LASL – IBM Meeting, June 11, 1957

- components
- machine organization
- look ahead calc.
- exchange

H. Musalle, S. Dunwell, W. Wolensky, Brooks, Buchholtz, Sweeny, Cocke, Griffith, Blaauw, Codd, Carlson, Lazarus, Wood, E. Voorhees, Kolsky

(Dunwell)

Circuits group has been moved –
Tape program " ".

Components:
- production questions. NPN, PNP, to drivers.
- fabrication techniques.
- mem. program – lacks drivers –

Safety margins in circuits are great – can use present slow transistors – packaging.

Schedule

1. broad organization of computer
2. internal organization – machine performance for specific problems – instr. set.
3. Instr. Set needs to be broadly used. (avoid 701-702 ~~type~~ split)
   - auto prog – common.

organization. "3 in one Study"

1. LASL
2. Extension to "Commercial" machines.
3. Harvest

Standardized: - transistors good for any machine,
- circuit designs
- memories - input to standardize
- internal equipment & communication -

question of ALU - Bus system.

~~Instr. Decoder~~ Instr. decoder - common
registers - common

Fred Brooks:

Registers - very expensive but not specialized.

Components: B R S H

LASL	B+S
BusShips	B+H
704 upl.	B+R

known R → 0 in decisions

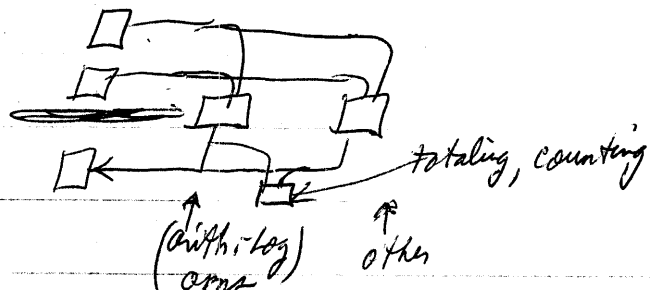
B versatility - flexibility

define B: 1. variable field length
2. variable byte size 1, 4, 6, 8 bits (but any binary combination is possible.)
(decimal - alphabetical)

all arith, & logical ops.

Floating Point System.

Harvest Streaming:

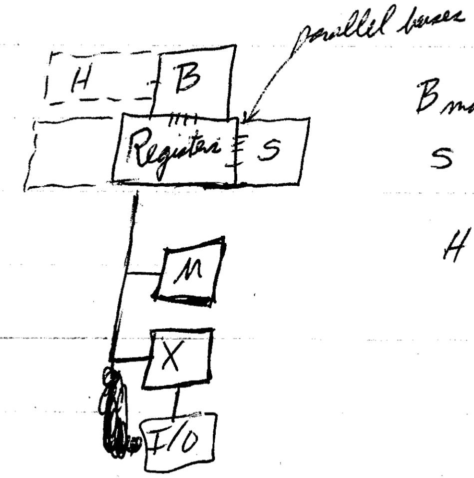


conclusions: 1. B+H is separable with only about 5% (2% for S)

2. variable length use combinations of 1, 4, 6, 8,

serial is faster for ~20 bits or less,

Parallel " " " greater than 48 bits, — parallel would take more hardware. Serial = $\frac{2}{5}$ parallel



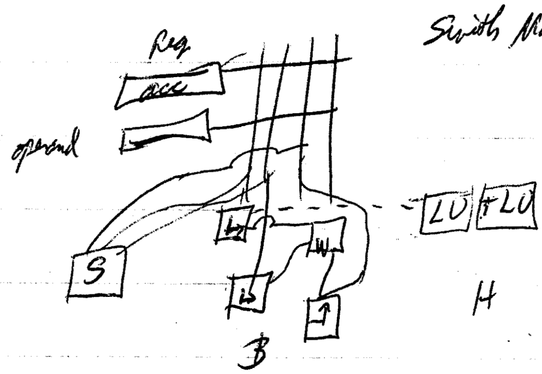
B machine uses these registers for arith opns,

S uses same registers.

H can be attached to B — serial busses or to Reg. (parallel reg.)

