PROBABILISTIC TIMING OF COMBINATIONAL CIRCUITS

William Bricken
April, 1995

This memo contains notes on circuit configuration and timing as a function of the probability of input values. Motivation: situated silicon

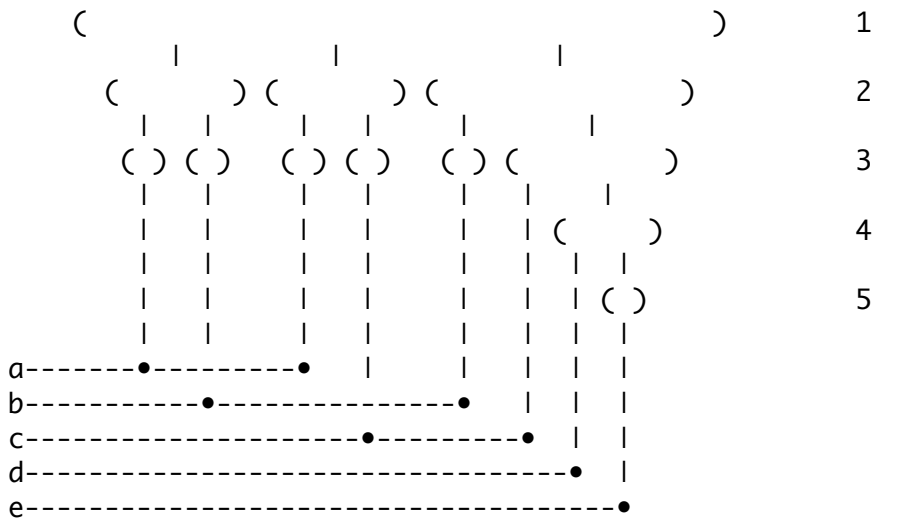
Consider circuitry that can alter its configuration dynamically depending on the distribution of incoming signals/values, thus responding to input circumstances with optimal layout. Here are some preliminary notes on logic structure which is sensitive to the probability distribution of the incoming values. The example of a full-adder follows general considerations of propagation time through combinatorial distinction networks.

Time as Depth

Assume a parens expression is minimized by Extract. Consider this expression:

E = (((a) (b)) ((a) (c)) ((b) (c (d (e)))))

Each parens stands for a dnode in a dnet, with letters referring to signals from arbitrary structures or subcircuits. Here is the graph of the example:



The DEPTH of a variable is the number of parens one must cross to reach the outermost space. Each crossing takes one dnode-time-tick.

The distribution of time ticks under all evaluations of each variable represents the complete sequential evaluation-time of the expression.

The BDD (binary decision diagram) of the above circuit is:

```
a=( )                          E = (b c)
      b=( )                          E =
      b=                             E= (c)
a=                             E = ((b) (c (d (e))))
      b=( )                          E = c (d (e))
            c=( )                          E = ( )
            c=                             E = d (e)
                  d=( )                          E = ( )
                  d=                             E = (e)
      b=                             E =
```

Let the probability of each variable value be .5, indexed as (1/2)^n

binding	*probability*	*index*	*value*
a=b=()	.25	2	E =
a=b=	.25	2	E = ( )
a=(b)=()			
c=()	.125	3	E =
c=	.125	3	E = ( )
a=(b)=			
c=()	.125	3	E =
c=			
d=()	.0625	4	E = ( )
d=			
e=()	.03125	5	E =
e=	.03125	5	E = ( )

Totals for probability of output value;

```
E = () at  .46875
E =    at  .53125
```

Also associated with the evaluation of an expression is the time-probability of evolution. To count time, begin the initial cross at variable binding time.

In the example:

```
a=b=( )

      Tick 0      ( ((a)  (b)) ((a) (c)) ((b) (c (d (e)))) )
      Tick 1      ( (())(())) ((())(c)) ((())(c (d (e)))) )
      Tick 2      ( (        ) (    (c)) (    (c (d (e)))) )
      Tick 3      <void>

(a)=(b)=( )

      Tick 1      ( (( )  ( )) (( ) (c)) (( ) (c (d (e)))) )
      Tick 2      (                                        )

a=(b)=c=( )

      Tick 1      ( ((())( )) ((())(())) (() (() (d (e)))) )
      Tick 2      (           (        )                   )
      Tick 3      <void>

a=(b)=(c)=()

      Tick 1      ( ((())( )) ((()) ( )) (( ) (  (d (e)))) )
      Tick 2      (                                        )

(a)=b=c=()

      Tick 1      ( (( )(())) (( )(())) ((()) (()(d (e)))) )
      Tick 2      (                     (                ) )
      Tick 3      <void>

(a)=b=(c)=d=()

      Tick 1      ( (( )(())) (( ) ( )) ((()) (  (()(e)))) )
      Tick 2      (                     (     (         )) )
      Tick 3      (                                        )

(a)=b=(c)=(d)=e=()

      Tick 1      ( (( )(())) (( ) ( )) ((()) (  ( (()))) ) )
      Tick 2      (                     (     (  (     ))) )
      Tick 3      (                     (                ) )
      Tick 4      <void>

(a)=b=(c)=(d)=(e)=()

      Tick 1      ( (( )(())) (( ) ( )) ((()) (  (  ( )))) )
      Tick 2      (                     (     (         )) )
      Tick 3      (                                        )
```

