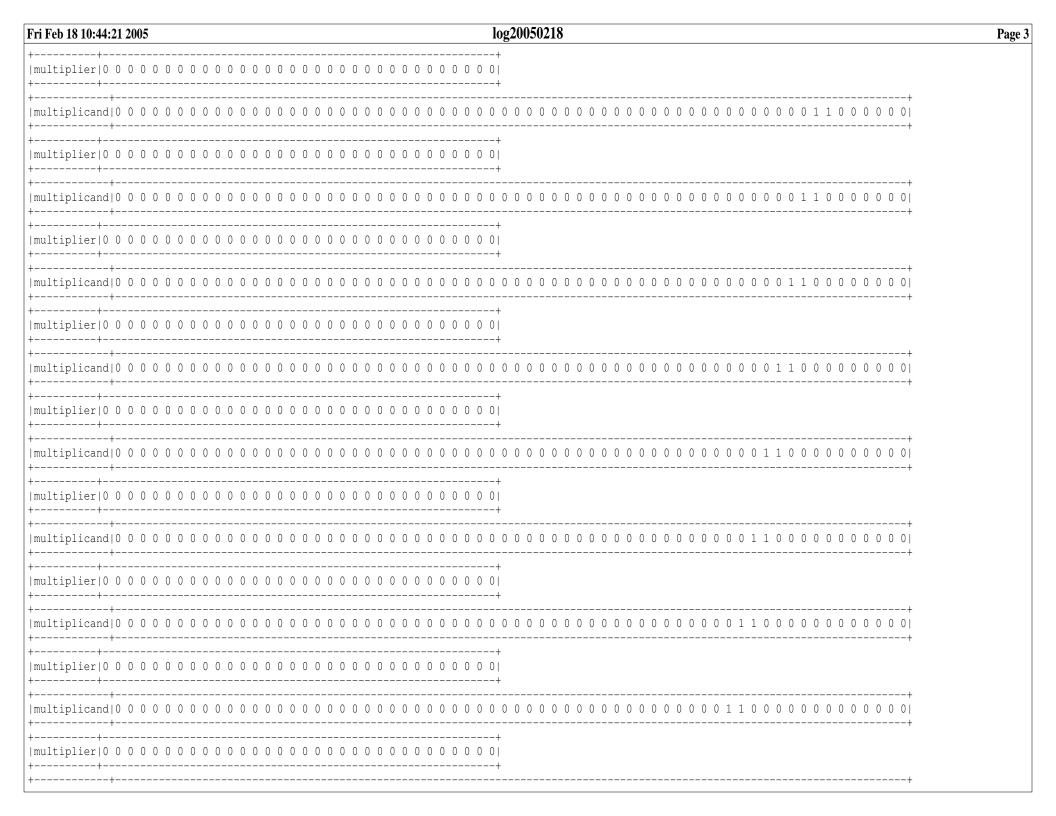
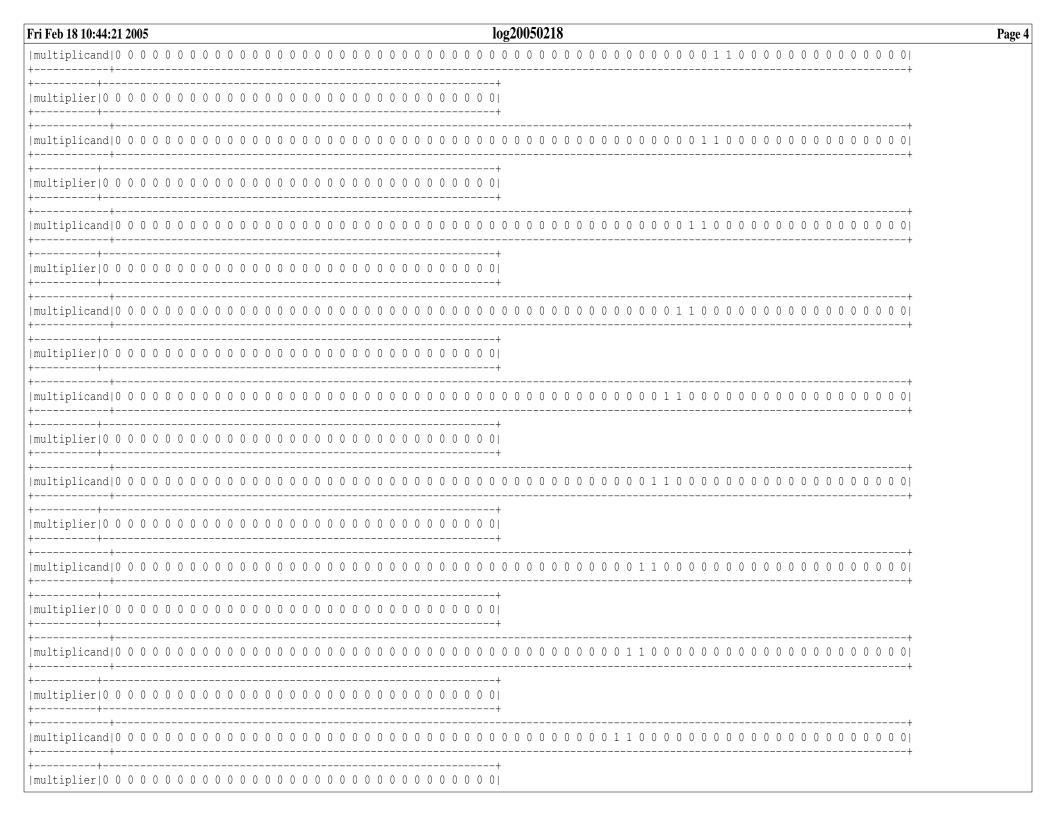
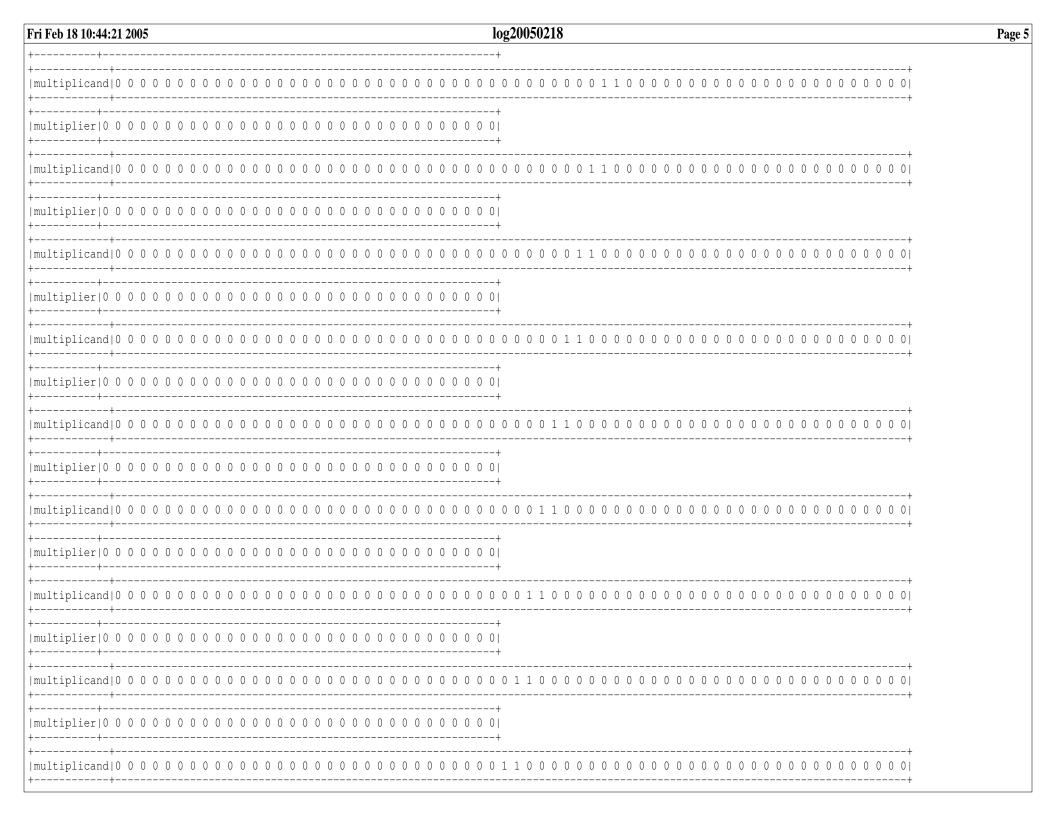
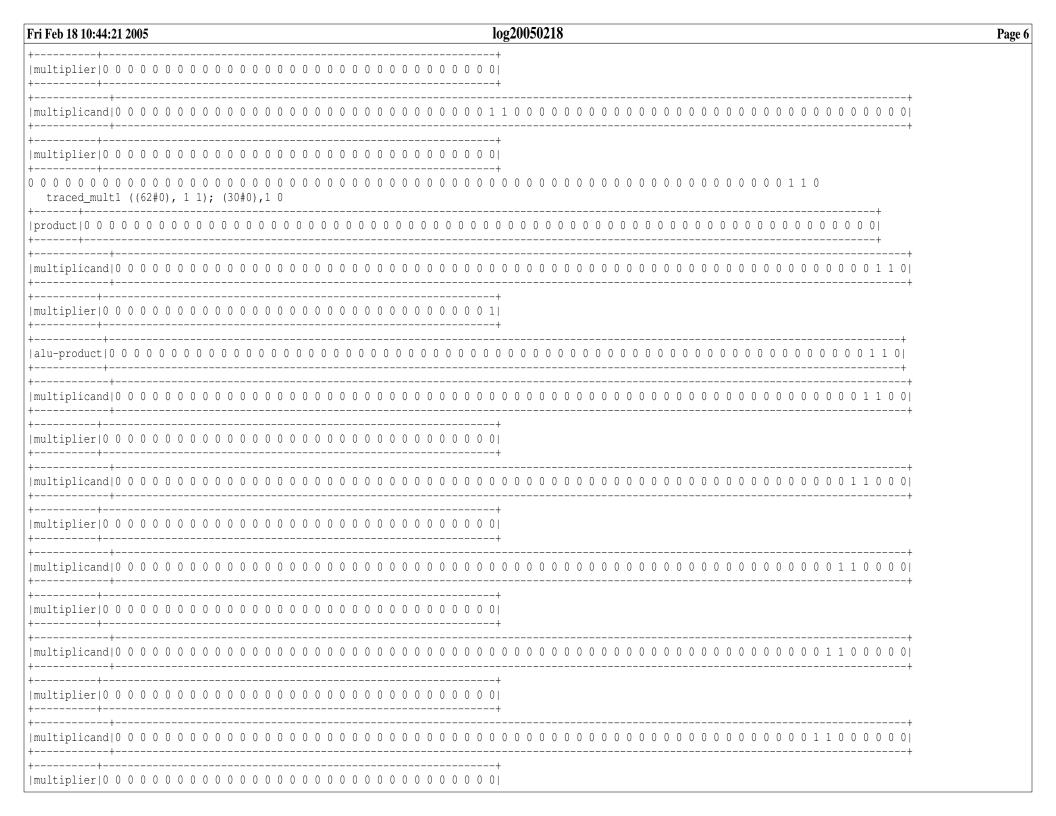
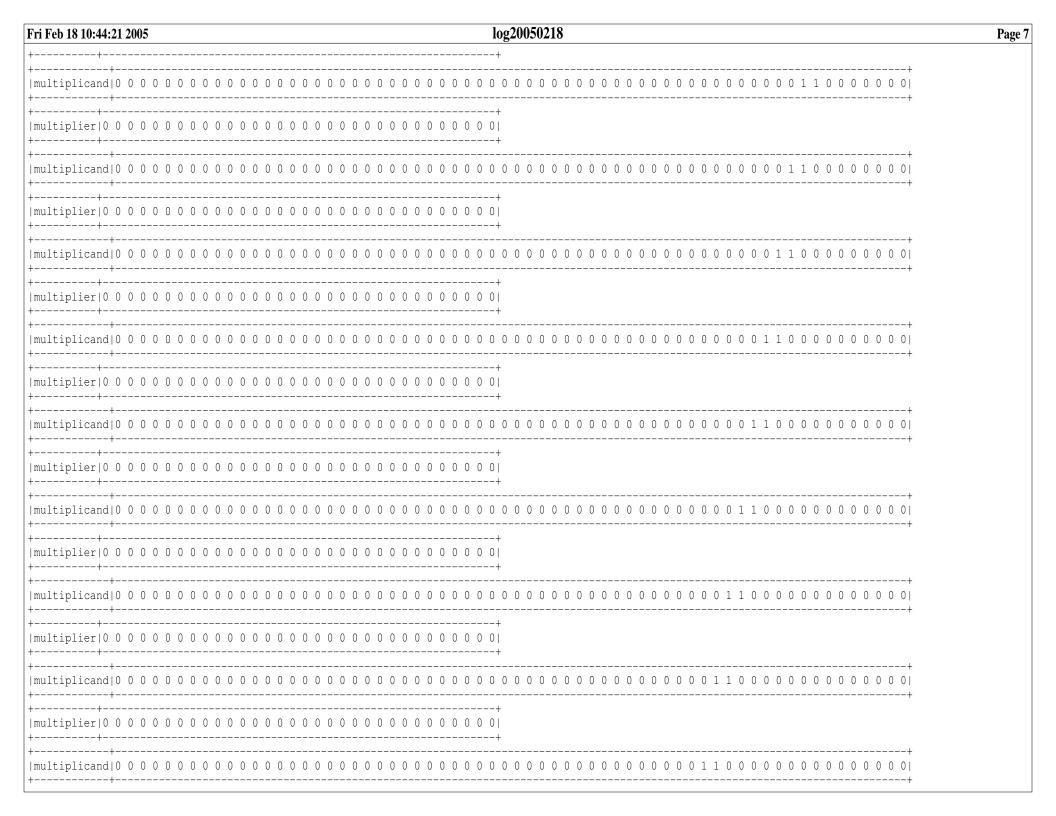
