```
#|
```



```
|#

;              Proof of the Correctness of a LOG2 Program
```

EVENT: Start with the library "mc20-2" using the compiled version.

```
#|

The following C function computes the integer logarithm (base 2) of a
nonnegative integer.  We proved the correctness of the binary of this
C function.  The binary is produced by Gnu C compiler.

The proof described here was worked out with Matt Kaufmann and Bill
Pierce.

/* computes the integer logarithm of a nonnegative integer. */
log2(int n)
{
  int log = 0;
  while (n > 1) {
    log++;
```

```
    n /= 2;}
  return(log);
}
```

Here is the MC68020 assembly code of the above LOG2 program. The code is generated by "gcc -O".

```
0x22ce <log2>:          linkw fp,#0
0x22d2 <log2+4>:        movel d2,sp@-
0x22d4 <log2+6>:        movel fp@(8),d0
0x22d8 <log2+10>:       clrl d1
0x22da <log2+12>:       bra 0x22e6 <log2+24>
0x22dc <log2+14>:       addql #1,d1
0x22de <log2+16>:       tstl d0
0x22e0 <log2+18>:       bge 0x22e4 <log2+22>
0x22e2 <log2+20>:       addql #1,d0
0x22e4 <log2+22>:       asrl #1,d0
0x22e6 <log2+24>:       movel #1,d2
0x22e8 <log2+26>:       cmpl d0,d2
0x22ea <log2+28>:       blt 0x22dc <log2+14>
0x22ec <log2+30>:       movel d1,d0
0x22ee <log2+32>:       movel fp@(-4),d2
0x22f2 <log2+36>:       unlk fp
0x22f4 <log2+38>:       rts

<log2>:       0x4e56  0x0000  0x2f02  0x202e  0x0008  0x4281  0x600a  0x5281
<log2+16>:    0x4a80  0x6c02  0x5280  0xe280  0x7401  0xb480  0x6df0  0x2001
<log2+32>:    0x242e  0xfffc  0x4e5e  0x4e75

'(78      86      0       0       47      2       32      46
  0       8       66      129     96      10      82      129
  74      128     108     2       82      128     226     128
  116     1       180     128     109     240     32      1
  36      46      255     252     78      94      78      117)
|#

; in Nqthm, log2 is defined as:
```

DEFINITION:
LOG2-CODE
= '(78 86 0 0 47 2 32 46 0 8 66 129 96 10 82 129 74 128
108 2 82 128 226 128 116 1 180 128 109 240 32 1 36
46 255 252 78 94 78 117)

```
; we define the Nqthm counterpart of log2.
```

DEFINITION:
log2 (n, *log2*)
= **if** $1 < n$ **then** log2 ($n \div 2$, $1 + $ *log2*)
else *log2* **endif**

```
; the clock.
```

DEFINITION:
log2-t0 (n)
= **if** $1 < n$ **then** splus (7, log2-t0 ($n \div 2$))
else 7 **endif**

DEFINITION: log2-t (n) = splus (5, log2-t0 (n))

```
; an induction hint.
```

DEFINITION:
log2-induct (s, n, *log2*)
= **if** $1 < n$ **then** log2-induct (stepn (s, 7), $n \div 2$, $1 +$ *log2*)
else **t** **endif**

```
; the preconditions on the initail state.
```

DEFINITION:
log2-statep (s, n)
= ((mc-status (s) = 'running)
∧ evenp (mc-pc (s))
∧ rom-addrp (mc-pc (s), mc-mem (s), 40)
∧ mcode-addrp (mc-pc (s), mc-mem (s), LOG2-CODE)
∧ ram-addrp (sub (32, 8, read-sp (s)), mc-mem (s), 16)
∧ (n = iread-mem (add (32, read-sp (s), 4), mc-mem (s), 4))
∧ ($n \in \mathbf{N}$))

```
; an intermediate state.
```

DEFINITION:
log2-s0p (s, n, *log2*)
= ((mc-status (s) = 'running)
∧ evenp (mc-pc (s))
∧ rom-addrp (sub (32, 24, mc-pc (s)), mc-mem (s), 40)
∧ mcode-addrp (sub (32, 24, mc-pc (s)), mc-mem (s), LOG2-CODE)
∧ ram-addrp (sub (32, 4, read-an (32, 6, s)), mc-mem (s), 16)
∧ (n = iread-dn (32, 0, s))
∧ (*log2* = iread-dn (32, 1, s))
∧ int-rangep (*log2* + n, 32)
∧ (*log2* $\in \mathbf{N}$)
∧ ($n \in \mathbf{N}$))

```
; from the initial state to s0:  s --> s0.
```

THEOREM: log2-s-s0
$\text{log2-statep}(s,\ n) \rightarrow \text{log2-s0p}(\text{stepn}(s,\ \mathtt{5}),\ n,\ \mathtt{0})$