Time of output:

2-ticks	.375
3-ticks	.59375
4-ticks	.03125

Expectation:

2*3/8 + 3*19/32 + 4*1/32 = .75 + 1.78125 + .125 = 2.65625 ticks

[NOTE: assigning probabilities to input values is stronger than "don't care" since we can assign a degree of caring.]

Time is a function of the configuration of the expression, which can be transformed. We want to minimize the EXPECTED TIME (Et) of evaluation, relative to the number of variables and dnodes, and the expected value of inputs (Ev).

Example of Full-adder

Flat (sum of products) sum:

((a) b c) ((b) a c) ((c) a b) ((a)(b)(c))

Flat carry:

((a)(b)) ((a)(c)) ((b)(c))

Flat, non-sharing full-adder:

sum: 12 input lines, 10 dnodes, Et = 2 Ev = .5

a= ((b) c) ((c) b)
b= c

	E=	T=	P=
(a)=(b)=(c)=( )		2	3
(a)=(b)=c=( )	( )	2	3
(a)=b=(c)=( )	( )	2	3
(a)=b=c=( )		2	3
a=(b)=(c)=( )	( )	2	3
a=(b)=c=( )		2	3
a=b=(c)=( )		2	3
a=b=c=( )	( )	2	3

carry: 6 inviars, 9 dnodes, Et = 2.125 Ev = .5

	E=	T=	P=
(a)=(b)=(c)=( )		2	3
(a)=(b)=c=( )		2	3
(a)=b=(c)=( )		2	3
(a)=b=c=( )	( )	2	3
a=b=(c)=( )	( )	2	3
a=(b)=c=( )	( )	2	3
a=(b)=(c)=( )		2	3
a=b=c=( )	( )	3	3

[NOTE: if both values are expected at the same time after eval, then the *slowest* of the Ets determines the circuit time.

Probabilistically Reconfigured Full-adder

The flat adder works best when Ev =.5

But assume Ea=.25 and Eb=.25. and Pcarry=.0625

sum: 12 in-vars, 10 dnodes, Et = 2 Ev = .391

			Ea=.75	*Ea=.25*
	E=	T=	P=	
(a)=(b)=(c)=( )		2	.029	.526
(a)=(b)=c=( )	( )	2	.035	.035
(a)=b=(c)=( )	( )	2	.082	.176
(a)=b=c=(		2	.105	.012
a=(b)=(c)=( )	( )	2	.082	.176
a=(b)=c=( )		2	.105	.012
a=b=(c)=( )		2	.246	.059
a=b=c=( )	( )	2	.316	.004

carry: 6 in-vars, 9 dnodes, Et = 2.125 Ev = .087

	E=	T=	P=	
(a)=(b)=(c)=( )		2	3	
(a)=(b)=c=( )		2	3	
(a)=b=(c)=( )		2	3	
(a)=b=c=( )	( )	2	3	.012
a=b=(c)=( )	( )	2	3	.059
a=(b)=c=( )	( )	2	3	.012
a=(b)=(c)=( )		2	3	
a=b=c=( )	( )	3	3	.004

Contrast with deeper version:

Deep sum: 6 in-vars, 6 dnodes,

(o (a b c)) ((a)(b)(c)) =

(((a)(b)) ((a)(c)) ((b)(c)) (a b c)) ((a)(b)(c))

Deep carry:

o = ((a)(b)) ((a)(c)) ((b)(c))

Et-carry: .996*2 + .004*3 = 2.04

and Et-sum = 2.47

	E=	T=	P=
(a)=(b)=(c)=( )		2	.526
(a)=(b)=c=( )	( )	3	.035
(a)=b=(c)=( )	( )	3	.176
(a)=b=c=( )		3	.012
a=(b)=(c)=( )	( )	3	.176
a=(b)=c=( )		3	.012
a=b=(c)=( )		3	.059
a=b=c=( )	( )	3	.004

Summary of results for adder

Configuration	*Prob-of-HI-input*	*sum-time*	*carry-time*
flat	.5	2	2.125
flat	.25	2	2.125
deep	.25	2.47	2.04

Summary: The probability distribution of input values is related to the average timing of a circuit. When this distribution is known (or can be dynamically computed), circuit layout can be specifically optimized.