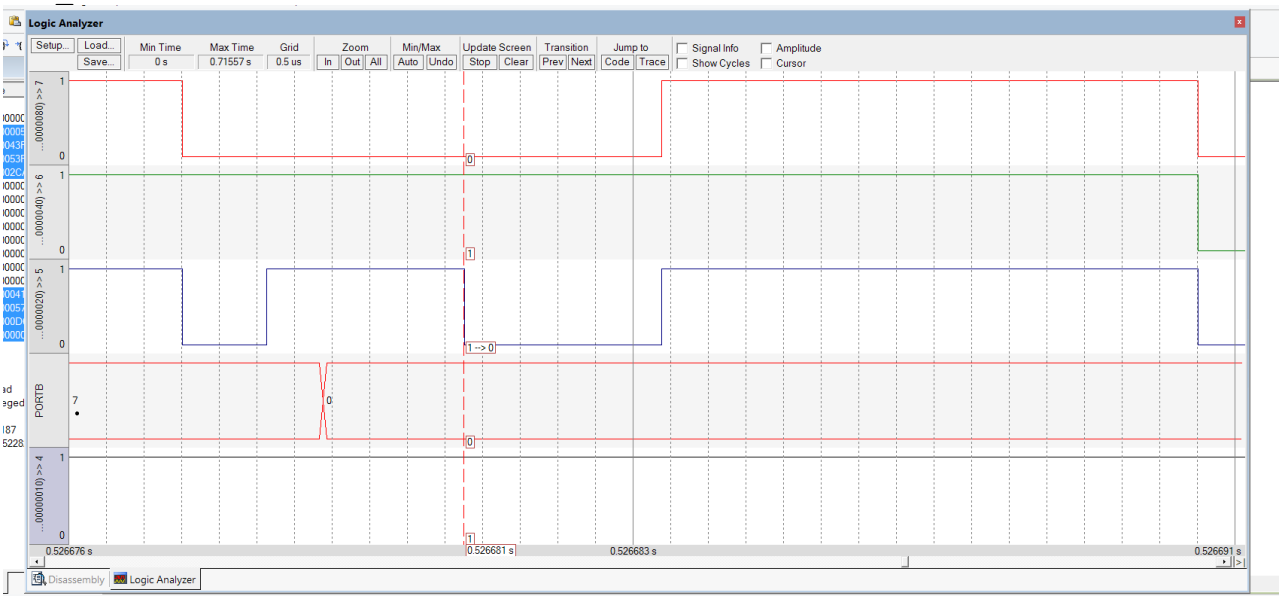
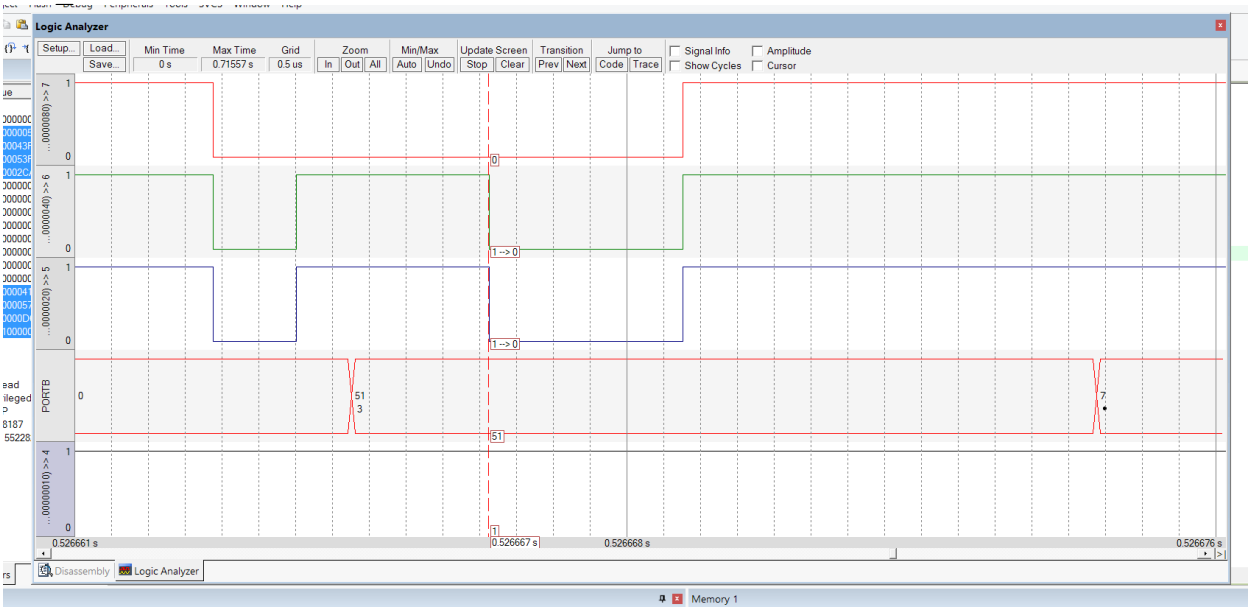
Ben Fu and Rochelle Roberts
Lab 7 Report

LCD_WriteData



LCD_WriteCommand



```
; LCD.s
; Student names: change this to your names or look very silly
; Last modification date: change this to the last modification date or
look very silly

; Runs on LM4F120 or TM4C123
; EE319K lab 7 device driver for the Kentec EB-LM4F120-L35
; This is the lowest level driver that interacts directly with
hardware
; As part of Lab 7, students need to implement these three functions
;
;  Data pin assignments:
;  PB0-7   LCD parallel data input
;
;  Control pin assignments:
;  PA4     RD  Read control signal             ----------------------
--
;  PA5     WR  Write control signal            | PA7 | PA6 | PA5 | PA4
|
;  PA6     RS  Register/Data select signal     | CS  | RS  | WR  | RD
|
;  PA7     CS  Chip select signal              ----------------------
--
;
;  Touchpad pin assignments:
;  PA2     Y-                                  -------------  --------
-----
;  PA3     X-                                  | PA3 | PA2 |  | PE5 |
PE4 |
;  PE4     X+    AIN9                          | X-  | Y-  |  | Y+  |
X+  |
;  PE5     Y+    AIN8                          -------------  --------
-----

      EXPORT   LCD_GPIOInit
      EXPORT   LCD_WriteCommand
      EXPORT   LCD_WriteData

      AREA    |.text|, CODE, READONLY, ALIGN=2
      THUMB
      ALIGN
SYSCTL_RCGC2_R          EQU   0x400FE108
GPIO_PORTA_DATA_R       EQU   0x400043FC
GPIO_PORTA_DIR_R        EQU   0x40004400
GPIO_PORTA_AFSEL_R      EQU   0x40004420
GPIO_PORTA_DEN_R        EQU   0x4000451C
GPIO_PORTB_DATA_R       EQU   0x400053FC
GPIO_PORTB_DIR_R        EQU   0x40005400
GPIO_PORTB_AFSEL_R      EQU   0x40005420
GPIO_PORTB_DEN_R        EQU   0x4000551C

; ************** LCD_GPIOInit ****************************
```

```
; Initializes Ports A and B for Kentec EB-LM4F120-L35
; Port A bits 4-7 are output to four control signals
; Port B bits 0-7 are output data is the data bus
; Initialize all control signals high (off)
;  PA4     RD  Read control signal             --------PA6 | PA5 | PA4
|
;  PA6     RS  Register/Data select signal     | CS  | RS  | WR  | RD
|
;  PA7     CS  Chip select signal    -----------------
;  PA5     WR  Write control signal            | PA7 |           -----
--------------------
; wait 40 us
; Invariables: This function must not permanently modify registers R4
to R11
LCD_GPIOInit
      LDR R1, =SYSCTL_RCGC2_R
      LDR R0, [R1]
      ORR R0, #0x03         ;initialize Port A and B clock
