64 bit Antelope

where Antelope has not gone before

Daniel Quinlan, BRTT

Why?

- large files
- limits on memory (~ 2 Gbytes)
- database size
- use additional physical memory
- support available on all platforms (almost)
- 32 bit more difficult on SuSE Linux
- Matlab supports only 64 bit on Solaris

How does source change?

- ints generally become longs
- string and byte array indexes become long
  - gettbl(Tbl *list, long i)
  - memdup(void *a, long n)
  - long pushstr(void **vstack, char *s)
  - bitset(Bitvector *bvec, long i)

Datascope integers -> 64 bit

```
long nsamp, n ;
dbgetv(db, “wfdisc”, “nsamp”, &nsamp, ...
dbquery(db, dbFIELD_COUNT, &n)
```

But not every int

Still (32 bit) int

- file descriptors
- boolean return codes
- network protocol (eg, orb) ints
- ints in saved data structures (for compatibility)
- pktid stays int (max 2G packets in orb => ~ 1000 Gbyte orb of 500 byte packets)

traces

- generally, nsamp -> 64 bits
- but not in
  - trfilter_segs, trfilter_pkts
  - msdget/msdput

Variable argument lists

More subtle problems

- compiler doesn't know the types of arguments, so it can't coerce arguments to the right type.
- In particular, the convention of using 0 to terminate a variable argument list is often wrong, eg

```
dbgetv(db, 0, "nsamp", &nsamp, 0 ) ;
```

- dbgetv wants a pointer (to a string), so it gets 64 bits from the stack, but only 32 bits of 0 were put onto the stack.

Problems

- `strtbl()`
- `concatpaths()`
- `strconcat()`
- `strjoin()`
- `dbgetv()`
- `dbputv()`
- `dbaddv()`

Its own special case: dbquery

```
int ntables ;

dbquery(db, dbTABLE_COUNT, &ntables);
```

Other issues

- existing longs are often meant as 32 bit integers
- long long should probably stay as long in 64 bit
- replace with int64_t, uint64_t

But format has to change: %lld vs %ld, depending on 32 vs 64 bit compilation

Where ints hold pointers

- This doesn’t work any more. Use lint.

missing include files

```
s = strdup("");
```

without

```
#include <string.h>
```

s points to garbage.

Pay attention to complaints like

```
warning: implicit function declaration:
```

Consistency

- generate prototypes
  - -auxinfo (gcc)
  - -xP (Solaris cc)
  - protoize
- put them into include files
- include these files, both in the implementation and where the routines are used.

Miscellaneous

- use %p to print pointers
- look carefully at constants like 0xffffffff -- promotion to 64 bit will almost certainly be wrong.
- new script 64bit attempts to point out, may fix many common problems like the ending 0)

Jettison K&R c style

- no implicit int -- e.g. Don't use

```
main(argc, char **argv)
```

- no inline prototypes

```
double atof() ;
```

Other changes

- CVS -> git
  - http://youtube.com/watch?v=4XpnKHJAok8
- c99
- certify

Schema change Rules

- **Easy**
  - size/format
  - add new fields
- **Harder**
  - change names
- **Hardest**
  - change keys
  - change tables

Problems

- compatibility with other flat file css3.0 implementations

schema changes

- nsamp: 8 digits
- id fields: 8 digits

non-64 bit changes

- make dir/dfile larger
- make grname and srname larger

Bolder changes

- change ondate/offdate to time/endtime
- eliminate jdate

Bold (and unlikely) changes

- redo calibration table (again)
- eliminate stage table altogether
- eliminate instrument and sensor table
- add net, loc codes to primary key