COMENTARIO TÉCNICO

Buceando en los MCUs Freescale.....

Por Ing. Daniel Di Lella
Dedicated Field Application Engineer
EDUDEVICES
www.edudevices.com.ar
dilella@arnet.com.ar

www.edudevices.com.ar

"Matemática de Punto Flotante"

Por el CETAD – UNLP
Coordinador Ing. Jose Rapallini
Participantes:
Sebastián Ledesma
e-mail: sledesma@barcala.ing.unlp.edu.ar
Federico Costantino,
e-mail: fcostant@barcala.ing.unlp.edu.ar
Jorge R. Osio,
e-mail: josio@gioia.ing.unlp.edu.ar

4ta. y última Parte.

Nota: La presente serie de artículos está basada en el trabajo realizado por el Centro de Técnicas Analógicas – Digitales (CETAD) de la universidad Nacional de La Plata, y constituye una Nota de Aplicación para los MCU´s de 8 Bits de las familias HC908 y HC9S08 de Freescale Semiconductor.

Concluyendo con esta serie de artículos sobre matemática de punto flotante en MCUs de 8 Bits, se detallarán otras sub rutinas que componen la aplicación de ejemplo y las conclusiones finales del trabajo aquí presentado.

Subrutina de división:

```
$BASE    10T

          org      $80

exp1      ds       1
exp2      ds       1
numA      db       $00,$7F,$FF,$FF
denB      db       $00,$01,$16,$C2
acum      ds       3
resp      ds       3
var       ds       1
tempsig   ds       1
expo      ds       1


          org      $EE00

          mov      #24,var
          clr      tempsig
clrresp   ldx      #3           ;Pone a cero todo el espacio para la mantisa
otra      clr      resp-1,x
          dbnzx    otra

*****************************************************************************
;Primera parte: Chequeo de condiciones particulares
*****************************************************************************

           lda         numA                ;recupero y salvo el exponente de numA
           ldx         numA+1
           aslx
           rola
           sta         exp1
           lda         denB                ;recupero y salvo el exponente de denB
           ldx         denB+1
           aslx
           rola
           sta         exp2
           lda         numA+1        ;Me fijo si la mantisa de numA es cero y la guardo
           and         #$7F          ;en una posición de memoria temporal
           add         numA+2
           adc         numA+3
           adc         #0
           sta         acum
           lda         denB+1        ;Me fijo si la mantisa de denB es cero y la guardo
           and         #$7F          ;en una posición de memoria temporal
           add         denB+2
           adc         denB+3
           adc         #0
           sta         acum+1
           add         acum  ;Si las dos mantisas son cero puede ser inf/inf, 0/0
           adc         #0    ;que como resultado es QNaN,pero si es 0/inf o inf/0
           bne         cheqnum      ;el resultado no es igual
           lda         exp1
           cmp         exp2
           bne         cheqnum
           lda         exp1
           beq         salto
           inca
           bne         cheqnum
salto      jmp         QNaN
cheqnum    lda         exp1                ;Chequeo si el numerador es NaN
           add         #$1
           bcc         cheqden
           lda         acum
           beq         cheqden
           jmp         SNaN
cheqden    lda         exp2                ;Chequeo si el denominador es NaN
           add         #$1
           bcc         uncheq
           lda         acum+1
           beq         uncheq
           jmp         SNaN
uncheq     lda         acum.     ;Si numA es cero el resultado es cero y si es infinito
           bne         mascheq   ;el resultado es infinito
           lda         exp1
```

```
            beq         cero
            add         #$1
            bcs         infin
mascheq     lda         acum+1
            bne         INI          ;Si denB es cero el resultado es infinito y si es
            lda         exp2         ;infinito el resultado es cero
            beq         infin
            inca
            bcs         cero
             bra        INI

*******************************************************************************

cero        mov      #0,resp-1           ;El resultado es cero