      STR R0, [R1]
      NOP
      NOP
      LDR R1, =GPIO_PORTA_DIR_R
      LDR R0, [R1]
      ORR R0, #0xF0              ;set portA bits[7-4] to 1
      STR R0, [R1]
      LDR R1, =GPIO_PORTB_DIR_R
      LDR R0, [R1]
      ORR R0, #0xFF              ;set portB bits[7-0] to 1
      STR R0, [R1]
      LDR R1, =GPIO_PORTA_AFSEL_R
      LDR R0, [R1]
      AND R0, #0x0F              ;turn off portA AFSEL
      STR R0, [R1]
      LDR R1, =GPIO_PORTB_AFSEL_R
      LDR R0, [R1]
      AND R0, #0x00              ;turn off portB AFSEL
      STR R0, [R1]
      LDR R1, =GPIO_PORTA_DEN_R
      LDR R0, [R1]
      ORR R0, #0xF0              ;enable
      STR R0, [R1]
      LDR R1, =GPIO_PORTB_DEN_R
      LDR R0, [R1]
      ORR R0, #0xFF              ;enable
      STR R0, [R1]
    LDR R1, =GPIO_PORTA_DATA_R
      LDR R0, [R1]
      ORR R0, #0xF0              ;make control signals high (OFF)
      STR R0, [R1]
      ;delay sequence
      MOV R1,#640                     ;set R1 to a 1us delay
delay
```

```
      SUB R1,#1
      CMP R1,#0
      BNE delay
      BX  LR

;* * * * * * * * End of LCD_GPIOInit * * * * * * * *


; ************** LCD_WriteCommand ************************
; - Writes an 8-bit command to the LCD controller
; - RS low during command write
; 8-bit command passed in R0
; 1) LCD_DATA = 0x00;    // Write 0 as MSB of command
; 2) LCD_CTRL = 0x10;    // Set CS, WR, RS low
; 3) LCD_CTRL = 0x70;    // Set WR and RS high
; 4) LCD_DATA = command; // Write 8-bit LSB command
; 5) LCD_CTRL = 0x10;    // Set WR and RS low
; 6) wait 2 bus cycles
; 7) LCD_CTRL = 0xF0;    // Set CS, WR, RS high
; ******************************************************
; Invariables: This function must not permanently modify registers R4
to R11
LCD_WriteCommand
      AND R3, #0
      LDR R2, =GPIO_PORTB_DATA_R ;R2=address PortB,data
      STRB R3, [R2]                    ;write 0 as MSB of command
      LDR R1, =GPIO_PORTA_DATA_R ;R1=address PortA,ctrl
      LDR R3, [R1]
      AND R3, #0x10                    ;set CS, RS, and WR low (RD
always high for command)
      STRB R3, [R1]
      ORR R3, #0x70                    ;set RS and WR High
      STRB R3, [R1]
      STRB R0, [R2]                    ;write command (in R0) to
LCD_DATA
      LDR R2, [R1]
      AND R2, #0x10                    ;set RS and WR low
      STRB R2, [R1]
      NOP
      NOP                                   ;waiting two bus cycles
      LDR R3, [R1]
      ORR R3, #0xF0                    ;set CS, RS, WR high
      STRB R3, [R1]
      BX  LR

;* * * * * * * * End of LCD_WriteCommand * * * * * * * *

