Embedding Lua into LabVIEW

- LabVIEW
- Uses for Lua
- Preemptive and cooperative script scheduling
- Issues & future

Albert-Jan Brouwer
acj.brouwer@citengineering.com

About LabVIEW

- Development environment for data acquisition, test & measurement
- Libraries
- Widgets
- Drivers
- Graphical data-flow language (demo)
  - Compiled
  - Statically typed
- Lacks run-time programmability and control over execution: need a scripting language
- LuaVIEW (http://www.citengineering.com/LuaVIEW)

Uses for Lua

For users

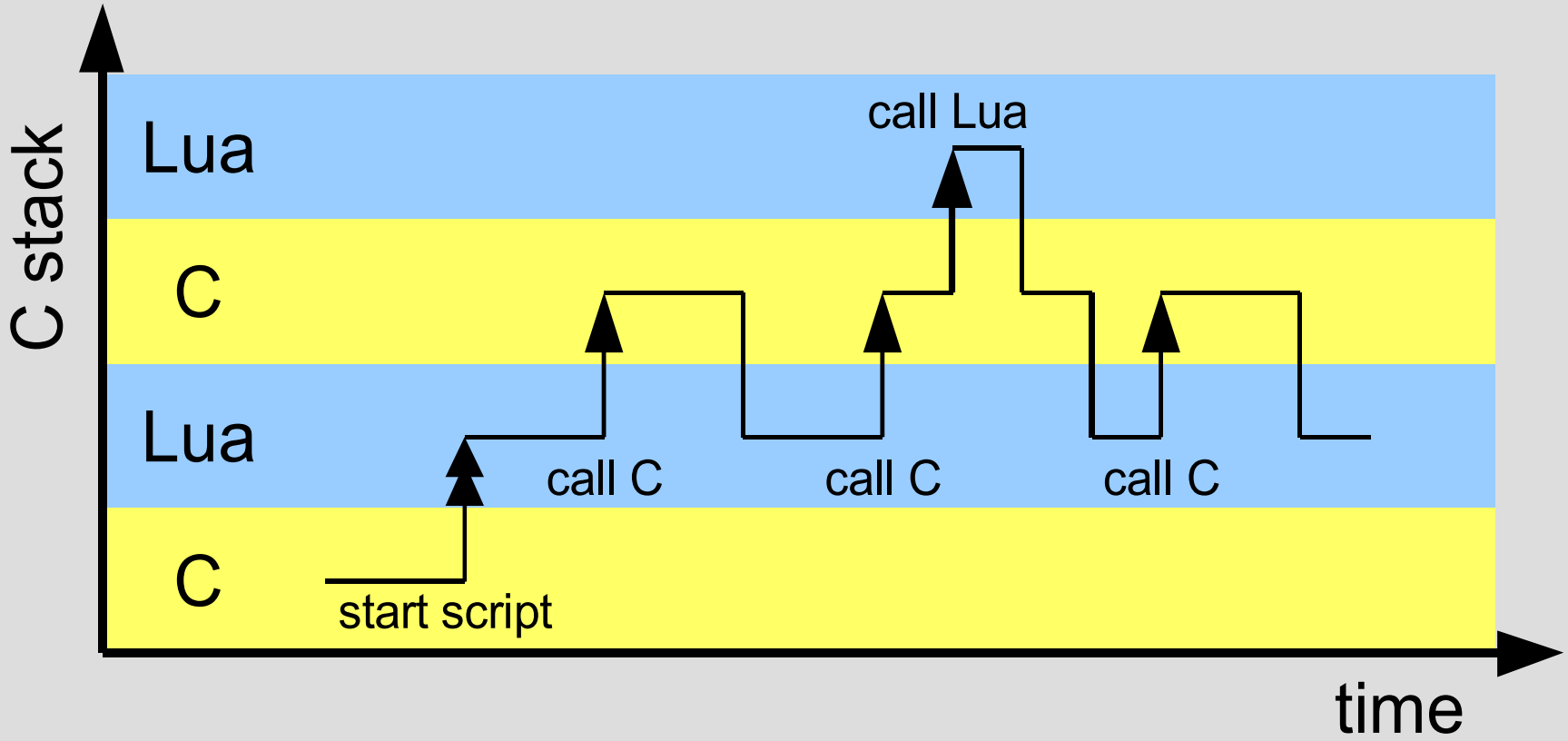
- Test scripting
- Query scripts
- Feedback/polling loops
- Single syntax

For application programmers

- Lua as a library
  - Expression evaluation
  - Serialization
- Application glue
  - Initialization
  - Configuration
  - Runlevels
  - Unit tests & reuse