```
load'mult.ijs'
 alu 64
3:0
NB. a and b are each 64-bit summands
NB. the result is a 64-bit sum
'a b' =. y.
(64#2) rep (base x: a) + base x: b
а
base a
4.29497e9
 signed value a
|value error: signed_value
   signed_value a
 load'arith.ijs'
 signed value a
_2
 a = : (64#2) \text{ rep } _2
b
alu 64 a;b
c = : (64#2) rep _1
 С
alu 64 b;c
alu 64 3 4
not_equal
~:
 last
1&from
 а
1 from a
 0 from a
 _1 from a
0
 mult1
3:0
('multiplicand'; 'multiplier') =. y.
product=. 64#0
count=. 32
while. O not_equal count
do. control =. last multiplier
  if. control
```

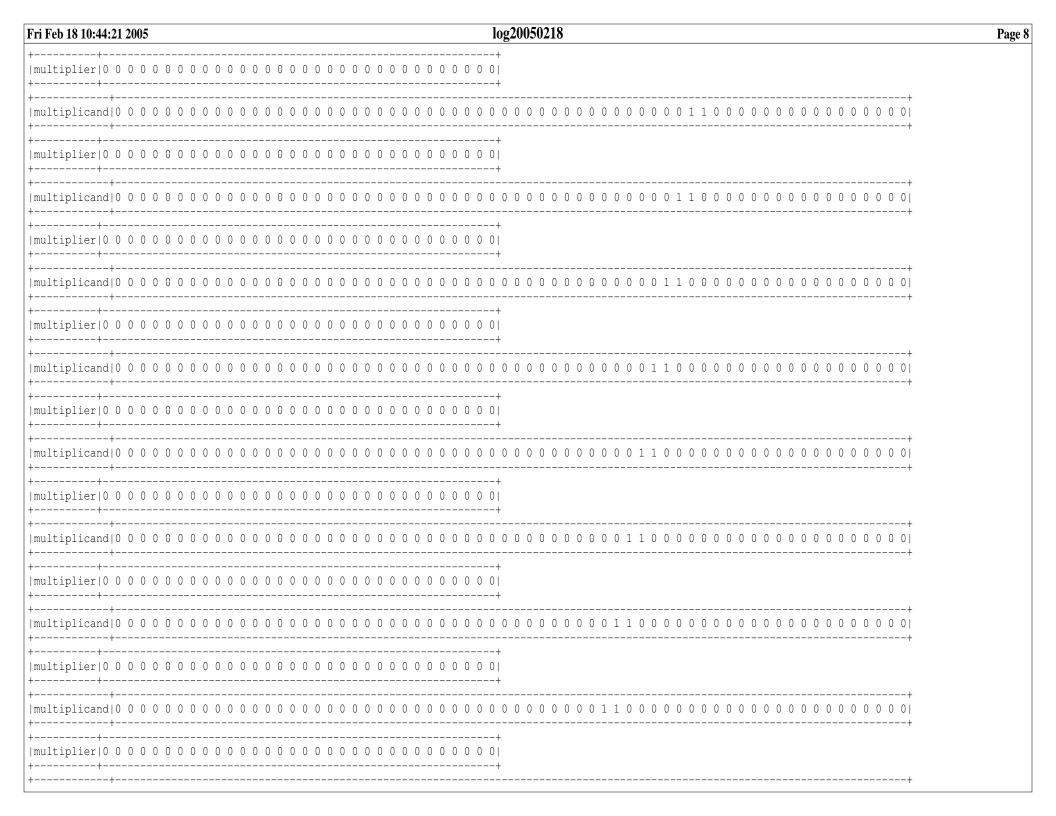


