Project 1: Alarm Clock

- Project 1 content
- 7-segment display
- Questions left for you

Project 1 Content

- Goals of this project:
  - Learn how to use Pmod external devices
  - Learn the method of displaying digits using PmodSSD
- Minimal Hardware:
  - PIC32 Kit board, PmodSSD*2
- Inputs:
  - Two on-board buttons: Btn1 and Btn2.
- Outputs:
  - Two external PmodSSDs displaying the clock or the alarm
  - Four on-board LEDs showing the modes of the device

Alarm Clock Functions

Five modes:

- Mode 1 - Initial
  - Clock = 00:00. Alarm = 12:00.
- Mode 2 – Set Alarm
  - Program each digit of the alarm one by one.
- Mode 3 – Set Time
  - Program each digit of the alarm one by one.
- Mode 4 – Dislay Time
  - Display the current time.
- Mode 5 - Alarming
  - Flash LEDs to indicate that time equals alarm time.

Outputs

Modes	7-seg display	LED4	LED3	LEDs 2 & 1
Initial	Clock Value	OFF	OFF	OFF
Set Alarm	Alarm Value	OFF	ON	The digit under programming
Set Time	Clock Value	ON	OFF	The digit under programming
Display Time	Clock Value	ON	ON	OFF
Alarming	Clock Value	Flashing		

Inputs and State Transitions

	Mode	Btn1	Btn2	Btn 1 & Btn 2
1	Initial	Mode 4	Mode 4	No effect
2	Set Alarm	Set next digit, enter Mode 4 if pressed 4 times	Current digit++	Mode 1
3	Set Time	Set next digit, enter Mode 4 if pressed 4 times	Current digit++	Mode 1
4	Display Time	Mode 2	Mode 3	Mode 1
5	Alarming	Mode 4	Mode 4	Mode 1

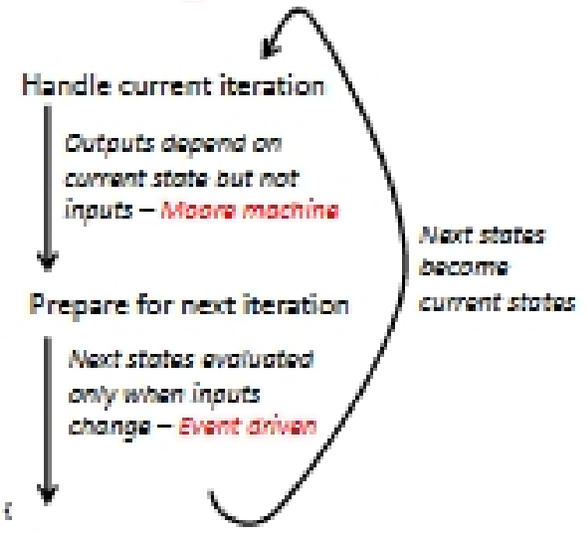
Some Notes

- When you are setting the alarm, time will not stop even if it is not displayed.
- When time = alarm time, on board LEDs will flash for 10 seconds unless you turn the alarm off. Flashing the two SSDs is not required, but you will get extra points for doing it.
- Because your clock will increment at a high speed (frequency for increasing minutes is 5 Hz), displaying seconds is not required and will be ignored.

Well-Structured Code

Decoupling output and next state logic!

```
int locked = 0;
while(1)  {
    // output logic
    switch (mode) {
    case A:
        …
    case B:
        …
    }
    // next state logic
    if (Btn1) {
        …
    }
    else if (Btn2) {
        …
    }
    else if (!Btn1 && !Btn2) {
        …
    }
}
```



Simpler to code, easier to debug!

Questions left for you

In Project 0, we learned the approach to holding state transitions until both buttons are released.

```
int locked = 0;
while(1)  {
    if (Btn1 && !locked) {
        …
        locked = 1;
    }
    else if (Btn2 && !locked) {
        …
        locked = 1;
    }
    else if (!Btn1 && !Btn2)
        locked = 0;
}
```

This ensures that no matter how long a button is pressed, the display mode changes at most once (only when locked == 0).

Pressing two buttons were ignored in project 0.

Questions left for you

In Project 1, the controller enters the initial mode when both buttons are pressed.
Yet in the real world, you almost always end up with one button pressed earlier than the other.

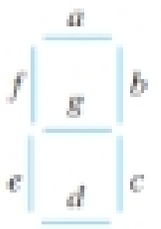
```
int locked = 0;
while(1)  {
    if (Btn1 && !locked) {
        …
        locked = 1;
    }
    else if (Btn2 && !locked) {
        …
        locked = 1;
    }
    else if (!Btn1 && !Btn2)
        locked = 0;
}
```

If Btn1 is pressed earlier, pressing Btn2 will not be recognized since Btn1 has not been released yet.

Q: How to let your controller enter the Initial Mode properly?
Is there any change needed for locking the state transitions?

7-segment Display

LED display



Example

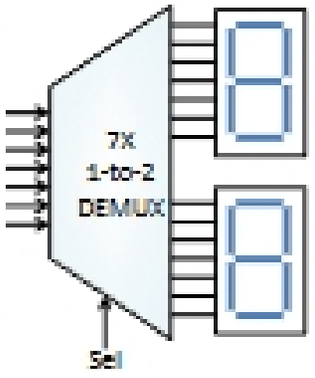
#	Display	Segments ON
1		b,c
3		a,b,c,d,g
9		a,b,c,d,f,g

Hex #	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1
A	1	1	1	0	1	1	1
b	0	0	0	1	1	1	1
C	1	0	0	1	1	1	0
D	0	1	1	1	0	0	1
E	1	0	0	1	1	1	1
F	1	0	0	0	1	1	1

Can be stored as a loop up table

Questions left for you

- Each PmodSSD is able to display two digits at the same time.
- For each digit, seven inputs are required to control all the seven segments.
- So it seems that fourteen inputs are needed for displaying two digits.
- Yet there are only 8 inputs available for each PmodSSD (despite GND and VCC)
  - *Depending on the value of "Sel", inputs are sent to different 7-seg LEDs receive inputs*



Q: Using these limited 8 inputs, how to display both digits at the same time?