/\*

\* main2.c

\*

\* Created: 4/4/2020 6:34:41 PM

\* Author: Nathan Potvin

\*/

#include <avr/io.h>

#include <stdbool.h>

#include <avr/interrupt.h>

#include "PSerial.h"

#include "debug.h"

#define TIM5 0x2000

#define TIM4 0x1000

#define TIM3 0x0800

#define TIM2 0x0040

#define TIM1 0x0008

#define TIM0 0x0020

#define USART3 0x0400

#define USART2 0x0200

#define USART1 0x0100

#define USART0 0x0002

#define ADC\_ 0x0001

#define SPI 0x0004

#define TWI 0x0080

volatile bool sleeping;

/\*

\* end the sleep on the button press

\*/

ISR(INT0\_vect)

{

sleeping = false;

}

/\*

\* fucntions to execute each sleep mode

\*/

void no\_sleep()

{

while(sleeping);

}

void idle()

{

SMCR |= 0b000<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

void adcnrm()

{

SMCR |= 0b001<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

void power\_down()

{

SMCR |= 0b010<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

void power\_save()

{

SMCR |= 0b011<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

void standby()

{

SMCR |= 0b110<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

void ext\_standby()

{

SMCR |= 0b111<<SM0;

SMCR |= 0b1<<SE;

\_\_asm\_\_("sleep");

}

// array of the sleep modes

void (\*sleep\_modes[7])() = {no\_sleep, idle, adcnrm, power\_down

, power\_save, standby, ext\_standby};

*uint16\_t* modules[14] = {0x0000, TIM0, TIM1, TIM2, TIM3, TIM4, TIM5

, USART0, USART1, USART2, USART3, ADC\_, SPI, TWI};

// array of modules that can be disabled for power savings

// with the PRR register

void (\*sleep\_mode)();

*uint16\_t* module\_disable\_vect;

/\*

\* disable modules by setting PRR according to the module\_disable\_vect

\*/

void disable\_modules()

{

PRR0 = (*uint8\_t*) (module\_disable\_vect & 0x00ff);

PRR1 = (*uint8\_t*) ((module\_disable\_vect >> 8) & 0x00ff);

}

/\*

\* enable all modules in the PRR register

\*/

void enable\_modules()

{

PRR0 = 0x00;

PRR1 = 0x00;

}

/\*

\* reset the state

\*/

void reset()

{

PSerial\_open(0, 9600, SERIAL\_8E2);

sleep\_mode = no\_sleep;

module\_disable\_vect = 0x0000;

enable\_modules();

DDRD &= ~(0x01);

PORTD |= 0x01;

*uint8\_t* isc0 = 0b10;

EICRA = (isc0 << ISC00);

EIMSK = (0b1 << INT0) | (0b1 << INT0);

*sei*();

}

/\*

\* print the current sleep configuration to the terminal

\*/

void print\_state()

{

print\_s("Modules:");

print\_s("\n\r TIM0 : ");

module\_disable\_vect & TIM0 ? print\_c('d') : print\_c('e');

print\_s("\n\r TIM1 : ");

module\_disable\_vect & TIM1 ? print\_c('d') : print\_c('e');

print\_s("\n\r TIM2 : ");

module\_disable\_vect & TIM2 ? print\_c('d') : print\_c('e');

print\_s("\n\r TIM3 : ");

module\_disable\_vect & TIM3 ? print\_c('d') : print\_c('e');

print\_s("\n\r TIM4 : ");

module\_disable\_vect & TIM4 ? print\_c('d') : print\_c('e');

print\_s("\n\r TIM5 : ");

module\_disable\_vect & TIM5 ? print\_c('d') : print\_c('e');

print\_s("\n\r USART0 : ");

module\_disable\_vect & USART0 ? print\_c('d') : print\_c('e');

print\_s("\n\r USART1 : ");

module\_disable\_vect & USART1 ? print\_c('d') : print\_c('e');

print\_s("\n\r USART2 : ");

module\_disable\_vect & USART2 ? print\_c('d') : print\_c('e');

print\_s("\n\r USART3 : ");

module\_disable\_vect & USART3 ? print\_c('d') : print\_c('e');

print\_s("\n\r ADC\_ : ");

module\_disable\_vect & ADC\_ ? print\_c('d') : print\_c('e');

print\_s("\n\r SPI : ");

module\_disable\_vect & SPI ? print\_c('d') : print\_c('e');

print\_s("\n\r TWI : ");

module\_disable\_vect & TWI ? print\_c('d') : print\_c('e');

print\_s("\n\r");

print\_s("Sleep mode:\n\r");

if (sleep\_mode == no\_sleep)

{

print\_s(" no\_sleep\n\r");

}

else if (sleep\_mode == idle)

{

print\_s(" idle\n\r");

}

else if (sleep\_mode == adcnrm)

{

print\_s(" adcnrm\n\r");

}

else if (sleep\_mode == power\_down)

{

print\_s(" power\_down\n\r");

}

else if (sleep\_mode == power\_save)

{

print\_s(" power\_save\n\r");

}

else if ( sleep\_mode == standby)

{

print\_s(" standby\n\r");

}

else if (sleep\_mode == ext\_standby)

{

print\_s(" ext\_standby\n\r");

}

else

{

print\_s(" Not a valid sleep\_mode\n\r");

}

while (!(UCSR0A & (1<<TXC0)));

}

int main()

{

for (int i = 0; i < 7; ++i)

{

for (int j = 0; j < 14; ++j)

{

reset();

sleep\_mode = sleep\_modes[i];

module\_disable\_vect = modules[j];

print\_state();

disable\_modules();

sleeping = true;

sleep\_mode();

}

reset();

sleep\_mode = sleep\_modes[i];

module\_disable\_vect = modules[1] | modules[2] | modules[3]

| modules[4] | modules[5] | modules[6]

| modules[7] | modules[8] | modules[9]

| modules[10] | modules[11] | modules[12]

| modules[13];

print\_state();

disable\_modules();

sleeping = true;

sleep\_mode();

reset();

sleep\_mode = sleep\_modes[i];

module\_disable\_vect = modules[1] | modules[2] | modules[3] | modules[4]

| modules[5] | modules[6];

print\_state();

disable\_modules();

sleeping = true;

sleep\_mode();

reset();

sleep\_mode = sleep\_modes[i];

module\_disable\_vect = modules[7] | modules[8] | modules[9] | modules[10];

print\_state();

disable\_modules();

sleeping = true;

sleep\_mode();

}

}