Generación de números aleatorios

Los números aleatorios son muy útiles en muchas aplicaciones. Aunque existen varias maneras de generar números aleatorios, siempre es deseable construir una secuencia que sea de máxima longitud antes de que se repita (una secuencia pseudoaleatoria).

Para números de 16 bits es evidente que la longitud de dicha secuencia es 2 elevado a la 16, es decir 65536 números. McCraken describe un algoritmo y tiene la siguiente forma:

$X_{n+1} = (2053_d * X_n + 13849_d) \bmod 2_d^{16}-1$

Donde :

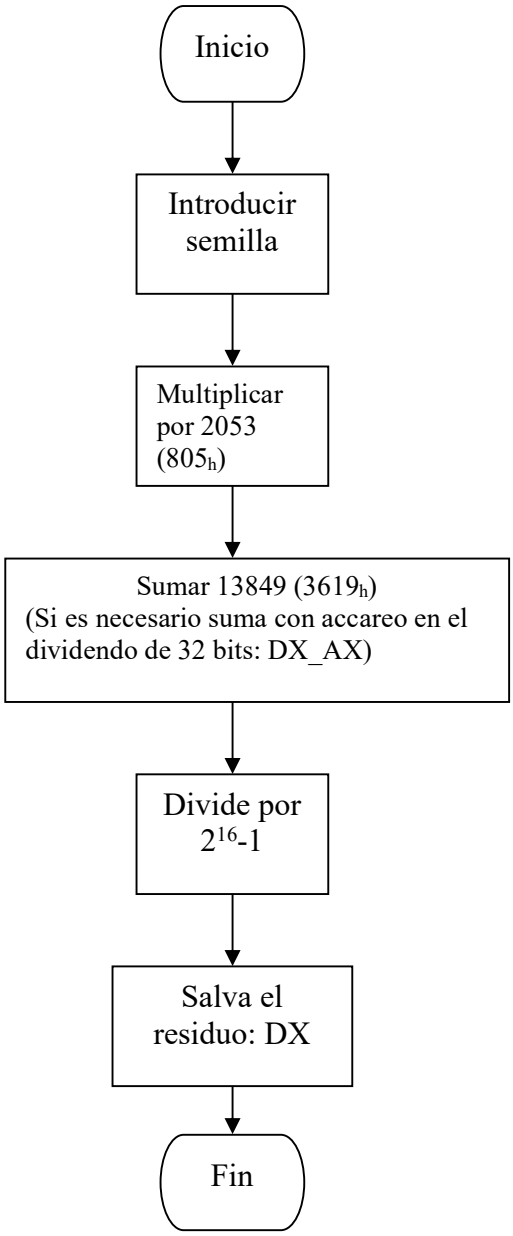
X_n es un número de entrada aleatoria

X_{n+1} es el resultado de la operación

$2053_d = 805_h$

$13849_{d=} 3619_h$

Diagrama de flujo del procedimiento ALAETORIO



Ejemplo para entender el uso del algoritmo:

Suponga la semilla FFFF , cargar en AX= FFFF , en DX=0000 y en BX= 0805. Esto implica el uso de una multiplicación palabra por palabra.

FFFF * 0805 = 0804 F7FB

El resultado quedará

DX=0804 AX=F7FB CF=0

Cargar la constante aditiva en BX=3619
Sumar de la siguiente manera : ADD AX, BX

F7FB + 3619= 2E14

Esta suma genera un acarreo enciende la CF=1

Sumar de la siguiente manera: ADC DX, 0
Esta suma se realiza de la siguiente forma: 0804 + 0 +1= 0805

Entonces DX=805 AX=2E14

Ahora cargar en BX= FFFF
Realizar la división a 16 bits, el residuo quedará en DX listo para ser utilizado.

