CS 219: Homework #8

Due on November 2nd, 2016 at 4:00pm

Dr. Egbert

Matthew J. Berger

Problem 1

1.) Write a program called SUB64 to subtract the 64-bit integer in memory locations 0x40001000 and 0x40001004 from the 64-bit integer in 0x40001010 and 0x40001014. Store the result in memory location 0x40001020 and 0x40001024.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.

SUB64 ldr   r0, =0x40001000  ; upper 32 bits
      ldr   r1, [r0]
      ldr   r0, =0x40001004  ; lower 32 bits
      ldr   r2, [r0]
      ldr   r3, =0x40001010  ; upper 32 bits
      ldr   r3, [r3]
      ldr   r4, =0x40001014  ; lower 32 bits
      ldr   r4, [r4]
      subs  r5, r3, r0       ; subtract lower bits, could result in carry
      subc  r6, r2, r1       ; subtract upper bits with respect for carry
      str   r6, =0x40001010  ; store upper 32 bits of result
      str   r5, =0x40001014  ; store lower 32 bits of result

;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 2

2.) Write a program called COMBINE that combines the low-order nibbles of the four bytes in memory locations 0x40001000 to 0x40001003 into a single 16-bit word. The nibbles should be ordered low-to-high in the result beginning with the data from location 0x40001000. Store the result as 16-bits in memory location 0x40001004.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart ldr r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
COMBINE ldr r0, =0x40001000
        ldr r0,[r0]          ; load the value in r0 into r0
        mov r0,r0,lsl#28     ; left shift until only the nibble is left
        mov r0,r0,lsr#16     ; right shift into a new set of 4 bits
        ldr r1, =0x40001001 ; load the memory location into r1
        ldr r1,[r1]          ; load the value in r1 into r1
        mov r1,r1,lsl#28     ; left shift unitl only the nibble is left
        mov r1,r1,lsr#20     ; right shift into a new set of 4 bits
        ldr r2, =0x40001002 ; load the memory location into r2
        ldr r2,[r2]          ; load the value in r2 into r2
        mov r2,r2,lsl#28     ; left shift until only the nibble is left
        mov r2,r2,lsr#24     ; right shift into a new set of 4 bits
        ldr r3, =0x40001003 ; load the memory location into r3
        ldr r3,[r3]          ; load the value in r3 into r3
        mov r3,r3,lsl#28     ; left shift until only the nibble is left
        mov r3,r3,lsr#28     ; right shift into a new set of 4 bits
        add r4,r3,r2         ; add two groups of nibbles together
        add r5,r1,r0         ; add two groups of nibbles together
        add r6,r5,r4         ; add the final two groups of nibbles together
        ldr r0, =0x40001004
        str r6, [r0]
;*******************************************************************
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                  ;END directive to show nothing more in file
```

Problem 3

3.) Write a program called FIND to find the larger of two signed bytes. Assume the two bytes are in memory locations 0x40001000 and 0x40001001. Store the larger of the two in memory location 0x40001002.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
FIND ldr r0, =0x40001000  ; load a memory location into register 0
     ldr r0, [r0]         ; load the value from memory into register 0
     ldr r1, =0x40001000  ; load a memory location into register 0
     ldr r1, [r1]         ; load the value from memory into register 0
     cmp r0, r1           ; compare the two registers
     bgt big              ; branch if greater than
     ldr r0, =0x40001002  ; load the memory to store
     str r1, [r0]         ; store r1
     bal stop             ; stop the program

big  ldr r1, =0x40001002  ; load the memory to store
     str r0, [r1]         ; store r0

;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                 ;END directive to show nothing more in file
```

