APPLICATIONS OF LUA-BASED EMBEDDED SCRIPTING WITHIN EPICS AT LANSCE

Jeff Hill

LANSCE

APPLICATIONS OF LUA EMBEDDED SCRIPTING – OUTLINE

- Lua, a Brief Introduction (review)
- EPICS Integration of Lua milestones (review)
- Lua Beam Species Filtering at LANSCE
- Conclusions

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA A BRIEF INTRODUCTION (REVIEW)

- Lua *embeddable* language was created in 1993
  - By members of the Computer Graphics Technology Group (Tecgraf) at the Pontifical Catholic University of Rio de Janeiro, in Brazil.
- "Lua" (pronounced **LOO-ah**) means "Moon" in Portuguese
- Interpreted, compiled at load-time to byte-code
- A mixture of C-like and Pascal-like syntax
- Dynamic typed, automated conversion between string and nummeric types
- Efficient virtual machine execution, small footprint, incremental garbage collection, easily interfaced with C code
- Liberal MIT license
- Some negatives also, see my talk at Michigan EPICS meeting
  - In particular, variables are globally scoped by default

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- Lua 5.2.3, the current release, embedded inside of EPICS base
  - Built by the EPICS build system
  - This is the current released version of Lua
    - It has the upgraded support for integer primitive types

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- Lua based subscription filtering in the CA server
  - Event queue is order correct
  - Based on C++ 11 shared pointer
    - Subset of boost included in EPICS base supporting prior compilers

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- Lua based subscription filtering in the CA server
  - Filters specified as channel name postfix
    - Invoking Lua methods supplied when the IOC boots
  - Each client attaching to the server
    - Instantiates an independent Lua context

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- Alternative EPICS SHELL
  - In contrast, a fully functionality scripting language
    - Powerful libraries, built-in and community
- An environment well proven for use in
  - Configuration
  - Scripting
  - Rapid-prototyping

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- EPICS IOC shell can invoke, and pass arguments to, Lua scripts
- Lua scripts can invoke, and pass arguments to
  - Any of the commands registered into EPICS IOC shell
  - We can, for example, instantiate records within a Lua for loop

APPLICATIONS OF LUA EMBEDDED SCRIPTING – EPICS INTEGRATION OF LUA MILESTONES

- Currently we have two computational record-level building block components
  - EPICS calc record
    - Excellent rapid prototyping, but limited functionality
  - EPICS subroutine record
    - Excellent efficiency, but possibly less popular for rapid prototyping
- A new Lua based record provides
  - Comprehensive functionality set
  - A reasonable compromise runtime execution efficiency
  - The rapid prototyping we depend on with the calc record
    - Upgrade in-place
      - Runtime code updates via CA puts to lua record fields
  - And, hopefully the heavy lifting comes for free with Lua

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- LANSCE Requirements
- At LANSCE flavors are specified by
  - A set of timing system gates that shall logically be present
  - A set of timing system gates that shall logically not be present
- At LANSCE we schedule unique flavors for each beam pulse in a 120 entry map
  - Beam pulses occur at 120 Hz rate, flavor map index increments at this rate
  - Our flavor map repeats at 1 Hz rate , flavor map index returns to zero at this rate
- EPICS CA flavor subscription update rates, no more than 4 Hz
  - At LANSCE flavored data are typically waveforms
- Real-time lock between data arrival and timing-system in embedded systems

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- LANSCE Implementation
- Beam species subscriptions specified on the CA client side
- Flavored subscriptions don't require modifications to existing client side tools in the control room
    - Flavor is specified in a CA channel name postfix
- Flavored subscriptions are decimated to satisfy update rate requirements
    - The CA server selects 4 entries in our flavor map for any specified flavor
    - The same 4 entries are selected on all IOCs so we can have synchronous data
    - Managed network consumption and synchronization of flavors between IOCs
- RTEMS real time OS in embedded systems