; ************** LCD_WriteData ***************************
; - Writes 16-bit data to the LCD controller
; - RS high during data write
; 16-bit data passed in R0
; 1) LCD_DATA = (data>>8);  // Write MSB to LCD data bus
```

```
; 2) LCD_CTRL = 0x50;       // Set CS, WR low
; 3) LCD_CTRL = 0x70;       // Set WR high
; 4) LCD_DATA = data;       // Write LSB to LCD data bus
; 5) LCD_CTRL = 0x50;       // Set WR low
; 6) wait 2 bus cycles
; 7) LCD_CTRL = 0xF0;       // Set CS, WR high
; ***************************************************
; Invariables: This function must not permanently modify registers R4
to R11
LCD_WriteData
      MOV R1, R0
      AND R1, #0xFF00            ;mask for MSB
      LSR R1, #8                 ;R1=shifted MSB
      AND R0, #0x00FF            ;R0=LSB
      LDR   R3, =GPIO_PORTB_DATA_R          ;R3=address PORTB data
      STRH R1, [R3]              ;write MSB to LCD data bus
      LDR R2, =GPIO_PORTA_DATA_R       ;R2=address PORTA data
      LDR R1, [R2]
    AND R1, #0x50                ;set CS, WR low
      STRB R1, [R2]
      ORR R1, #0x70              ;set WR high
      STRH R1, [R2]              ;store in PORTA data
      STRH R0, [R3]              ;write LSB to LCD data bus
      LDR R1, [R2]
      AND R1, #0x50
      STRH R1, [R2]              ;set WR low
      NOP
      NOP
      LDR R1, [R2]
      ORR R1, #0xF0
      STRH R1, [R2]
      BX  LR

;* * * * * * * * End of LCD_WriteData * * * * * * * *



    ALIGN                           ; make sure the end of this
section is aligned
    END                             ; end of file
```

```
; IO.s
; Student names: change this to your names or look very silly
; Last modification date: change this to the last modification date or
look very silly

; Runs on LM4F120 or TM4C123
; EE319K lab 7 device driver for the switch and LED
; You are allowed to use any switch and any LED,
; although the Lab suggests the SW1 switch PF4 and Red LED PF1

; As part of Lab 7, students need to implement these three functions

;  This example accompanies the book
;  "Embedded Systems: Introduction to ARM Cortex M Microcontrollers"
;  ISBN: 978-1469998749, Jonathan Valvano, copyright (c) 2013
;
;Copyright 2013 by Jonathan W. Valvano, valvano@mail.utexas.edu
;   You may use, edit, run or distribute this file
;   as long as the above copyright notice remains
;THIS SOFTWARE IS PROVIDED "AS IS".  NO WARRANTIES, WHETHER EXPRESS,
IMPLIED
;OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF
;MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS
SOFTWARE.
