SNC/SEQ Developments

Michael Laznovsky

SLAC

overview

- sequencer internals via PVs
- seqPvShow()
- SNL local-variable access API
- edm-based state-level debugger
- future/ongoing

seq internal-state PVs

- seq program internal state available via PVs ...work at SLAC by Kuhkee Kim (now at KBSI)
- available since 9/2003 in version 2.0.6
- code in src/dev/devSequencer.c
- updates asynchronously at 10Hz via internal scan thread; use “I/O Intr” record scan
- doesn’t distinguish between multiple invocations of the same program (only first is accessible)

via “stringin” record

```
record(stringin,"$(user):nSS")
{
        field(DTYP,"Sequencer Internals")
        field(INP,"@sncExample:nStateSets")
        field(SCAN,“I/O Intr")
}

record(stringin,"$(user):nASSIGN")
{
        field(DTYP,"Sequencer Internals")
        field(INP,"@sncExample:nAssign")
        field(SCAN,“I/O Intr")
}

etc.
```

seq internals

stringin record INP field format:

string is of form <seqName>.**<xxx>**, where **<xxx>** may exhibit further structure, e.g. <seqName>.**nStatesets**, <seqName>.<ssName>.**nStates**

- <seqName>:<ssName>
- <seqName>:<ssName>.**threadId**
- <seqName>:<ssName>.**threadIdHex**
- <seqName>:<ssName>.**timeElapsed**
- <seqName>:<ssName>.**nStates**
- <seqName>:<ssName>.**prevState**
- <seqName>:<ssName>.**nextState**
- <seqName>:<ssName>.**currentState**
- <seqName>.**nStateSets**
- <seqName>.**nAssgin**
- <seqName>.**nConnect**
- <seqName>.**nChans**
- <seqName>.**nQueues**
- <seqName>.**pQueues**
- <seqName>.**nlogFile**
- <seqName>.**threadPriority**
- <seqName>.**varSize**

example

```
record(stringin, "$(IOC):SDB$(N):SEQ") {
  field(DTYP,"Sequencer Internals")
  field(INP,"@zz99Control:ss1.currentState")
  field(SCAN,"I/O Intr")
}

% camonitor ZZ99:SDB1:SEQ
 ZZ99:SDB1:SEQ                  06/01/06 09:31:04.679082779 ss1_init
 ZZ99:SDB1:SEQ                  06/01/06 09:31:16.749157851 ss1_update
 ZZ99:SDB1:SEQ                  06/01/06 09:31:17.779172017 ss1_delay123
 ZZ99:SDB1:SEQ                  06/01/06 09:31:18.809246599 ss1_update
 ZZ99:SDB1:SEQ                  06/01/06 09:31:19.839219015 ss1_delay123
 ZZ99:SDB1:SEQ                  06/01/06 09:31:20.869182430 ss1_update
 ZZ99:SDB1:SEQ                  06/01/06 09:31:21.689297930 ss1_delay123
 ZZ99:SDB1:SEQ                  06/01/06 09:31:22.929213761 ss1_update
 ZZ99:SDB1:SEQ                  06/01/06 09:31:23.769233178 ss1_delay123
 ZZ99:SDB1:SEQ                  06/01/06 09:31:24.799274010 ss1_update
 ZZ99:SDB1:SEQ                  06/01/06 09:31:25.839421092 ss1_delay123
 ZZ99:SDB1:SEQ                  06/01/06 09:31:26.879261175 ss1_update
...
```

seqPvShow()

- dump list of macro-expanded PVs to stdout/file
- show single thread's PVs (via thread id), or all threads if id==0
- basically a per-thread "dbl"
- setting env var turns on automatic dump at each seq thread startup

```
epics> seqShow
Program Name     Thread ID  Thread Name      SS Name
freqControl      4e0200     freqCtlLX        freq_ctrl
                 4f6c28     freqCtlLX_1      sens_ctrl

epics> seqPvShow 4e0200
program freqControl 4e0200
LB40:X:FREQ:SCAN_CUT
LB40:X:FREQ:SCAN_FIT
LB40:X:PHASE:SCAN_CUT
LB40:X:PHASE:SCAN_FIT
LB40:XY:TUNE:KNOB_DELTA
LB40:XY:TUNE:KNOB_STATUS
...
```

snl local-variable access

- additional snc code output
- iocsh command-line functions
- access API

additional snc output

- for example, for this source code:

```
float  amp;
double beta;
short  num;
int    mask;
double ww;
double theta;
```

- snc adds a new table to the generated *.c file (re-entrant example):

```
/* Local Variables (some may also be connected to PVs above)
 *   ... does not include escaped declarations
 */
static struct seqVar seqVar[NUM_VARS] = {
  /* name type_i type_s class dim1 dim2 initial address */
  { "amp",        5, "float",    0,   1,   1, NULL, (void *)OFFSET(struct UserVar, amp) },
  { "beta",       6, "double",   0,   1,   1, NULL, (void *)OFFSET(struct UserVar, beta) },
  { "num",        2, "short",    0,   1,   1, NULL, (void *)OFFSET(struct UserVar, num) },
  { "mask",       3, "int",      0,   1,   1, NULL, (void *)OFFSET(struct UserVar, mask) },
  { "ww",         6, "double",   0,   1,   1, NULL, (void *)OFFSET(struct UserVar, ww) },
  { "theta",      6, "double",   0,   1,   1, NULL, (void *)OFFSET(struct UserVar, theta) }
};
```

seqVarShow (tid,varName)

- display local var info
- ... ("tid" == 0) -> show all local-var info for *all* threads
- ... ("varName" == 0) -> show all local-var info for *single* thread
- ... ("varName" != 0) -> show single var value only

```
long epicsShareAPI seqVarShow (
        epicsThreadId       tid,        /* thread id */
        char               *varName)    /* var name */

epics> seqVarShow

Program: zz99Control
  Name                  Type            [n][n]   Address Value
  ------------------------------------------------------------
  amp                   float                    247128  "0"
  beta                  double                   247130  "0”
  dd                    int             [10][25] 255270  "0"
  num                   short                    247138  "0"
  mask                  int                      24713c  "0"
  w                     double                   247140  "0"
  theta                 double                   247148  "0"
```

local-variable access API

- **get pointer to start of local-variable table, and var count**

```
long epicsShareAPI seqVarGetPtrAndCount (
     epicsThreadId       tid,        /* thread id */
     SVAR              **ppSV,       /* [returned] ptr to struct array */
     int                *pNV)        /* [returned] ptr to count */
```

- **get single local-variable value**

```
long epicsShareAPI seqVarGet (
     epicsThreadId       tid,        /* thread id */
     char               *varName,    /* var name */
     char               *pStr)       /* [returned] char buf for value */
```

- **set single local-variable value**

```
long epicsShareAPI seqVarPut (
     epicsThreadId       tid,        /* thread id */
     char               *varName,    /* var name */
     char               *pStr)       /* value */
```

local-variable access API

- iterator versions (“SVAR” is defined in seqCom.h):

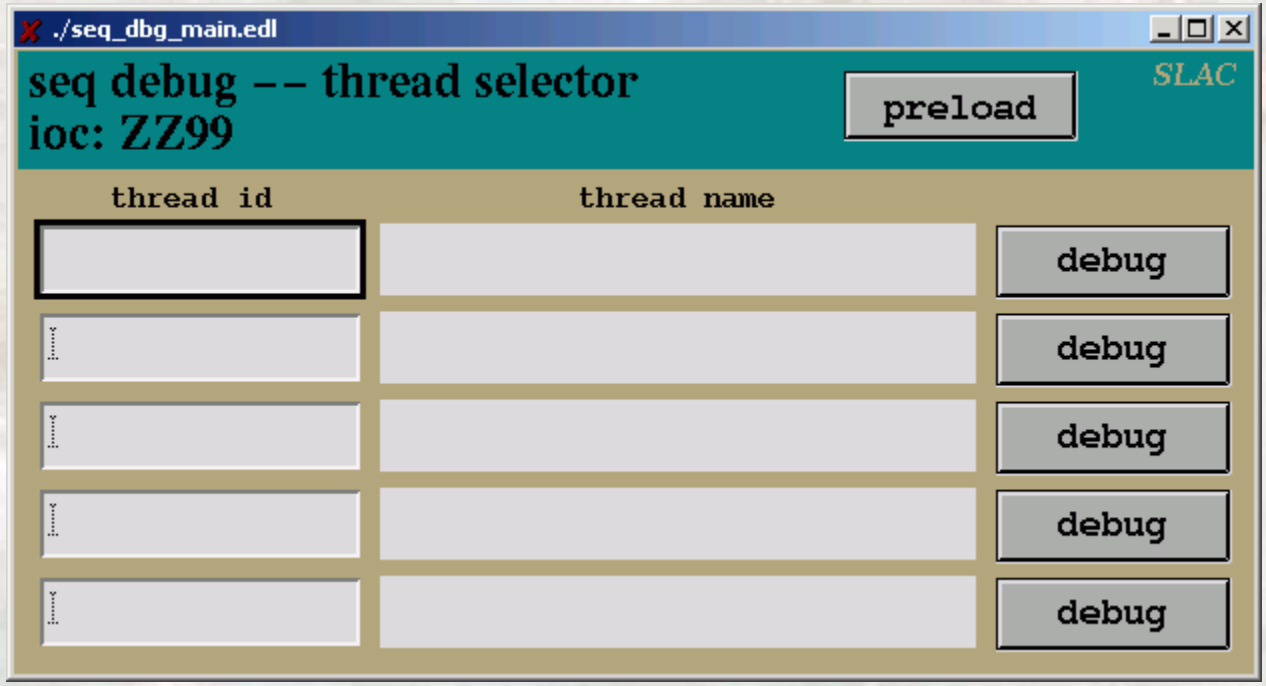
```
long epicsShareAPI seqVarGetInit (epicsThreadId tid)

long epicsShareAPI seqVarGetNext (epicsThreadId tid,
                                  SVAR          **ppSV)
```