(~3.0 µs serial float
~0.6 µs parallel)

Switch Matrix



registers addressible

B has one inst. lookahead

S has more ---

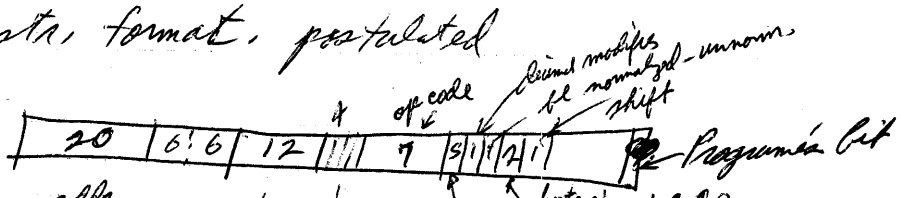
How does machine appear to programmer?

S will do fl. arith very fast parallel. B does all other in serial.

a prog. written for B would run on B+S except faster, serial fl. pt. hardware would be waste

Similarly B+S prog. can run on B with some interpretive subroutines.

Instr. format, postulated



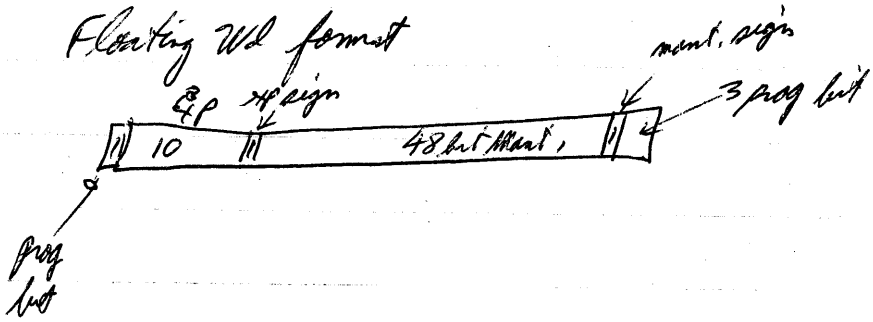
addr. can be 20, 26, or 27 long.

indx can be indirect by refering to another wd

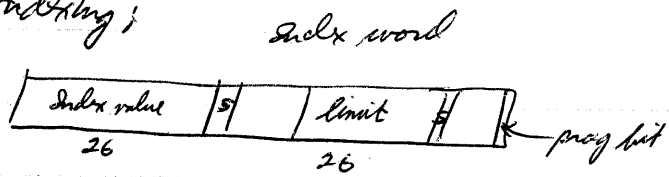
B has decimal adder.

B shift register - addressed operand can be shifted wrt. accumulator

Floating Wd format



Indexing:

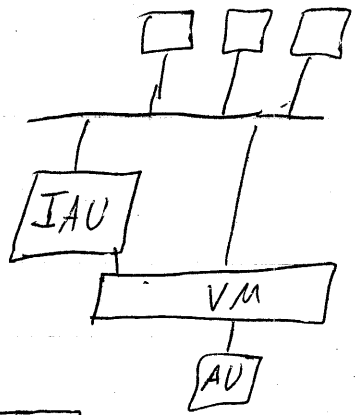


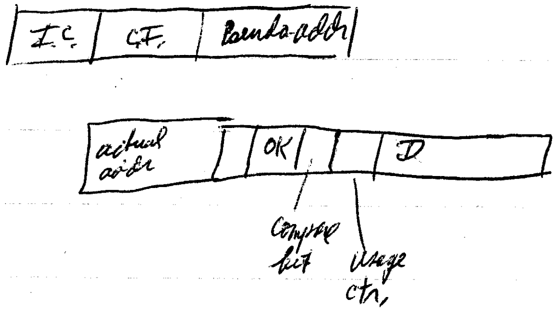
(increment in this position)

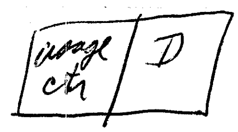
question: geometric indexing

pre-store, post-store

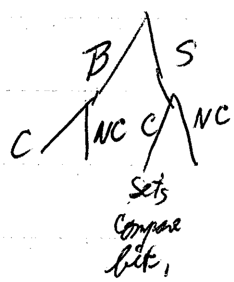
(cache)







for Bring-Store orders test compare or not vs. wds in pr. mem.



get effective prestore
post-store -

parameters

No. look ahead 1, 2, 4, 8, ... 32

Mem. times separately (table lookup)

arith. times

bus time 0.1 & up.

