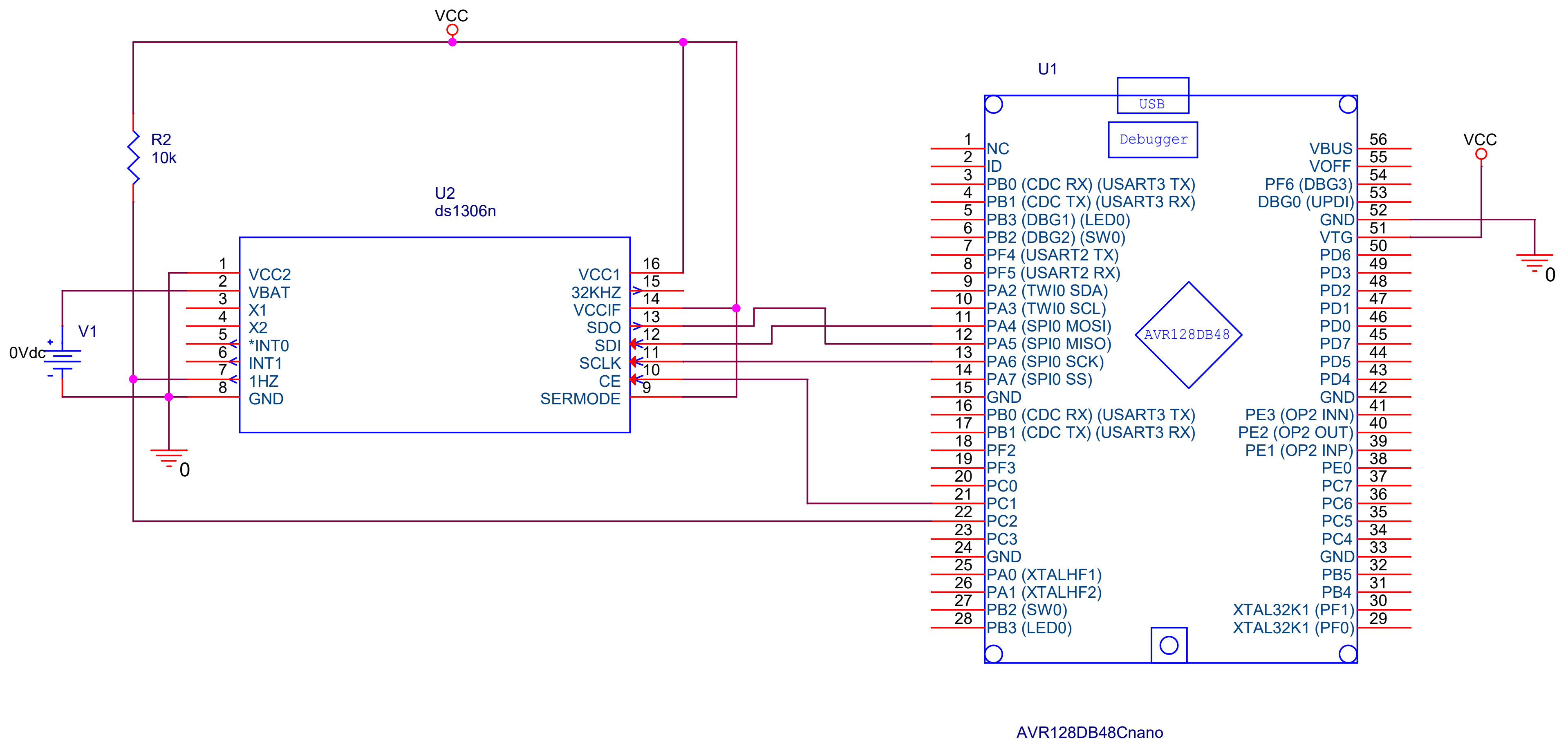
Judah Ben-Eliezer

112352727
4/29/2021

Prelab 12:

Data Logger: Asynchronous Serial Interrupt Table-Driven FSM User Interface



Title
DS1306 RTC SPI wiring

Size	Document Number	Rev
A	<Doc>	<RevCo

Date: Thursday, April 29, 2021 Sheet 1 of 1

```c
/*
 * fsm_ui.h
 *
 * Created: 4/29/2021 5:43:27 PM
 *  Author: Judah Ben-Eliezer
 */ 


#ifndef FSM_UI_H_
#define FSM_UI_H_

typedef enum {s, h, m, e, digit, enter, eol} key;
typedef enum {idle, set, hours, minutes, seconds} state;

typedef void (*task_fn_ptr) ();

typedef struct
{
    key keyval;
    state next_state;
    task_fn_ptr tf_ptr;
} transition;

void set_fn();
void hours_fn();
void minutes_fn();
void seconds_fn();
void digit_fn();
void enter_hours_fn();
void enter_minutes_fn();
void enter_seconds_fn();
void error_fn();


state fsm_ui(state ps, key keyval);



#endif /* FSM_UI_H_ */
```

```c
/*
 * fsm_ui.c
 *
 * Created: 4/29/2021 5:41:27 PM
 *  Author: Judah Ben-Eliezer
 */

#include <avr/io.h>
#include "fsm_ui.h"

void set_fn() {}

void hours_fn()
{
    /* print current hour */
}

void minutes_fn()
{
    /* print current minute */
}

void seconds_fn()
{
    /* print current second */
}

void digit_fn()
{
    /* read digits into buffer */
}

void enter_hours_fn()
{
    /* write new value for hours */
}

void enter_minutes_fn()
{
    /* write new value for minutes */
}

void enter_seconds_fn()
{
    /* write new value for seconds */
}

void error_fn()
{
    /* output error message */
}

```

```c
const transition idle_transitions[] =   //subtable for idle transitions.
{
    //input         next_state          task
    {s,             set,                set_fn},
    {h,             idle,               error_fn},
    {m,             idle,               error_fn},
    {e,             idle,               error_fn},
    {digit,         idle,               error_fn},
    {enter,         idle,               error_fn},
    {eol,           idle,               error_fn}
};

const transition set_transitions[] =    //subtable for set transitions.
{
    //input         next_state          task
    {s,             set,                set_fn},
    {h,             hours,              hours_fn},
    {m,             minutes,            minutes_fn},
    {e,             seconds,            seconds_fn},
    {digit,         idle,               error_fn},
    {enter,         idle,               error_fn},
    {eol,           idle,               error_fn}
};

const transition hour_transitions[] =   //subtable for hour transitions.
{
    //input         next_state          task
    {s,             set,                set_fn},
    {h,             idle,               error_fn},
    {m,             idle,               error_fn},
    {e,             idle,               error_fn},
    {digit,         digit,              digit_fn},
    {enter,         idle,               enter_hours_fn},
    {eol,           idle,               error_fn}
};

const transition minute_transitions[] = //subtable for minute transitions.
{
    //input         next_state          task
    {s,             set,                set_fn},
    {h,             idle,               error_fn},
    {m,             idle,               error_fn},
    {e,             idle,               error_fn},
    {digit,         digit,              digit_fn},
    {enter,         idle,               enter_minutes_fn},
    {eol,           idle,               error_fn}
};

const transition second_transitions[] = //subtable for second transitions.
{
    //input         next_state          task
    {s,             set,                set_fn},
    {h,             idle,                error_fn},
    {m,             idle,                error_fn},
    {e,             idle,                error_fn},
    {digit,         digit,               digit_fn},
    {enter,         idle,                enter_seconds_fn},
    {eol,           idle,                error_fn}
};

const transition* transitions[5] = // table of all transitions.
{
    idle_transitions,
    set_transitions,
    hour_transitions,
    minute_transitions,
    second_transitions
};



state fsm_ui(state ps, key keyval)
{
    int i = 0;
    for (; (transitions[ps][i].keyval != keyval) && (transitions[ps][i].keyval != 
      eol); ++i);

    transitions[ps][i].tf_ptr();

    return transitions[ps][i].next_state;
}

```