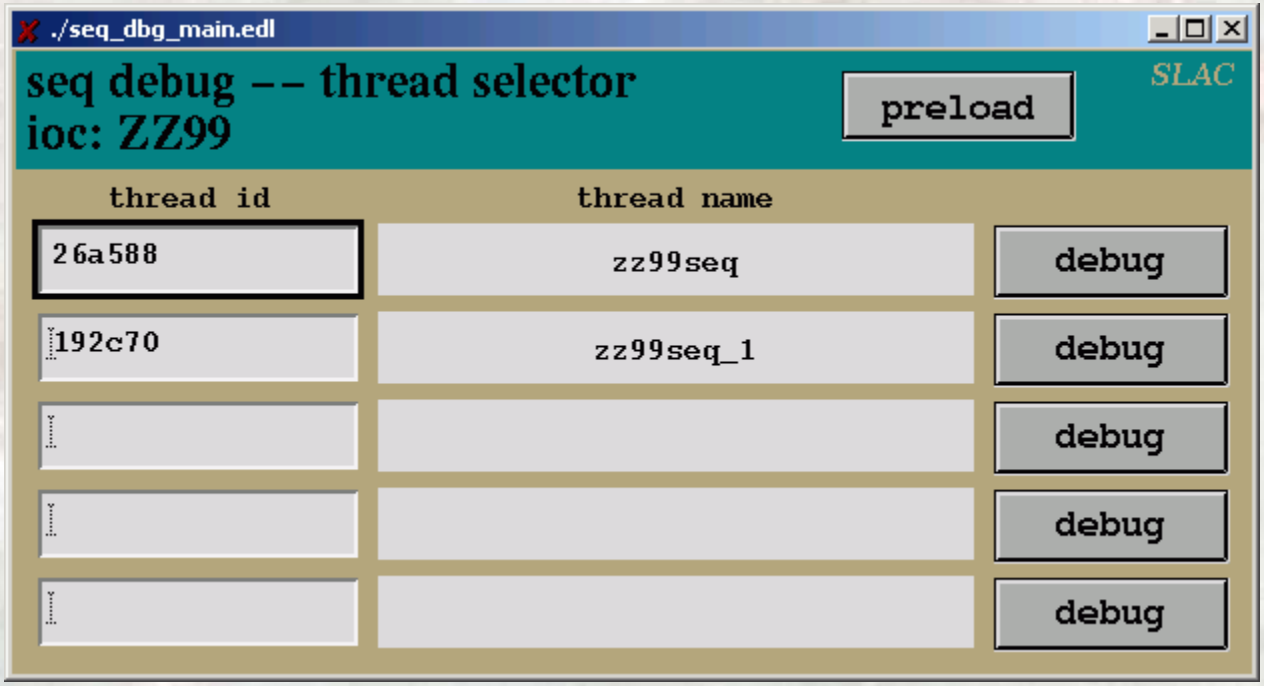
local-variable get/put example

```
epics> seqShow
Program Name      Thread ID  Thread Name       SS Name

zz99Control       267ef0     zz99seq           ss1
                  191a88     zz99seq_1         ss2

epics> seqVarGet 267ef0,"beta"
"0"

epics> seqVarPut 267ef0,"beta","100"

epics> seqVarGet 267ef0,"beta"
"100"
```

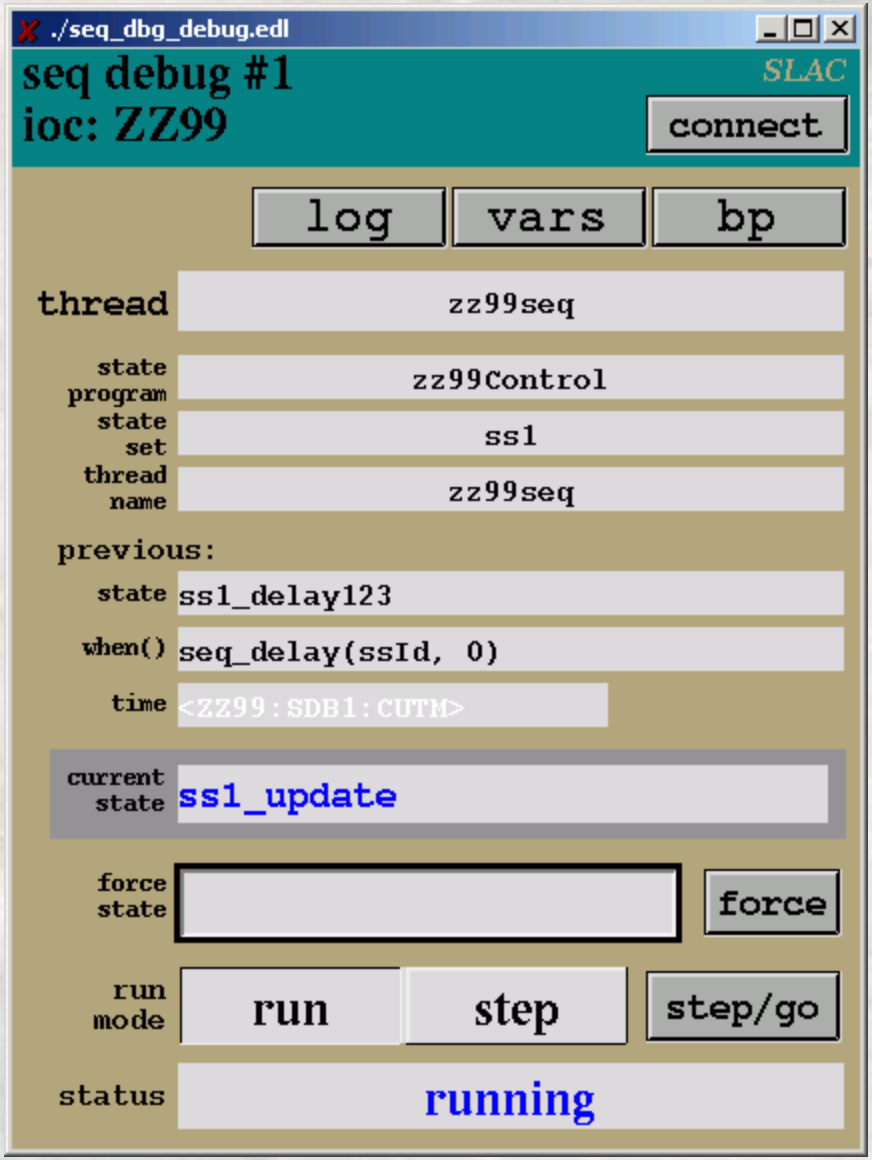
state-level debugger

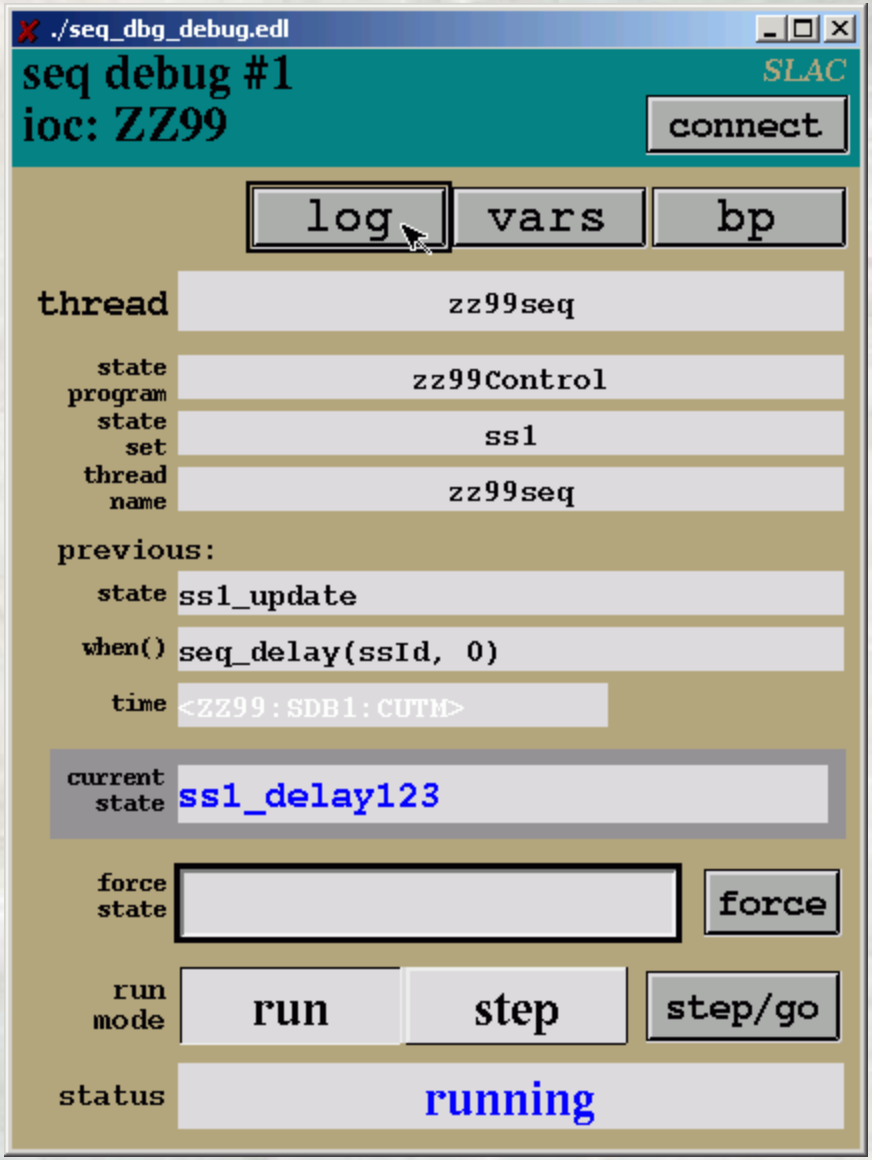
- edm gui backed by small(~ish) db built around **gensub** records
- view/debug multiple threads (currently up to 5)
- view/modify local variables
- force state transitions
- halt / single-step / breakpoints (now: states; later: expressions?)
- trace execution to screen/stdout/file
  - log state names
  - log when() clauses
  - log local-variable values