Ejemplo: Desplegar 10 números aleatorios entre 0 y 9

```
;programa para tasm QUE OBTIENE UN NUMERO ALEATORIO
   ; ESTE PROGRAMA CALCULA UN NUMERO ALEATORIO BASADO
   ; EN OTRO NUMERO ALEATORIO PREVIO O SEMILLA, COLOCADO EN DX
   ;Y LA SALIDA SE OBTIENE EN AX, EL NUMERO ALEATORIO ES DE 16 BITS

; Definicion de stack
STACKSG segment para stack 'stack'
        DB 20 DUP (0)
STACKSG ENDS

;DEFINICION DE AREAS DE TRABAJO
DATASG SEGMENT PARA 'DATA'
MEN1 DB 'SEMILLA PARA EL NUMERO$'
MEN2 DB 'ADIOS$'
DATASG ENDS

CODESG SEGMENT PARA 'CODE'
PRINCI PROC FAR
   ASSUME SS:STACKSG, DS:DATASG,CS:CODESG
   ;PROTOCOLO
   PUSH DS
   SUB AX,AX
   PUSH AX
   MOV AX,SEG DATASG
   MOV DS,AX

   ;INICIA PROGRAMA
   MOV CX,10
OTRO:   PUSH CX
   CALL SEMILLA
   CALL ALEATORIO
   CALL ESCALANDO
   CALL LEE
   POP CX
   LOOP OTRO
   CALL SALIR_DOS
PRINCI ENDP


SEMILLA PROC
PUSH AX
MOV AH,2CH ; SERVICIO 2CH OBTIENE LA HORA ACTUAL EN EL SISTEMA
INT 21H  ; RETORNA CH=HORAS, EN FORMATO 00-23, MEDIANOCHE=0
         ; CL MINUTOS 00-59
         ; DH SEGUNDOS 00-59
         ; DL CENTESIMAS DE SEGUNDO 00-99
POP AX
RET
SEMILLA ENDP
```

```
ALEATORIO PROC
; XN+1=(2053*XN + 13849)MOD (2**16-1)
; RETORNA EL NUMERO PSEUDOALEATORIO EN AX

MOV AX,DX   ;CARGANDO A AX EL NUMERO SEMILLA tomado de la int 21H serv
            ;     2CH
MOV DX,0    ;CARGANDO CERO EN LA POSICION MAS SIGNIFICATIVA DEL
            MULTIPLICANDO
MOV BX,2053 ; MULTIPLICADOR
MUL BX
MOV BX,13849 ;CARGA EN BX LA CONSTANTE ADITIVA
CLC
ADD AX,BX   ; SUMA PARTES MENOS SIGNIFICATIVAS DEL RESULTADO
ADC DX,0    ; SUMA EL ACARREO SI ES NECESARIO
MOV BX,0FFFFH     ; CARGAR LA CONSTANTE 2**16-1
DIV BX
MOV AX,DX    MUEVE EL RESIDUO  AX
RET
ALEATORIO ENDP


ESCALANDO PROC
   ; ESCALANDO EL NUMERO PSEUDOALEATORIO OBTENIDO
   MOV DX,0
   MOV BX,0AH ;NUMEROS ALEATORIOS ENTRE 0 Y 9
   DIV BX
   MOV AX,DX
   ADD AX,3030H ; CONVIRTIENDO EL DATO BINARIO A ASCII
   MOV DL,AH
   MOV DH,AL
   CALL ESCRIBE
   MOV DL,DH
   CALL ESCRIBE
   RET
ESCALANDO ENDP


LEE PROC
   MOV AH,01
   INT 21H
RET
LEE ENDP


ESCRIBE PROC
   MOV AH,02
   INT 21H
 RET
ESCRIBE ENDP

SALIR_DOS PROC
   MOV AH,4CH
   INT 21H
RET
SALIR_DOS ENDP


CODESG ENDS
```

```
END PRINCI
```

INSTRUCCIÓN CLC: Limpia la bandera de acarreo

INSTRUCCION ADC

Propósito: Adición con acarreo.

Sintaxis:

ADC destino, fuente

Lleva a cabo la suma de dos operandos y suma uno al resultado en caso de que la bandera CF esté activada, esto es, en caso de que exista acarreo.

El resultado se guarda en el operando destino.

Ejemplo:

ADC ah,bl ; la operación que realiza es: AH = (AH + BL) + CF

ADC ax,bx ; la operación que realiza es: AX = (AX + BX) + CF

ADC eax,ebx ; la operación que realiza es: EAX = (EAX + EBX) + CF