/* ******************************************************************************
   |* Module Name:    SPI_DRIVER.h
   |* Module Author:  Jon Peterson
   |* Start Date:     1/12/2005
   |* End Date:
   |* Revised By:     Leo Wan
   |* Date:           3/28/2006
   |*
   |* uController:    HCS12NE64 or HCS12DP256
   |*
   |* Description:    This file (with its companion SPI_DRIVER.c) includes
   |*                  everything necessary for communicating with a device on the
   |*                  Serial Peripheral Interface.
   |*                  
   |* Notes:          MAKE SURE to check/set the General Configuration Options.
   |*                  The areas of interest are:  "Registers", and
   |*                  "SPI Baud Rate Register"
   |*
   |******************************************************************************

   |* Revision History:
   |*     1. 01/12/2005 - I started the basic layout for the driver module.
   |*
   |******************************************************************************

//Define the header file
#define SPI_DRIVER_H    1

/*******************************************************************************
   |*                      Global Function Exports                         *|
   |************************************************************************/
extern void InitSPI(void);          //This function initializes the SPI
extern void WriteSPI(INT8U data);   //This function writes to the SPI
extern INT8U ReadSPI(void);         //This function reads from the SPI
/*******************************************************************************
   |*                   General Configuration Options                      *|
   |************************************************************************/
//Registers & PORTS
#define SPI_PORT        PTS         //Set the SPI PORT
#define SPI_DDR         DDRS        //Set the data direction register
#define CTL_REG1        SPI0CR1     //Set the SPI Control Register #1
#define CTL_REG2        SPI0CR2     //Set the SPI Control Register #2
#define BAUD_RATE_REG   SPI0BR      //Set the SPI Baud Rate
#define SPI_STATUS      SPI0SR      //Set the SPI status register
#define SPI_DATA        SPI0DR      //Set the SPI data register

//SPI Control Register #1 (SPI0CR1)
#define SPI_INTERRUPTS  0           //1=Interrupts Enabled, 0=Disabled
#define SPI_ENABLE      1           //1=SPI enabled, 0=Disabled
#define TX_INTERRUPTS   0           //1=Transmit Int. enabled, 0=Disabled
#define MASTER_SLAVE    1           //1=SPI is master, 0=SPI is slave
#define CLOCK_POLARITY  0           //1=active high clk, 0=active low clk
#define CLOCK_PHASE     0           //1=sample even edge, 0=sample odd
#define SLAVE_ENABLE    1           //1=SPI selects slave, 0=doesn't
#define DATA_DIRECTION  0           //1=LSB transferred 1st, 0=MSB 1st

//SPI Control Register #2 (SPI0CR2)
#define MODF_ENABLE     1           //1=SS pin + MODF, 0=SS is not used
#define OUTPUT_BUFFER   0           //1=output buffer enabled, 0=Disabled
#define CLOCK_MODE      0           //1=stop in wait mode, 0=normal oper.
#define SERIAL_MODE     0           //1=bidirectional, 0=normal operation
//SPI Baud Rate Register (SPIBR), NE64: pg 281 DP256:
#define BAUD_RATE 0x21 //2.08333 MHz (bus clock = 25MHz)
#define BAUD_RATE 0x10 //6.25 MHz (bus clock = 25MHz)
#define BAUD_RATE 0x51 //1.04167 MHz (bus clock = 25Mhz)

/************************************************************************|  
|*                     General Defined Constants                        *|
|*                     General Defined Constants                        *|
#define ENABLED         1
#define DISABLED        0

//Used with SPI Control Register #1
#define MASTER_MODE     1       //SPI Mode
#define ACTIVE_LOW      1       //Active Low clock mode
#define EVEN_EDGES      1       //Clock phase set to even edges

//Used with SPI Control Register #2
#define STOP_CLOCK      1       //Stop clock in wait mode
#define STOP_CLOCK      1       //Stop clock in wait mode
#define BIDIRECTIONAL   1       //Serial mode = bidirectional

//SPI Status Register (SPIRS)
#define TX_BUFFER_FULL  0       //Indicates that the TX buffer is full
#define RX_BUFFER_EMPTY 0       //Indicates that the RX buffer is empty

/************************************************************************|  
|*                       General Defined Macros                         *|
|*                       General Defined Macros                         *|
#define TX_STATUS()     (SPI_STATUS & SPTEF)
#define RX_STATUS()     (SPI_STATUS & SPIF)
#define START_SPI()     (WriteSPI(0x00))
#define START_SPI()     (WriteSPI(0x00))
#define ENABLE_SLAVE()  (SPI_PORT |= SS)
#define DISABLE_SLAVE() (SPI_PORT &= ~SS)

/************************************************************************|  
|*                           Register Masks                             *|
|*                           Register Masks                             *|
#define SPIE            0x80       //SPI Interrupt Enable bit
#define SPE             0x40       //SPI System Enable bit
#define SPTIE           0x20       //SPI Transmit Interrupt enable
#define MSTR            0x10       //SPI Master/Slave Mode select bit
#define CPOL            0x08       //SPI Clock Polarity bit
#define CPHA            0x04       //SPI Clock Phase bit
#define SSOE            0x02       //Slave Select Output enable
#define LSBFE           0x01       //LSB-First Enable

//SPI Control Register #2 (SPICR2)
#define MODFEN          0x10       //Mode fault enable bit
#define BIDIROE         0x08       //Output enable for bidirectional mode
#define SPISWAI         0x02       //SPI Stop in Wait Mode bit
#define SPC0            0x01       //Serial Fin Control Bit 0

//SPI Status Register (SPISR)
#define SPIF            0x80       //SPIF Interrupt Flag
#define SPTEF           0x20       //SPI Transmit Empty Interrupt Flag
#define MODF            0x10       //Mode Fault Flag

/************************************************************************|  
|*                              Port Masks                              *|
|*                              Port Masks                              *|
135 //Masks for Port-S
#define SS 0x80  //Slave Select pin
#define SCK 0x40  //Slave Clock
#define MOSI 0x20  //Master Out Slave In
#define MISO 0x10  //Master In Slave Out