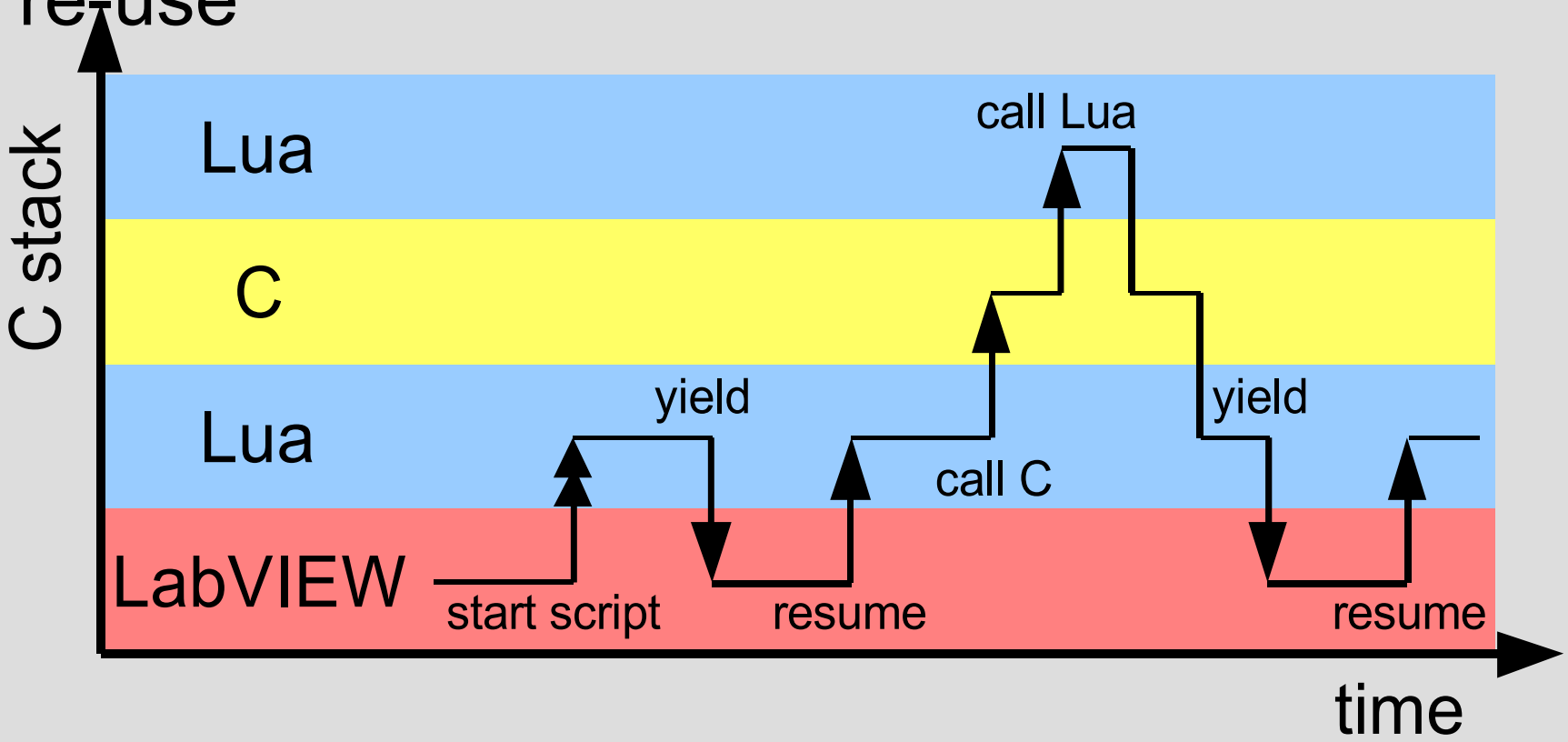
Embedding into LabVIEW is problematic

- Small number of threads
- LabVIEW is “stackless” on account of dataflow scheduling: cannot call LabVIEW



Solution: yield

- As of Lua 5: lua_newthread(), lua_resume(), and lua_yield()
- Yield to call and yield periodically for thread re-use



