Name: Torkom Pailevanian

Function: Converters

CGROUP GROUP CODE

CODE SEGMENT PUBLIC 'CODE'

ASSUME CS:CGROUP

; Dec2String

;

; Description:

; The function is passed a 16-bit signed value to convert to

; decimal (at most 5 digits plus sign) and store as a string. The ; string contains the <null> terminated decimal representation of ; the value in ASCII. The resulting string is stored starting at

; the memory location indicated by the passed address

;

; Operation:

; The function first checks to see if the digit is positive or

; negative. The function starts dividing with the largest power

; of 10 possible (10000) and loops dividing the number by the

; power of 10, the quotient is a digit and the remainder is used

; in the next iteration of the loop. Each loop iteration divides ; the power of 10 by 10 until it is 0. At that point the number

; has been converted to decimal.

;

; Arguments:

; AX - binary value to convert to decimal

; SI – address to start storing string

;

; Return Value:

; None: String Array written to memory

;

; Local Variables:

; Remainder – remainder from division operation

; pwr10 – power of 10 being divided by

; arg – copy of number to convert

;

; Shared Variables: None

; Global Variables: None

;

; Input: None

; Output: None

;

; Error Handling: None

;

; Algorithms:

; Repeatedly divide by powers of 10 and get the remainders

; Data Structures:

; Output String

;

; Registers Changed: AX, SI, DX, CX

; Stack Depth:

;

; Author: Torkom Pailevanian

; Last Modified: 10/19/2014

;Pseudo Code

; pwr10 = 10000

; IF(negative)

; add “–“ to string

; REPEAT

; remainder = arg modulo pwr10

; arg = arg/pwr10

; add to string remainder + 0x30 for ascii value

; pwr10 = pwr10/10

; UNTIL (quotient = 0 OR pwr10 = 0)

; add <null> to string

Dec2String PROC NEAR

PUBLIC Dec2String

RET

Dec2String ENDP

; Hex2String

;

; Description:

; The function is passed a 16-bit unsigned value (n) to convert to ; hexadecimal (at most 4 digits) and store as a string. The string ; should contain the <null> terminated hexadecimal representation ; of the value in ASCII. The resulting string is stored starting

; at the memory location indicated by the passed address

;

; Operation:

; This function rotates the argument by 4 bits and does a bitwise ; and with 0x000F to get the value of last 4 digits. It then

; takes the equivalent ASCII hex value and adds to the string

;

; Arguments:

; AX - binary value to convert to hex

; SI – address to start storing string

;

; Return Value:

; None: String Array written to memory

;

; Local Variables:

; arg – copy of number to convert to hex

; result – value of AND operation

;

; Shared Variables: None

; Global Variables: None

;

; Input: None

; Output: None

;

; Error Handling: None

;

; Algorithms:

; Rotate number and do bitwise AND with 0x000F and store the ASCII ; equivalent value

;

; Data Structures:

; Output String

;

; Registers Changed: AX, SI, BX

; Stack Depth:

;

; Author: Torkom Pailevanian

; Last Modified: 10/19/2014

;

; Pseudo Code

; FOR (i = 0 to 3, i++)

; result = AND arg with 0x000F

; IF(result between 0 and 9)

; add result + 0x30 to string

; ELSE

; add result – 10 + 0x41 to string

; Rotate arg 4 bits right

; add <null> to string

Hex2String PROC NEAR

PUBLIC Hex2String

RET

Hex2String ENDP

CODE ENDS

END