Meeting with Dunwell, on Performance   March 21, '58

Bloch   Dias   Wallenski
McSorley   Blaauw   Koleky
Foss   Bloch   Cocke

New Rolling Frame

Sigma AU: - understood - cant make goals with tools we have today.
- but we can't give up here.
U. of Ill. propose faster speeds. - probably possible.
Standard Plug. units - frames
however IBM cannot put out AU a factor of 2 or 3 below others.
will depart from S.C. if necessary. ~~one~~ one way or another circuits
will appear - og. machine so

(Bloch) Basic-Sigma Interactions
- have defined LA interaction with I box. - some Timing has not.
instr. preparation now limited to 0.9 μs rate. - any improvement is
expensive. Basic with core storage Tentative 0.8 cycle 0.4 fetch.
- several Things in parallel, use 0.3 Transfer 0.6 for add
stand to improve, clocked to cores.
If index mem were faster - Transfer ~~0~~ is 0.3 anyway.
might gain 0.1 in instr prep. rate. - would reorganize design to clock at 0.3.
(Steve) Core array could run faster - may need freon - etc to pull down
0.3 is max prep rate.
simultaneous fetch + store a possibility.

(Blaauw) rate can be unused.

(Kolsky) Two types of I instr.
(1) Steady indexing rate — read out time main effect
(2) Incr., C+Br. type — total cycle.

suggest: multiple accumulators.
- going back to full wds.

(Dirac) what are cost figures of putting in

(Dunwell) one will lose .3 + .4 μs in going to another frame to H.S. reg.
instead of core mem next to unit.
need mem right in AU.
- capacity 16 to 256.

cryogenic a.u. - or mem speed disappears.

(Block) 3 in 1 concept. how would I box be redesigned?
(K) indexing ops could be broken up & fed to L.A.
fetch & store separately.
(Locke) - an instr mem tied directly to IAU. (1) need faster indexing (2) initial fetch time low.
higher speed -
- don't allow instr store except by transmit ops.
cut down compares - etc.

(Dunwell) assume core indexes connected to IAU only built in direct bus
to get 0.6 rate, half micro mem rate 0.5 μs

also assume an xtra core mem bussed to AU.
is this all one needs?

Index Cores 16 — now in.
Instr Storage ~~[illegible]~~ 2048
Fast, Data 64 — exclude instrs.
general Data 16384

lose homogenity of mem. complicates Fortran, etc.

Instr. buffer reg. — not necessarily in. — needs to be evaluated.
question of buffer as sigma only
— code not an exact mirror of machine —

(costs) (1) Buss Time
(2) Index Time
(3) AU Time.
(4) checking

changes which have reduced speed: (1) more complex design
(2) slower circ. speeds.
(3) slower mem.

are restrictions too severe on special mems?
— we should evaluate how much this does gain?

(D) Can we build core array directly into I box?
— is already done.
can look-ahead improve jump situation

(K) propose we have Branch assume I Br F will be taken
assume I Br N will not be taken
need Ind ≤0 >0 and + will look into — —

question of I box indicators is Br. on, ~~off~~ exception also or only on AU ?

→. I box ~~all~~ should do the testing branch indicators,

look into → what about H, L, E on floating compares ? also need HE, LE?

Intermediate data results
— would have to be part of look ahead.

→ [we should try case where intermediate storage is in main mem.
indexes in core.

1. Purpose

2. Rolling Frame

3. Sigma AU — separate prob. / circuit speeds — will take best can get; memory cycle time — " " true both for indexing cores & data & instr. mem.

4. Blosk: Basic-Sigma interactions are fairly well laid out except for some timing

instr. prep rate 0.9 μs — any improvement appears to be [illegible] type.

0.4 0.8 ... transfer 0.3 add 0.6

uses core — several ops in parallel, clocked to core cycles.

going faster than 0.3 not worth while as long as there is 0.3 [illegible]

0.3 is max prep rate with present circuit speeds. regen cycle is covered by add.

Stores from LA distort indexing rate seriously.

5. Kolsky: Two Types of I Box instrs.

(1) steady indexing rate: read out time is important no.

(2) Increment or C + Br Type — total cycle is imp.

Store from LA still done here.

Suggestions

1. to avoid mem. refs to indices — multiple accumulators — advantages / disadvantages

2. another core array — lose memory logicity — look into

sim shows 5% effect

3. Instr buffer reg, not [illegible] decide as yet.

4. IBox should test branch ind.

5. Possibility of Branching off & on — need more indicators — look into.

must not make any part intentionally slow — may have branch thru [illegible]

6. read-out time of instr & data mem.