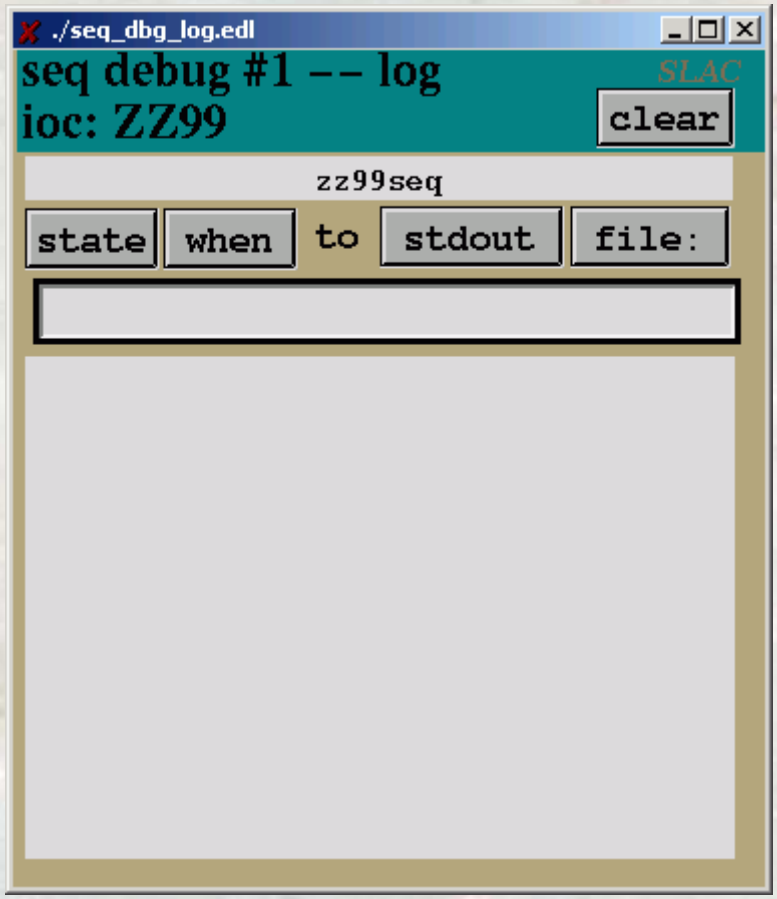

./seq_dbg_main.edl
seq debug -- thread selector
ioc: ZZ99
preload
SLAC
thread id
thread name
debug
debug
debug
debug
debug

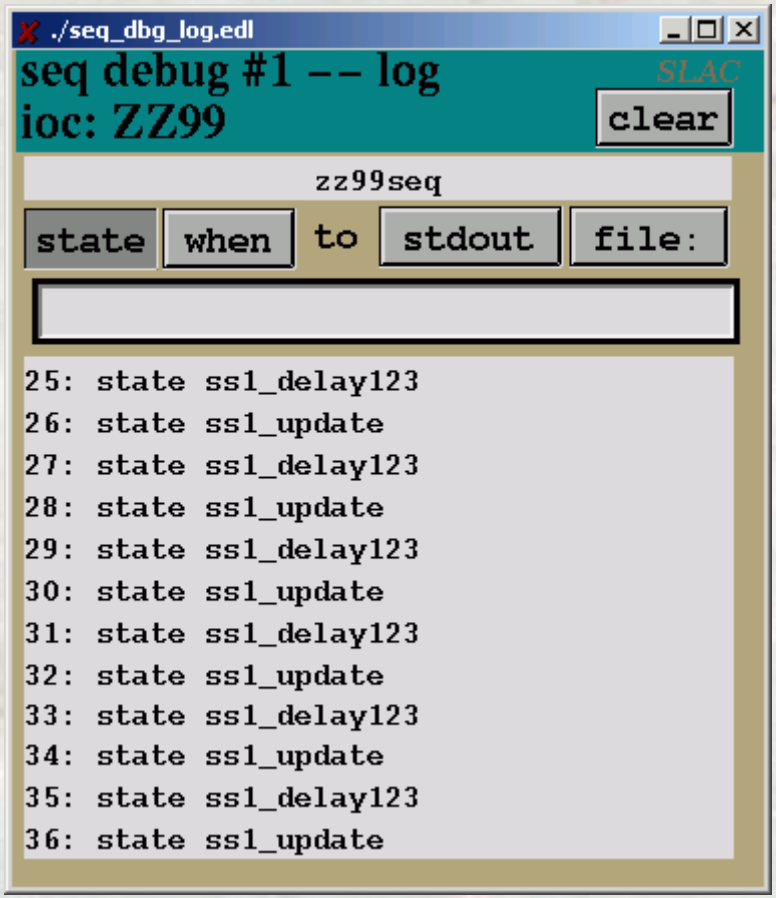

./seq_dbg_main.edl
seq debug -- thread selector
ioc: ZZ99
SLAC
preload
thread id
thread name
26a588
zz99seq
debug
192c70
zz99seq_1
debug
debug
debug
debug

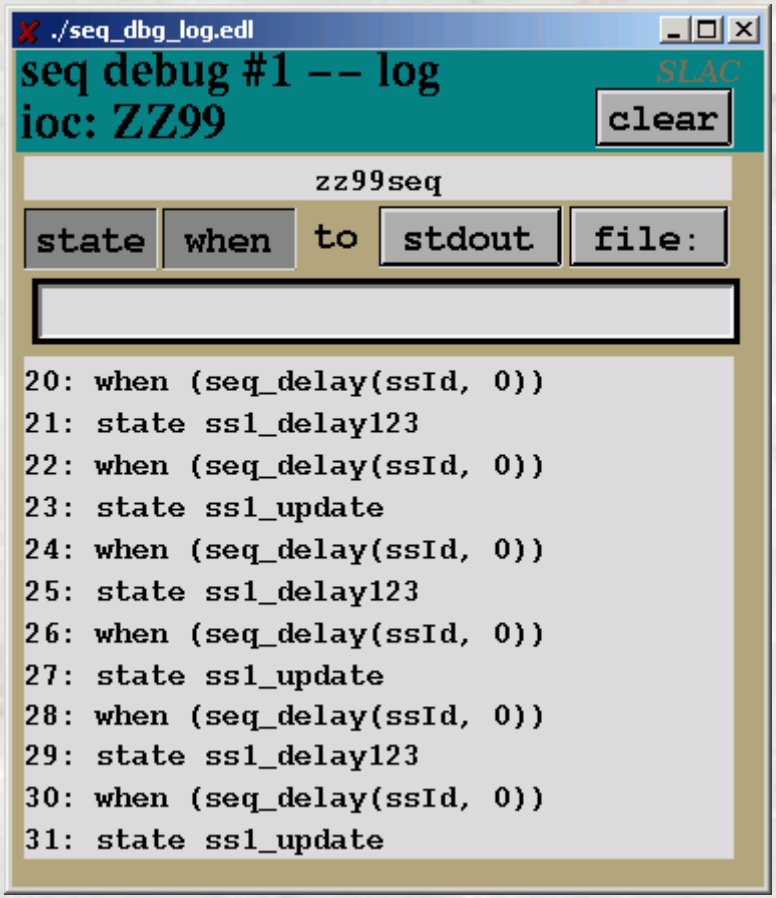

./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
state program
state set
thread name
previous:
state
when()
time
<ZZ99:SDB1:CUTM>
current state
force state
force
run mode
run
step
step/go
status

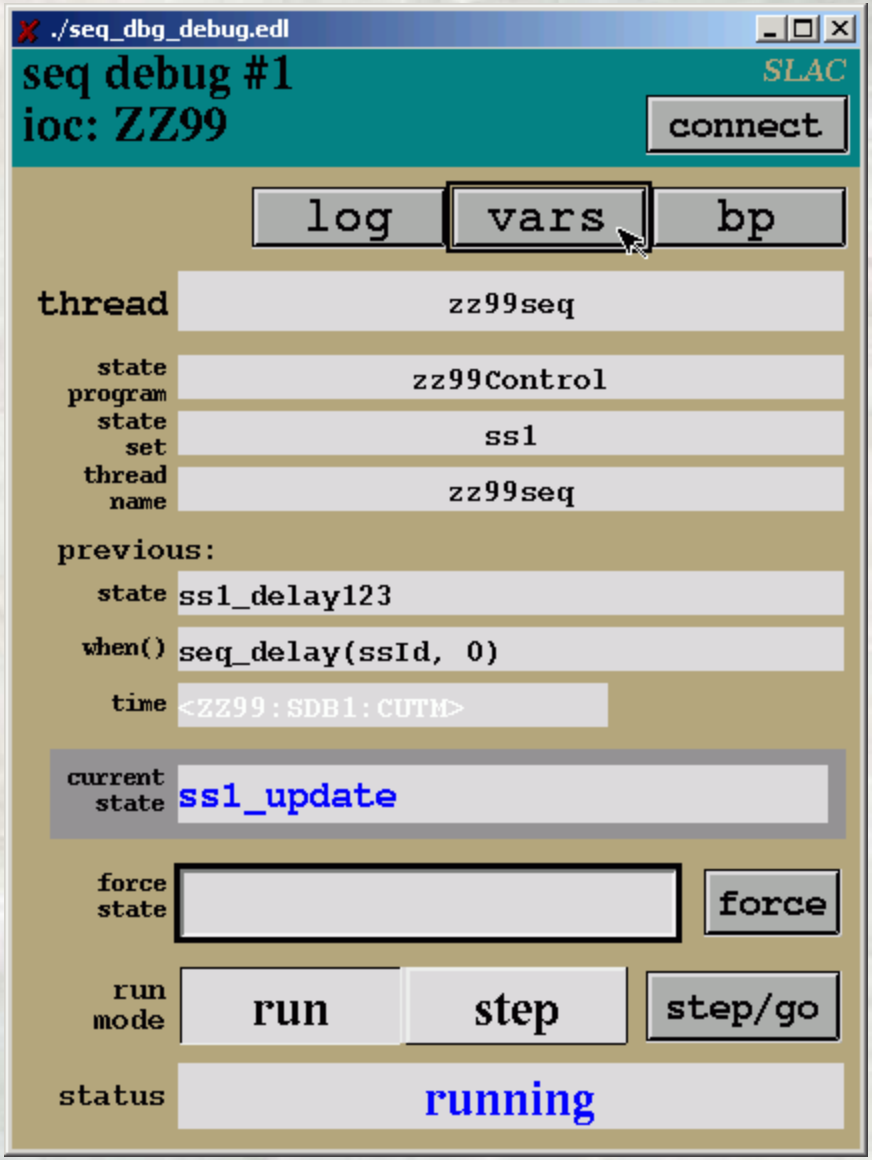

./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_delay123
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
force
run mode
run
step
step/go
status
running