;VALVANO SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL,
INCIDENTAL,
;OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.
;For more information about my classes, my research, and my books, see
;http://users.ece.utexas.edu/~valvano/

; negative logic SW2 connected to PF0 on the Launchpad
; red LED connected to PF1 on the Launchpad
; blue LED connected to PF2 on the Launchpad
; green LED connected to PF3 on the Launchpad
; negative logic SW1 connected to PF4 on the Launchpad

        EXPORT   IO_Init
        EXPORT   IO_Touch
        EXPORT   IO_HeartBeat

GPIO_PORTF_DATA_R  EQU 0x400253FC
GPIO_PORTF_DIR_R   EQU 0x40025400
GPIO_PORTF_AFSEL_R EQU 0x40025420
GPIO_PORTF_PUR_R   EQU 0x40025510
GPIO_PORTF_DEN_R   EQU 0x4002551C
GPIO_PORTF_LOCK_R  EQU 0x40025520
GPIO_PORTF_CR_R    EQU 0x40025524
GPIO_PORTF_AMSEL_R EQU 0x40025528
GPIO_PORTF_PCTL_R  EQU 0x4002552C
GPIO_LOCK_KEY      EQU 0x4C4F434B  ; Unlocks the GPIO_CR register

SYSCTL_RCGC2_R     EQU 0x400FE108
```

```
        AREA    |.text|, CODE, READONLY, ALIGN=2
        THUMB

;------------IO_Init------------
; Initialize GPIO Port for a switch and an LED
; Input: none
; Output: none
; Invariables: This function must not permanently modify registers R4
to R11
IO_Init
    LDR    R0,=SYSCTL_RCGC2_R              ;enable PORTF
      LDR   R1,[R0]
      ORR   R1,#0x20
      STR   R1,[R0]
      NOP
      NOP

      LDR   R0,=GPIO_PORTF_DIR_R       ;set PF0 and PF4 as inputs, PF3-
1 as outputs
      LDR   R1,[R0]
      ORR   R1,#0x0E
      STR   R1,[R0]

      LDR   R0,=GPIO_PORTF_AFSEL_R           ;clear PF4-0
      LDR   R1,[R0]
      AND   R1,#0xE1
      STR   R1,[R0]

      LDR   R0,=GPIO_PORTF_DEN_R       ;enable PF4-0
      LDR   R1,[R0]
      ORR   R1,#0x1E
      STR   R1,[R0]

      LDR   R0,=GPIO_PORTF_PUR_R             ;enable PUR for PF0 and PF4
      LDR   R1,[R0]
      ORR   R1,#0x10
      STR   R1,[R0]

    BX  LR
;* * * * * * * * End of IO_Init * * * * * * * *

;------------IO_HeartBeat------------
; Toggle the output state of the  LED.