THEOREM: log2-s-s0-else
$\text{log2-statep}(s,\ n)$
$\rightarrow\ ((\text{linked-rts-addr}(\text{stepn}(s,\ \mathtt{5})) = \text{rts-addr}(s))$
$\quad \wedge\ (\text{linked-a6}(\text{stepn}(s,\ \mathtt{5})) = \text{read-an}(\mathtt{32},\ \mathtt{6},\ s))$
$\quad \wedge\ (\text{read-rn}(\mathtt{32},\ \mathtt{14},\ \text{mc-rfile}(\text{stepn}(s,\ \mathtt{5})))$
$\qquad = \ \text{sub}(\mathtt{32},\ \mathtt{4},\ \text{read-sp}(s)))$
$\quad \wedge\ (\text{rn-saved}(\text{stepn}(s,\ \mathtt{5})) = \text{read-dn}(\mathtt{32},\ \mathtt{2},\ s)))$

THEOREM: log2-s-s0-rfile
$(\text{log2-statep}(s,\ n) \wedge \text{d3-7a2-5p}(rn))$
$\rightarrow\ (\text{read-rn}(oplen,\ rn,\ \text{mc-rfile}(\text{stepn}(s,\ \mathtt{5})))$
$\quad = \ \text{read-rn}(oplen,\ rn,\ \text{mc-rfile}(s)))$

THEOREM: log2-s-s0-mem
$(\text{log2-statep}(s,\ n) \wedge \text{disjoint}(x,\ k,\ \text{sub}(\mathtt{32},\ \mathtt{8},\ \text{read-sp}(s)),\ \mathtt{16}))$
$\rightarrow\ (\text{read-mem}(x,\ \text{mc-mem}(\text{stepn}(s,\ \mathtt{5})),\ k) = \text{read-mem}(x,\ \text{mc-mem}(s),\ k))$

```
; s0 --> exit.
; base case: s0 --> exit.
```

THEOREM: log2-s0-sn-base
$(\text{log2-s0p}(s,\ n,\ log2) \wedge (\mathtt{1} \not< n))$
$\rightarrow\ ((\text{mc-status}(\text{stepn}(s,\ \mathtt{7})) = \mathtt{'running})$
$\quad \wedge\ (\text{mc-pc}(\text{stepn}(s,\ \mathtt{7})) = \text{linked-rts-addr}(s))$
$\quad \wedge\ (\text{iread-dn}(\mathtt{32},\ \mathtt{0},\ \text{stepn}(s,\ \mathtt{7})) = log2)$
$\quad \wedge\ (\text{read-rn}(\mathtt{32},\ \mathtt{14},\ \text{mc-rfile}(\text{stepn}(s,\ \mathtt{7}))) = \text{linked-a6}(s))$
$\quad \wedge\ (\text{read-rn}(\mathtt{32},\ \mathtt{15},\ \text{mc-rfile}(\text{stepn}(s,\ \mathtt{7})))$
$\qquad = \ \text{add}(\mathtt{32},\ \text{read-an}(\mathtt{32},\ \mathtt{6},\ s),\ \mathtt{8})))$

THEOREM: log2-s0-sn-base-rfile
$(\text{log2-s0p}(s,\ n,\ log2) \wedge (\mathtt{1} \not< n) \wedge \text{d2-7a2-5p}(rn) \wedge (oplen \leq \mathtt{32}))$
$\rightarrow\ (\text{read-rn}(oplen,\ rn,\ \text{mc-rfile}(\text{stepn}(s,\ \mathtt{7})))$
$\quad = \ \textbf{if}\ \text{d3-7a2-5p}(rn)\ \textbf{then}\ \text{read-rn}(oplen,\ rn,\ \text{mc-rfile}(s))$
$\qquad \textbf{else}\ \text{head}(\text{rn-saved}(s),\ oplen)\ \textbf{endif})$

THEOREM: log2-s0s-n-base-mem
$(\text{log2-s0p}(s,\ n,\ log2) \wedge (\mathtt{1} \not< n))$
$\rightarrow\ (\text{read-mem}(x,\ \text{mc-mem}(\text{stepn}(s,\ \mathtt{7})),\ k) = \text{read-mem}(x,\ \text{mc-mem}(s),\ k))$

```
; induction case: s0 --> s0.
```

THEOREM: log2-rangep-la
(int-rangep $(m + n,$ 32) $\wedge$ (1 $< n$))
$\rightarrow$ int-rangep (1 + $(m + (n \div 2))$, 32)

THEOREM: log2-s0-s0
(log2-s0p (s, n, *log2*) $\wedge$ (1 $< n$))
$\rightarrow$ (log2-s0p (stepn (s, 7), $n \div$ 2, 1 + *log2*)
$\wedge$ (read-rn (*oplen*, 14, mc-rfile (stepn (s, 7)))
= read-rn (*oplen*, 14, mc-rfile (s)))
$\wedge$ (linked-a6 (stepn (s, 7)) = linked-a6 (s))
$\wedge$ (linked-rts-addr (stepn (s, 7)) = linked-rts-addr (s))
$\wedge$ (read-mem (x, mc-mem (stepn (s, 7)), k)
= read-mem (x, mc-mem (s), k))
$\wedge$ (rn-saved (stepn (s, 7)) = rn-saved (s)))