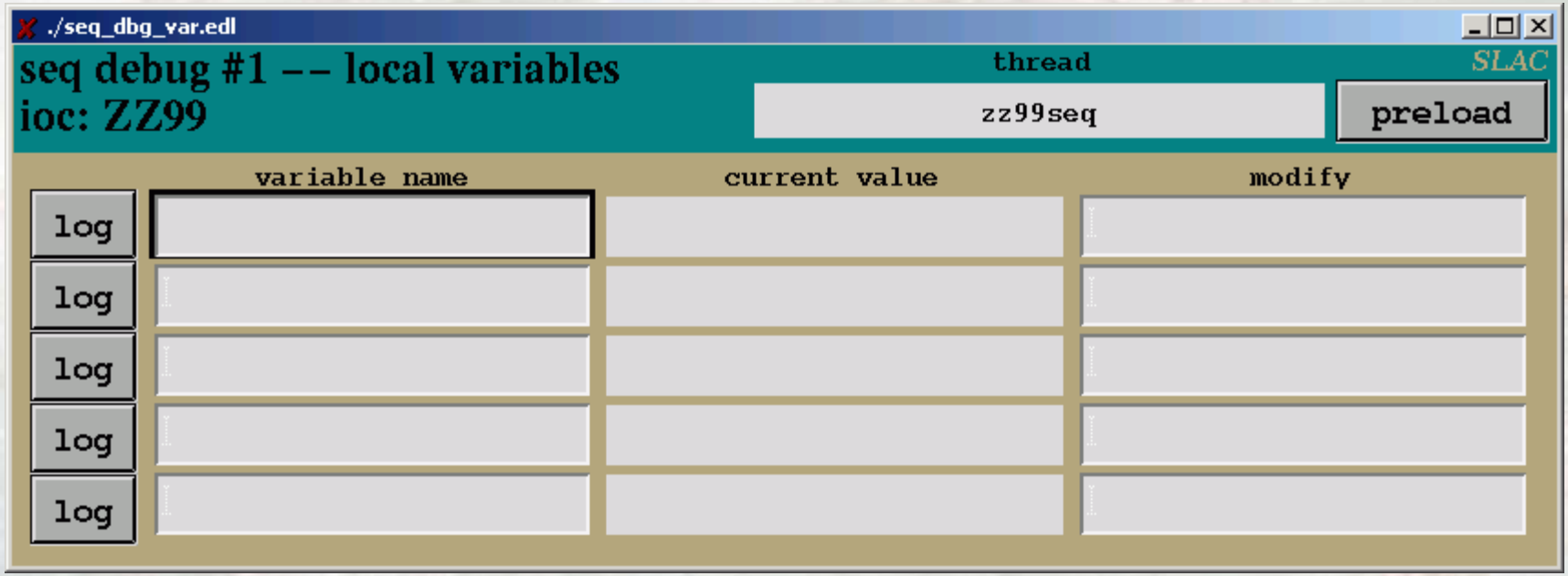

./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_update
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_delay123
force state
force
run mode
run
step
step/go
status
running

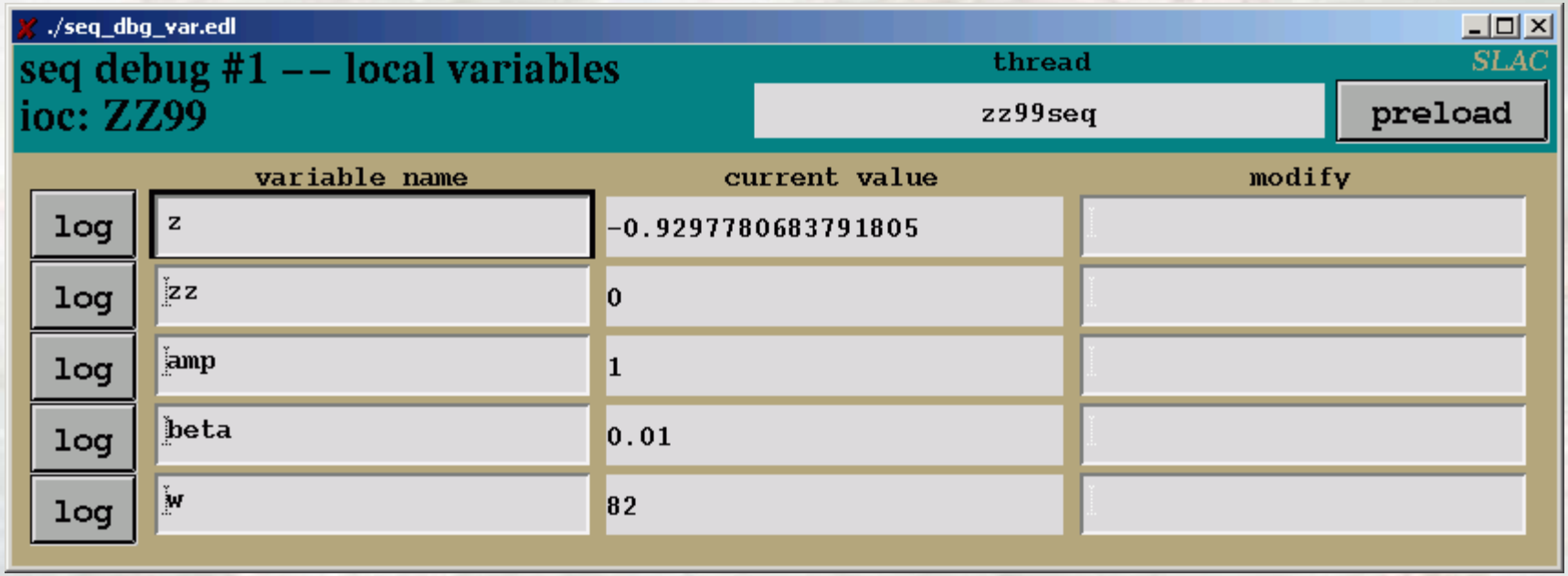

./seq_dbg_log.edl
seq debug #1 -- log
ioc: ZZ99
SLAC
clear
zz99seq
state
when
to
stdout
file:

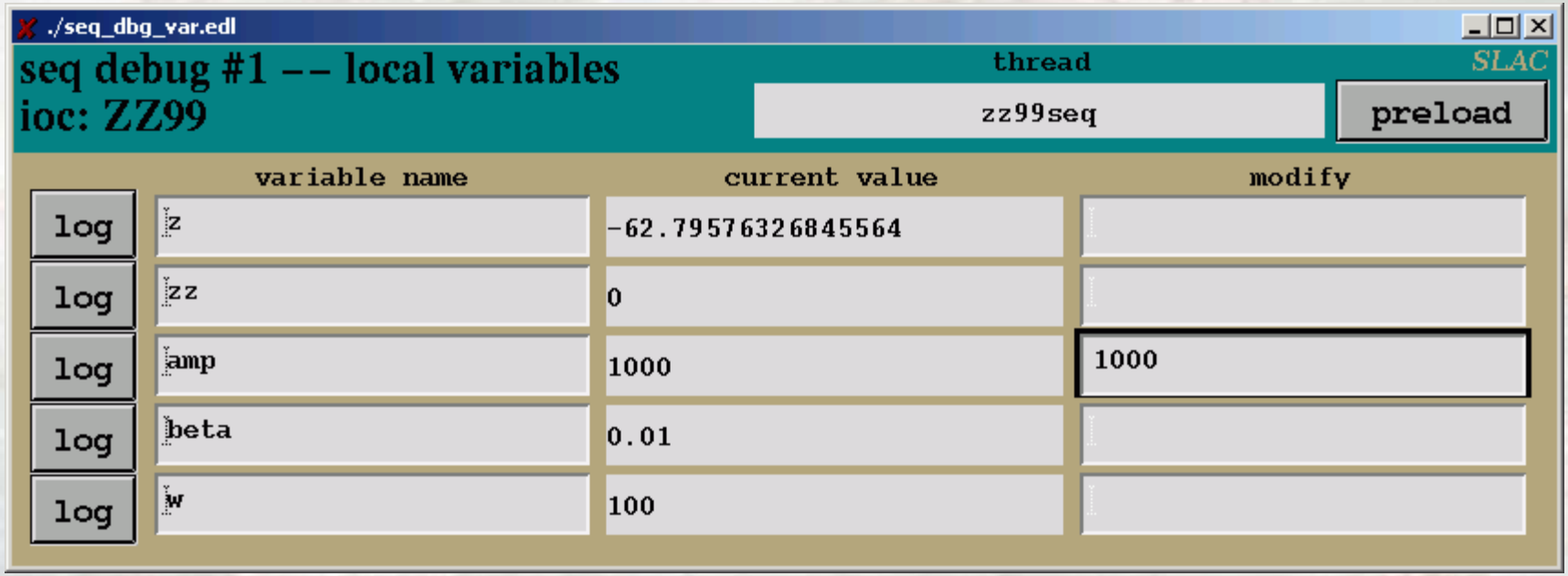

./seq_dbg_log.edl
seq debug #1 -- log
ioc: ZZ99
SLAC
clear
zz99seq
state
when
to
stdout
file:
25: state ss1_delay123
26: state ss1_update
27: state ss1_delay123
28: state ss1_update
29: state ss1_delay123
30: state ss1_update
31: state ss1_delay123
32: state ss1_update
33: state ss1_delay123
34: state ss1_update
35: state ss1_delay123
36: state ss1_update

