[E³]

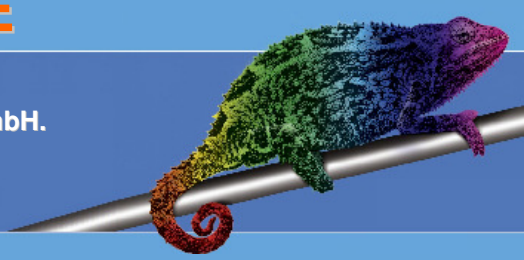
The Third Evolution™

Programmable, RGB-backlit LCD Keyswitches

APPLICATION NOTE

PIC SOURCE CODE

[E³]



PIC Source Code

The following Assembler source is taken from our firmware, which is used in our DS00000 DemoBoard prototyping kit. The DemoBoard is operated by a PIC16F627/8.

```
;*****************************************************************
;               Sample Code, derived from DemoBoard              *
;*****************************************************************
;    Filename:      Sample source PIC.asm                        *
;    Date:          13.02.2004                                   *
;    File Version:  First Revision 26.01.2004                    *
;    Author:        Reinhard Engstler                            *
;    Company:       [E³] Engstler Elektronik Entwicklung GmbH    *
;*****************************************************************
;    Files required: p16f628.inc                                 *
;********************************************** ***************
;    Notes:  actual code snippets not tested                     *
;            just cut and paste from working demoboard           *
;*****************************************************************

      list      p=16f628      ; list directive to define processor
      #include p16f628.inc    ; processor specific variable definitions

 __CONFIG _CP_OFF & _WDT_OFF & _BODEN_ON & _PWRTE_ON & _INTRC_OSC_NOCLKOUT &
_MCLRE_OFF & _LVP_OFF

; '__CONFIG' directive is used to embed configuration data within .asm file.
; The lables following the directive are located in the respective .inc file.
; See respective data sheet for additional information on configuration word.

;*****************************************************************
;                   Memory Definitions
;*****************************************************************
        cblock 0x20
KeyMask           ; PortMask for writing LCD Data
SABitCount        ; Bitcounter for LCWrite
SABuf       ; transmit buffer for LCD
LCParity          ; count '1' bits for parity generation
```

```
PortAMask          ; save point for TRISA
PortBMask          ; save point for TRISB
Time               ; count Time intterupts
TimeOut            ; Count Time intterupts until Power OFF
TimeOuth           ; High Byte Count
RunCMD             ; Samples RunCMDes on every Timer IRQ
wpattern           ; which pattern to be used
Keyold             ; last key press
bcount             ; how many bytes to go
vTest              ; general purpose test
Colour             ; hand over for Colour value
tmp                ; guess what
Pointer:2          ; Pointer into Program Memory to read tables
RGB:3              ; 7bit values for RedGreenBlue
      endc

w_temp             equ    0x7E   ; variables used for context saving
STATUS_temp equ    0x7F   ; memory always mapped to bank 0

;*****************************************************************
;           Bit Definitions
;*****************************************************************
;           Defines
;*****************************************************************

#DEFINE SA_CLK          PORTB,3 ; LC Clock on RB3
                        ; Can't be changed, as PWM is used!!!!!!!
                        ; for permanent clock tests
#DEFINE SA_DATA         PORTA   ; Data Lines on RA2, RA3
#DEFINE Key1            02      ; RA2 handles Data of Key1
#DEFINE Key2            03      ; RA3 handles Data of Key1

#DEFINE LED_OFF               0x00

#DEFINE Dark_Red              0x10
#DEFINE Red                   0x20
#DEFINE Bright_Red            0x30
;
```

```
#DEFINE Dark_Green              0x04
#DEFINE Green                   0x08
#DEFINE Bright_Green            0x0c
;
#DEFINE Dark_Blue               0x01
#DEFINE Blue                    0x02
#DEFINE Bright_Blue             0x03

#DEFINE Dark_White              0x15
#DEFINE White                   0x2a
#DEFINE Bright_White            0x3f

#DEFINE SA_CMD_WrDisplay        0x40  ; Set Adr of Display (3 nibbles) and Bitmap
(128, 216, 512 nibbles)
#DEFINE SA_CMD_SetColour        0x41  ; Set Colour awaits one param: '00rrggbb'
#DEFINE SA_CMD_EndTr            0x43  ; end of current command execute

;################################################################
      ORG     0x000             ; processor reset vector
      goto    COLD              ; go to beginning of program

      ORG     0x004             ; interrupt vector location
      retfie                    ; return from interrupt

