Clock design

1. <u>Objectives</u>

a. Ability to measure precisely time intervals for time accounting of (for example)

- i. core memory usage
- ii. channel usage (time delay from connect to interrupt.
- iii other devices.

Precision for this purpose means ability to define an interval within a 1-100 μsec range; preferably as small as possible. Range of intervals to be measured: 100 μsec – 10 min with this acc.
10 min – 10 yr with 1 sec acc.

Restriction: the beginning and end of any of these intervals may be noted by <u>different</u> processors.

b. Ability to determine calendar time to within 1 sec of local standard time.

c. Ability to obtain a processor interrupt after a program specified (~~program~~ time interval. Interrupt signal should occur within 0.5 μs of specified time, preferably more precisely. Time interval may vary from a few tens of μs. to many ~~minutes or~~ hours.

d. All clocks should use a common time unit; preferably μsec.

e. Use same clock for short (μs) and long (weeks) time intervals

Possible Techniques

1. Calendar clock, 32 bits, (52 bits for 100 years) pulse increments, I/P scan

Approach:

1. 32 (52) bit register counting 1/μsec will last 100 years.

2. 36 bit register counting down to zero can generate an interrupt in up to 9 hrs.

Where to put them

1. I/P [illegible]: problems of synchronization & space in 645.

2. Memory Controller: entail redesign of 645 memory; would require some new CPU instructions to access the register. (N.B.: cannot be a bit in memory cell, because of interlace. In a new system this would probably be best home for a calendar clock.

3 Drum: not acceptable from a product standpoint; also system should run without a drum.

4. Special adapter on board of GIOC; reads into memory on basis of some bit flipping, e.g., every 64 μsec.
   a. Would use 5% of a GIOC capacity
   b. Would use 2% of one memory controller capacity.

Calender Clock

1. Basic clock – 72 bit register of which leftmost 20 bits are wired zeros.

   Right most 52 bits contain ~~you~~ an integer = # of μsecs since Jan 1, 1900.

   This number will not overflow the 52 bit allotment until Sept in the year 2042.

2. The clock is incremented 1/μsec; as the right-most bit is a wired zero + the clock changes every 2 μsec.

3. Variable tap on some bit near right governs storing of the 72 bits as a double word in core memory.

4. Core memory location is set in ~~24~~ 23 toggles on side of clock; word is stored starting at an even location.

5. Clock adapter uses a direct channel; no interrupts are generated.

6. Setting: Operator can get to three dials:

1. year since 1900 (in octal)
2. days this year " "
3. Seconds today. " "

To establish 2 + 3, he has a small book of tables. Table I contains 366 + 365 entries; (for regular + leap years)

Table II ~~contains 24 pages: 1/hour.~~
~~Each page contains 3600 entries~~ 60 entries; this
is 24 X 60 = 1640 entries

hour \ minute	1	2	3	59	
0100	1	60	120		
0200	3601	3660	3720		etc.
0300					
2400				86,340	

He sets up year, days, seconds, and waits for next minute; then he presses the read-in button. This button

1. changes the Calendar clock to his value.
2. Interrupts via GIBC + Status word.

In general, it must be possible for the operator to, easily, set the calendar clock to within 1 sec of local standard time.

Internal wake-up clock

36-bit register containing an integer in μsec. This register is counted down 1/μsec. If it goes to zero (or negative) an interrupt is generated back thru GIOC.

Program may write into this register at any time via GIOC.

This clock is synchronized to the Calendar clock.

6. Setting clock: Operator can get to ~~52~~ (32) toggle switches, (or 11 octal switches) He has a small book of tables giving octal values ~~seconds since 1900~~ of the left most 32 of the 52 bit counter for each ~~second~~ minute since 1900.

Table has ~~100 pages (1/year.)~~

365 pages/year; each page has 24×60 entries of 11 digit octal numbers.

He may now ~~set~~ set the ~~time~~ calendar to within ±0.5 ~~sec~~ of local standard time. He presses a button & reads his toggles in & causes an interrupt to the operating system via the GIOC.

[Need reference to known documents; request quotation, delivery date + price or provide specs. This is a standard operation from R. Miller]

Need info on needed delivery.