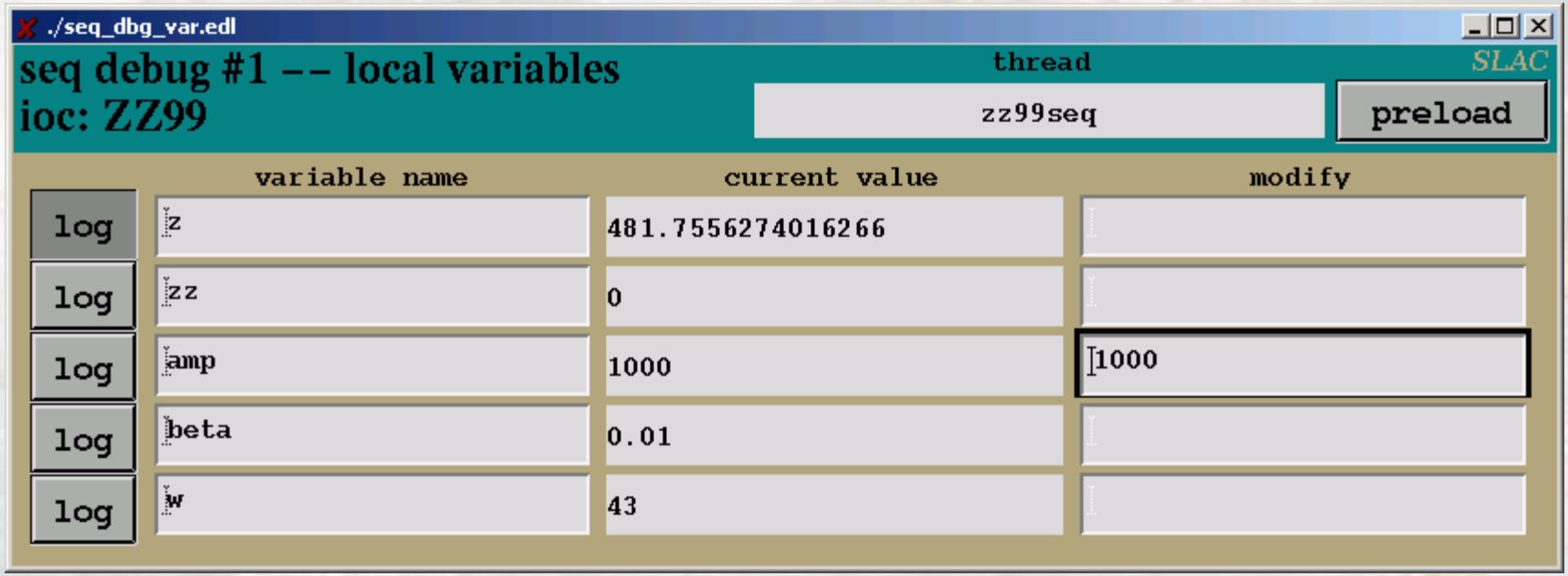

./seq_dbg_log.edl
seq debug #1 -- log
ioc: ZZ99
SLAC
clear
zz99seq
state
when
to
stdout
file:
20: when (seq_delay(ssId, 0))
21: state ss1_delay123
22: when (seq_delay(ssId, 0))
23: state ss1_update
24: when (seq_delay(ssId, 0))
25: state ss1_delay123
26: when (seq_delay(ssId, 0))
27: state ss1_update
28: when (seq_delay(ssId, 0))
29: state ss1_delay123
30: when (seq_delay(ssId, 0))
31: state ss1_update

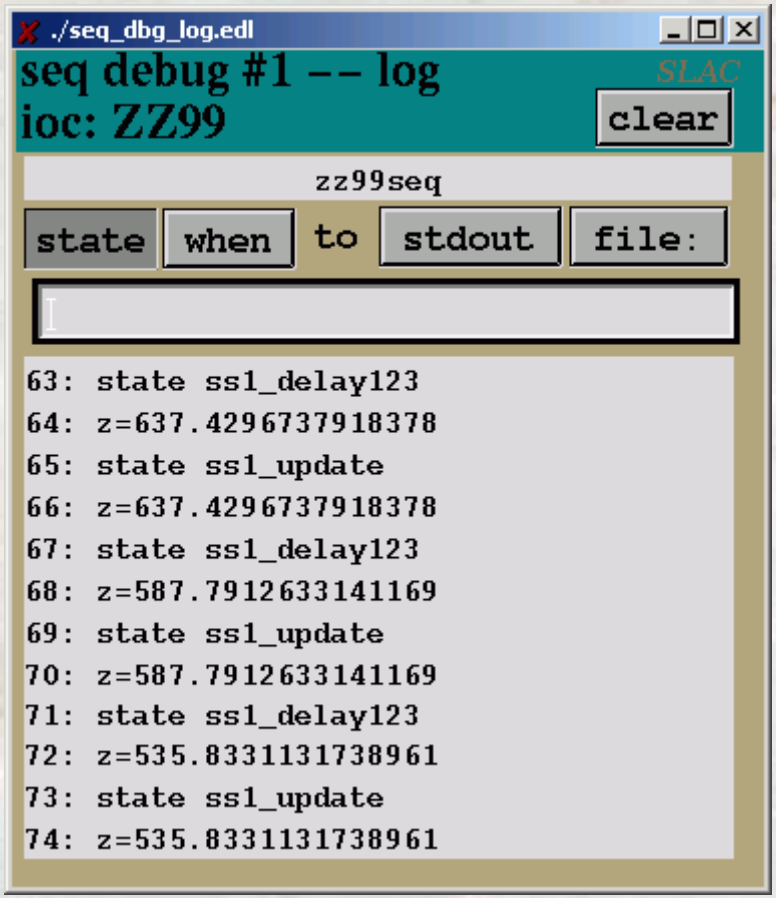

./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_delay123
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
force
run mode
run
step
step/go
status
running

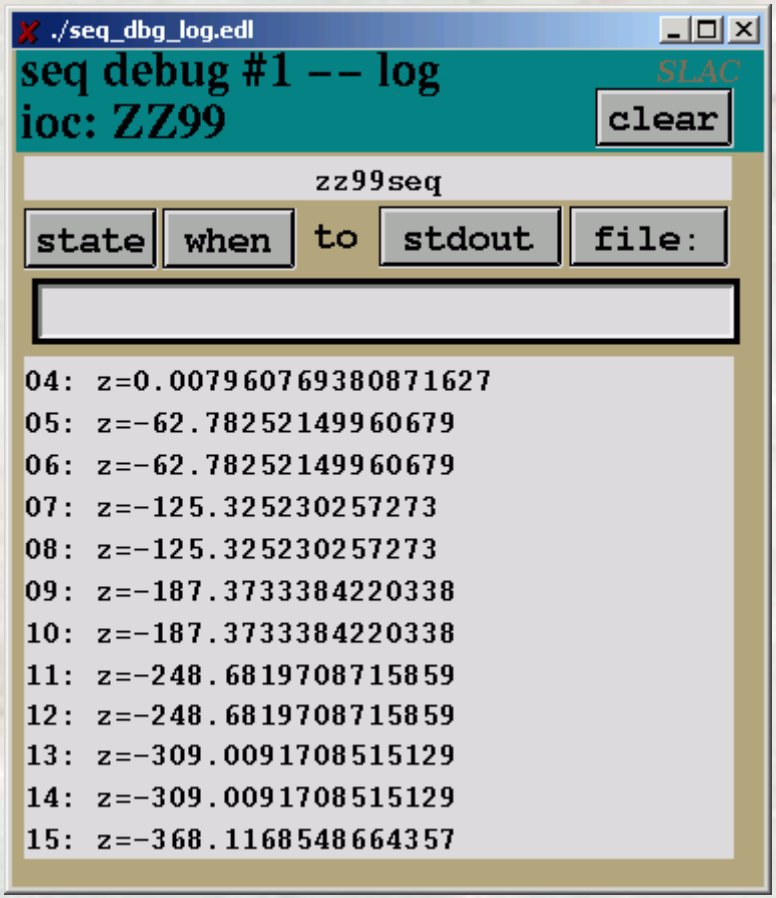

./seq_dbg_var.edl
seq debug #1 -- local variables
ioc: ZZ99
thread
SLAC
zz99seq
preload
variable name
current value
modify
log
log
log
log
log