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Architecture
- Data produced by FPGA Signal processing into multiplexed Avalon packet streams
  - Hardware produces N packets per beam-pulse
    - Each packet is channel-number-tagged
    - Multiplexing hardware enforces ordered arrival of per-channel packets
  - Scatter-gather DMA of packets into Nios2 FPGA-soft-core DDR RAM
  - Identical software DMA support software leveraged in multiple systems
    - Low-level-RF feedback controls, BPPMs, current-monitors, radiation-monitors

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Real-time lock between DMA data and timing-system is required
  - Packets arrive in ordered sequence
    - Software doesn't necessarily start at the correct place in this sequence
  - DMA driver therefore runs in two modes, and can transition between them at any time
    - Acquiring real-time lock mode
    - Real-time locked mode
  - Acquiring real-time lock mode runs at one interrupt per-packet
  - Real-time locked mode runs at one interrupt per beam-pulse
  - Transition to locked mode occurs
    - When the packet with end-of-series channel number arrives
  - Transition to acquiring-lock state
    - If any of the received packets fail to have expected sequentially ordered channel number
  - It was necessary to lower the network daemon's priority in EPICS RTEMS startup code to get this type of real-time synchronization to work in a high priority thread

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

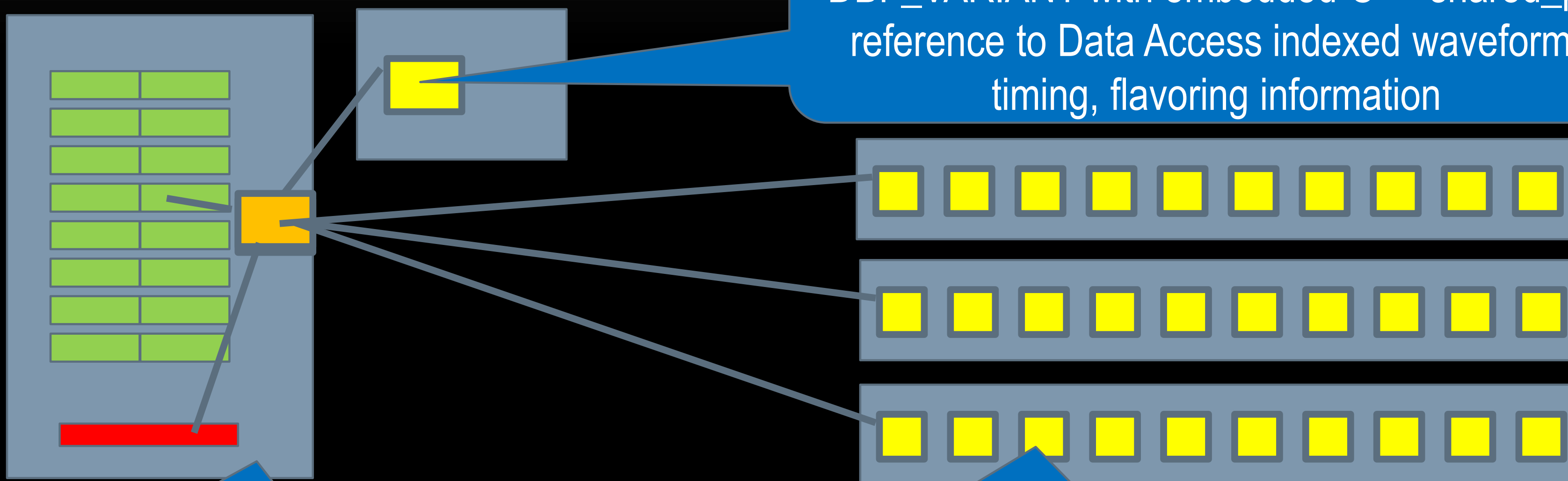
- Performance
- Data arrival rate versus FPGA-embedded softcore processor clock-rate
  - 120 Hz * 16 channels * 2048 elements * 4 bytes per element * 8 bits per byte
    - 126 Mbps Avalon stream packets to DDR RAM incoming data rate
  - 180-220 MHz Nios2 FPGA-soft-core clock rate
    - 6.4 Gbps processor to DDR RAM throughput
    - 1 Gbps LAN interface with 120 Mbps processor-limited throughput