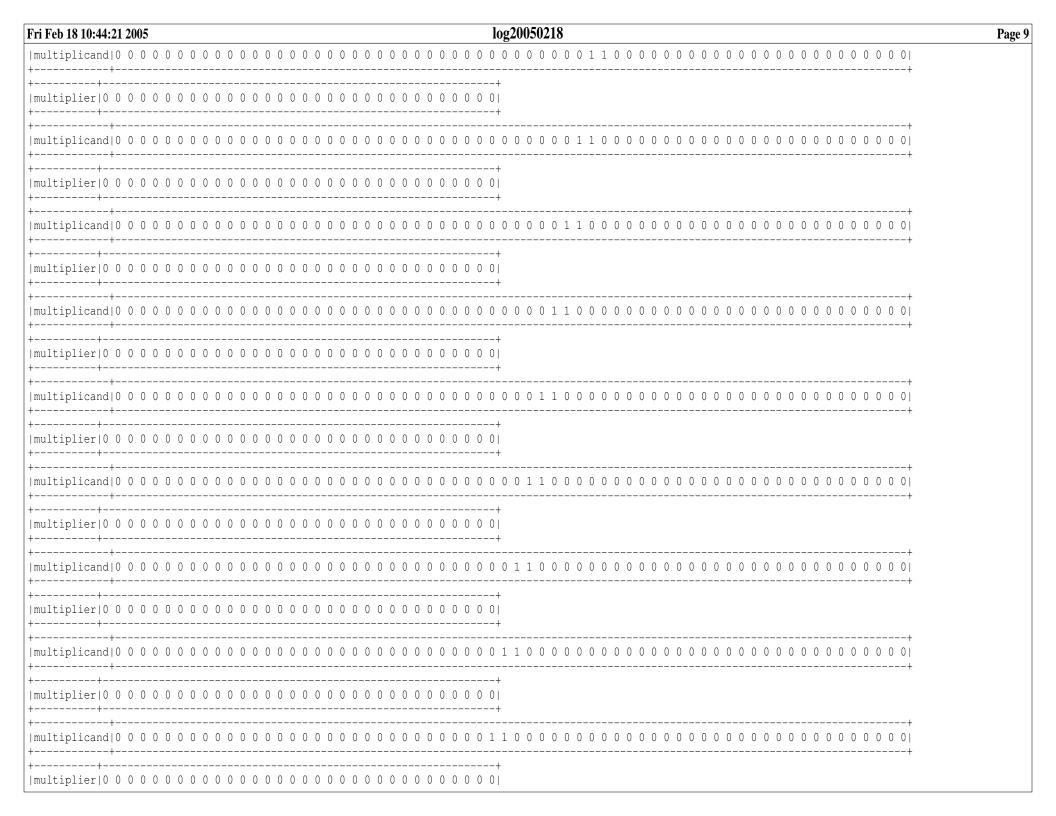


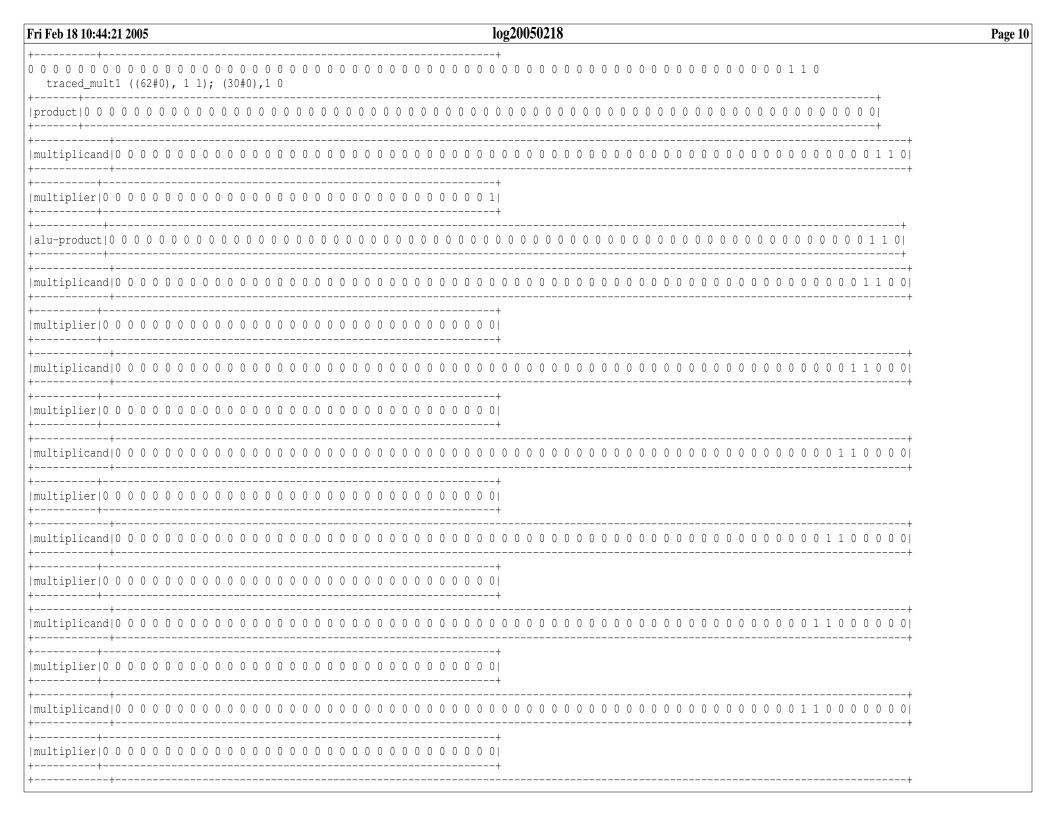


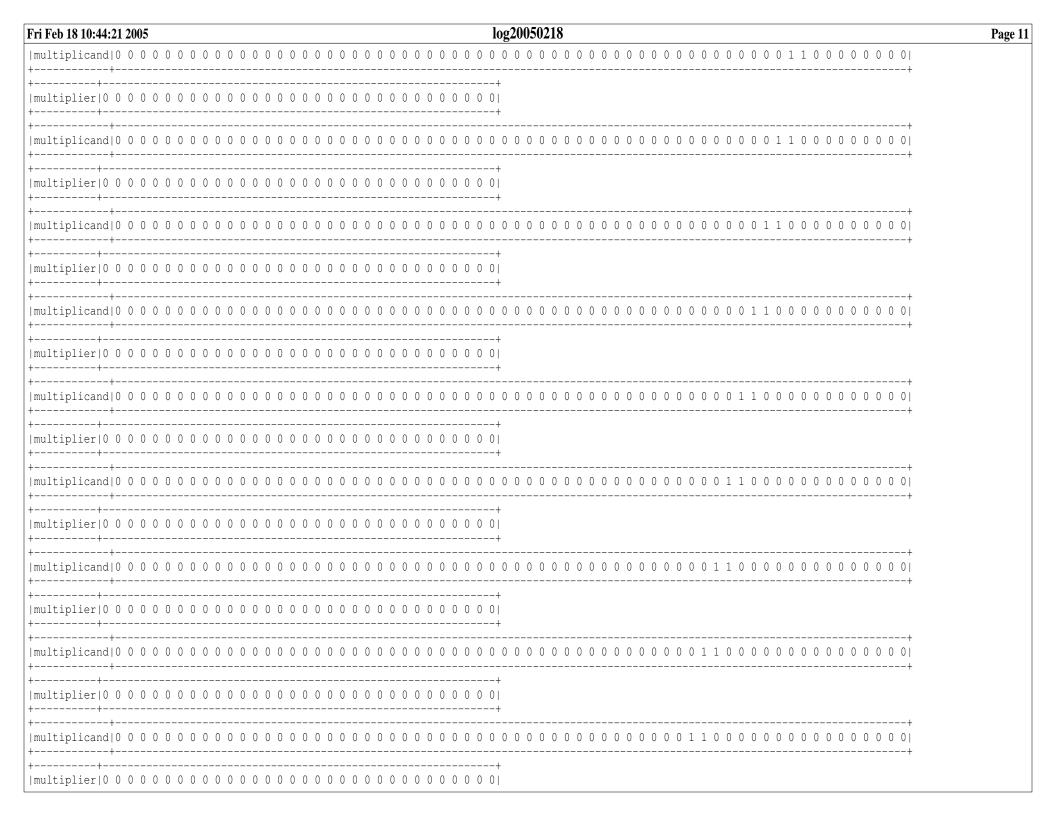


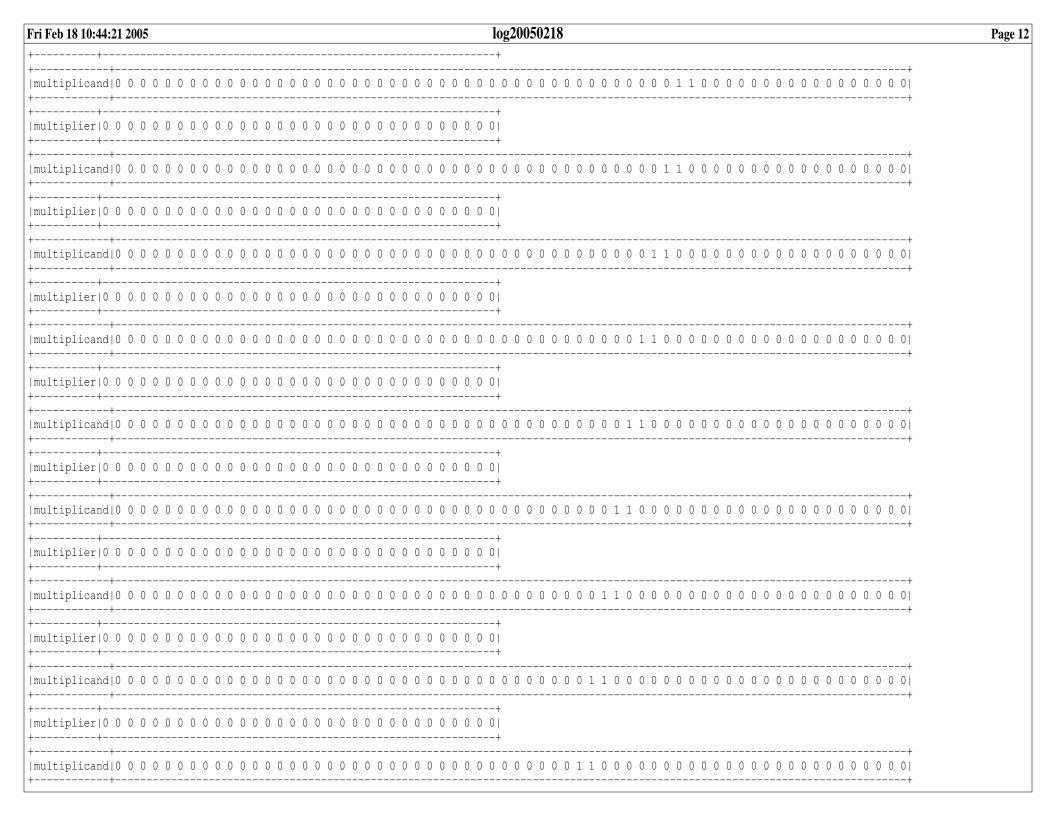