            jmp      fin

SNaN        mov      #$7F,resp-1         ;Uno o ambos numeros son no validos
            mov      #$81,resp
            jmp      fin

QNaN        mov      #$7F,resp-1         ; La operacion da valor no valido
            mov      #$C0,resp
            jmp      fin

infin       mov      #$7F,resp-1         ;El resultado es +/- infinito
            mov      #$80,resp
            bra      signo

*******************************************************************************

signo     bclr     7,resp-1            ;Rutina que determina el signo
          lda      numA
          eor      denB
          and      #$80
          beq      aca
          bset     7,resp-1
aca       jmp      fin

desnA     clc
          ldx      #$3
cachi     rol      (acum-1),x
          dbnzx    cachi
          brset    7,acum,nova
          lda      expo
          sub      #$1
          sta      expo
          bcc      desnA
          dec      expo-1
          bra      desnA
nova      rts

desnB     clc
          ldx      #$3
daba      rol      denB,x
          dbnzx    daba
          brset    7,denB+1,noviene
          lda      expo
          add      #$1
          sta      expo
          bcc      desnB
          inc      expo-1
          bra      desnB
noviene   rts

*******************************************************************************
;Segunda parte: Análisis y calculo del exponente del resultado
*******************************************************************************

INI       lda     exp1
          bmi     EXPOS
          sub     exp2
          bcc     todobien

          cmp     #$68
          bls     cero
          cmp     #$80
          bhi     todobien
```

```
          mov     #$FF,expo-1
          bra     todobien

expos     sub     exp2
          bcs     todobien
          cmp     #$80
          bhi     infin

todobien  add     #127
          sta     expo

*******************************************************************************
;Tercer Parte: División
*******************************************************************************

divi      bset    7,(denB+1)
          lda     exp2
          bne     bnorm
          bsr     desnB
bnorm     mov     numA+1,acum
          mov     numA+2,acum+1
          mov     numA+3,acum+2
          bset    7,acum
          lda     exp1
          bne     comp
          bsr     desnA
          lda     expo-1
          deca
          bne     nose
          jmp     infin
nose      clc
comp      bcs     restar
          clrx
sigo      lda     acum,x
          cmp     denB+1,x
          bhi     restar              ; es mayor (c)or(z)=0 =>salta
          blo     despla              ; es menor (c)=1
          incx
          cpx     #3
          bne     sigo

restar    ldx     #3
          clc
aca3      lda     acum-1,x
          sbc     denB,x
          sta     acum-1,x
          decx
          bne     aca3
          sec
          bra     restó


despla    clc
restó     ldx     #6
mas       rol     acum-1,x
          decx
          bne     mas
          dec     var
          beq     movex
          bra     comp

movex     lda     expo-1
          beq     etiq
limpiar   clrx
          clc
          lda     #3
pako      ror     resp,x
          incx
          deca
          bne     pako
          inc     expo
          bne     limpiar
          bra     desnorm
etiq      lda     expo
          beq     desnorm
detect    brset   7,resp,normal
          ldx     #3
```

```
mus       rol     resp-1,x   ;Si supongo que el resultado es 1,xxxx o en todo caso
          decx               ;x,1xxx no debería tener que hacerlo mas de una vez
          bne     mus
          dec     expo
          bne     normal
desnorm   lda     #3
          clrx                ;por la ubicación de la coma
aca5      ror     resp,x
          incx
          deca
          bne     aca5
normal    rol     resp
aca6      mov     expo,resp-1
          lda     #2
          clrx
aca4      ror     resp-1,x
          incx
          deca
          bne     aca4
          jmp     signo

fin       swi


          org     $FFFA

          dw      $EE00
          dw      $EE00
          dw      $EE00
```