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Performance
- Interrupt service routine is very short by design
  - It only clears interrupts and sets a semaphore
- DMA daemon is responsible for
  - Packet validation
  - Timing link validation
  - Waveform memory management
  - DMA device management
- DMA daemon uses about 6% of the CPU, independent of CA client load
- Record processing uses about 10% of the CPU, minimally impacted by CA client load

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Memory management, key to efficiency



Database record VAL field is type DBF_VARIANT with embedded C++ shared_ptr reference to Data Access indexed waveform, timing, flavoring information

One optimized memory allocation per 16 waveforms performed DMA daemon

CA Server upgraded Event Queues, order preserving, based on C++ shared_ptr

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- CA Channel name postfix Lua code
  - This code can serve one of two purposes
    - The channel name postfix Lua code is a per-subscription-update filter
      - This code is executed by Lua for each and every subscription update
        - Returns false, then the subscription update isn't sent
        - Returns true then the subscription update is sent
    - The channel name postfix code is a factory
      - This code is executed by Lua when the channel is created
        - Returns a Lua function
          - This function is employed as the per-subscription-update filter
        - Returns a Lua object
          - A method on this object serves as the per-subscription-update filter

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- CA Channel name postfix Lua code syntax examples
  - Syntax borrows from scheme of Lua long comments
  - This approach has the benefit of avoidance of escape character sequences

```
xxxChannelName % { Lua channel or filter factory source code }"
xxxChannelName % {{ Lua channel or filter factory source code }}"
xxxChannelName % {={ Lua channel or filter factory source code }=}"
xxxChannelName % {=={ Lua channel or filter factory source code
}==}" ...
xxxChannelName % [ Lua filter source code ]" syntax
xxxChannelName % [[ Lua filter source code ]]" syntax
xxxChannelName % [=[ Lua filter source code ]=]" syntax
xxxChannelName % [==[ Lua filter source code ]==]" syntax ...
```

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Error handling copies Lua stack trace back to client application

```
$ camonitor "53ML001V00%{dog()}"
CA.Client.Exception...............................................
    Warning: "Not supported by attached service"
    Context: "host=m53lfcm.lcs.net:5064 ctx=PV ( 53ML001V00 ) Lua Factory:1: attempt to
call a nil value (global 'dog')
stack traceback:
                [C]: in global 'dog'
                PV ( 53ML001V00 ) Lua Factory:1: in main chunk
                [C]: in ?
                [C]: in ?"
    Current Time: Sat Sep 17 2016 18:30:24.709222557
..................................................................
```

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Lua code for simple counting decimator, loaded at IOC startup

```
function decimatorFactory ( channelName )
        function filtFac ( channel, lowDelta, highDelta, timeout )
                local count = 0
                function filter ( ch )
                        count = count + 1
                        return (count % 30) == 0
                end
                return filter;
        end
        local chan = {
                filterFactory = filtFac
        }
        return chan
end
```

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Channel Access channel name invoking the decimatorFactory () to control the update rate

```
camonitor "53ML001V00%{decimatorFactory ()}"
```

APPLICATIONS OF LUA EMBEDDED SCRIPTING – LUA EVENT FILTERING AT LANSCE

- Syntax of LANSCE flavor specification (factory syntax)
  - This specifies that gate MBEG shall be present
  - This specifies that gates LBEG and H-GX shall *not* be present

```
xxxChannelName % { flavor( “MPEG no LBEG H-GX” ) } "
```

- Syntax of LANSCE flavor specification (factory syntax)
  - This specifies only that gate MBEG shall be present

```
xxxChannelName % { flavor(“MPEG”) } "
```

THE EPICS LUA SCRIPT RECORD – CONCLUSION

- Lua *embeddable* scripting language capabilities have been integrated into EPICS
  - At LANSCE Lua-based CA server event queue filtering is used to implement beam species filtering
  - Filtering is implemented using a quite general Lua scripting language based approach which makes it suitable for multiple sites and projects
  - Flavored decimation is necessary to manage network bandwidth consumption and for synchronizing updates between IOCs