```
;**** **** **** **** ****
;
; BLHeli program for controlling brushless motors in helicopters and
multirotors
;
; Copyright 2011, 2012 Steffen Skaug
; This program is distributed under the terms of the GNU General Public
License
;
; This file is part of BLHeli.
;
; BLHeli is free software: you can redistribute it and/or modify
; it under the terms of the GNU General Public License as published by
; the Free Software Foundation, either version 3 of the License, or
; (at your option) any later version.
;
; BLHeli is distributed in the hope that it will be useful,
; but WITHOUT ANY WARRANTY; without even the implied warranty of
; MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
; GNU General Public License for more details.
;
; You should have received a copy of the GNU General Public License
; along with BLHeli.  If not, see <http://www.gnu.org/licenses/>.
;
;**** **** **** **** ****
;
; XP 7A3A hardware definition file
;
;**** **** **** **** ****



;*********************
; Device SiLabs F330
;*********************
$include (c8051f330.inc)

;**** **** **** **** ****
; Uses internal calibrated oscillator set to 24Mhz
;**** **** **** **** ****

;**** **** **** **** ****
; Constant definitions
;**** **** **** **** ****
CSEG AT 1A40h
Eep_ESC_Layout:         DB    "#XP7AXP3A#          " ; ESC layout tag
CSEG AT 1A50h
Eep_ESC_MCU:                  DB    "#BLHELI#F330#   "     ; Project and
MCU tag (16 Bytes)

ONE_S_CAPABLE                 EQU   1     ; Set to 1 if ESC can operate at
1S
PORT3_EXIST             EQU   0     ; Set to 1 if MCU has port3
```

```
COMP1_USED              EQU   0     ; Set to 1 if MCU has comparator 1 and
it is being used
LOCK_BYTE_ADDRESS_16K   EQU   3FFFh ; Address of lock byte if 16k flash
size
LOCK_BYTE_ADDRESS_8K    EQU   1DFFh ; Address of lock byte if 8k flash size
DUAL_BEC_VOLTAGE        EQU   0     ; Set to 1 if dual BEC voltage is
supported
DAMPED_MODE_ENABLE            EQU   01    ; Damped mode disabledenabled
NFETON_DELAY                  EQU   505   ; Wait delay from pfets off to
nfets on
PFETON_DELAY                  EQU   1     ; Wait delay from nfets off to
pfets on
COMP_PWM_HIGH_ON_DELAY EQU   10    ; Wait delay from pwm on until
comparator can be read (for high pwm frequency)
COMP_PWM_HIGH_OFF_DELAY       EQU   15    ; Wait delay from pwm off until
comparator can be read (for high pwm frequency)
COMP_PWM_LOW_ON_DELAY  EQU   4     ; Wait delay from pwm on until
comparator can be read (for low pwm frequency)
COMP_PWM_LOW_OFF_DELAY EQU   5     ; Wait delay from pwm off until
comparator can be read (for low pwm frequency)
HIGH_DRIVER_PRECHC_TIME       EQU   0     ; Time between commutations use
to precharge the high side driver (for all nfet ESCs)
ADC_LIMIT_L             EQU   85171 ; Power supply measurement ADC value
for which main motor power is limited (low byte)
ADC_LIMIT_H             EQU   01    ; Power supply measurement ADC value
for which main motor power is limited (2 MSBs)
TEMP_LIMIT              EQU   109246      ; Temperature measurement ADC
value for which main motor power is limited (low byte, assuming high byte
is 1)
TEMP_LIMIT_STEP         EQU   46    ; Temperature measurement ADC value
increment for which main motor power is further limited
MAIN_SPOOLUP_TIME       EQU   7     ; Main motor spoolup time

;**** **** **** **** ****
; ESC specific defaults
;**** **** **** **** ****
; T.Hiro for HP03s 13300kv
;DEFAULT_PGM_MAIN_STARTUP_PWR            EQU 11      ; 1=0.031 2=0.047
3=0.063 4=0.094 5=0.125 6=0.188    7=0.25  8=0.38  9=0.50  10=0.75 11=1.00
12=1.25 13=1.50
DEFAULT_PGM_MAIN_STARTUP_PWR            EQU 13      ; 1=0.031 2=0.047
3=0.063 4=0.094 5=0.125 6=0.188    7=0.25  8=0.38  9=0.50  10=0.75 11=1.00