Subrutina de Comunicación serie "SCI_TXTR.INC":

```
*******************************************************************************
* ;SACA_CARACTER - transmite la variable caracter pin TXD *;(Definido como salida) *
*******************************************************************************

saca_caracter:
                stx     aux  ;[3] guardo valor de X
                ldx     #9   ;[2] bits por caracter (incluido bit start)
                clc          ;[1] borra carry para enviar bit de start
saca_bit_dato:
                bcc     saca_0    ;[3] si cy=0, salta para sacar 0
                bset    1,ptb     ;[4] si no saca 1
                bra     otro_bit  ;[3]
saca_0:
                bclr    1,ptb     ;[4]
                bra     otro_bit  ;[3]
otro_bit:
                lda     #baud_tx  ;[2]
                jsr     delay_7a  ;[7a+9]demora=7*31+9=226T
                ror     caracter  ;[4] busca siguiente bit (va al
                                  ;carry)
                decx                    ;[1]
                bne     saca_bit_dato   ;[3] 8T vuelvo a sacar el sig
                                        ;bit
saca_bit_stop:
                lda     #baud_tx   ;[2] Valor para TX a 19200 baudios
                lsra               ;[1] A=baud_tx/2 divido en 2 a baud_tx
                bset    1,ptb      ;[4]  __| sube línea para bit de STOP
                jsr     delay_7a   ;[7a+9] demora=16T=6,51useg
                ldx     aux        ;[3]recupero valor de X
                rts                ;[4]
```

```
;-------------------------------------------------------------------------;-----------
; SELECCION DE VELOCIDAD DE TRANSMISION
;-------------------------------------------------------------------------;-----------
baud_tx         equ     $20     ;Tabla de Selección de Baud Rate:
*******************************************************************************
* LEE_CARACTER - recive un dato por el pin RXD (siempre b0 de *cualquier puerto *RXD=0)
*
*******************************************************************************
lee_caracter:
                stx     aux                 ;[3]guardo el valor de X
            lda     #$8                 ;[2]cant de bits del caracter
            sta     cuenta_bit          ;[3]
            clrx                        ;[1]
lee_bit_start:
                brset   0,ptb,*             ;[5]espera el bit de start
                lda     #baud_rx            ;[2] ($08)
                lsra                        ;[1] divido/2 el baud rate
                jsr     delay_7a            ;[7A+9]
                brclr   0,ptb,lee_b_dato    ;[5]=24T->9,76useg (mitad del pulso)
                bra     lee_bit_start       ;[3]
lee_b_dato:
                lda     #baud_rx      ;[2]\
                nsa                   ;[3] \
                nsa                   ;[3]  > Demora p/complementar 12 cicl.T
                nsa                   ;[3] /  q/faltan a la 1ra lectura (bit 0)
                nop                   ;[1]/
lee_bit_dato:
                lda     #baud_rx      ;[2]
                nop                   ;[1]
                nop                   ;[1]
                nop                   ;[1]
                jsr     delay_7a      ;[7a+9]
                lda     ptb             ;[3] ->> Lee dato del b0,ptb (RXD)
                rora                    ;[1]
                ror     caracter        ;[4]
                dec     cuenta_bit      ;[4]
                bne     lee_bit_dato    ;[3]
lee_bit_stop:
                lda     #baud_rx        ;[3]
                sub     #$02            ;[2] NOTA: p/57600 es (sub #$01)
                jsr     delay_7a        ;[7a+9]
                brset   0,ptb,rx_ok     ;[5] hasta aquí (7a+31)T A=1->38T=15,46us
                lda     #$24            ; '$' -> caracter de error
                sta     caracter
                bset    0,pta           ;si hay error enciende el led
rx_ok:
                ldx     aux             ;[3] recupero el valor de X
                rts                     ;[4]


;-------------------------------------------------------------------------------
; SELECCION DE VELOCIDAD DE RECEPCION
;-------------------------------------------------------------------------------
baud_rx         equ     $20     ;Tabla de Selección de Baud Rate:
*******************************************************************************
* delay_7a - modulo de demora, sirve para calcular el tiempo en la *velocidad de *
*******************************************************************************
delay_7a:
                nop                     ;[1]
                nop                     ;[1]
                tsta                    ;[1]
                deca                    ;[1]
                bne     delay_7a        ;[3]= 7 cycles
                rts                     ;[4]=7a+4 cycles
*******************************************************************************
```

