;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;; FILENAME: PushButtonTimer.asm

;; Version: 2.6, Updated on 2015/3/4 at 22:27:47

;; Generated by PSoC Designer 5.4.3191

;;

;; DESCRIPTION: Timer16 User Module software implementation file

;;

;; NOTE: User Module APIs conform to the fastcall16 convention for marshalling

;; arguments and observe the associated "Registers are volatile" policy.

;; This means it is the caller's responsibility to preserve any values

;; in the X and A registers that are still needed after the API functions

;; returns. For Large Memory Model devices it is also the caller's

;; responsibility to perserve any value in the CUR\_PP, IDX\_PP, MVR\_PP and

;; MVW\_PP registers. Even though some of these registers may not be modified

;; now, there is no guarantee that will remain the case in future releases.

;;-----------------------------------------------------------------------------

;; Copyright (c) Cypress Semiconductor 2015. All Rights Reserved.

;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

include "m8c.inc"

include "memory.inc"

include "PushButtonTimer.inc"

;-----------------------------------------------

; Global Symbols

;-----------------------------------------------

export PushButtonTimer\_EnableInt

export \_PushButtonTimer\_EnableInt

export PushButtonTimer\_DisableInt

export \_PushButtonTimer\_DisableInt

export PushButtonTimer\_Start

export \_PushButtonTimer\_Start

export PushButtonTimer\_Stop

export \_PushButtonTimer\_Stop

export PushButtonTimer\_WritePeriod

export \_PushButtonTimer\_WritePeriod

export PushButtonTimer\_WriteCompareValue

export \_PushButtonTimer\_WriteCompareValue

export PushButtonTimer\_wReadCompareValue

export \_PushButtonTimer\_wReadCompareValue

export PushButtonTimer\_wReadTimer

export \_PushButtonTimer\_wReadTimer

export PushButtonTimer\_wReadTimerSaveCV

export \_PushButtonTimer\_wReadTimerSaveCV

; The following functions are deprecated and subject to omission in future releases

;

export wPushButtonTimer\_ReadCompareValue ; deprecated

export \_wPushButtonTimer\_ReadCompareValue ; deprecated

export wPushButtonTimer\_ReadTimer ; deprecated

export \_wPushButtonTimer\_ReadTimer ; deprecated

export wPushButtonTimer\_ReadTimerSaveCV ; deprecated

export \_wPushButtonTimer\_ReadTimerSaveCV ; deprecated

export wPushButtonTimer\_ReadCounter ; obsolete

export \_wPushButtonTimer\_ReadCounter ; obsolete

export wPushButtonTimer\_CaptureCounter ; obsolete

export \_wPushButtonTimer\_CaptureCounter ; obsolete

AREA project1\_RAM (RAM,REL)

;-----------------------------------------------

; Constant Definitions

;-----------------------------------------------

;-----------------------------------------------

; Variable Allocation

;-----------------------------------------------

AREA UserModules (ROM, REL)

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_EnableInt

;

; DESCRIPTION:

; Enables this timer's interrupt by setting the interrupt enable mask bit

; associated with this User Module. This function has no effect until and

; unless the global interrupts are enabled (for example by using the

; macro M8C\_EnableGInt).

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None.

; RETURNS: Nothing.

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_EnableInt:

\_PushButtonTimer\_EnableInt:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

PushButtonTimer\_EnableInt\_M

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_DisableInt

;

; DESCRIPTION:

; Disables this timer's interrupt by clearing the interrupt enable

; mask bit associated with this User Module.

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: Nothing

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_DisableInt:

\_PushButtonTimer\_DisableInt:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

PushButtonTimer\_DisableInt\_M

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_Start

;

; DESCRIPTION:

; Sets the start bit in the Control register of this user module. The

; timer will begin counting on the next input clock.

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: Nothing

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_Start:

\_PushButtonTimer\_Start:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

PushButtonTimer\_Start\_M

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_Stop

;

; DESCRIPTION:

; Disables timer operation by clearing the start bit in the Control

; register of the LSB block.

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: Nothing

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_Stop:

\_PushButtonTimer\_Stop:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

PushButtonTimer\_Stop\_M

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_WritePeriod

;

; DESCRIPTION:

; Write the 16-bit period value into the Period register (DR1). If the

; Timer user module is stopped, then this value will also be latched

; into the Count register (DR0).

;-----------------------------------------------------------------------------

;

; ARGUMENTS: fastcall16 WORD wPeriodValue (LSB in A, MSB in X)

; RETURNS: Nothing

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_WritePeriod:

\_PushButtonTimer\_WritePeriod:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

mov reg[PushButtonTimer\_PERIOD\_LSB\_REG], A

mov A, X

mov reg[PushButtonTimer\_PERIOD\_MSB\_REG], A

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_WriteCompareValue

;

; DESCRIPTION:

; Writes compare value into the Compare register (DR2).

;

; NOTE! The Timer user module must be STOPPED in order to write the

; Compare register. (Call PushButtonTimer\_Stop to disable).

;-----------------------------------------------------------------------------

;

; ARGUMENTS: fastcall16 WORD wCompareValue (LSB in A, MSB in X)

; RETURNS: Nothing

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_WriteCompareValue:

\_PushButtonTimer\_WriteCompareValue:

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

mov reg[PushButtonTimer\_COMPARE\_LSB\_REG], A

mov A, X

mov reg[PushButtonTimer\_COMPARE\_MSB\_REG], A

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_wReadCompareValue

;

; DESCRIPTION:

; Reads the Compare registers.