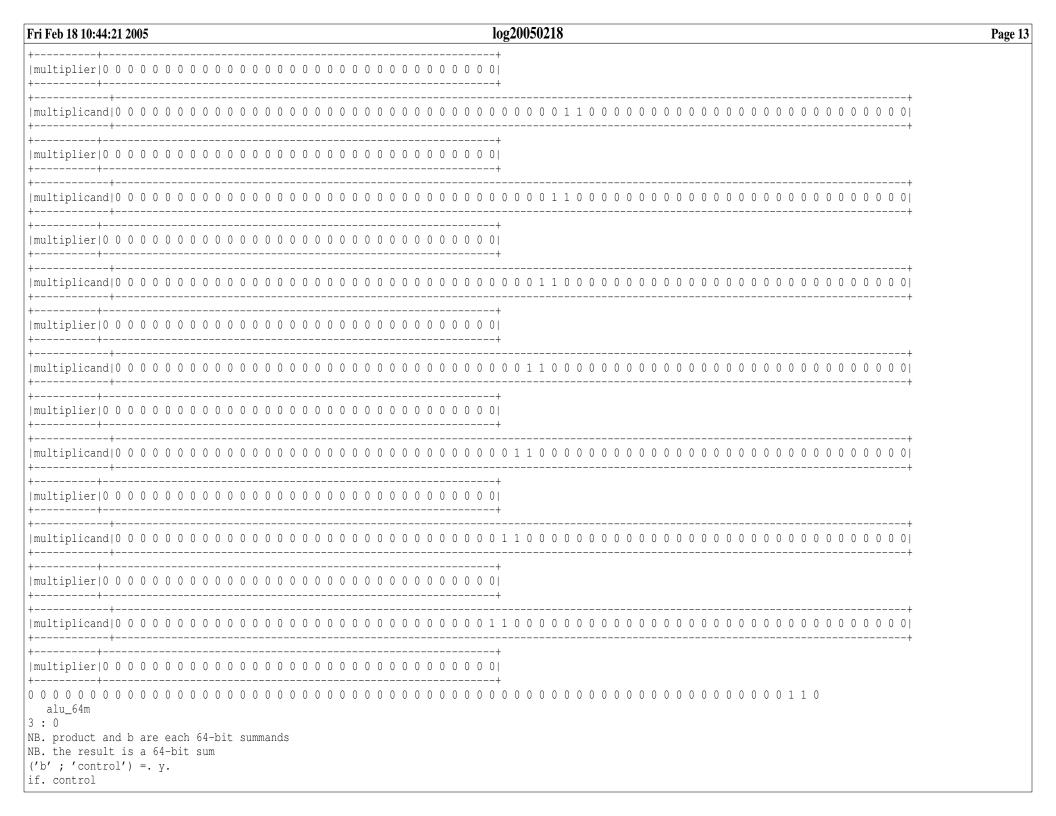












```
do. product =: (64#2) rep (base x: product) + base x: b
end.
 signed_value mult1m ((62#0), 1 1); (30#0), 1 0
 alu_32
3 : 0
NB. a and b are each 32-bit summands
NB. the result is a 32-bit sum
('a' ; 'b') = . y.
(32#2) rep (base x: a) + base x: b
 a = : (31 # 1), 0
 b=: (30#0), 1 0
 alu_32 a;b
c = : 32 # 1
 alu 32 b;c
b
base alu 32 100 ; 28
128
 mult2 ((30#0), 1 1); (30#0), 1 0
mult2 (0,31#1):0,31#1
base mult2 (0,31#1);0,31#1
4611686014132420609
 base 0,31#1x
2147483647
 (base 0,31#1x)^2
4611686014132420609
 base mult2m ((30#0), 1 1); (30#0), 1 0
```