Word Clock Functional Breakdown

States

Clock Mode- Default Mode, always performed while checking for entry to change mode

Read Time and Date

Display Time

Check Date and Display BDay Message if Bday match

Change Modes

Top Button 2s press to enter mode

In Mode Top Button Cycles Change Item

Mode Variables

5 Bytes, Year, Month, Date, Hour, Minute

At button press

Save current value for change item

Move to next change item

Top Button 2s Press to Save Data and exit change mode

Enable chip writing

Write out values

Disable Chip Writing

Year Change

Read Current Value

Display Current Value

Bottom Button Press increments Value

Start at 2015

Month Change

Read Current value

Display Current Value

Bottom Button Press Increments Value

At least 15ms depress

Start at January

Roll over after December

Day Change

Read Current Value

Display Current Value

Bottom Button Press Increments value

At least 15ms depress

Start at 01

Evaluate against current month for maximum date available

Roll back to 01 at end of max date available for current month

Time Change

Read Button Input

Display Change

Write Time

Date Change

Read Button Input

Display Change

Write Date

Hardcoded Elements

SPI

Clock

Input

Output

Chip Enable – RTC

Chip Enable – LEDs

RGB LED Output

B-day

2 Byte BCD, Month, Day

LEDs

2LED first, then 4 LED1, then 4LED2, then 3LED1, then 3LED2, then 3LED3

6 bytes total to write

4 LED addresses/order

2 Byte Binary Order, 10 values then 6 leading zeros

3 LED addresses/order

3 Byte Binary Order, 17 values then 7 leading zeros

2LED addresses/order

1 Byte Binary Order, 4 values then 4 leading zeros

RGB addresses/order

20 LEDs

Buttons

2 Buttons

Hold Top Button for 2 Sec to enter change mode

In Change Mode, press top button to cycle item being changed

Day/Month/Year/Hour/Minute

Bottom button will change item.

Display

Write 4 LED segments

Write 3 LED segments

Write 2 LED segments

Write RGB segments

Color rotation

writeTime(time)

Compare time value input and create necessary 6 byte value to output to LEDs

Output those 6 bytes to the LED drivers

Void createDisplayTime(byte minute, byte hour)

Compare minutes

Must find 0, 5, 10, 15, 20, 25, half and to/past

Create minute bitmask

Compare hours

Must fine 1-12

Create hour bitmask

Output 6 bytes to display

Call writeTime(6 bytes for LED chips, slaveselect)

Void writeTime(byte 2LED, 3LED0, 3LED1,3LED2,4LED0,4LED1, int slaveSelect)

Write slaveSelect = low;

SPI.transfer(2LED);

SPI.transfer(4LED0);

SPI.transfer(4LED1);

SPI.transfer(3LED0);

SPI.transfer(3LED1);

SPI.transfer(3LED2);

Write slaveSelect = high;

writeRGB()

rainbow theater chase – use a 50ms delay, 20 pixels

Need to rewrite this non-blocking in the main loop as necessary.

Maybe just do a quick compare of date in the main loop and change the color

//Theatre-style crawling lights with rainbow effect

void theaterChaseRainbow(uint8\_t wait) {

for (int j=0; j < 256; j++) { // cycle all 256 colors in the wheel

for (int q=0; q < 3; q++) {

for (int i=0; i < strip.numPixels(); i=i+3) {

strip.setPixelColor(i+q, Wheel( (i+j) % 255)); //turn every third pixel on

}

strip.show();

delay(wait);

for (int i=0; i < strip.numPixels(); i=i+3) {

strip.setPixelColor(i+q, 0); //turn every third pixel off

}

}

}

}

Read RTC

Read byte BCD hour

Read byte BCD minute

Read byte BCD date

Read byte BCD month

Return values

Time readClock (clock address, clock pin)

Read hours

Read minutes

Read Date

Read Month

Convert to necessary return format

Return time and date

byte readClock(byte address, int slaveSelect) {

byte incomingByte = 0; //Set up a byte to read in the incoming value

digitalWrite(slaveSelect, HIGH); //Select the slave we're reading from

SPI.transfer(address); //Write the address to be read out to the slave

incomingByte = SPI.transfer(0b00000000); //Read in the returned value from the slave

digitalWrite(slaveSelect, LOW); //Deselect the slave

return(incomingByte); //Return the value given by the slave

}

unsigned int BCDtoInt(byte input) {

unsigned int value = 0;

byte BCD = 0;

BCD = input;

BCD = BCD & 0b00001111; //Remove the Most Significant Nibble

value = BCD;

BCD = input;

BCD = BCD >> 4; //Shift the Nibble down

value = value + (BCD\*10); //Add the second nibble in as the tens value

return value;

}

Read Button Inputs for changes

Main loop:

Read RTC

Read Buttons for time/date change

Write Time At Set Interval

Use millis() with debounce timing for reading buttons

Use millis() timer for reading RTC

Compare date for RGB function and display if necessary

Boolean topButtonDown = false

Unsigned long topButtonTimer = 0

Unsigned long rtcReadTimer = 0

//Read and debounce top button to enter change time mode

If millis() – topButtonTimer >= 5UL “give button time to debounce”

And not topButtonDown and digitalRead(TopButton) == down

Then topButtonDown = true, topButtonTimer = millis()

If millis() – topButtonTimer >= 2000UL and topButtonDown

Then topButtonDown = false,

If digitaRead(TopButton) == down

Then call change functions

//Read the RTC and adjust display every minute of elapsed time

If millis() – rtcReadTimer >= 60000UL

Then read RTC, write Display, rtcReadTimer = millis()