True Random number Generation For Encryption

Details : New method developed for random number generation which utilizes the physical random elements like processor speed & system load to generate a set of random numbers and again manipulate on those to further increase the stochastic nature of the number generated.

Pseudocode

```
Import math , Import stdlib , Import time , Import stdio

Int a [10000]
Int b [10000]
Int selector,selector2

RAND MAX =size of random number

srand(time(milliseconds))
For i in range 10000
        a[i]=rand()

selector= rand()%10000

PRINT (1 pass random number is %d,a[selector])

For i in range 10000
        srand(a[i])
        b[i]=rand()   //we used a[i] to the seed of b[i]

 //now we use selector to select randomly again from random data

srand(time(milliseconds))

//we seed with time again to utilize the difference in execution time from previous loop

selector2= rand()%10000

PRINT (2 pass random number  is %d, b[selector2] )

#WE can repeat this process to increase randomness in multiple passes.
```

Key points and ideology

-a big array was taken ,and filled with random numbers but the seed was very dynamic [adds randomness] with millisecond resolution.

-now another array B is filled with random numbers but seeded with random numbers from the first array.

-selector element was created with random value which picked random element from the array.selector had a rand function used with different seed.

Hence 2 passes are made and and 2 different time stamps are used which will coincide only if the code was executed in null time,which is not possible. And no 2 processors have the same performance hence it also adds uncertainity.

All in all,the uncertainity in processors circuitry,processor load and speed have added to randomness.