

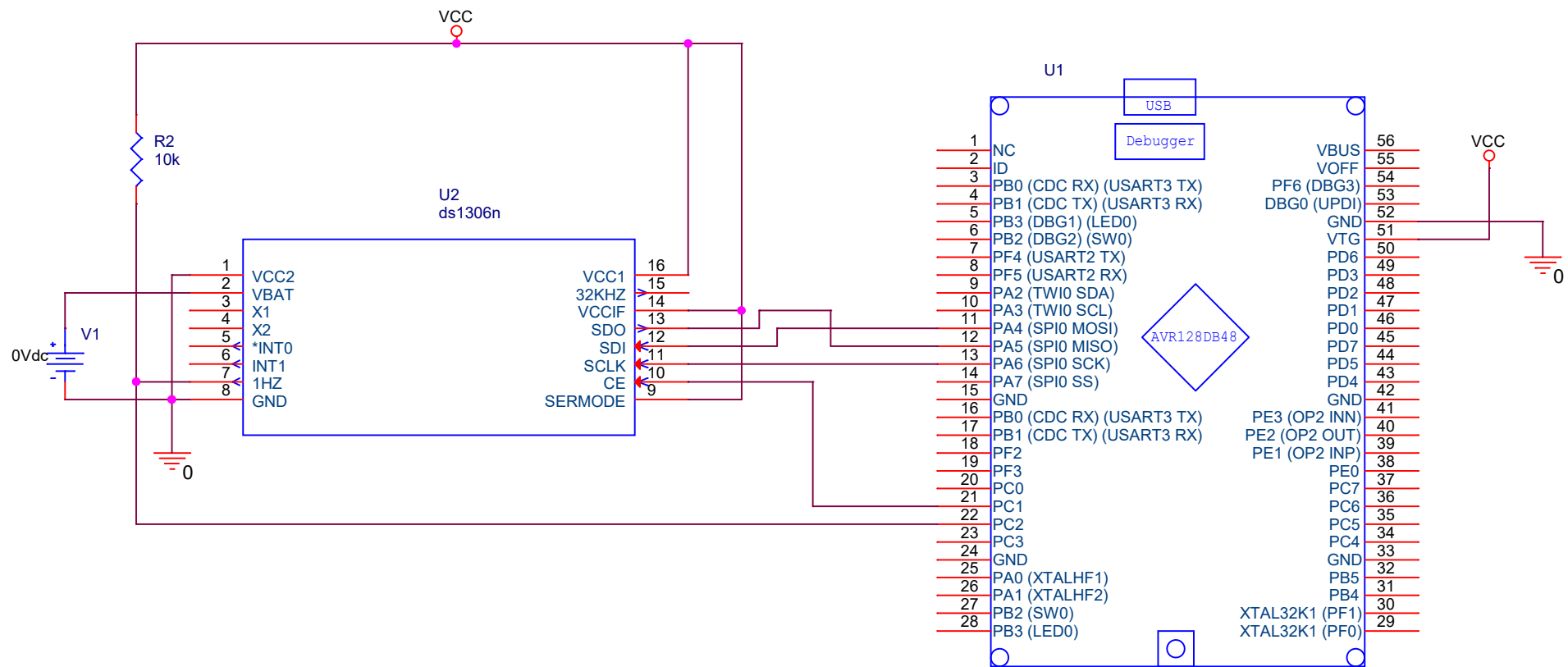
Judah Ben-Eliezer

112352727

4/29/2021

Prelab 12:

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



AVR128DB48Cnano

Title		
DS1306 RTC SPI wiring		
Size	Document Number	Rev
A	<Doc>	<RevCo
Date:	Thursday, April 29, 2021	Sheet 1 of 1

```
/*
 * 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, eof} 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_ */
```

```
1  /*
2   * fsm_ui.c
3   *
4   * Created: 4/29/2021 5:41:27 PM
5   * Author: Judah Ben-Eliezer
6   */
7
8  #include <avr/io.h>
9  #include "fsm_ui.h"
10
11 void set_fn() {}
12
13 void hours_fn()
14 {
15     /* print current hour */
16 }
17
18 void minutes_fn()
19 {
20     /* print current minute */
21 }
22
23 void seconds_fn()
24 {
25     /* print current second */
26 }
27
28 void digit_fn()
29 {
30     /* read digits into buffer */
31 }
32
33 void enter_hours_fn()
34 {
35     /* write new value for hours */
36 }
37
38 void enter_minutes_fn()
39 {
40     /* write new value for minutes */
41 }
42
43 void enter_seconds_fn()
44 {
45     /* write new value for seconds */
46 }
47
48 void error_fn()
49 {
50     /* output error message */
51 }
52
```

```
53 const transition idle_transitions[] = //subtable for idle transitions.
54 {
55     //input      next_state      task
56     {s,          set,            set_fn},
57     {h,          idle,           error_fn},
58     {m,          idle,           error_fn},
59     {e,          idle,           error_fn},
60     {digit,      idle,           error_fn},
61     {enter,      idle,           error_fn},
62     {eol,        idle,           error_fn}
63 };
64
65 const transition set_transitions[] = //subtable for set transitions.
66 {
67     //input      next_state      task
68     {s,          set,            set_fn},
69     {h,          hours,          hours_fn},
70     {m,          minutes,        minutes_fn},
71     {e,          seconds,        seconds_fn},
72     {digit,      idle,           error_fn},
73     {enter,      idle,           error_fn},
74     {eol,        idle,           error_fn}
75 };
76
77 const transition hour_transitions[] = //subtable for hour transitions.
78 {
79     //input      next_state      task
80     {s,          set,            set_fn},
81     {h,          idle,           error_fn},
82     {m,          idle,           error_fn},
83     {e,          idle,           error_fn},
84     {digit,      digit,          digit_fn},
85     {enter,      idle,           enter_hours_fn},
86     {eol,        idle,           error_fn}
87 };
88
89 const transition minute_transitions[] = //subtable for minute transitions.
90 {
91     //input      next_state      task
92     {s,          set,            set_fn},
93     {h,          idle,           error_fn},
94     {m,          idle,           error_fn},
95     {e,          idle,           error_fn},
96     {digit,      digit,          digit_fn},
97     {enter,      idle,           enter_minutes_fn},
98     {eol,        idle,           error_fn}
99 };
100
101 const transition second_transitions[] = //subtable for second transitions.
102 {
103     //input      next_state      task
104     {s,          set,            set_fn},
```

```
105     {h,          idle,          error_fn},
106     {m,          idle,          error_fn},
107     {e,          idle,          error_fn},
108     {digit,      digit,         digit_fn},
109     {enter,      idle,          enter_seconds_fn},
110     {eol,        idle,          error_fn}
111 };
112
113 const transition* transitions[5] = // table of all transitions.
114 {
115     idle_transitions,
116     set_transitions,
117     hour_transitions,
118     minute_transitions,
119     second_transitions
120 };
121
122
123
124 state fsm_ui(state ps, key keyval)
125 {
126     int i = 0;
127     for (; (transitions[ps][i].keyval != keyval) && (transitions[ps][i].keyval != ↵
128         eol); ++i);
129     transitions[ps][i].tf_ptr();
130
131     return transitions[ps][i].next_state;
132 }
133
```