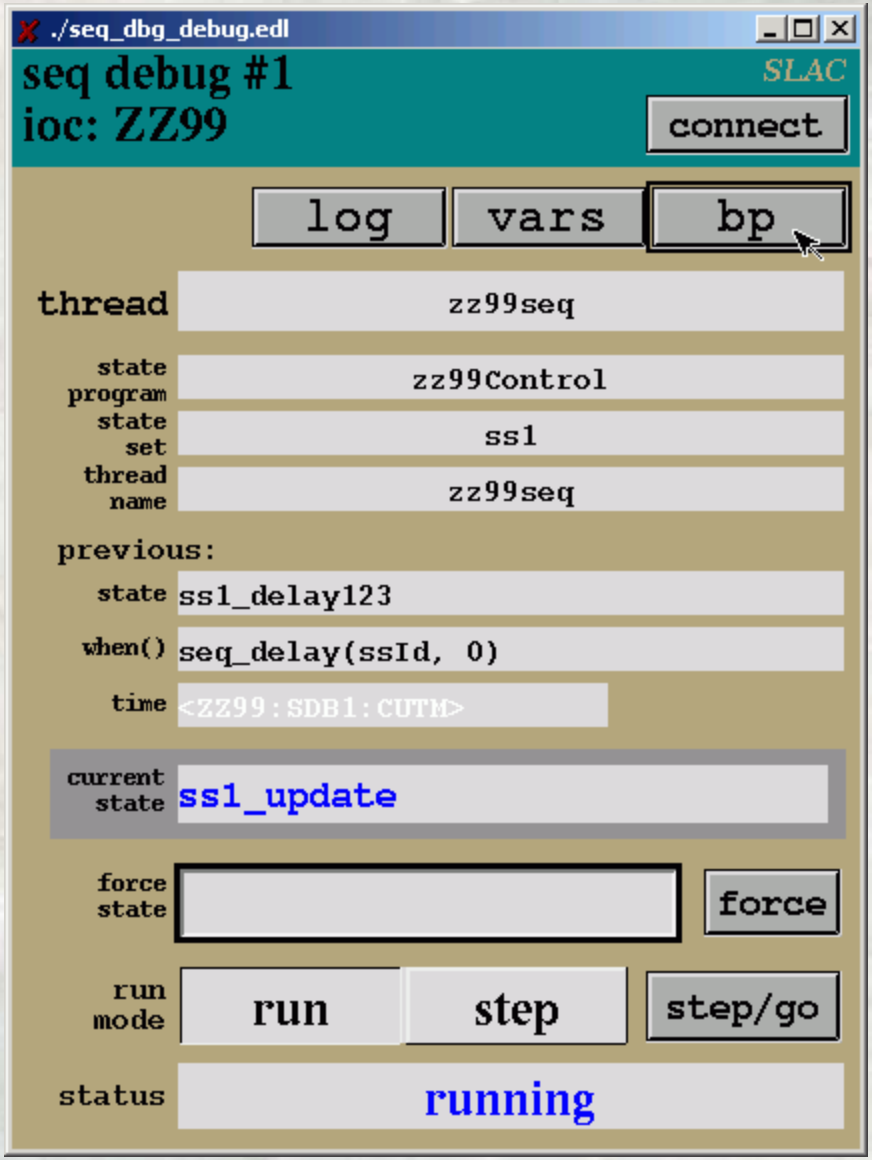

./seq_dbg_var.edl
seq debug #1 -- local variables
ioc: ZZ99
thread
SLAC
zz99seq
preload
variable name
current value
modify
log
z
-0.9297780683791805
log
zz
0
log
amp
1
log
beta
0.01
log
w
82

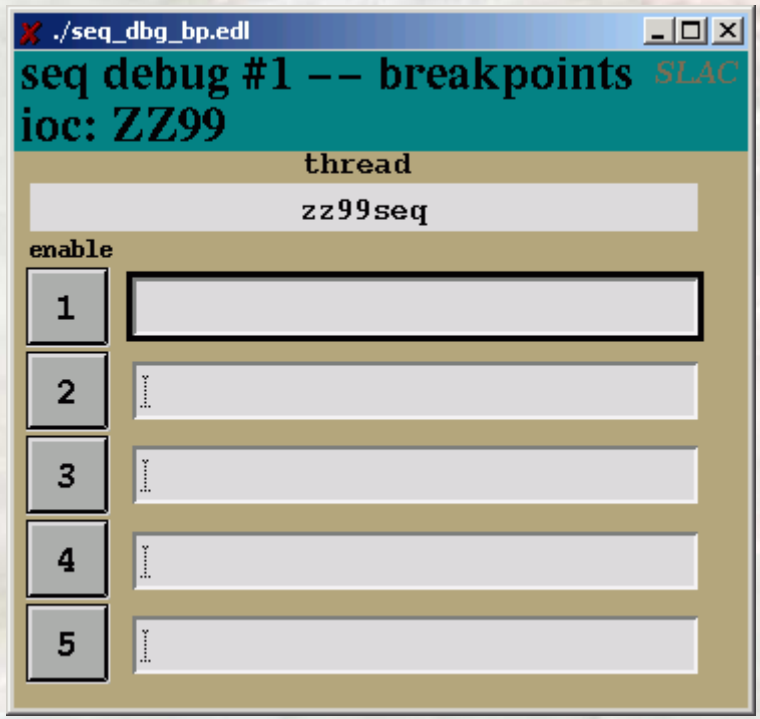

./seq_dbg_var.edl
seq debug #1 -- local variables
ioc: ZZ99
thread
SLAC
zz99seq
preload
variable name
current value
modify
log
z
-62.79576326845564
log
zz
0
log
amp
1000
1000
log
beta
0.01
log
w
100

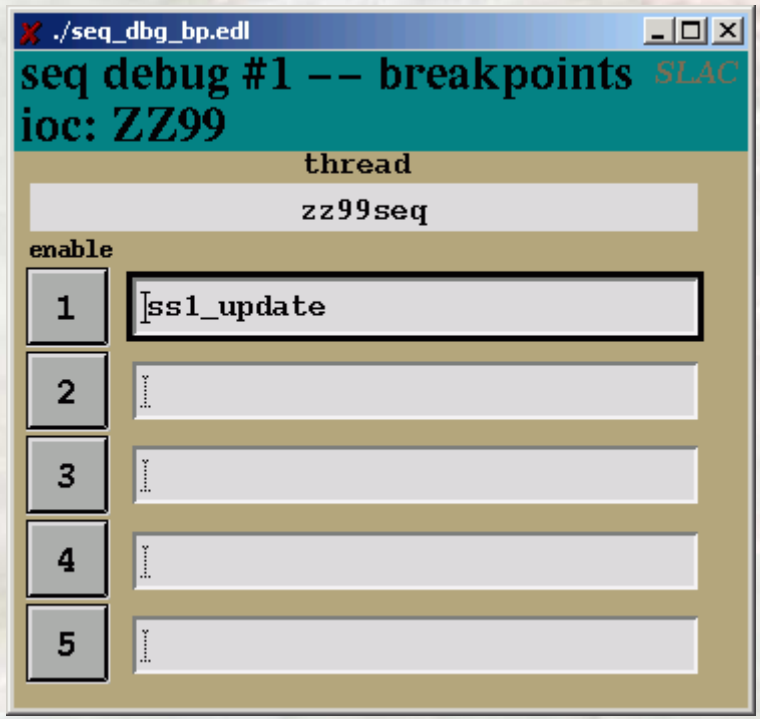

./seq_dbg_var.edl
seq debug #1 -- local variables
ioc: ZZ99
thread
SLAC
zz99seq
preload
variable name
current value
modify
log
z
481.7556274016266
log
zz
0
log
amp
1000
1000
log
beta
0.01
log
w
43

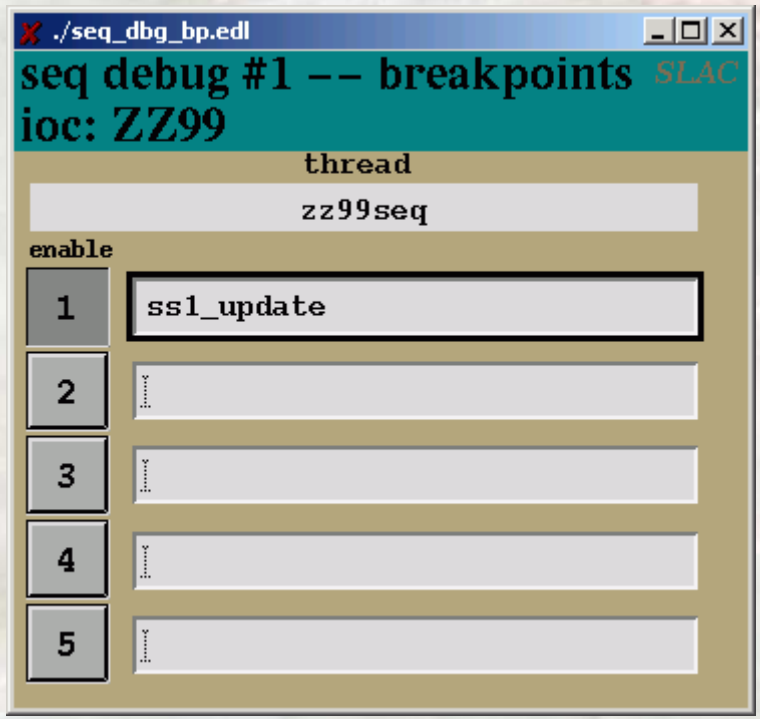

./seq_dbg_log.edl
seq debug #1 -- log
ioc: ZZ99
SLAC
clear
zz99seq
state
when
to
stdout
file:
63: state ss1_delay123
64: z=637.4296737918378
65: state ss1_update
66: z=637.4296737918378
67: state ss1_delay123
68: z=587.7912633141169
69: state ss1_update
70: z=587.7912633141169
71: state ss1_delay123
72: z=535.8331131738961
73: state ss1_update
74: z=535.8331131738961

