Auto-Calibrated GPS/RTC Si5351A FST4W & WSPR MEPT

Gene Marcus W3PM GM4YRE

This Manned Experimental Propagation Transmitter (MEPT) uses the very popular Arduino Nano and Si5351A clock generator board to generate LF – 10 meter FST4W or WSPR signals. Frequency and time accuracy are maintained by a highly accurate temperature compensated DS3231 real time clock (RTC) board or by using a GPS receiver.

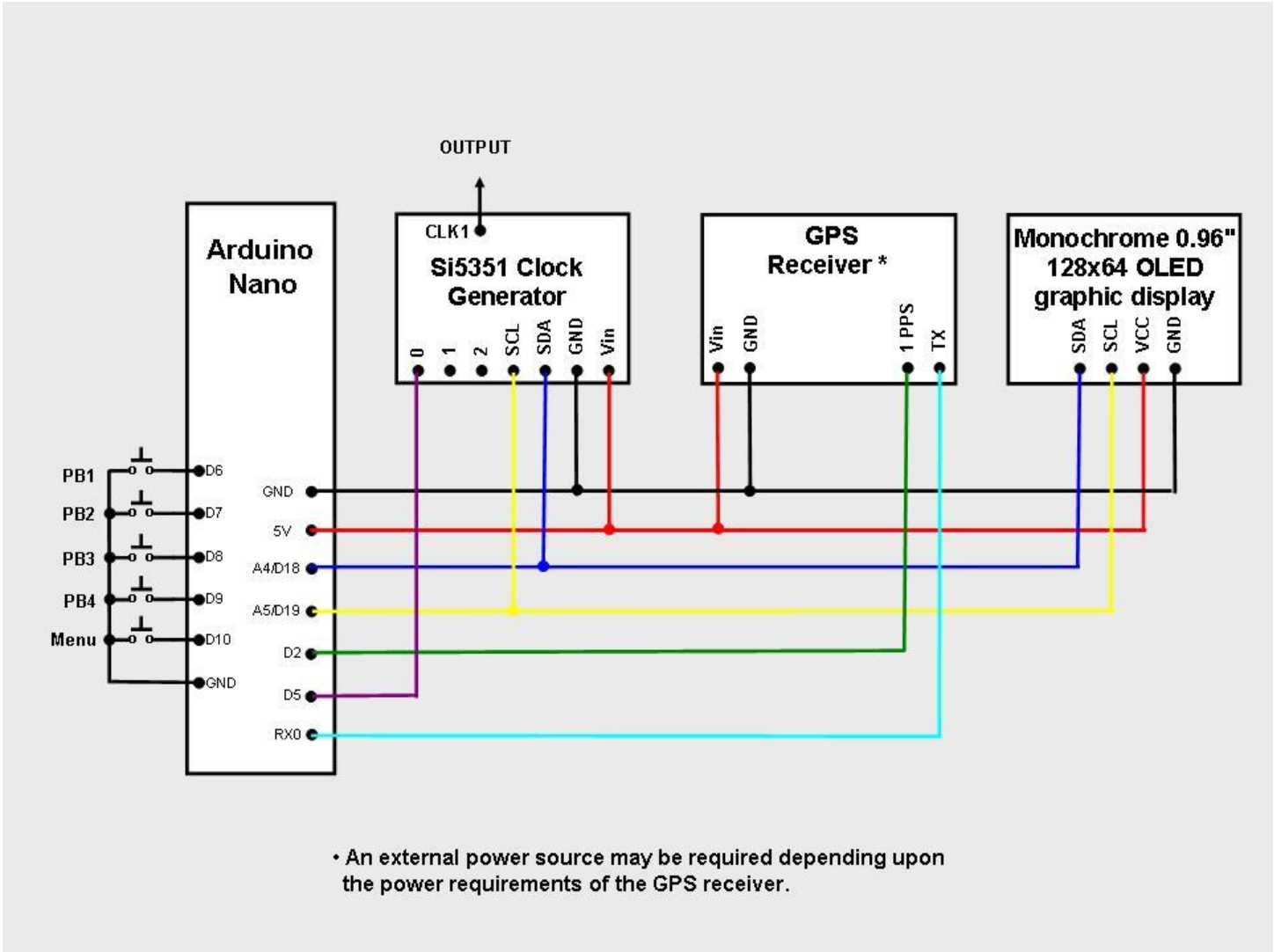

Figure 1 GPS Version

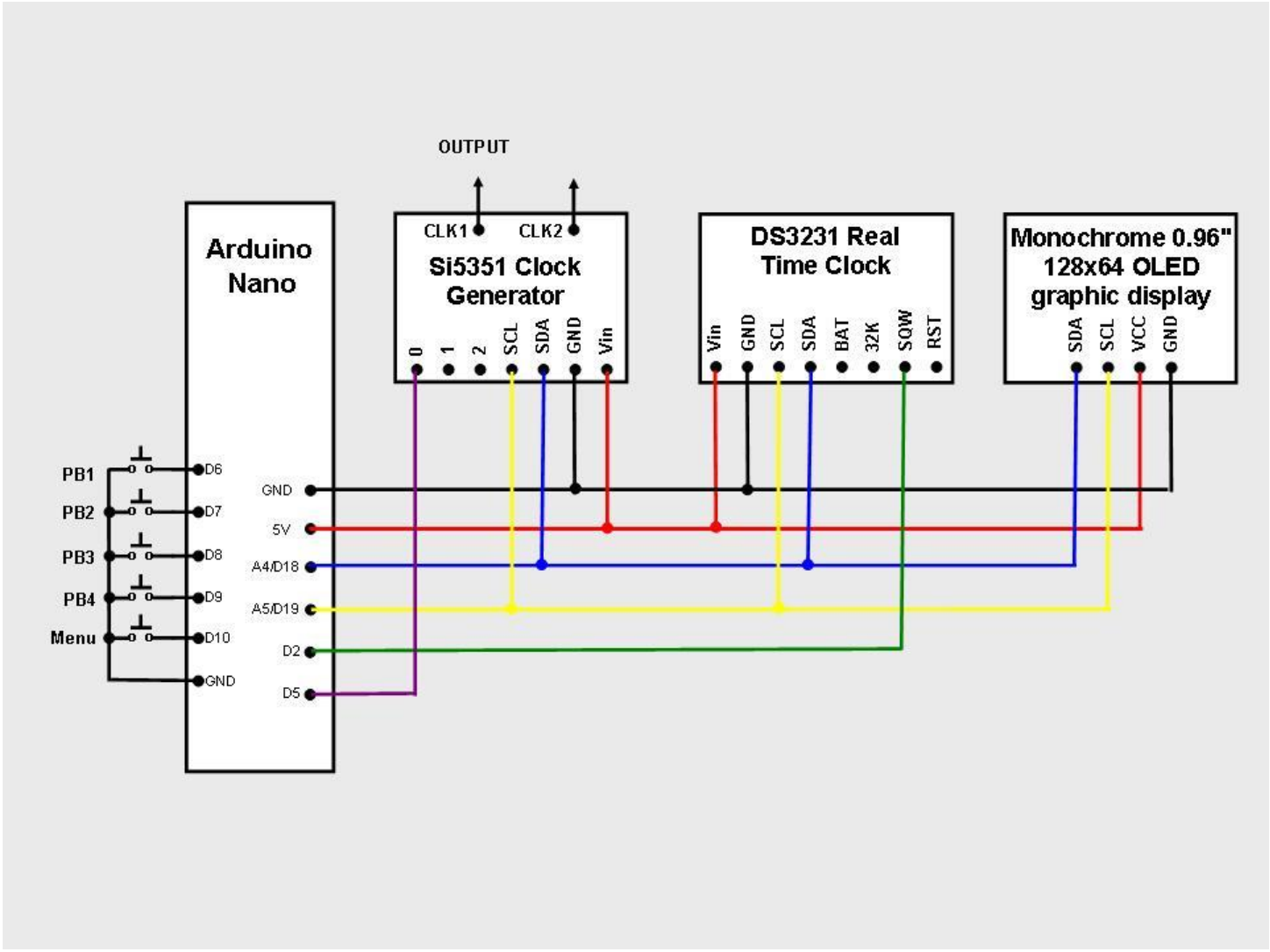

Figure 2 RTC Version

Introduction

As shown in Figures 1 and 2 the building blocks for this project consist of four boards: an Arduino Nano (*or Uno*) microcontroller, a Si5351A clock generator breakout board, a DS3231 real time clock (RTC) board or GPS receiver, and a 0.96 inch 128x64 serial I2C OLED display board.

The Si535a, DS3231, and OLED boards communicate with the Nano microcontroller over an I2C bus using only three wires. The boards were chosen because they are highly capable, inexpensive, well documented, and available over the internet from a wide variety of vendors. The completed unit draws little power and can be powered by the Nano microcontroller through the USB cable connected to a computer, a USB charger, or by some 5V USB battery backup chargers.

Unlike many other Si5251A based projects, this project does not require calibration. Both time and frequency accuracy are derived from a DS3231 real time clock (RTC) or GPS receiver. While the RTC version is less accurate, the builder may anticipate an uncertainty of about 2 parts per million from 0 – 40 degrees C (32 to 104 degrees F). Additional accuracy can be attained by adjusting the DS3231 aging offset.