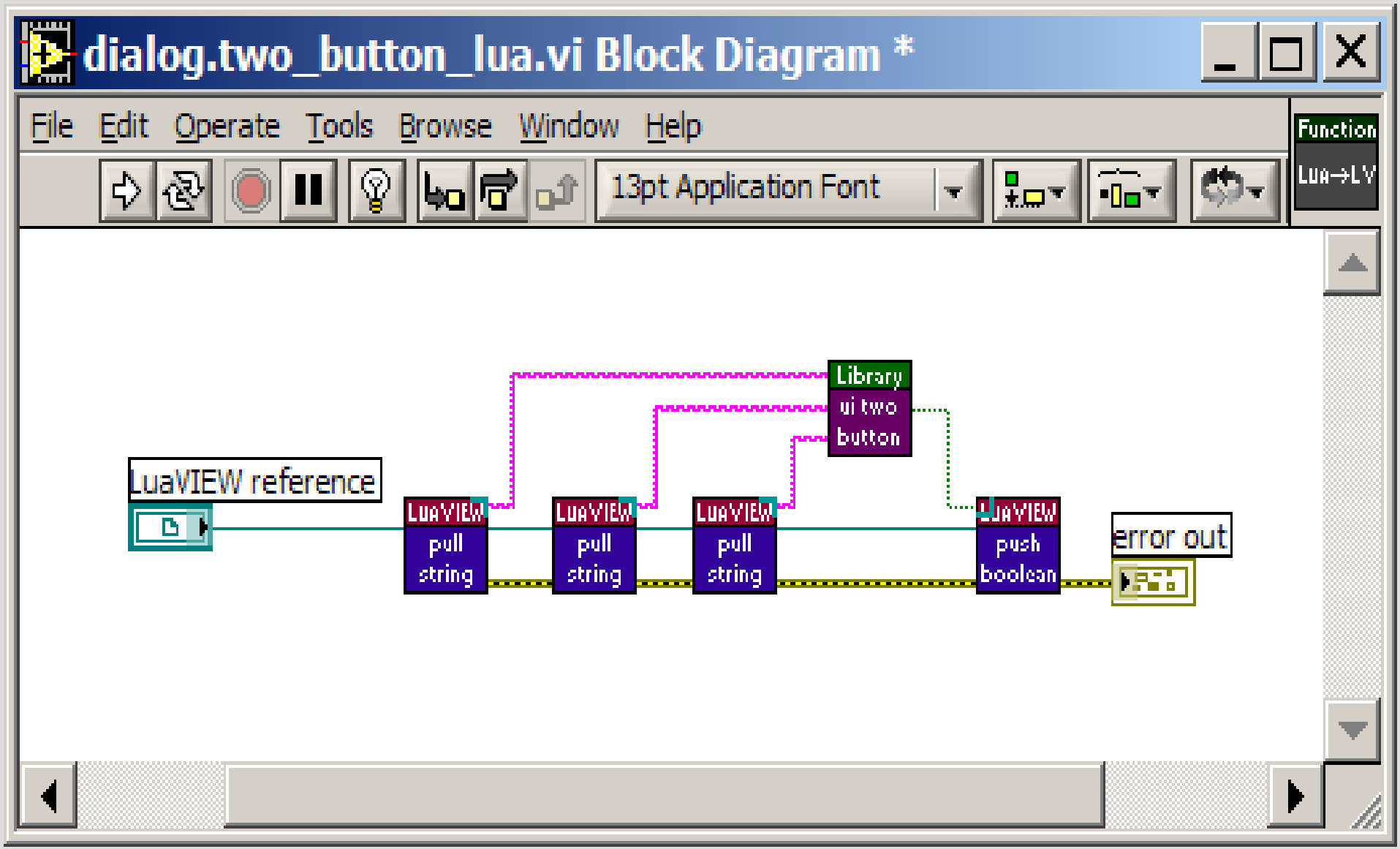
Preemptive and cooperative multitasking of scripts

- Making do with fewer OS threads
  - Reuse threads during protracted functionality
  - Embedded systems
  - Scalability
- Preemptive and cooperative on the C side
- Appears preemptive from the Lua side
- Automatic event handling
- Embedding in a stackless language

How does it work?

- Open a “base” and “thread” lua_State
- The Lua-side code runs on the “thread” lua_State
- Yield-to-call from an “adapter” C function: closures with as upvalue a reference to the actual function to be called when yielded
- Can pass arguments and results via the stack of the “base” lua_State: lua_xmove()
- Frequently yield from opcode-counting hook

Binding


dialog.two_button_lua.vi Block Diagram *
File Edit Operate Tools Browse Window Help
13pt Application Font
Function
Lua→LV
LuaVIEW reference
LuaVIEW
pull
string
LuaVIEW
pull
string
LuaVIEW
pull
string
Library
ui two
button
LuaVIEW
push
boolean
error out

Issues & future

- Cannot yield from Lua when inside a C call
  - The pcall() function in particular
  - No official API function to check this
- Cannot yield from meta methods
- Beware of thread-local storage

Questions?