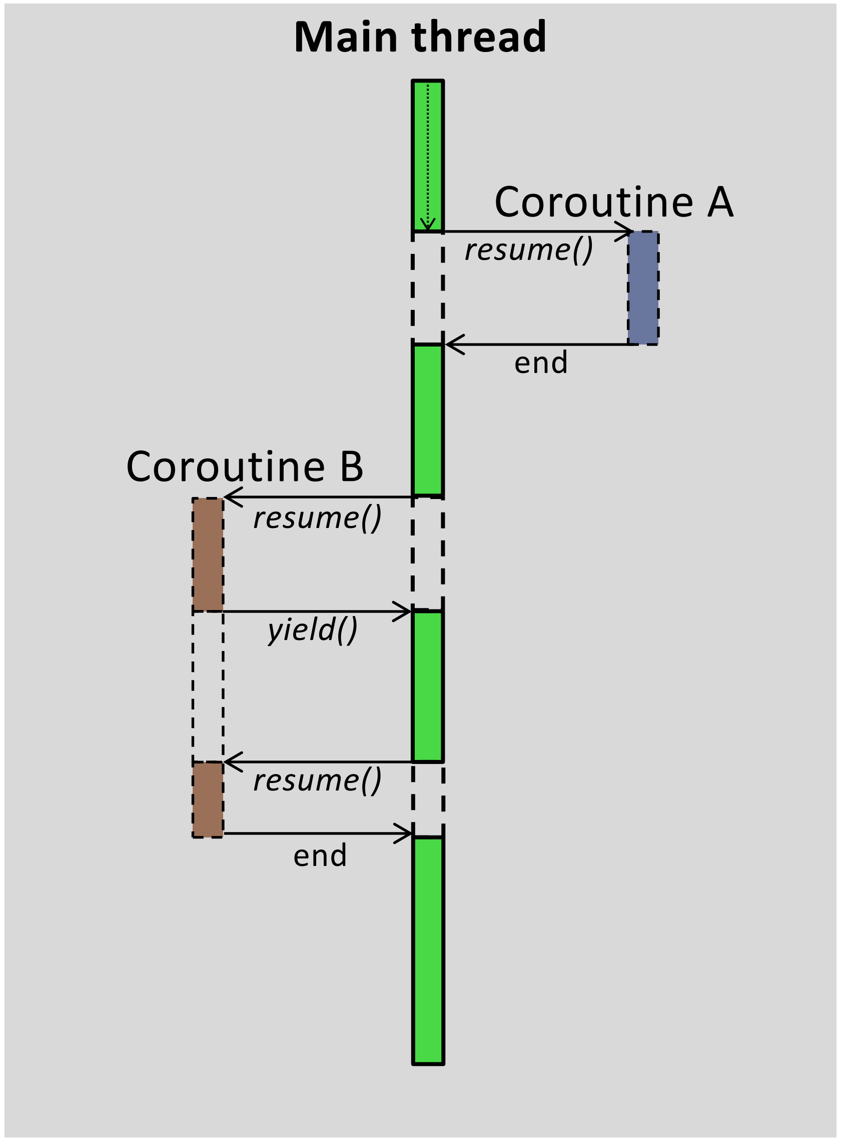
- *libwffit.so*

```
int wf_fit(
       int datapoints,
       float xpoints[],
       float ypoints[],
       double *center,
       double *peak);
```

