/\*

\* kernel.h

\*

\* Created: 4/17/2020 3:09:30 PM

\* Author: Nathan Potvin

\*/

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/atomic.h>

#include <stddef.h>

#include <stdint.h>

#include <stdbool.h>

#include "kernel\_config.h"

#ifndef KERNEL\_H\_

#define KERNEL\_H\_

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Kernel build parameters

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Define thread and stack constants

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define MAX\_THREADS 8

#define THREAD0 0

#define THREAD1 1

#define THREAD2 2

#define THREAD3 3

#define THREAD4 4

#define THREAD5 5

#define THREAD6 6

#define THREAD7 7

#define THREAD0\_MSK 0b00000001

#define THREAD1\_MSK 0b00000010

#define THREAD2\_MSK 0b00000100

#define THREAD3\_MSK 0b00001000

#define THREAD4\_MSK 0b00010000

#define THREAD5\_MSK 0b00100000

#define THREAD6\_MSK 0b01000000

#define THREAD7\_MSK 0b10000000

#define CANARY 0xaa

#define STACK\_POINTER ((volatile *uint8\_t* \*\*)(0x5d))

#define GCC\_STACK\_BASE (*uint8\_t* \*) RAMEND

#ifndef PREEMPTIVE

#define THREAD\_STACK\_CONTEXT\_SZ 18

#else

#define THREAD\_STACK\_CONTEXT\_SZ 33

#endif /\* PREEMPTIVE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Define system timer parameters

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define USEC\_PER\_MILLIS 1000

#ifdef PREEMPTIVE

# if TIME\_SLICE < (1<<8)\*64

# define PRESCALLER\_SELECT 0b100

# define TIMER2\_PRESCALLER 64

# define USEC\_PRE\_TIME\_SLIZE (TIME\_SLICE / TIMER2\_PRESCALLER)

# else

# define PRESCALLER\_SELECT 0b101

# define TIMER2\_PRESCALLER 128

# define USEC\_PRE\_TIME\_SLIZE (TIME\_SLICE / TIMER2\_PRESCALLER)

# endif

# define STR2(x) #x

# define STR(x) STR2(x)

# pragma message "Using timer 2 prescaler: " STR(TIMER2\_PRESCALLER)

#else /\* PREEMPTIVE \*/

# define PRESCALLER\_SELECT 0b101

# define TIMER2\_PRESCALLER 128

#endif /\* PREEMPTIVE \*/

#ifndef \_\_ASSEMBLER\_\_

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Kernel data structures

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

typedef void (\*PTHREAD)();

typedef struct

{

*uint8\_t* stack0[T0\_STACKSZ];

*uint8\_t* stack1[T1\_STACKSZ];

*uint8\_t* stack2[T2\_STACKSZ];

*uint8\_t* stack3[T3\_STACKSZ];

*uint8\_t* stack4[T4\_STACKSZ];

*uint8\_t* stack5[T5\_STACKSZ];

*uint8\_t* stack6[T6\_STACKSZ];

*uint8\_t* stack7[T7\_STACKSZ];

} stack\_struct;

typedef struct

{

volatile *uint8\_t* \*stack\_ptr;

*uint8\_t* \*stack\_base;

*uint8\_t* \*canary\_ptr;

PTHREAD entry\_pnt;

} thread\_ctrl\_struct;

typedef struct

{

*uint8\_t* disable\_status;

*uint8\_t* delay\_status;

*uint16\_t* delay\_ctrs[MAX\_THREADS];

*uint8\_t* cur\_thread\_id;

*uint8\_t* cur\_thread\_msk;

} schedule\_ctrl\_struct;

typedef struct

{

stack\_struct stacks;

thread\_ctrl\_struct thread\_ctrl\_tbl[MAX\_THREADS];

schedule\_ctrl\_struct schedule\_ctrl;

volatile *uint32\_t* system\_time;

} kernel\_data\_struct;

kernel\_data\_struct kernel\_data;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Kernel function prototypes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void init();

void new(*uint8\_t*, PTHREAD, bool);

void delay(*uint16\_t*);

void disable(*uint8\_t*);

void enable(*uint8\_t*);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Cooperative kernel function prototypes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef PREEMPTIVE

void yield();

#endif /\* PREEMPTIVE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Preemptive kernel function prototypes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifdef PREEMPTIVE

void lock();

void unlock();

#endif /\* PREEMPTIVE \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Error function prototypes

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void stack\_overflow();

void uninitialized\_thread\_error();

#endif /\* \_\_ASSEMBLER\_\_ \*/

#endif /\* KERNEL\_H\_ \*/