12=1.25 13=1.50
DEFAULT_PGM_TAIL_STARTUP_PWR            EQU 11      ; 1=0.031 2=0.047
3=0.063 4=0.094 5=0.125 6=0.188    7=0.25  8=0.38  9=0.50  10=0.75 11=1.00
12=1.25 13=1.50
DEFAULT_PGM_MULTI_STARTUP_PWR      EQU 11      ; 1=0.031 2=0.047 3=0.063
4=0.094 5=0.125 6=0.188      7=0.25  8=0.38  9=0.50  10=0.75 11=1.00
12=1.25 13=1.50
; T.Hiro
; DEFAULT_PGM_MAIN_STARTUP_METHOD  EQU 1       ; 1=Stepped 2=Direct
DEFAULT_PGM_MAIN_STARTUP_METHOD    EQU 1       ; 1=Stepped 2=Direct
DEFAULT_PGM_TAIL_STARTUP_METHOD    EQU 1       ; 1=Stepped 2=Direct
DEFAULT_PGM_MULTI_STARTUP_METHOD   EQU 2       ; 1=Stepped 2=Direct
```

```
;*********************
; PORT 0 definitions *
;*********************
Rcp_In            EQU   7     ;i
Adc_Ip            EQU   6     ;i
Mux_A       EQU   5     ;i
;                 EQU   4     ;i
Mux_B       EQU   3     ;i
Comp_Com          EQU   2     ;i
Mux_C       EQU   1     ;i
Vref              EQU   0     ;i

P0_DIGITAL  EQU   NOT((1 SHL Mux_A)+(1 SHL Mux_B)+(1 SHL Mux_C)+(1 SHL
Comp_Com)+(1 SHL Adc_Ip)+(1 SHL Vref))
P0_INIT           EQU   0FFh
P0_PUSHPULL EQU   0
P0_SKIP           EQU   NOT(1 SHL Rcp_In) AND 0FFh

MACRO Get_Rcp_Capture_Values MACRO
      mov   Temp1, PCA0CPL0             ; Get PCA capture values
      mov   Temp2, PCA0CPH0
ENDM
MACRO Read_Rcp_Int MACRO
      mov   A, P0
      jnb   Flags3.PGM_RCP_PWM_POL, ($+4)      ; Is pwm polarity negative?
      cpl   A                                  ; Yes - invert
ENDM
MACRO Rcp_Int_Enable MACRO
      orl   PCA0CPM0, #01h                     ; Interrupt enabled
ENDM
MACRO Rcp_Int_Disable MACRO
      anl   PCA0CPM0, #0FEh             ; Interrupt disabled
ENDM
MACRO Rcp_Int_First MACRO
      anl   PCA0CPM0, #0CFh
      jb    Flags3.PGM_RCP_PWM_POL, ($+6)      ; Is pwm polarity positive?
      orl   PCA0CPM0, #20h                     ; Capture rising edge
      jnb   Flags3.PGM_RCP_PWM_POL, ($+6)      ; Is pwm polarity negative?
      orl   PCA0CPM0, #10h                     ; Capture falling edge
ENDM
MACRO Rcp_Int_Second MACRO
      anl   PCA0CPM0, #0CFh
      jb    Flags3.PGM_RCP_PWM_POL, ($+6)      ; Is pwm polarity positive?
      orl   PCA0CPM0, #10h                     ; Capture falling edge
      jnb   Flags3.PGM_RCP_PWM_POL, ($+6)      ; Is pwm polarity negative?
      orl   PCA0CPM0, #20h                     ; Capture rising edge
ENDM
MACRO Rcp_Clear_Int_Flag MACRO
      clr   CCF0                               ; Clear interrupt flag
ENDM
```

```
;*********************
; PORT 1 definitions *
;*********************
;                EQU   7     ;i
;                EQU   6     ;i
ApFETCpFET       EQU   5     ;o
BpFET       EQU   4     ;o
CpFETApFET       EQU   3     ;o
AnFETCnFET       EQU   2     ;o
BnFET       EQU   1     ;o
CnFETAnFET       EQU   0     ;o

P1_DIGITAL  EQU   (1 SHL AnFET)+(1 SHL BnFET)+(1 SHL CnFET)+(1 SHL
ApFET)+(1 SHL BpFET)+(1 SHL CpFET)
P1_INIT          EQU   0           (1 SHL ApFET)+(1 SHL BpFET)+(1 SHL
CpFET)           ; Setting pFET outputs turn them off
P1_PUSHPULL EQU   (1 SHL AnFET)+(1 SHL BnFET)+(1 SHL CnFET)+(1 SHL
ApFET)+(1 SHL BpFET)+(1 SHL CpFET)
P1_SKIP          EQU   0

MACRO AnFET_on MACRO
      mov   A, Current_Pwm_Limited
      jz    ($+12)
      jb    Flags3.PGM_DIR_REV, ($+5)
      setb  P1.AnFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      setb  P1.CnFET
ENDM
MACRO AnFET_off MACRO
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.AnFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      clr   P1.CnFET
ENDM
MACRO BnFET_on MACRO
      mov   A, Current_Pwm_Limited
      jz    ($+4)
      setb  P1.BnFET
ENDM
MACRO BnFET_off MACRO
      clr   P1.BnFET
ENDM
MACRO CnFET_on MACRO
      mov   A, Current_Pwm_Limited
      jz    ($+12)