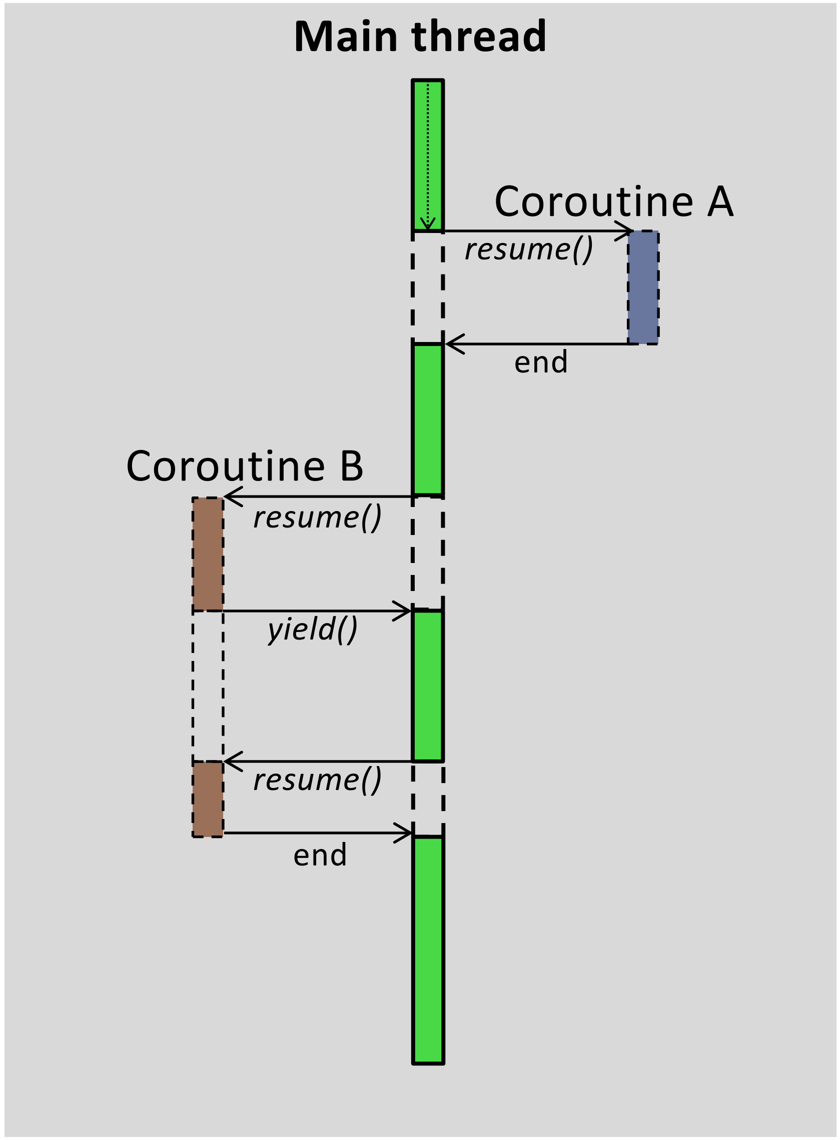
- *testwffit.lua*

```
alien=require “alien”
fitso=alien.load(“libwffit.so”)
fitfn=fitso.wf_fit
fitfn:types{“int”, “pointer”,
“pointer”, “ref double”, “ref
double”, ret=“int”}
```

Cooperative Threads



Cooperative Threads



<u>EPICS usage:</u>

- Scan thread as main thread, Lua record code as coroutine
- Explicit yield points within Lua code
- At each (re-)processing:
  - Input values are updated
  - Lua code is resumed

Implemented Features

Device support:

✓ai,ao,bi,bo,mbbi,mbbo,stringin,stringout,longin,longout,waveform

Record support:

✓luasub

Non-periodic record scanning:

✓Asynchronous callback for slow processing

✓I/O interrupt

✓Event

Others:

✓Extended device support, allowing runtime changes of INP/OUT fields

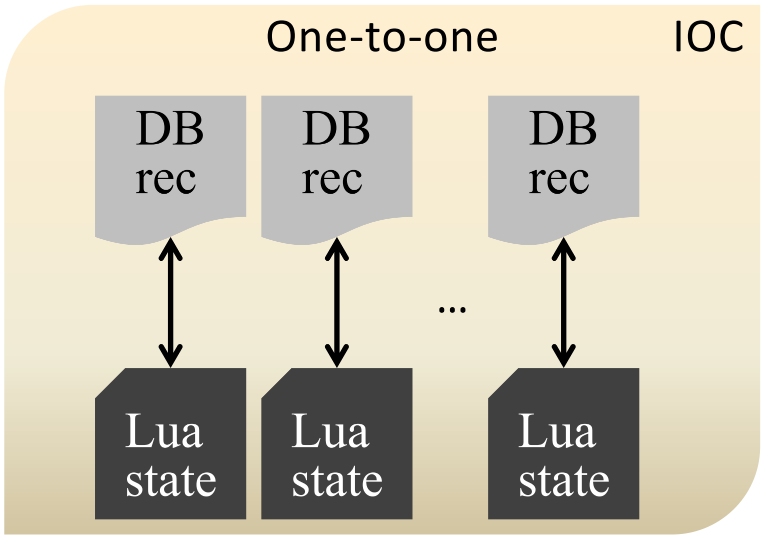
✓Worker-threads (via library)