Notas aclaratorias.

La Subrutinas de suma o resta utiliza un pin adicional del puerto A (A0) para seleccionar si la operación a realizar será una suma o una resta (si es una resta el pin recibirá un uno, si la operación es una suma el pin recibirá un cero. Las subrutinas de multiplicación y división no hacen uso de este pin, ya que están hechas por separado. El pin 0 y 1 del puerto B se usa para transmisión y recepción de datos. Los pines 4 y 5 del puerto A no están configurados en las subrutinas, pero están reservados para el uso del cristal.

Fe de erratas:

- La figura que muestra el circuito eléctrico tiene un error, pues el capacitor de 0,1uF que se encuentra entre la alimentación y tierra del microcontrolador, fue omitida en la placa física.

Limitaciones y posibles adaptaciones para otros MCUs de la familia.

Las operaciones matemáticas de punto flotante están hechas para números en punto flotante de simple precisión, por lo tanto cada número puede estar representado por 32 bits.

Otra de las limitaciones que tiene es que si ocurre un error en uno de los Byte's se debe esperar a que se envíen todos los datos para poder volver a enviar los valores correctos.

Una solución a esto sería poner un reset externo, de manera que si ocurre un error, se presiona el reset y se vuelve a realizar el envío de datos.

En cuanto a la detección de errores, la subrutina SCI solo tiene detección de errores de Encabezado (Frame), también se le debería poner detección de error por ruido.

Estas subrutinas están orientadas a los microcontroladores **HC908QY4A.**
Los HC908QT-QY son los mas pequeños de la Familia y son los únicos que no poseen modulo de transmisión SCI, por ello a las operaciones se les agregó una subrutina para transmisión serie. Por esto podemos decir que estas rutinas son fácilmente adaptables a toda la Familia **HC908** y al Familia **HC9S08.** Para hacer la adaptación se deben modificar los valores de los registros y las direcciones de inicio de RAM y Flash de acuerdo al microcontrolador utilizado. Además si se usa un microcontrolador con modulo SCI, directamente se saca la subrutina de comunicación y se configura el modulo SCI.

Herramientas de Hardware y Software utilizadas.

Herramientas Hardware.

1. **Placa de Desarrollo EVAL08QTY desarrollada por EDUDEVICES.**

2. **Computadora Personal.**

Se utiliza una PC, para el envío y recepción de números en punto flotante.
Los requerimientos mínimos de la PC son un puerto serie disponible, y poseer un software para comunicación serie.

Herramientas Software.

1. **Ambiente de Desarrollo Integrado Win IDE.**

WinIDE es un editor de aplicaciones que permite integración de varios programas diferentes en un mismo ambiente de desarrollo. En este ambiente se posee, un compilador, un simulador en circuito, un depurador en circuito y un programador.
El lector puede utilizar con mínimas modificaciones un entorno integrado más poderoso como el ***CodeWarrior 6.x*** con las mismas herramientas de hardware citadas en el proyecto.

2. **Software para comunicación por puerto serie "Hercules [9]".**

   El modo serial es una interfaz terminal sencilla para el puerto serial. Es útil para configurar dispositivos sobre RS-232. El modo predeterminado de configuración es **9600 8N1** para todos los dispositivos. El **SETUP mode / DATA mode** cambia los controles del pin de salida DRT (SETUP = log. 0). El Modo Serial es una interfaz terminal sencilla para el puerto serie.

3. **Software para comunicación por puerto serie RS232 Hex Com Tool v4.0.**

Este es el software mas apropiado para los propósitos del proyecto, ya que tiene una ventana de envío de datos y otra de recepción. Además permite el envío y recepción de datos en formato Hexadecimal, cosa que el software Hercules n permite. En cuanto a la configuración del software es la misma que la del Hercules.

Testeos realizados.

Sobre las subrutinas se realizaron todo tipo de testeos y simulaciones, abarcando todos los casos posibles en cuanto a al rango de valores en punto flotante que se pueden representas con 32 bits.

Las operaciones en punto flotante se pueden realizar con números mayores a 1, menores a 1, valores muy altos (+infinito) y valores muy bajos (-infinito) si uno de los valores de la operación es NAN, el resultado será NAN.

La comunicación serie fue testeada correctamente, y se detectaron menor cantidad de errores en la transmisión a 9600 y 19200 baudios. De cualquier manera la probabilidad de error es mayor al enviar dos números de 4 bytes cada uno, que si se enviara solo un byte.

Bibliografía.

[1] Norma IEEE P754.
[2] An American National Standard, "IEEE Standard for Binary Floating-Point Arithmetic", Approved July 26, 1985 Reaffirmed May 21, 1991. American National Standards Institute.
[3] Sergio Noriega, "Operaciones Matemáticas Con Números Binarios", Apuntes de Clases de Introducción a los sistemas Lógicos y Digitales, Departamento de electrotecnia, Facultad de ingeniería, UNLP, 2003.
[4] DAVID GOLDBERG ,"What Every Computer Scientist Should Know About Floating-Point Arithmetic ".
[5] Ing. Gabriel Dubatti, http://www.ingdubatti.com.ar/artmath.htm
[6] Nota de Aplicación AN1240 – Motorola Semiconductor by Scott George, CSIC MCU Product Engineering.
[7] CPU08 Central Processor Unit Reference Manual – Freescale Semiconductor. Order Nro.CPU08RM/AD.
[8] Nota de Aplicación NA_SyHDe - SyHDe, Soft y Hard Desarrollos.
[9]www.HW-group.com.

Nota de Radacción: El lector puede descargar este artículo y artículos anteriores de ***"Buceando..."*** desde la sección ***"Artículos Técnicos"*** en el sitio web de **EduDevices (www.edudevices.com.ar)**