/*
 * File:   Stepper.h
 * Author: Elkaim
 *
 * Software module to drive a stepper motor through a normal H-bridge in full-step drive
 * mode. The module uses TIMER3 and is capable of generated 1/2 to 20,000 steps per second.
 * The nominal port used is PORTZ and can be changed by changing the appropriate #defines
 * below.
 *
 * NOTE: Module uses TIMER3 for its interrupts. When using this module to directly drive
 *       the DRV8811 stepper board, it will be stepping at 1/2 the rate specified due to
 *       toggling the pin on every entry to the ISR (board steps on rising edge).
 *
 * STEPPER_TEST (in the .c file) conditionally compiles the test harness for the code.
 * Make sure it is commented out for module useage.
 *
 * Created on January 2, 2012, 9:36 PM
 */

#ifndef Stepper_H
#define Stepper_H

/*******************************************************************************
 * PUBLIC #DEFINES
 ******************************************************************************/

#ifndef SUCCESS
#define SUCCESS 0
#endif
#ifndef ERROR
#define ERROR -1
#endif

#ifndef TRUE
#define TRUE 1
#endif
#ifndef FALSE
#define FALSE 0
#endif

#define FORWARD 1
#define REVERSE 0

#define TRIS_COIL_A_ENABLE TRISFbits.TRISF1
#define TRIS_COIL_A_DIRECTION TRISEbits.TRISE4
#define TRIS_COIL_B_ENABLE TRISDbits.TRISD8
#define TRIS_COIL_B_DIRECTION TRISEbits.TRISE2
#define COIL_A_ENABLE LATFbits.LATF1
#define COIL_A_DIRECTION LATEbits.LATE4
#define COIL_B_ENABLE LATDbits.LATD8
#define COIL_B_DIRECTION LATEbits.LATE2

// Use these when using the DRV8811 Stepper Motor Driver
// you will need to set them to outputs and manipulate the values
#define STEPPER_BOARD_ENABLE
#define STEPPER_BOARD_DIRECTION
#define STEPPER_BOARD_STEP

/*******************************************************************************
 * PUBLIC FUNCTION PROTOTYPES
 ******************************************************************************/

/**
 *
 * Function: Stepper_Init(unsigned short int rate);
 *
 * @param rate - steps per second (0 is special case of 0.5Hz)
 *
 * @return SUCCESS or ERROR
 *
 * @brief Initializes the Stepper Module, sets up the stepper on PORTZ
 *
 * @note Rate 0 is a special case corresponding to 0.5Hz
 *
 * @author Gabriel Hugh Elkaim, 2012.01.28 23:21
 */
char Stepper_Init(unsigned short int rate);

/**
 *
 * Function: Stepper_ChangeStepRate(unsigned short int rate);
 *
 * @param rate - steps per second (0 is special case of 0.5Hz)
 *
 * @return SUCCESS or ERROR
 *
 * @brief Changes the step rate of the stepper motor.
 *
 * @author Gabriel Hugh Elkaim, 2012.01.28 23:21
 */
char Stepper_ChangeStepRate(unsigned short int rate);

/**
 *
 * Function: Stepper_SetSteps(char direction, unsigned int steps);
 *
 * @param direction - stepper direction (FORWARD or REVERSE)
 *
 * @param steps - number of steps to take
 *
 * @return SUCCESS or ERROR
 *
 * @brief Sets the number of steps and direction for the stepper drive. Wipes
 *        out any previous stepper SetSteps command.
 *
 * @author Gabriel Hugh Elkaim, 2012.01.28 23:21
 */
char Stepper_SetSteps(char direction, unsigned int steps);

/**
 *
 * Function: Stepper_IncrementSteps(char direction, unsigned int steps);
 *
 * @param direction - stepper direction (FORWARD or REVERSE)
 *
 * @param steps - number of steps to take
 *
 * @return SUCCESS or ERROR
 *
 * @brief Increments the number of steps and direction for the stepper drive. Adds
 *        to the remaining steps.
 *
 * @author Gabriel Hugh Elkaim, 2012.01.28 23:21
 */
char Stepper_IncrementSteps(char direction, unsigned int steps);

/**
 *
 * Function: Stepper_GetDirection(void);
 *
 * @return FORWARD or REVERSE
 *
 * @brief Returns the current direction
 *
 * @author Gabriel Hugh Elkaim, 2012.01.28 23:21
 */
char Stepper_GetDirection(void);

*/
@Function: Stepper_GetRemainingCount(void);

@return
number of remaining steps

@brief
Returns the number of remaining steps

@author
Gabriel Hugh Elkaim, 2012.01.28 23:21 */

unsigned int Stepper_GetRemainingCount(
);