COLD
      bcf     STATUS,5          ; set page to 0   (RP0)
      bcf     STATUS,6          ;             (RP1)
      call    InitREG           ; Initialise Register
      call    InitHW            ; Initialise HW
; --------------------
main
      movlw 0x0c
      movwf KeyMask             ; Key1 and Key2 active

      call  WaitKeyReleased     ; Wait for ....
      clrf  RunCMD              ; deactivate all flagged functions

      movlw .32                 ; wait some time
```

```
      call  WAIT              ; to ensure power OK and Keys ready

      incf  Colour, F         ; next Colour
        call    SA_SetColour  ; and show it

      movlw .32               ; wait some time
      call  WAIT              ; as last command already send endofcmd

      bsf   KeyMask, Key1     ; Select Key1
      bcf   KeyMask, Key2     ; DeSelect Key2
      call  SetBitMap6432     ; Setup Address Pointer
      call  TrBitMap          ; Write Bitmap to Keys

      goto  main
;###################################################################
GetData
      movf  Pointer,w         ; Get address byte high
      movwf PCLATH            ; Set Address Byte high to point to Data
      movf  Pointer+1,w       ; get address byte low
      movwf PCL               ; change Program counter to access data
;
IncPointer
      incf  Pointer+1,f       ; increment address byte low
      btfsc STATUS,Z          ; do we need to correct address byte high
      incf  Pointer,f         ; if do it
      return
;
SetBitMap6432
      movlw HIGH SA6432       ; get table address byte high
      movwf Pointer           ; and set Pointer
      movlw SA6432 & 0xff     ; get table address byte high
      movwf Pointer+1         ; and set Pointer
      movlw 0x00              ; set Byte Count for SA6432
      movwf bcount
      return
;
TrBitMap
      bcf   INTCON,7          ; no Interrupts allowed
```

```
                              ; in order to ensure proper data transfer
      movlw SA_CMD_WrDisplay  ; Start WriteDisplay sequence
      call  SAWrite           ; Transmit command
      movlw 0x00              ; Addressnibble 0: 0
      call  SAWrite           ;
      movlw 0x00              ; Addressnibble 1: 0
      call  SAWrite           ;
      movlw 0x00              ; Addressnibble 2: 0
      call  SAWrite           ;
;
Trloop      call  GetData     ; get data of Bitmap
      andlw 0x0f              ; as there is still the lower nibble
      call  SAWrite           ; and send to Key
      call  GetData           ; get same byte again
      andlw 0xf0              ; only 4 bits allowed per transmit -SA protocol
      movwf tmp               ; save data
      swapf tmp,w             ; data has to be in the lower nibble
      call  SAWrite           ; send next nibble of data

      call  IncPointer        ; Point to next Byte of Bitmap
      decfsz  bcount, F       ; Bitmap complete?
      goto    Trloop          ; no - nxt byte
      bsf   INTCON,7          ; allow Interrupts
      return

;*********************************************************
;           SAWrite
;*********************************************************
;
SAWrite                       ; write Data = W to SA Key
      movwf SABuf       ; save Data to be send in SABuf

      movlw   .8
      movwf   SABitCount      ; set the #bits to 8

bitout
      btfss       SABuf,7     ; check MSB
      goto  bit_low           ; Skip if Bit was '0'
```

```
      bsf   PORTA, Key1 ; Set Data Line of selected Key (1/2) high
      bsf   PORTA, Key2 ; This can be done with both keys even not selected!
      goto  sndbit
bit_low
      btfsc       KeyMask, Key1     ; if Key1 selected
      bcf   PORTA, Key1 ; send 0-bit to Key1
      btfsc       KeyMask, Key2     ; if Key2 selected
      bcf   PORTA, Key2 ; send 0-bit to Key2
sndbit
      bcf   SA_CLK              ; set clk low data is shifted in now
      bsf   SA_CLK              ; set clk high

      rlf   SABuf, F          ; rotate SABuf right
                              ;
      decfsz  SABitCount, F   ; 8 bits done?
      goto    bitout          ; no - nxt bit

      bsf   PORTA, Key1 ; Set Data Line of both Keys high
      bsf   PORTA, Key2 ; in case of auto clock

      retlw       0

      end
```

If you have additional questions, please contact techsupport@e3-keys.com.

Contact Information

[E³]
Engstler
Elektronik
Entwicklung
GmbH

Industriering 7
63868 Grosswallstadt
Germany

Phone: +49 (0) 6022 262570
Fax: +49 (0) 6022 262571
E-Mail: info@e3-keys.com