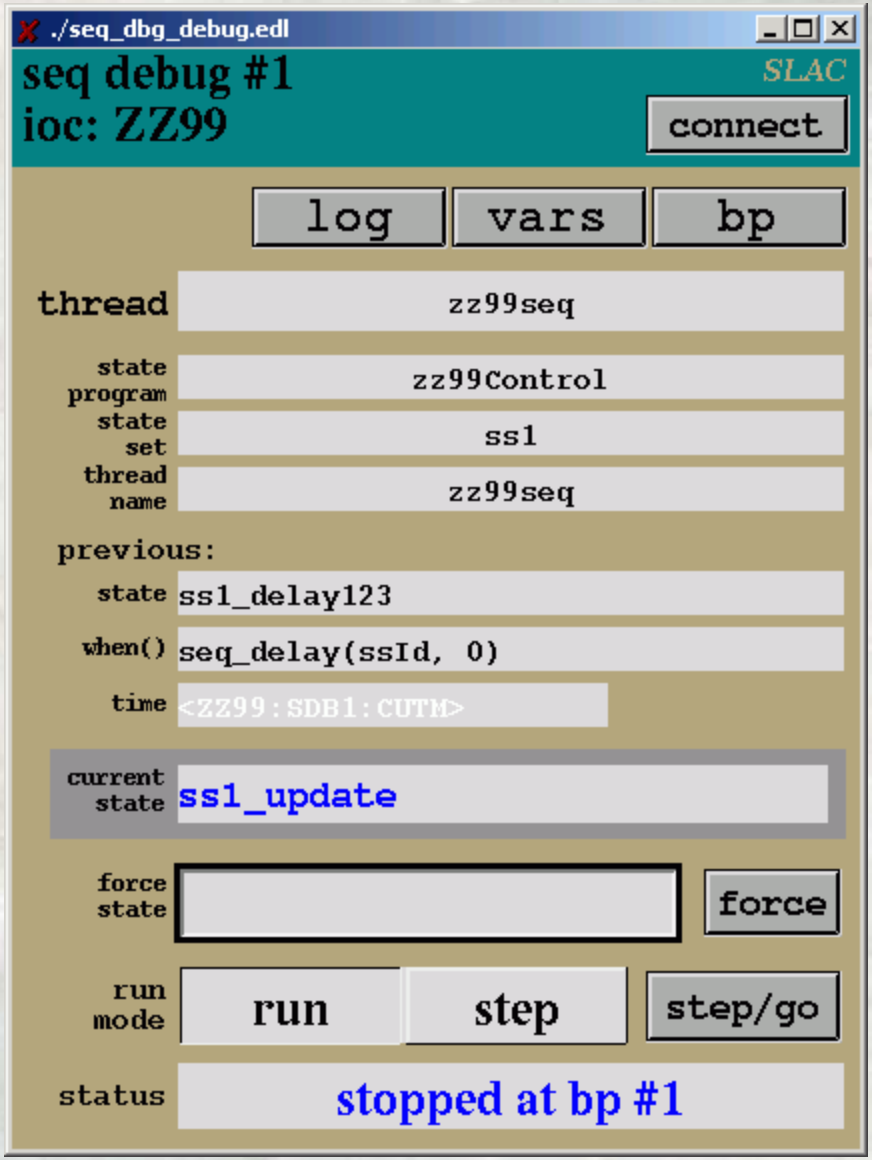

./seq_dbg_log.edl
seq debug #1 -- log
ioc: ZZ99
SLAC
clear
zz99seq
state
when
to
stdout
file:
04: z=0.007960769380871627
05: z=-62.78252149960679
06: z=-62.78252149960679
07: z=-125.325230257273
08: z=-125.325230257273
09: z=-187.3733384220338
10: z=-187.3733384220338
11: z=-248.6819708715859
12: z=-248.6819708715859
13: z=-309.0091708515129
14: z=-309.0091708515129
15: z=-368.1168548664357

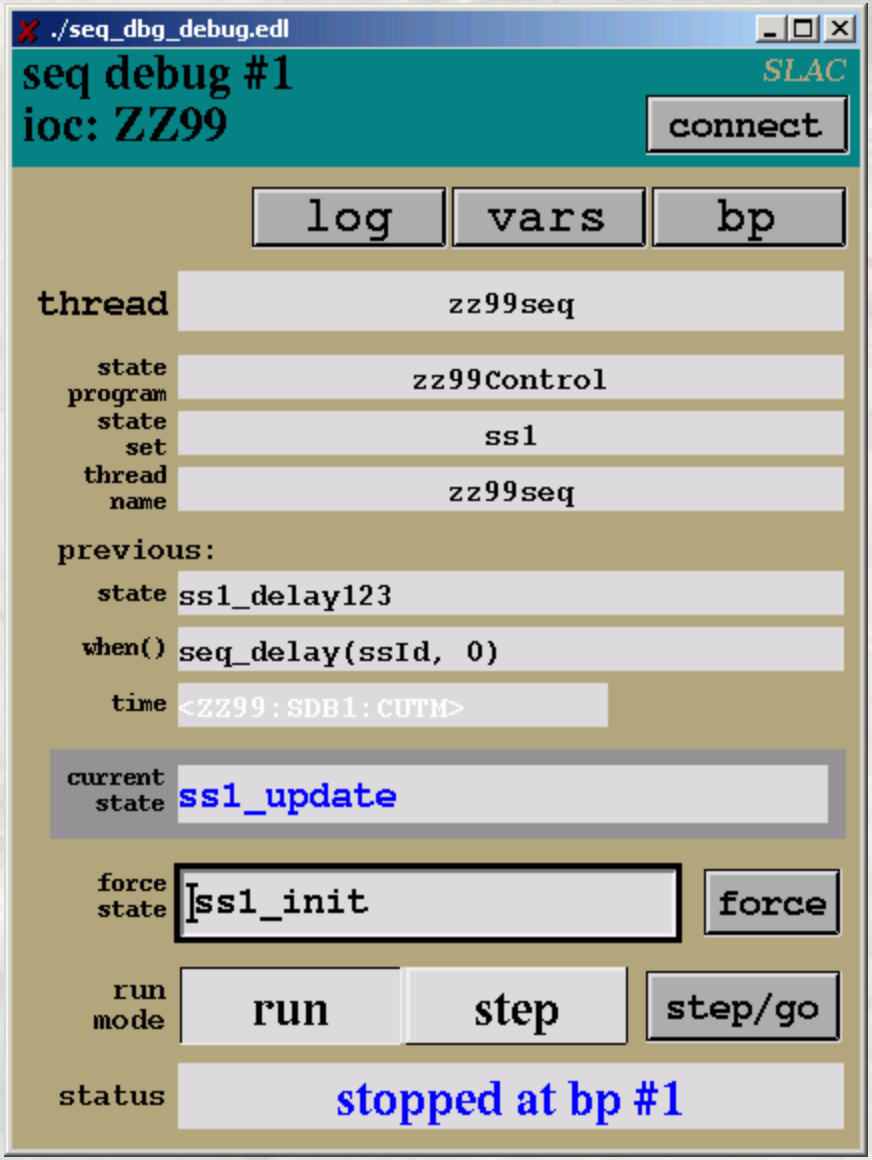

./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_delay123
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
force
run mode
run
step
step/go
status
running

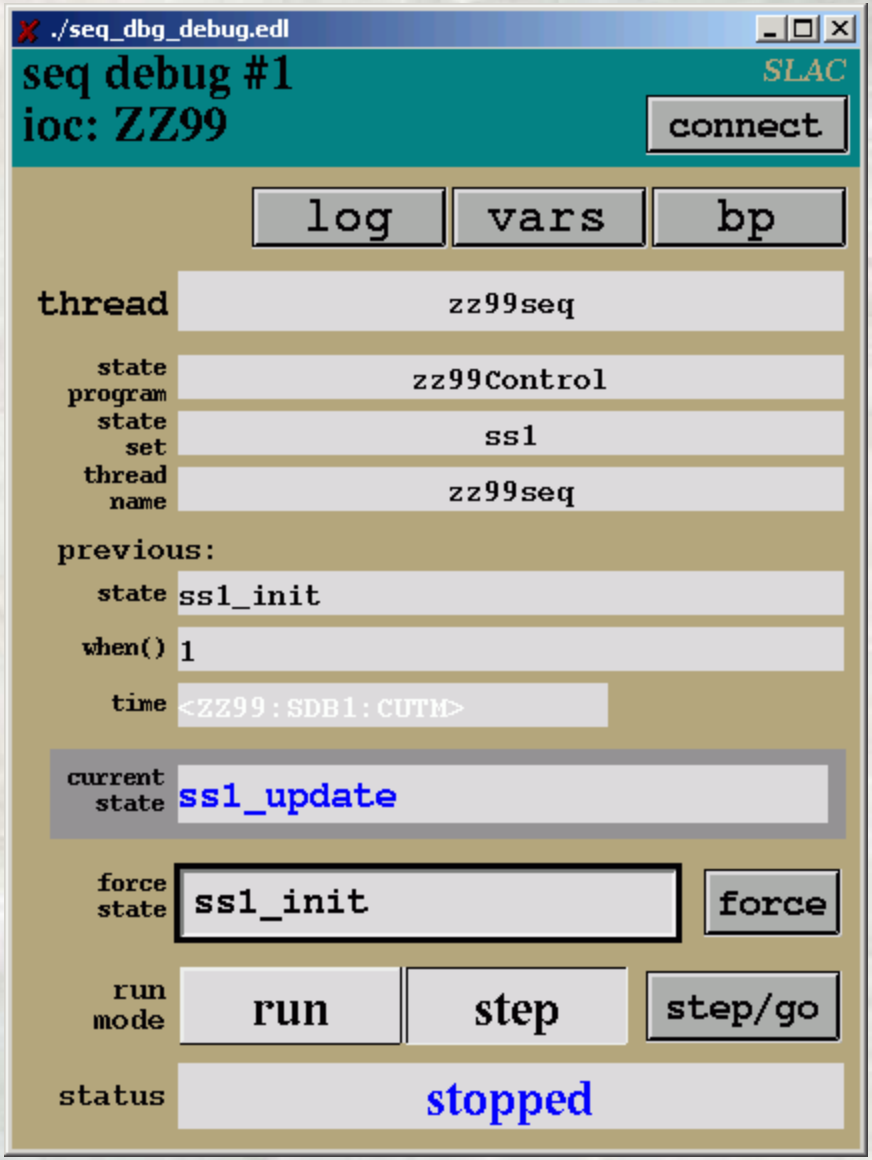

./seq_dbg_bp.edl
seq debug #1 -- breakpoints
SLAC
ioc: ZZ99
thread
zz99seq
enable
1
2
3
4
5