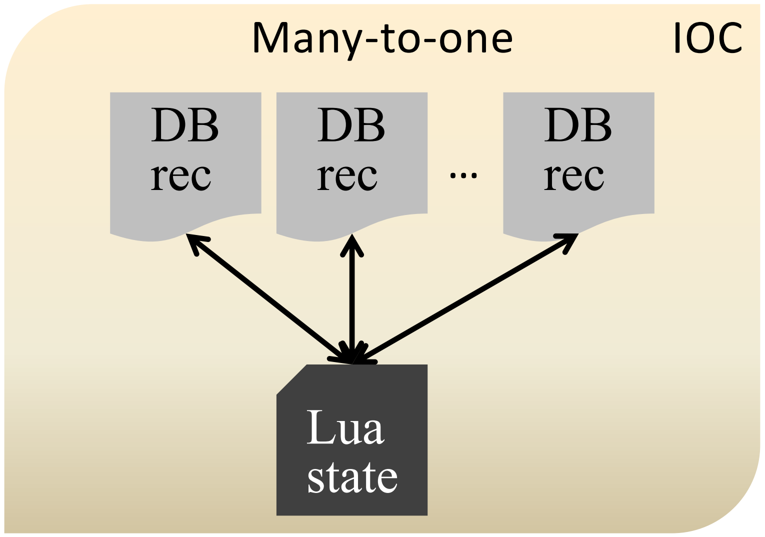
✓Cleanup at-exit callbacks

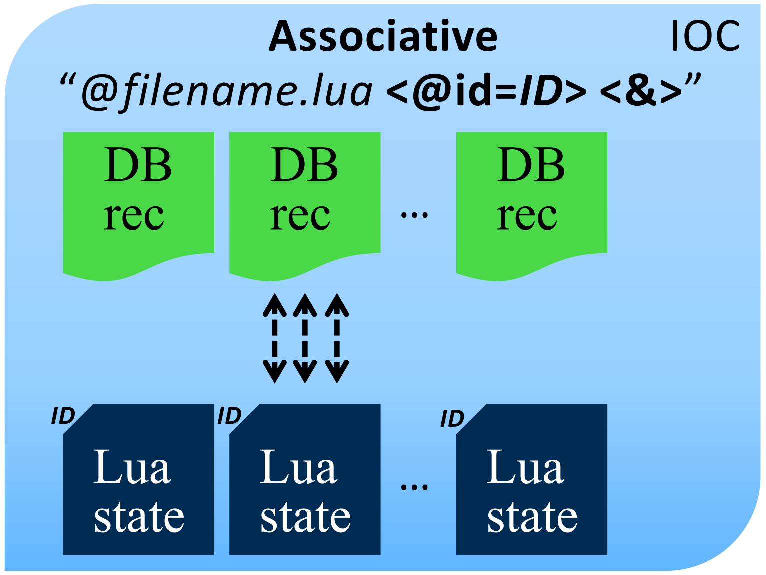
✓Report via *dbior* iocsh command

PAUL SCHERRER INSTITUT

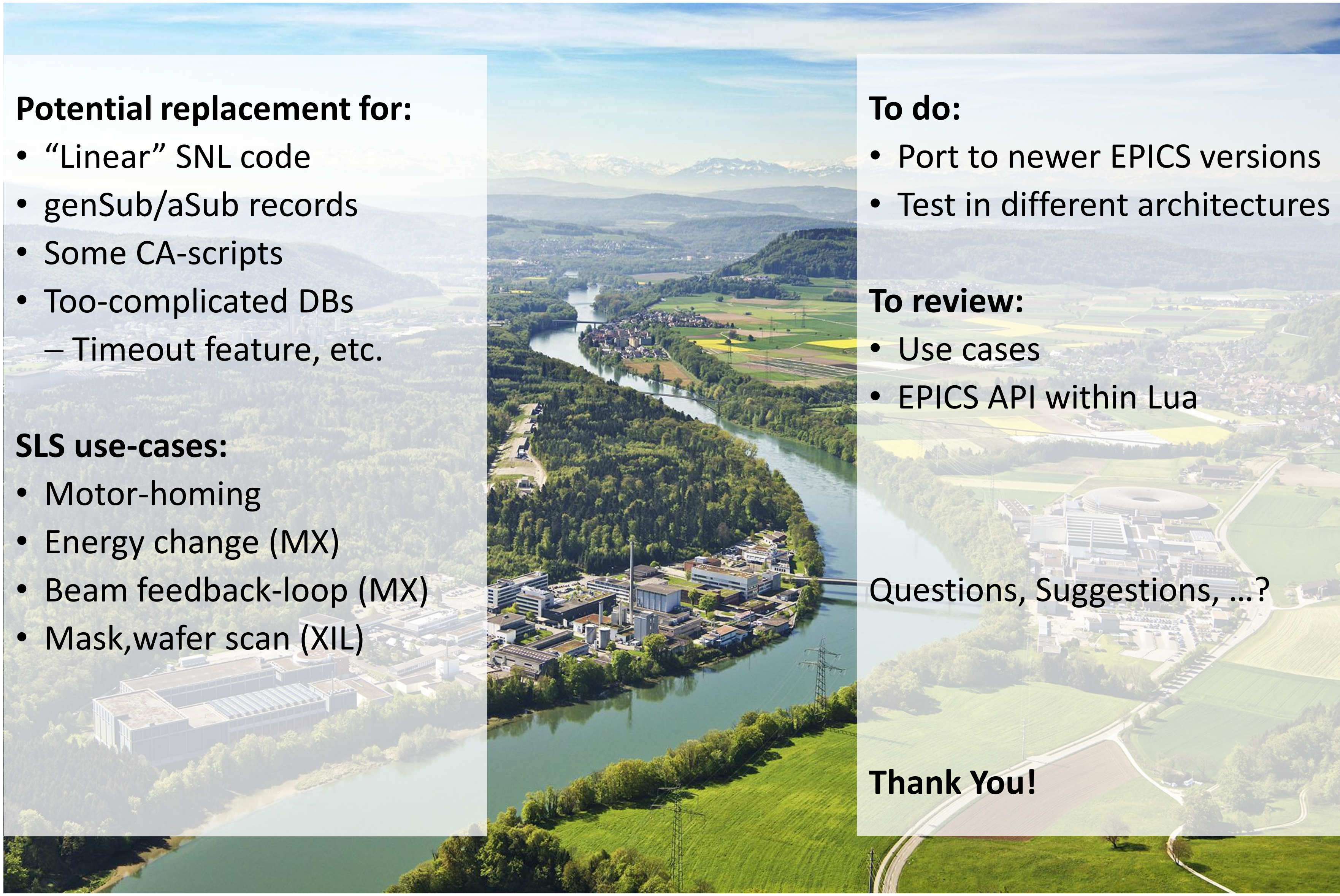
Records and Lua States Relationship







Summary



Potential replacement for:

- "Linear" SNL code
- genSub/aSub records
- Some CA-scripts
- Too-complicated DBs
  - Timeout feature, etc.

SLS use-cases:

- Motor-homing
- Energy change (MX)
- Beam feedback-loop (MX)
- Mask,wafer scan (XIL)

To do:

- Port to newer EPICS versions
- Test in different architectures

To review:

- Use cases
- EPICS API within Lua

Questions, Suggestions, …?

Thank You!