DAvE's Project Documentation

Project: **DaveTest.dav**

Controller: **XC866-4FR**

Compiler: **Keil**

Date: **09.02.2005 00:47:37**

**Please read this document carefully and note**

**the red-colored hints.**

**If you miss a file in the generated files list**

**maybe you have forgotten to select the**

**initialisation function of the related module.**

Generated Files:

**MAIN.H**

**MAIN.C**

**START\_XC.A51**

**IO.H**

**IO.C**

**T01.H**

**T01.C**

**T2.H**

**T2.C**

**CC6.H**

**CC6.C**

**DAVETEST.ASM**

Project Settings

Macros:

Functions:

Function:

**void MAIN\_vInit(void)**

Description:

*This function initializes the microcontroller. It is*

*assumed that the SFRs are in their reset state.*

Returnvalue:

**None**

Parameters:

**None**

Function:

**void main(void)**

Description:

*This is the main function.*

Returnvalue:

**None**

Parameters:

**None**

Initialization:

**Configuration of the System Clock:**

**- On Chip Osc is Selected**

**- PLL Mode, NDIV = 2**

**- input frequency is 9.4 MHz**

**Initialization of module 'GPIO'**

**Initialization of module 'Timer 0/1'**

**Initialization of module 'Timer 2'**

**Initialization of module 'Capture / Compare Unit 6 (CCU6)'**

GPIO

Macros:

Macro:

**IO\_ubReadPin(PinName)**

Description:

*This macro returns the status of the chosen portpin.*

*Note:*

*'PinName' identifies the pin of a bit-addressable port. The*

*default names can be changed in the port configuration*

*dialog.*

Returnvalue:

**Status of the chosen portpin**

Parameters:

**PinName:**

**Pin to be read**

Macro:

**IO\_vSetPin(PinName)**

Description:

*The chosen portpin is set to '1'.*

*Note:*

*'PinName' identifies the pin of a bit-addressable port. The*

*default names can be changed in the port configuration*

*dialog.*

Returnvalue:

**None**

Parameters:

**PinName:**

**Pin to be set to '1'**

Macro:

**IO\_vResetPin(PinName)**

Description:

*The chosen portpin is set to '0'.*

*Note:*

*'PinName' identifies the pin of a bit-addressable port. The*

*default names can be changed in the port configuration*

*dialog.*

Returnvalue:

**None**

Parameters:

**PinName:**

**Pin to be set to '0'**

Macro:

**IO\_vTogglePin(PinName)**

Description:

*The chosen portpin will be toggled.*

*Note:*

*'PinName' identifies the pin of a bit-addressable port. The*

*default names can be changed in the port configuration*

*dialog.*

Returnvalue:

**None**

Parameters:

**PinName:**

**Pin to be toggled**

Macro:

**IO\_ubReadPort(PortName)**

Description:

*This macro returns the value of the chosen port.*

*The following definitions for PortName are available:*

*P0, P1, P2, P3*

Returnvalue:

**Value of the port**

Parameters:

**PortName:**

**Port to be read**

Macro:

**IO\_vWritePort(PortName, Data)**

Description:

*The forwarded data is written to the chosen port.*

*The following definitions for PortName are available:*

*P0, P1, P3*

Returnvalue:

**None**

Parameters:

**PortName:**

**Port to be altered**

**Data:**

**Value to be written to the chosen port**

Macro:

**IO\_vSetInput(PortName, Mask)**

Description:

*This macro changes the direction of I/O port pins to input.*

*Parameter 'PortName' selects the port and parameter 'Mask'*

*selects the lines of this port.*

*Example: IO\_vSetInput(P1, 0x11) sets P1.0 and P1.4 to input.*

Returnvalue:

**None**

Parameters:

**PortName:**

**Port to be configured**

**Mask:**

**Mask for port pins to be configured**

Macro:

**IO\_vSetOutput(PortName, Mask)**

Description:

*This macro changes the direction of I/O port pins to*

*output. Parameter 'PortName' selects the port and parameter*

*'Mask' selects the lines of this port.*

*Example: IO\_vSetOutput(P1, 0x11) sets P1.0 and P1.4 to*

*output.*

Returnvalue:

**None**

Parameters:

**PortName:**

**Port to be configured**

**Mask:**

**Mask for port pins to be configured**

Functions:

Function:

**void IO\_vInit(void)**

Description:

*This is the initialization function of the IO function*

*library. It is assumed that the SFRs used by this library*

*are in their reset state.*

Returnvalue:

**None**

Parameters:

**None**

Initialization:

**Configuration of Port P0:**

**- no pin of port P0 is used**

**Configuration of Port P1:**

**- no pin of port P1 is used**

**Configuration of Port P2:**

**- no pin of port P2 is used**

**Configuration of Port P3:**

**P3.0:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.1:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.2:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.3:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.4:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.5:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.6:**

**- is used as general purpose output**

**- open drain output is selected**

**- the pin status is low level**

**- pull-up device is assigned**

**P3.7:**

**- is used as alternate output for the Output of Capture/Compare**

**channel 3**

**- push/pull output is selected**

**- pull-up device is assigned**

Timer 0/1

Macros:

Functions:

Function:

**void T01\_vInit(void)**

Description:

*This is the initialization function of the Timer 0/1*

*function library. It is assumed that the SFRs used by this*

*library are in their reset state.*

*The following SFR fields will be initialized for both*

*timers 0 and 1:*

*T0M - Timer 0 mode*

*T1M - Timer 1 mode*

*GATE0/GATE1 - gating control*

*TLx, THx - timer low/high registers*

*ETx - interrupt enable*

*TRx - timer run control (this is the final*

*action)*

Returnvalue:

**None**

Parameters:

**None**

Function:

**void T01\_viTmr0(void)**

Description:

*This is the service routine for the timer 0 interrupt. It*

*is called*

*- when TH0 overflows if timer 0 is in mode 0 or 1*

*- when TL0 overflows if timer 0 is in mode 2 or 3*

*Please note that you have to add application specific code*

*to this function.*

Returnvalue:

**none**

Parameters:

**None**

Function:

**void T01\_viTmr1(void)**

Description:

*This is the service routine for the timer 1 interrupt. It*

*is called*

*- when TH1 overflows if timer 1 is in mode 0 or 1 and timer*

*0 is not in mode 3*

*- when TL1 overflows if timer 1 is in mode 2 and timer 0 is*

*not in mode 3*

*- when TH0 overflows if timer 0 is in mode 3*

*Please note that you have to add application specific code*

*to this function.*

Returnvalue:

**none**

Parameters:

**None**

Initialization:

**Timer 0**

**Mode 0: 8-bit timer (TH0) with a divide-by-32 prescaler (TL0)**

**the timer 0 overflow is 65,362 µs**

**Timer 1**

**Mode 0: 8-bit timer (TH1) with a divide-by-32 prescaler (TL1)**

**the timer 1 overflow is 65,362 µs**

**Enable Timer 0 interrupt**

**Enable Timer 1 interrupt**

**Start Timer 0**

**Start Timer 1**

Timer 2

Macros:

Macro:

**T2\_vStartTmr()**

Description:

*This macro starts timer 2 by setting bit TR2.*

Returnvalue:

**None**

Parameters:

**None**

Macro:

**T2\_vStopTmr()**

Description:

*This macro stops timer 2 by clearing bit TR2.*

Returnvalue:

**None**

Parameters:

**None**

Macro:

**T2\_uwReadTmr()**

Description:

*This macro reads the contents of the T2 timer register. The*

*timer is not stopped.*

*Please take into account that an active timer 2 unit may*

*interfere with the sequence of high and low byte access.*

*Thus you should stop the timer first, unless you are sure*

*that the access takes place at a save moment.*

Returnvalue:

**16-bit timer register contents**

Parameters:

**None**

Macro:

**T2\_vSetRC2Reg(Value)**

Description:

*This macro writes the passed value into the reload/capture*

*timer 2 register (RC2H/RC2L).*

*Please take into account that an active timer 2 unit may*

*interfere with the sequence of high and low byte access.*

*Thus you should stop the timer first, unless you are sure*

*that the access takes place at a save moment.*

Returnvalue:

**None**

Parameters:

**Value:**

**16-bit timer reload value**

Macro:

**T2\_uwGetRC2Reg()**

Description:

*This macro reads the contents of the reload/capture timer 2*

*register (RC2H/RC2L).*

*Please take into account that an active timer 2 unit may*

*interfere with the sequence of high and low byte access.*

*Thus you should stop the timer first, unless you are sure*

*that the access takes place at a save moment.*

Returnvalue:

**16-bit reload/capture register contents**

Parameters:

**None**

Functions:

Function:

**void T2\_vInit(void)**

Description:

*This is the initialization function of the Timer 2 function*

*library. It is assumed that the SFRs used by this library*

*are in their reset state.*

*The following SFRs and SFR fields will be initialized:*

*T2\_RC2H/RC2L - reload/capture timer 2 register*

*ET2 - timer 2 interrupt enable*

*T2\_T2MOD - timer 2 mode register*

*CP/RL2 - Capture/Reload select*

*EXEN2 - External enable control*

*TR2 - Timer2 run control*

Returnvalue:

**None**

Parameters:

**None**

Function:

**void T2\_viTmr(void)**

Description:

*This is the service routine for the Timer 2 interrupt.*

*Depending on the selected operating mode it is called when*

*TF2 is set by an overflow or underflow of the timer 2*

*register or when EXF2 is set by a negative transition on*

*T2EX.*

*Please note that you have to add application specific code*

*to this function.*

Returnvalue:

**none**

Parameters:

**None**

Initialization:

**16-bit timer function with automatic reload when timer 2 overflows**

**Prescaler enabled - input clock = fPCLK/16**

**the timer 2 resolution is 0,766 µs**

**the timer 2 overflow is 50197,787 µs**

**timer 2 interrupt: enabled**

**timer 2 will be started**

Capture / Compare Unit 6 (CCU6)

Macros:

Functions:

Function:

**void CC6\_vInit(void)**

Description:

*This is the initialization function of the CCU6 function*

*library. It is assumed that the SFRs used by this library*

*are in their reset state.*

Returnvalue:

**None**

Parameters:

**None**

Function:

**void CC6\_viNodeI0(void)**

Description:

*This is the interrupt service routine for the CC6 node I0.*

*If the content of the corresponding compare timer*

*(configurable) equals the content of the capture/compare*

*register or if a capture event occurs at the associated*

*port pin, the interrupt request flag is set and an*

*interrupt is triggered (only if enabled).*

*Please note that you have to add application specific code*

*to this function.*

Returnvalue:

**None**

Parameters:

**None**

Function:

**void CC6\_vStartTmr(ubyte TimerNr)**

Description:

*This Function starts the selected CC6 timer. The timer*

*continues counting from where it stopped.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be started**

Function:

**void CC6\_vStopTmr(ubyte TimerNr)**

Description:

*This Function stops the selected CC6 timer. The content of*

*the timer register remains unchanged.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be stopped**

Function:

**void CC6\_vResetTmr(ubyte TimerNr)**

Description:

*This function sets the selected CC6 timer counter register*

*to zero. Timer is not stopped.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be cleared**

Function:

**void CC6\_vSetTmrPeriod(ubyte TimerNr, uword Value)**

Description:

*This function loads the period register of the selected CC6*

*timer.*

*Note: Call the function CC6\_vEnableShadowTransfer() to*

*enable the shadow transfer.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be used**

**Value:**

**16-bit Value with which the CC6 timer reload register is to**

**be loaded**

Function:

**void CC6\_vSetDeadTimePeriod(ubyte ubValue)**

Description:

*This function loads the counter of the dead time register.*

Returnvalue:

**None**

Parameters:

**ubValue:**

**Counter value (ubValue = 0x01 .. 0x3F)**

Function:

**void CC6\_vEnableShadowTransfer(ubyte TimerNr)**

Description:

*This function enables the transfer from the shadow*

*registers to the actual CC6 registers.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be used**

Function:

**void CC6\_vLoadChannelShadowRegister(ubyte ChName, uword**

Description:

*This function loads the selected CC6 channel shadow*

*register with the parameter Value.*

*The following definitions are available for RegName:*

*CC6\_CHANNEL\_0 .. CC6\_CHANNEL\_3 (see @Defines in header file)*

Returnvalue:

**None**

Parameters:

**ChName:**

**Name of the CC6 channel to be loaded**

**Value:**

**16-bit Value with which the CC6 shadow register is to be**

**loaded**

Function:

**void CC6\_vSetStatusBit(ubyte ChName)**

Description:

*This function sets the capture / compare modifikation*

*status bit of the selected CC6 channel.*

*The following definitions are available for ChName:*

*CC6\_CHANNEL\_0 .. CC6\_CHANNEL\_3 (see @Defines in header file)*

Returnvalue:

**None**

Parameters:

**ChName:**

**Name of the CC6 channel to be used**

Function:

**void CC6\_vResetStatusBit(ubyte ChName)**

Description:

*This function resets the capture / compare modifikation*

*status bit of the selected CC6 channel.*

*The following definitions are available for ChName:*

*CC6\_CHANNEL\_0 .. CC6\_CHANNEL\_3 (see @Defines in header file)*

Returnvalue:

**None**

Parameters:

**ChName:**

**Name of the CC6 channel to be used**

Function:

**void CC6\_uwSetTmrCounter(ubyte TimerNr, uword Value)**

Description:

*This function loads the selected CC6 timer counter register*

*with the parameter Value . It is possible to use this*

*function only if the selected timer is stopped.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**None**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be used**

**Value:**

**16-bit value with which the CC6 counter register is to be**

**loaded**

Function:

**uword CC6\_uwGetChannelShadowRegister(ubyte ChName)**

Description:

*This function returns the content of the CC6*

*capture/compare shadow register of the selected channel.*

*The following definitions are available for RegName:*

*CC6\_CHANNEL\_0 .. CC6\_CHANNEL\_3 (see @Defines in header file)*

Returnvalue:

**Current 16-bit value of the CC6 shadow register**

Parameters:

**ChName:**

**Name of the CC6 channel to be readed**

Function:

**uword CC6\_uwGetTmrCounter(ubyte TimerNr)**

Description:

*This function returns the contents of the selected CC6*

*timer counter register.*

*The following definitions for TimerNr are available:*

*CC6\_TIMER\_12*

*CC6\_TIMER\_13*

Returnvalue:

**16-bit counter value**

Parameters:

**TimerNr:**

**Name of the CC6 timer to be used**

Function:

**uword CC6\_uwGetChannelRegister(ubyte ChName)**

Description:

*This function returns the content of the CC6*

*capture/compare register of the selected channel.*

*The following definitions are available for RegName:*

*CC6\_CHANNEL\_0 .. CC6\_CHANNEL\_3 (see @Defines in header file)*

Returnvalue:

**Current 16-bit value of the CC6 register**

Parameters:

**ChName:**

**Name of the CC6 channel to be readed**

Initialization:

**Configuration of CCU6 module input signals:**

**- signal CC60 is not used**

**- signal CC61 is not used**

**- signal CC62 is not used**

**- signal #CTRAP is not used**

**- signal CCPOS0 is not used**

**- signal CCPOS1 is not used**

**- signal CCPOS2 is not used**

**- signal T12HR is not used**

**- signal T13HR is not used**

**Configuration of CCU6 Timer 12:**

**- prescaler factor is 1**

**- timer 12 run bit is reset**

**- single shot mode is disabled**

**- timer 12 works in edge aligned mode**

**- interrupt on period match is disabled**

**- interrupt on one match is disabled**

**Configuration of CCU6 Timer 13:**

**- prescaler factor is 128**

**- timer 13 run bit is reset**

**- trigger control is disabled**

**- single shot mode is disabled**

**- interrupt on period match is enabled**

**- interrupt on compare match is disabled**

**Configuration of Multi Channel Mode:**

**- multi channel mode is disabled**

**Configuration of CCU6 Channel 0:**

**- channel 0 is disabled**

**Configuration of CCU6 Channel 1:**

**- channel 1 is disabled**

**Configuration of CCU6 Channel 2:**

**- channel 2 is disabled**

**Configuration of CCU6 Channel 3:**

**- alternate output function COUT63 for the PWM signal generated by T13**

**is enabled**

**- the trap functionality of the pin COUT63 is disabled**

**- the compare output COUT63 drives passive level while CC63ST is ’0’**

**- the passive level of the output COUT63 is '0'**

**- T13 output is not inverted**

**- enable shadow transfer to T12 and T13**

**Configuration of CCU6 trap control:**

**- a trap can only be generated by SW by setting the bit TRPF**

**- the trap state is left when a zero-match of T12 (while counting up)**

**is detected (synchronization to T12)**

**- bit TRPF is automatically cleared by HW (according to TRPPEN, TRPM0**

**and TRPM1)**

**- trap interrupt is disabled**

**Configuration of CCU6 interrupt control:**

**- for channel 0 interrupts is node I0 selected**

**- for channel 1 interrupts is node I0 selected**

**- for channel 2 interrupts is node I0 selected**

**- for correct hall event interrupt is node I0 selected**

**- for error interrupts is node I0 selected**

**- for T12 interrupts is node I0 selected**

**- for T13 interrupts is node I0 selected**

**Configuration of the used CCU6 Channel Port Pins:**

**Configuration of the used CCU6 Channels Interrupts:**

**- capture/compare interrupt node 0 is enabled**

**- capture/compare interrupt node 1 is disabled**

**- capture/compare interrupt node 2 is disabled**

**- capture/compare interrupt node 3 is disabled**