      jb    Flags3.PGM_DIR_REV, ($+5)
      setb  P1.CnFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      setb  P1.AnFET
ENDM
MACRO CnFET_off MACRO
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.CnFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
```

```
      clr   P1.AnFET
ENDM
MACRO All_nFETs_Off MACRO
      clr   P1.AnFET
      clr   P1.BnFET
      clr   P1.CnFET
ENDM

MACRO ApFET_on MACRO
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.ApFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      clr   P1.CpFET
ENDM
MACRO ApFET_off
      jb    Flags3.PGM_DIR_REV, ($+5)
      setb  P1.ApFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      setb  P1.CpFET
ENDM
ApFET_off MACRO BpFET_on
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.ApFETBpFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      clr   P1.CpFET
ENDM
BpFET_on MACRO BpFET_off
      setb  P1.BpFET
ENDM
BpFET_off MACRO CpFET_on
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.BpFETCpFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      clr   P1.ApFET
ENDM
CpFET_on MACRO CpFET_off
      jb    Flags3.PGM_DIR_REV, ($+5)
      setb  P1.CpFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      setb  P1.ApFET
ENDM
CpFET_off MACRO
      jb    Flags3.PGM_DIR_REV, ($+5)
      clr   P1.CpFET
      jnb   Flags3.PGM_DIR_REV, ($+5)
      clr   P1.ApFET
ENDM
All_pFETs_Off MACRO
      clr   P1.ApFET
      clr   P1.BpFET
      clr   P1.CpFET
ENDM
All_pFETs_On MACRO
      setb  P1.ApFET
```

```
      setb  P1.BpFET
      setb  P1.CpFET
ENDM
MACRO All_pFETs_On
      clr   P1.ApFET
      clr   P1.BpFET
      clr   P1.CpFET
ENDM

MACRO Set_Comp_Phase_A MACRO
      jb    Flags3.PGM_DIR_REV, ($+6)
      mov   CPT0MX, #21h      ; Set comparator multiplexer to phase A
      jnb   Flags3.PGM_DIR_REV, ($+6)
      mov   CPT0MX, #01h
ENDM
MACRO Set_Comp_Phase_B MACRO
      mov   CPT0MX, #11h      ; Set comparator multiplexer to phase B
ENDM
MACRO Set_Comp_Phase_C MACRO
      jb    Flags3.PGM_DIR_REV, ($+6)
      mov   CPT0MX, #01h      ; Set comparator multiplexer to phase C
      jnb   Flags3.PGM_DIR_REV, ($+6)
      mov   CPT0MX, #21h
ENDM
MACRO Read_Comp_Out MACRO
      mov   A, CPT0CN         ; Read comparator output
      cpl   A                 ; Invert output
ENDM


;*********************
; PORT 2 definitions *
;*********************
DebugPin          EQU   0     ;o

P2_PUSHPULL EQU   (1 SHL DebugPin)


;***********************
; MCU specific macros *
;***********************
MACRO Interrupt_Table_Definition MACRO
CSEG AT 0                     ; Code segment start
      jmp reset
CSEG AT 0Bh             ; Timer0 interrupt
      jmp   t0_int
CSEG AT 2Bh             ; Timer2 interrupt
      jmp   t2_int
CSEG AT 5Bh             ; PCA interrupt
      jmp   pca_int
CSEG AT 73h             ; Timer3 interrupt
      jmp   t3_int
ENDM
```

```
MACRO Initialize_Xbar MACRO
      mov   XBR1, #41h  ; Xbar enabled, CEX0 routed to pin Rcp_In

ENDM

MACRO Initialize_Adc MACRO
      mov   REF0CN, #0Eh      ; Set vdd (3.3V) as reference.07h ; Enable
temp sensor, bias and biasreference buffer
      mov   ADC0CF, #58h      ; ADC clock 2MHz
      mov   AMX0P, #Adc_Ip    ; Select positive input
      mov   AMX0N, #11h ; Select negative input as ground
      mov   ADC0CN, #80h      ; ADC enabled
ENDM
MACRO Set_Adc_Ip_Volt MACRO
      mov   AMX0P, #Adc_Ip    ; Select positive input
ENDM
MACRO Set_Adc_Ip_Temp MACRO
      mov   AMX0P, #10h ; Select temp sensor input
ENDM
MACRO Start_Adc MACRO
      mov   ADC0CN, #90h      ; ADC start
ENDM
MACRO Get_Adc_Status MACRO
      mov   A, ADC0CN
ENDM
MACRO Read_Adc_Result MACRO
      mov   Temp1, ADC0L
      mov   Temp2, ADC0H
ENDM
MACRO Stop_Adc MACRO
ENDM
```