Problem 4

4.) Write a program called LSHIFT to shift logically the 32-bit contents of memory location 0x40001000 left according to the 8-bit shift count stored in memory location 0x40001004 and store the results at memory address 0x40001008.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
LSHIFT ldr  r0, =0x40001000 ; load number to shift by
       ldr  r0, [r0]        ; load number from value
       ldr  r1, =0x40001004 ; load number to shift
       ldrh r1, [r1]        ; load number from value
       mov  r1,r1, lsl r0   ; start shifting
       ldr  r0, =0x40001008 ; load memory location
       strh r1, [r0]
;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                 ;END directive to show nothing more in file
```

Problem 5

5. Write a program called FIND8 to find the largest unsigned 8-bit word in a list. The list begins at address 0x40001004. The length of the list is stored in an 8-bit variable at address 0x40001000. Store the largest entry in memory location 0x40001002.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
FIND8   ldr  r0, =0x40001000 ; load memory location
        ldrb r1, [r0]        ; load byte into r1
        ldr  r0, =0x40001004 ; load memory l ocation
        ldr  r2, =0          ; initialize to 0
loopst  ldrb r3, [r0,+r1]    ; load byte into r3 using offset of r0 and r1
        cmp  r2,r3           ; compare values
        bhi  f8cpl           ; branch if high
        mov  r2,r3           ; move r3 into r2

f8cpl   adds r1,#-1          ; decrement by 1
        bpl  loopst          ; branch if positive or equal to 0
        ldr  r0, =0x40001002 ; load memory location
        strb r2, [r0]        ; store byte
;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                 ;END directive to show nothing more in file
```

Problem 6

6.) Write a program called FIND32 to find the largest unsigned 32-bit word in a list. The list begins at address 0x40001010. The length of the list is stored in an 8-bit variable at address 40001000H. Store the largest entry in memory location 0x40001004.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.

FIND32  ldr r0, =0x40001010 ; load memory location
        ldrh r1, [r0]       ; load byte into r1

        ldr r0, =0x4000100H ; load memory location
        ldr r2, =0          ; initialize to 0

loopst  ldrh r3, [r0,+r1]   ; load byte into r3 using offset of r0 and r1
        cmp r2,r3           ; compare values
        bhi f8cpl           ; branch if high
        mov r2,r3           ; move r3 into r2

f8cpl   adds r1,#-1         ; decrement by 1
        bpl loopst          ; branch if positive or equal to 0
        ldr r0, =0x40001004 ; load memory location
        strh r2, [r0]       ; store byte
;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 7

7.) Write a program called SCAN to scan a list of unsigned bytes and find the smallest and largest entries in the list. The length of the list is stored in a 16-bit variable at addresses 0x40001002. The list begins at address 0x40001010. Store the smallest byte at address 0x40001000 and the largest byte at address 0x40001001.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*******************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
; begin initialize data
                LDR r4,= 0x25           ;set first operand = 0x100
                LDR r5,= 0x50           ;set second operand to 0x50
                STRB r4,[r3,#0]         ;write first operand to memory
                STRB r5,[r3,#1]         ;write second operand to memory
; end initialize data
                LDRB r0,[r3,#0]         ;get first operand
                LDRB r1,[r3,#1]         ;get second operand
                ADD     r2,r0,r1                ;form the sum of the two values
                STRB r2,[r3,#2]         ;save sum
;               STRB r2,[r3,#3]         ;save sum
;*******************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 8

8.) Write a program called COUNT to count the number of characters in a null-terminated ASCII string that are equal to a KEY. The KEY is stored in memory location 0x40001000. The string is stored in memory beginning at address 0x40001010. Store the 8-bit count in memory location 0x40001004. (Assume the maximum count is 255.)

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*********************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
COUNT ldr r0,=0x40001000 ;set pointer to key
      ldr r1,[r0]        ;store value of r0 into r1
      ldr r0,=040001004  ;load memory location
      ldr r2,=0          ;initialize to 0
      ldr r3,=0          ;initialize to 0 again

again ldr r4,[r0,+r2]    ;load r4 with an offset
      cmp r4, r1         ;compare the numbers
      bne goup           ;branch if not equal
      add r3, r3, #1     ;increment and add

goup  cmp r4, #0         ;compare r4 with "null"
      bpl again          ;if greater than 0, go again

      ldr r0,=0x40001002 ;load memory location
      str r3,[r0]        ;store in r3
;*********************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 9