indexer time

index levels 1, 2, 3,

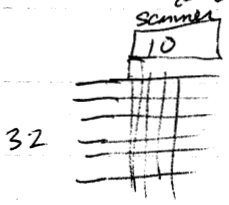
2nd day:

Tuesday June 11, 1957

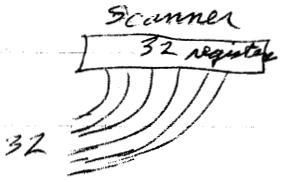
Changes in Exchange (Buckholtz)

Cross pt. switch

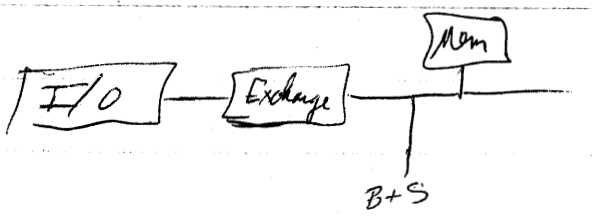
old system



new system



new system can handle 10 channels as well as all — more flexible,

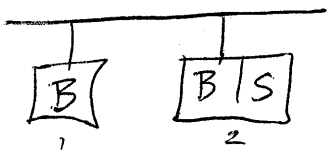


assumes only 1 instruction counter (not a dual computer)

(Carlson)

0. I/O Computer (Secondary) (Simultaneous Oper assumed)
1. radix conversion fl. bin. ⇄ fl. dec.
2. Interruption buffer to main computer
3. Decimal arith., probs, -- Data processing --
4. Search & Sort,
5. Code & Data transfer

another possibility



where 2 is masked on interrupt — 1 is not.

---

discussion: masking example when we are in an interrupt subroutine we don't want to be interrupted again,

---

Addressing: (Calcomp) floating point — more general than

A ≡ mathematical symbol

$a, b, c, \cdots$ parameters

$C(A)$ ≡ value of symbol

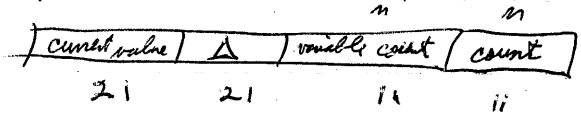
$(A, X) = f(x)$

X ≡ variable, A = base addr, ← A now needs range only thru few hundred.

$(A, X_1, X_2, \cdots) = f(x_1, x_2 \cdots)$ fn. of several variables, dimensionality,

$C\left[(A, X_2 \cdots)\right]$ = function table

Index reg equiv. to variable



limit = $n'\Delta$

21

or can set $n = 0$ & count n' down

problems,

1. Indexing one or two words? (index & increment)
2. count or limit in index?
3. base address (small no. or large no.?)

1. Is it acceptable to have instructions of the pre-load, post-store type apply to floating point type instructions only. , pre-store

2. Is it a sufficient advantage to have only multiple accumulator operation and not pre-store, post-store, pre-load operations.

3. How large should the second address be. (what size)

4. Is it acceptable to have geometric indexing for floating point operations only.

5. What is the smallest number of index registers acceptable. for geometric indexing What is an optimum.

6. If there is a choice between geometric indexing and multiple accumulators, which is preferred. (second addressing vs. geom. ind.)

7. Would it be desirable to address high-speed registers both as geometric indices and multiple accumulators. Desirable to share part each way

8. If the operand address would not span full memory, what size of address would be acceptable. (base address)

9. On the same assumptions, what maximum index address would be desired. (i.e. span full memory.) most extended version of STRETCH word address

10. Is it acceptable to concentrate the effort of obtaining very high-speeds upon floating point arithmetic rather than other data manipulations. ← (can go as high as 2 or 3 μs Variable field also)

11. Is the floating point format acceptable.
    a. Mantissa 48 + 1
    b. Exponent 10 + 1
    c. Tag bits 4

12. Is a zero tag bit acceptable if it would speed up operations.

13. If it were possible to use and specify variable field length floating point operations, would this be usefull. (in one order) for memory space reasons mainly — less speed

identical zero. (e.g. input data)

cancellation of mantissa

Discussion of question list:

→ input, code comp., input/output { examine these in detail coding of problem

a writeup on chapt 5 - 6 will come out next week