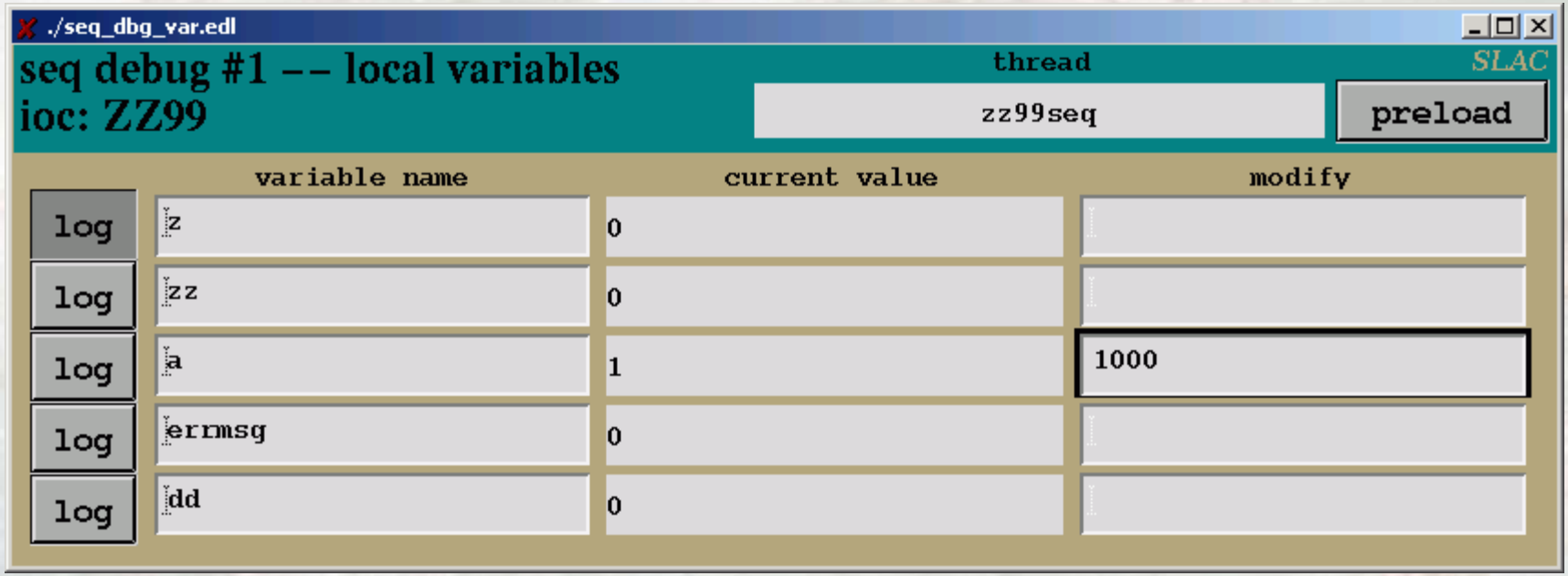

./seq_dbg_bp.edl
seq debug #1 -- breakpoints
SLAC
ioc: ZZ99
thread
zz99seq
enable
1
ss1_update
2
3
4
5


./seq_dbg_bp.edl
seq debug #1 -- breakpoints
SLAC
ioc: ZZ99
thread
zz99seq
enable
1
ss1_update
2
3
4
5


./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_delay123
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
force
run mode
run
step
step/go
status
stopped at bp #1


./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_delay123
when()
seq_delay(ssId, 0)
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
ss1_init
force
run mode
run
step
step/go
status
stopped at bp #1


./seq_dbg_debug.edl
seq debug #1
ioc: ZZ99
SLAC
connect
log
vars
bp
thread
zz99seq
state program
zz99Control
state set
ss1
thread name
zz99seq
previous:
state
ss1_init
when()
1
time
<ZZ99:SDB1:CUTM>
current state
ss1_update
force state
ss1_init
force
run mode
run
step
step/go
status
stopped

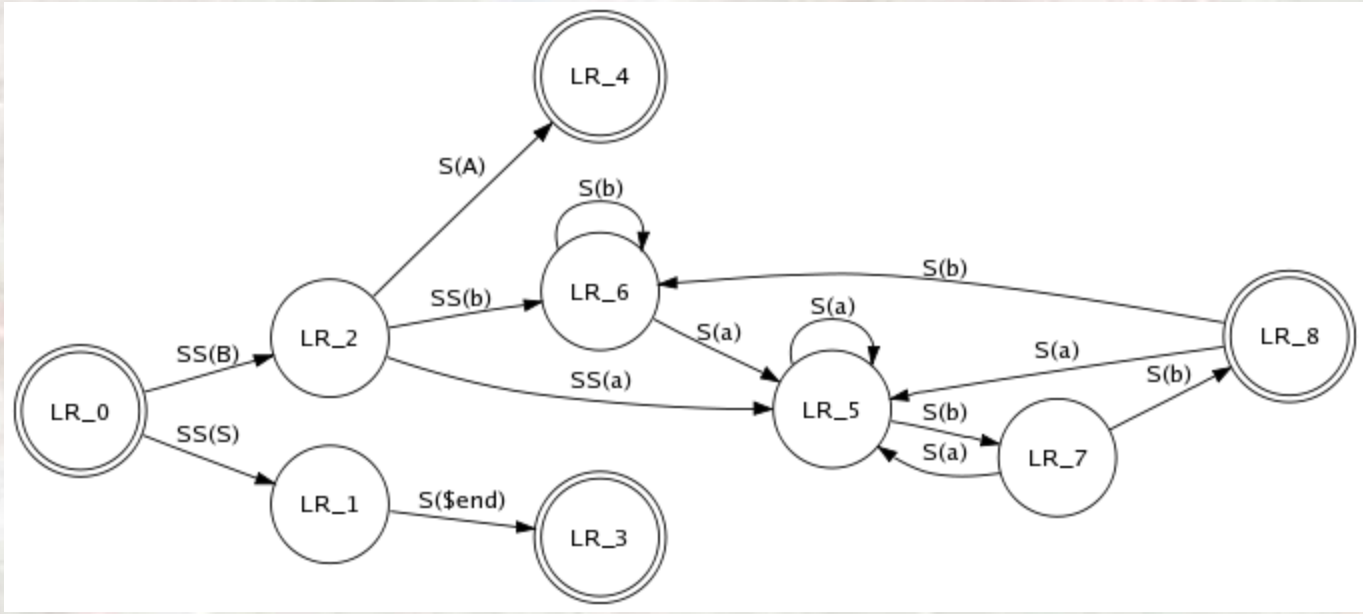

./seq_dbg_var.edl
seq debug #1 -- local variables
ioc: ZZ99
thread
SLAC
zz99seq
preload
variable name
current value
modify
log
z
0
log
zz
0
log
a
1
1000
log
errmsg
0
log
dd
0

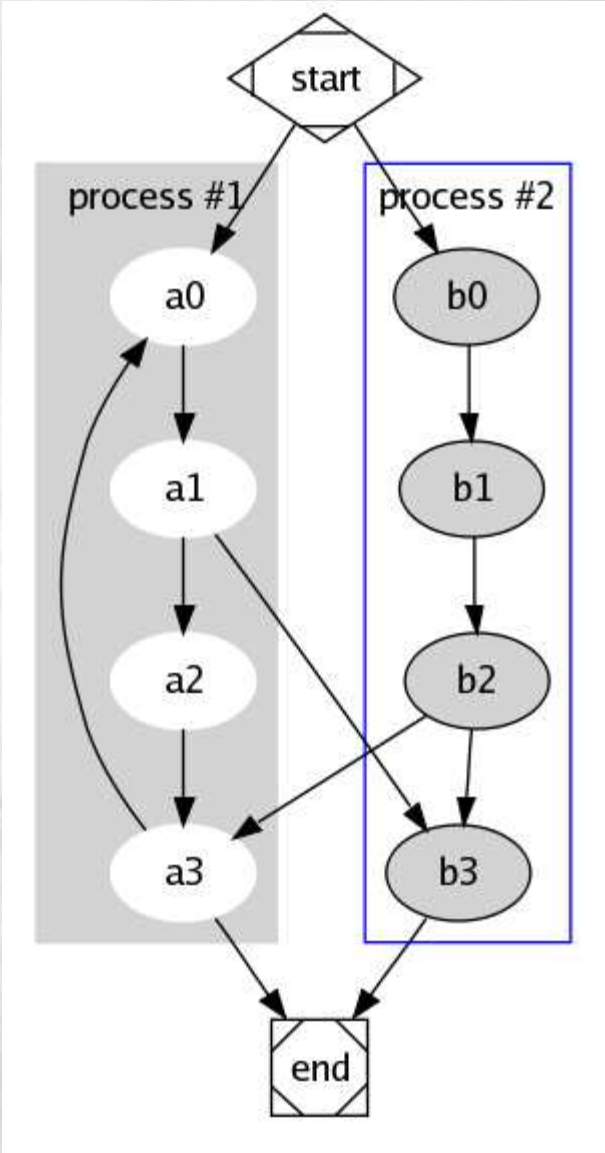
seqDbg* api
(proposed)

```
seqDbgRun (run | stop)
seqDbgStep (count)
seqDbgForce (state)
seqDbgBpSet (state | expr)
seqDbgBpEnable (…)
seqDbgBpClear (…)
seqDbgBpWait (…)
……
```

future/on-going

- graphical state program output (graphviz/...)
- master/slave api for redundant IOCs
- suggestions?


LR_4
S(A)
S(b)
LR_6
S(b)
LR_8
SS(b)
S(a)
S(a)
LR_2
S(a)
SS(B)
S(b)
LR_0
SS(a)
LR_5
S(b)
SS(S)
S(a)
LR_7
LR_1
S($end)
LR_3


start
process #1
process #2
a0
b0
a1
b1
a2
b2
a3
b3
end

fin

comments/suggestions to
lazmo @ slac.stanford.edu