9.) Write a program called ONES to determine the number of bits equal to one in a 32-bit variable. The 32-bit variable is in memory location 0x40001004. Store the 8-bit counter in memory location 0x40001000

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*********************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
; begin initialize data
                LDR r4,= 0x25           ;set first operand = 0x100
                LDR r5,= 0x50           ;set second operand to 0x50
                STRB r4,[r3,#0]         ;write first operand to memory
                STRB r5,[r3,#1]         ;write second operand to memory
; end initialize data
                LDRB r0,[r3,#0]         ;get first operand
                LDRB r1,[r3,#1]         ;get second operand
                ADD     r2,r0,r1                ;form the sum of the two values
                STRB r2,[r3,#2]         ;save sum
;               STRB r2,[r3,#3]         ;save sum
;*********************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 10

10.) Write a subroutine called STRLEN that determines the length of a null-terminated ASCII string. Pass the 32-bit start address of the string to the subroutine in register R0. Return the length, excluding the null byte, in register R7. All registers (except R7) should return to the calling program unchanged.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*********************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
; begin initialize data
                LDR r4,= 0x25           ;set first operand = 0x100
                LDR r5,= 0x50           ;set second operand to 0x50
                STRB r4,[r3,#0]         ;write first operand to memory
                STRB r5,[r3,#1]         ;write second operand to memory
; end initialize data
                LDRB r0,[r3,#0]         ;get first operand
                LDRB r1,[r3,#1]         ;get second operand
                ADD     r2,r0,r1                ;form the sum of the two values
                STRB r2,[r3,#2]         ;save sum
;               STRB r2,[r3,#3]         ;save sum
;*********************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 11

11.) Write a subroutine called REPLACE that processes a null-terminated string of decimal characters and replaces leading zeros with spaces. Pass the 32-bit address of the string to the subroutine in register R0.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*********************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
; begin initialize data
                LDR r4,= 0x25           ;set first operand = 0x100
                LDR r5,= 0x50           ;set second operand to 0x50
                STRB r4,[r3,#0]         ;write first operand to memory
                STRB r5,[r3,#1]         ;write second operand to memory
; end initialize data
                LDRB r0,[r3,#0]         ;get first operand
                LDRB r1,[r3,#1]         ;get second operand
                ADD     r2,r0,r1                ;form the sum of the two values
                STRB r2,[r3,#2]         ;save sum
;               STRB r2,[r3,#3]         ;save sum
;*********************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```

Problem 12

12.) Write a program called UNPACK to convert the 16-bit BCD variable in memory locations 0x40001000 and 0x40001001 to four ASCII characters with the high-order digit first, beginning in memory location 40001004H.

Solution:

```
;Simple Keil tools demonstration program
;Modified from J.R. Gibson (7/14/07)
;Modified by D. Egbert ver. 2.0 1/28/10
;
        AREA    Assembly1, CODE, READONLY
        EXPORT  __main
__main
        ENTRY
mystart LDR r3,= 0x40001000     ;point to RAM location
;
;*********************************************************************
; For other programs (projects) replace the code below with new code.
; Leave the code outside the ******* lines as is for all programs.
; begin initialize data
                LDR r4,= 0x25           ;set first operand = 0x100
                LDR r5,= 0x50           ;set second operand to 0x50
                STRB r4,[r3,#0]         ;write first operand to memory
                STRB r5,[r3,#1]         ;write second operand to memory
; end initialize data
                LDRB r0,[r3,#0]         ;get first operand
                LDRB r1,[r3,#1]         ;get second operand
                ADD     r2,r0,r1                ;form the sum of the two values
                STRB r2,[r3,#2]         ;save sum
;               STRB r2,[r3,#3]         ;save sum
;*********************************************************************
;
; an infinite loop because the processor continues to fetch instructions
stop    BAL     stop
        END                     ;END directive to show nothing more in file
```