THEOREM: log2-s0-s0-rfile
(log2-s0p (s, n, *log2*) $\wedge$ (1 $< n$) $\wedge$ d3-7a2-5p (*rn*))
$\rightarrow$ (read-rn (*oplen*, *rn*, mc-rfile (stepn (s, 7)))
= read-rn (*oplen*, *rn*, mc-rfile (s)))

```
; put together.
```

THEOREM: log2-s0-sn
log2-s0p (s, n, *log2*)
$\rightarrow$ ((mc-status (stepn (s, log2-t0 (n))) = 'running)
$\wedge$ (mc-pc (stepn (s, log2-t0 (n))) = linked-rts-addr (s))
$\wedge$ (iread-dn (32, 0, stepn (s, log2-t0 (n))) = log2 (n, *log2*))
$\wedge$ (read-rn (32, 14, mc-rfile (stepn (s, log2-t0 (n))))
= linked-a6 (s))
$\wedge$ (read-rn (32, 15, mc-rfile (stepn (s, log2-t0 (n))))
= add (32, read-an (32, 6, s), 8))
$\wedge$ (read-mem (x, mc-mem (stepn (s, log2-t0 (n))), k)
= read-mem (x, mc-mem (s), k)))

THEOREM: log2-s0-sn-rfile
(log2-s0p (s, n, *log2*) $\wedge$ d2-7a2-5p (*rn*) $\wedge$ (*oplen* $\leq$ 32))
$\rightarrow$ (read-rn (*oplen*, *rn*, mc-rfile (stepn (s, log2-t0 (n))))
= **if** d3-7a2-5p (*rn*) **then** read-rn (*oplen*, *rn*, mc-rfile (s))
else head (rn-saved (s), *oplen*) **endif**)

```
; correctness.
```

THEOREM: log2-correct
log2-statep (s, n)
$\rightarrow$ ((mc-status (stepn (s, log2-t (n))) = 'running)

$\wedge \quad (\text{mc-pc}\,(\text{stepn}\,(s,\ \text{log2-t}\,(n))) = \text{rts-addr}\,(s))$
$\wedge \quad (\text{iread-dn}\,(\mathtt{32},\ \mathtt{0},\ \text{stepn}\,(s,\ \text{log2-t}\,(n))) = \text{log2}\,(n,\ \mathtt{0}))$
$\wedge \quad (\text{read-an}\,(\mathtt{32},\ \mathtt{6},\ \text{stepn}\,(s,\ \text{log2-t}\,(n))) = \text{read-an}\,(\mathtt{32},\ \mathtt{6},\ s))$
$\wedge \quad (\text{read-an}\,(\mathtt{32},\ \mathtt{7},\ \text{stepn}\,(s,\ \text{log2-t}\,(n)))$
$\qquad = \quad \text{add}\,(\mathtt{32},\ \text{read-an}\,(\mathtt{32},\ \mathtt{7},\ s),\ \mathtt{4})))$

THEOREM: log2-rfile
$(\text{log2-statep}\,(s,\ n) \wedge \text{d2-7a2-5p}\,(rn) \wedge (oplen \leq \mathtt{32}))$
$\rightarrow \quad (\text{read-rn}\,(oplen,\ rn,\ \text{mc-rfile}\,(\text{stepn}\,(s,\ \text{log2-t}\,(n))))$
$\qquad = \quad \text{read-rn}\,(oplen,\ rn,\ \text{mc-rfile}\,(s)))$

THEOREM: log2-mem
$(\text{log2-statep}\,(s,\ n) \wedge \text{disjoint}\,(x,\ k,\ \text{sub}\,(\mathtt{32},\ \mathtt{8},\ \text{read-sp}\,(s)),\ \mathtt{16}))$
$\rightarrow \quad (\text{read-mem}\,(x,\ \text{mc-mem}\,(\text{stepn}\,(s,\ \text{log2-t}\,(n))),\ k) = \text{read-mem}\,(x,\ \text{mc-mem}\,(s),\ k))$

```
; the correctness of the Nqthm function log2.
```

THEOREM: log2-log
$(i \in \mathbf{N}) \rightarrow (\text{log2}\,(n,\ i) = (i + \log\,(\mathtt{2},\ n)))$

```
; 2^log2(n) <= n.
```

THEOREM: log2-thm1
$(\mathtt{1} < n) \rightarrow (n \not< \exp\,(\mathtt{2},\ \text{log2}\,(n,\ \mathtt{0})))$

```
; n < 2^(log2(n)+1).
```

THEOREM: log2-thm2
$n < \exp\,(\mathtt{2},\ 1 + \text{log2}\,(n,\ \mathtt{0}))$

Index