; Input: none
; Output: none
; Invariables: This function must not permanently modify registers R4
to R11
IO_HeartBeat
    LDR    R0,=GPIO_PORTF_DATA_R      ;toggle PF3-1
      LDR   R1,[R0]
      EOR   R1,#0x0E
```

```
      STR   R1,[R0]
    BX  LR

;* * * * * * * * End of IO_HeartBeat * * * * * * * *

;------------IO_Touch------------
; wait for release and touch of the switch
; Input: none
; Output: none
; This is a public function
; Invariables: This function must not permanently modify registers R4
to R11
IO_Touch
      LDR   R0,=GPIO_PORTF_DATA_R
      LDR   R1,[R0]
      AND   R1,#0x10              ;see if switch is originally pressed
      CMP   R1,#0x00
      BEQ   Pressed
      CMP   R1,#0x0E
      BEQ   Pressed
NotPressed                        ;wait for switch to be pressed
      LDR   R1,[R0]
      CMP   R1,#0x00
      BEQ   Done
      CMP   R1,#0x0E
      BEQ   Done
      B     NotPressed
Pressed
      LDR   R1,[R0]
      CMP   R1,#0x10              ;wait for switch to be not pressed
      BEQ   Done
      CMP   R1,#0x1E
      BEQ   Done
      B     Pressed
Done
    BX  LR
;* * * * * * * * End of IO_Touch * * * * * * * *

    ALIGN                           ; make sure the end of this
section is aligned
    END                             ; end of file
```

```
; print.s
; Student names: change this to your names or look very silly
; Last modification date: change this to the last modification date or
look very silly
; Runs on LM4F120 or TM4C123
; EE319K lab 7 device driver for the Kentec EB-LM4F120-L35
;
; As part of Lab 7, students need to implement these two functions

;  Data pin assignments:
;  PB0-7   LCD parallel data input
;
;  Control pin assignments:
;  PA4     RD  Read control signal             ----------------------
--
;  PA5     WR  Write control signal            | PA7 | PA6 | PA5 | PA4
|
;  PA6     RS  Register/Data select signal     | CS  | RS  | WR  | RD
|
;  PA7     CS  Chip select signal              ----------------------
--
;
;  Touchpad pin assignments:
;  PA2     Y-                                  ------------  --------
-----
;  PA3     X-                                  | PA3 | PA2 |  | PE5 |
PE4 |
;  PE4     X+    AIN9                          | X-  | Y-  |  | Y+  |
X+  |
;  PE5     Y+    AIN8                          ------------  --------
-----

    IMPORT   LCD_OutChar
    IMPORT   LCD_Goto
    IMPORT   LCD_OutString
    EXPORT   LCD_OutDec
    EXPORT   LCD_OutFix

     AREA  DATA, ALIGN=2
Link1 SPACE 8
Link2 SPACE 8
Counter    SPACE 8
    AREA    |.text|, CODE, READONLY, ALIGN=2
    THUMB
    ALIGN


;-----------------------LCD_OutDec-----------------------
; Output a 32-bit number in unsigned decimal format
; Input: R0 (call by value) 32-bit unsigned number
; Output: none
```

```
; Invariables: This function must not permanently modify registers R4
to R11
LCD_OutDec
      LDR   R1,=Link1        ;save LR
      STR   LR,[R1]
      LDR   R1,=Counter           ;initialize counter to be used in
program
      MOV   R2,#0
      STR   R2,[R1]

n     EQU   0                     ;bind n as a variable on stack

      MOV   R2,#0
      MOV   R1,#12                ;R1 is counter
InitializeStack_Dec
      STR   R2,[SP]               ;clear all locations (set to null)
      SUB   SP,#8            ;allocate 10 32-bit numbers (null
terminated on both sides)
      SUB   R1,#1
      CMP   R1,#0
      BNE   InitializeStack_Dec

      ADD   SP,#16                ;make sure SP points to top of stack
(right address = 0x200003C8)
      MOV   R3,#10000        ;R3 = 10^4
      MOV   R12,#10
      MUL   R3,R3
      MUL   R3,R12                ;R3 = 10^9

Divide_Dec
      CMP   R0,R3            ;see if R3 is too big to be divided
      BLO   NotEnough_Dec
      B     Print_Dec
NotEnough_Dec
      UDIV R3,R12
      CMP   R3,#0
      BEQ   PrintNull_Dec
      B     Divide_Dec
Print_Dec
      CMP   R0,R3            ;see if character is a 0
      BLO   PrintZero_Dec
      UDIV R1,R0,R3          ;get first char into R0
      MUL   R2,R1,R3         ;R2 = (R1/R3)*R3 without the remainder
      SUB   R0,R2            ;R0 = remainder (new number)
      ADD   R1,#0x30         ;convert to ASCII