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: fastcall16 WORD wCompareValue (value of DR2 in the X & A registers)

; SIDE EFFECTS:

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

PushButtonTimer\_wReadCompareValue:

\_PushButtonTimer\_wReadCompareValue:

wPushButtonTimer\_ReadCompareValue: ; this name deprecated

\_wPushButtonTimer\_ReadCompareValue: ; this name deprecated

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

mov A, reg[PushButtonTimer\_COMPARE\_MSB\_REG]

mov X, A

mov A, reg[PushButtonTimer\_COMPARE\_LSB\_REG]

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_wReadTimerSaveCV

;

; DESCRIPTION:

; Returns the value in the Count register (DR0), preserving the

; value in the compare register (DR2).

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: fastcall16 WORD wCount (value of DR0 in the X & A registers)

; SIDE EFFECTS:

; 1) May cause an interrupt, if interrupt on Compare is enabled.

; 2) If enabled, Global interrupts are momentarily disabled.

; 3) The user module is stopped momentarily while the compare value is

; restored. This may cause the Count register to miss one or more

; counts depending on the input clock speed.

; 4) The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

; THEORY of OPERATION:

; 1) Read and save the Compare register.

; 2) Read the Count register, causing its data to be latched into

; the Compare register.

; 3) Read and save the Counter value, now in the Compare register,

; to the buffer.

; 4) Disable global interrupts

; 5) Halt the timer

; 6) Restore the Compare register values

; 7) Start the Timer again

; 8) Restore global interrupt state

;

PushButtonTimer\_wReadTimerSaveCV:

\_PushButtonTimer\_wReadTimerSaveCV:

wPushButtonTimer\_ReadTimerSaveCV: ; this name deprecated

\_wPushButtonTimer\_ReadTimerSaveCV: ; this name deprecated

wPushButtonTimer\_ReadCounter: ; this name deprecated

\_wPushButtonTimer\_ReadCounter: ; this name deprecated

CpuFlags: equ 0

wCount\_MSB: equ 1

wCount\_LSB: equ 2

RAM\_PROLOGUE RAM\_USE\_CLASS\_2

mov X, SP ; X <- stack frame pointer

add SP, 3 ; Reserve space for flags, count

mov A, reg[PushButtonTimer\_CONTROL\_LSB\_REG] ; save the Control register

push A

mov A, reg[PushButtonTimer\_COMPARE\_LSB\_REG] ; save the Compare register

push A

mov A, reg[PushButtonTimer\_COMPARE\_MSB\_REG]

push A

mov A, reg[PushButtonTimer\_COUNTER\_LSB\_REG] ; synchronous copy DR2 <- DR0

; This may cause an interrupt!

mov A, reg[PushButtonTimer\_COMPARE\_MSB\_REG] ; Now grab DR2 (DR0) and save

mov [X+wCount\_MSB], A

mov A, reg[PushButtonTimer\_COMPARE\_LSB\_REG]

mov [X+wCount\_LSB], A

mov A, 0 ; Guess the global interrupt state

tst reg[CPU\_F], FLAG\_GLOBAL\_IE ; Currently Disabled?

jz .SetupStatusFlag ; Yes, guess was correct

mov A, FLAG\_GLOBAL\_IE ; No, modify our guess

.SetupStatusFlag: ; and ...

mov [X+CpuFlags], A ; StackFrame[0] <- Flag Reg image

M8C\_DisableGInt ; Disable interrupts globally

PushButtonTimer\_Stop\_M ; Disable (stop) the timer

pop A ; Restore the Compare register

mov reg[PushButtonTimer\_COMPARE\_MSB\_REG], A

pop A

mov reg[PushButtonTimer\_COMPARE\_LSB\_REG], A

pop A ; restore start state of the timer

mov reg[PushButtonTimer\_CONTROL\_LSB\_REG], A

pop A ; Return result stored in stack frame

pop X

RAM\_EPILOGUE RAM\_USE\_CLASS\_2

reti ; Flag Reg <- StackFrame[0]

.ENDSECTION

.SECTION

;-----------------------------------------------------------------------------

; FUNCTION NAME: PushButtonTimer\_wReadTimer

;

; DESCRIPTION:

; Performs a software capture of the Count register. A synchronous

; read of the Count register is performed. The timer is NOT stopped.

;

; WARNING - this will cause loss of data in the Compare register.

;-----------------------------------------------------------------------------

;

; ARGUMENTS: None

; RETURNS: fastcall16 WORD wCount, (value of DR0 in the X & A registers)

; SIDE EFFECTS:

; May cause an interrupt.

;

; The A and X registers may be modified by this or future implementations

; of this function. The same is true for all RAM page pointer registers in

; the Large Memory Model. When necessary, it is the calling function's

; responsibility to perserve their values across calls to fastcall16

; functions.

;

; THEORY of OPERATION:

; 1) Read the Count register - this causes the count value to be

; latched into the Compare registers.

; 2) Read and return the Count register values from the Compare

; registers into the return buffer.

;

PushButtonTimer\_wReadTimer:

\_PushButtonTimer\_wReadTimer:

wPushButtonTimer\_ReadTimer: ; this name deprecated

\_wPushButtonTimer\_ReadTimer: ; this name deprecated

wPushButtonTimer\_CaptureCounter: ; this name deprecated

\_wPushButtonTimer\_CaptureCounter: ; this name deprecated

RAM\_PROLOGUE RAM\_USE\_CLASS\_1

mov A, reg[PushButtonTimer\_COUNTER\_LSB\_REG] ; synchronous copy DR2 <- DR0

; This may cause an interrupt!

mov A, reg[PushButtonTimer\_COMPARE\_MSB\_REG] ; Return DR2 (actually DR0)

mov X, A

mov A, reg[PushButtonTimer\_COMPARE\_LSB\_REG]

RAM\_EPILOGUE RAM\_USE\_CLASS\_1

ret

.ENDSECTION

; End of File PushButtonTimer.asm