      STR   R1,[SP,#n]       ;push ASCII character to stack
      ADD   SP,#8            ;increment the SP
      B     Next_Dec

PrintZero_Dec
      MOV   R1,#0x30         ;print "0"
      STR   R1,[SP,#n]       ;push 0x30 (ASCII for 0) to stack
```

```
      ADD   SP,#8
      B     Next_Dec

Next_Dec
      CMP   R3,#1           ;see if last char has been outputted
(checking the ones place)
      BEQ   Done_Dec
      UDIV R3,R12               ;R3 = R3/10 (check next place)
      B     Print_Dec

PrintNull_Dec
      MOV   R0,#0x30
      BL    LCD_OutChar
      B     Finished_Dec

Done_Dec
RecalcSP
      SUB   SP,#8           ;subtract by how many times SP was
incremented
      LDR   R1,[SP]             ;check if string has reached null
terminated ending
      CMP   R1,#0
      BNE   RecalcSP

      ADD   SP,#8           ;SP points to correct address
OutChar_Dec
      LDR   R0,[SP,#n]      ;output character onto LCD
      BL    LCD_OutChar
      ADD   SP,#8           ;increment SP
      LDR   R1,[SP]             ;check for null termination
      CMP   R1,#0
      BNE   OutChar_Dec

      B     Finished_Dec

Finished_Dec
      ADD   SP,#8           ;deallocate so SP points to bottom of stack
      LDR   R1,[SP]
      CMP   R1,#0
      BEQ   Finished_Dec    ;continue until SP points to one past
0x20000418

      SUB   SP,#8           ;reset SP to 0x20000418
      LDR   R1,=Link1       ;restore LR
      LDR   LR,[R1]

      BX    LR

;* * * * * * * * End of LCD_OutDec * * * * * * * *

; ----------------------LCD _OutFix----------------------
; Output characters to LCD display in fixed-point format
```

```
; unsigned decimal, resolution 0.001, range 0.000 to 9.999
; Inputs:  R0 is an unsigned 32-bit number
; Outputs: none
; E.g., R0=0,    then output "0.000 "
;       R0=3,    then output "0.003 "
;       R0=89,   then output "0.089 "
;       R0=123,  then output "0.123 "
;       R0=9999, then output "9.999 "
;       R0>9999, then output "*.*** "
; Invariables: This function must not permanently modify registers R4
to R11
LCD_OutFix
Digit1     EQU  0
Digit2     EQU  8
Digit3     EQU  16
Digit4     EQU  24

     LDR  R1,=Link2       ;save LR
     STR  LR,[R1]

     SUB  SP,#24               ;allocate space for variables
     MOV  R1,#9999        ;see if the character in R0 is greater than
9999
     CMP  R0,R1
     BHI  TooBig

     MOV  R3,#1000        ;R3 = 10^3
     MOV  R12,#10
     MOV  R1,R0           ;transfer number into R1

Print_Outfix
     CMP  R1,R3           ;see if character is a 0
     BLO  PrintZero_Outfix
     UDIV R0,R1,R3        ;get first char into R0
     MUL  R2,R0,R3        ;R2 = (R1/R3)*R3 without the remainder
     SUB  R1,R2           ;R1 = remainder (new number)
     ADD  R0,#0x30        ;convert to ASCII
     STR  R0,[SP]              ;push ASCII character to stack
     ADD  SP,#8
     B    Next_Outfix
PrintZero_Outfix
     MOV  R0,#0x30        ;print a 0
     STR  R0,[SP]              ;push 0x30 (ASCII for 0) to stack
     ADD  SP,#8
     B    Next_Outfix
Next_Outfix
     CMP  R3,#1           ;see if last char has been outputted
     BEQ  Done_Outfix
     UDIV R3,R12               ;R3 = R3/10
     B    Print_Outfix
TooBig
     MOV  R0,#0x2A        ;"*"
```

```
      BL    LCD_OutChar
      MOV   R0,#0x2E        ;"."
      BL    LCD_OutChar
      MOV   R0,#0x2A        ;"*"
      BL    LCD_OutChar
      MOV   R0,#0x2A        ;"*"
      BL    LCD_OutChar
      MOV   R0,#0x2A        ;"*"
      BL    LCD_OutChar
      B     Finished_Outfix
Done_Outfix
      SUB   SP,#32
      LDR   R0,[SP,#Digit1]
      BL    LCD_OutChar
      MOV   R0,#0x2E
      BL    LCD_OutChar
      LDR   R0,[SP,#Digit2]
      BL    LCD_OutChar
      LDR   R0,[SP,#Digit3]
      BL    LCD_OutChar
      LDR   R0,[SP,#Digit4]
      BL    LCD_OutChar
      B     Finished_Outfix
Finished_Outfix
      ADD   SP,#24              ;deallocate stack
      LDR   R1,=Link2       ;restore LR
      LDR   LR,[R1]

    BX     LR
;* * * * * * * * End of LCD_OutFix * * * * * * * * *


    ALIGN                           ; make sure the end of this
section is aligned
    END                             ; end of file
```