The hardware for the RTC version is compatible with the Multifunction Project (QEX July/August 2020) found at http://www.knology.net/~gmarcus/. The GPS version requires one additional serial data connection.

Construction

NOTE: If you chose to build the RTC version be sure your project uses the DS3231SN. Some vendors substitute the DS3231M version which is not as accurate as the "N" version.

Construction of the unit is not critical provided adequate RF techniques are used. Do not use long unshielded wires for RF and 1pps connections. I first built the system using a solderless breadboard without any problems. The circuit was then transferred to a piece of perfboard and placed in a plastic project box. Mounting the unit in a metal cabinet is the preferred way to house this project. To improve short and long term accuracy the unit should be kept at a constant temperature away from any drafts.

The Si5351A, DS3231 RTC, and the display modules are powered from the 5 volt pin of the Arduino. You can use the USB programming cable, a separate 5VDC source connected to the "+5V" pin, or 7-12VDC source connected to the Arduino Nano's "VIN" pin to power the unit. Builders of the GPS version may require an external 5 V power supply for the GPS receiver.

Software Installation and Setup

NOTE: The sketch contains two variables which hold unique data symbols pertaining to your callsign, gridsqare, and power level. The variables are named "WSPRsymbols" and "FST4Wsymbols". Examples containing data for "W3PM EM64 37dBm" are shown below.

```
const byte  WSPRsymbols[162] = {
  3, 1, 0, 0, 2, 2, 0, 0, 1, 2, 0, 0, 3, 3, 1, 2, 2, 2, 3, 2, 0, 1, 0, 3, 3, 1, 1, 2, 0, 2, 0, 0,
  0, 2, 3, 0, 0, 3, 2, 3, 2, 0, 2, 0, 2, 2, 3, 2, 3, 3, 2, 2, 3, 3, 0, 1, 2, 2, 0, 3, 1, 2, 1, 2,
  2, 2, 2, 3, 1, 2, 1, 0, 1, 0, 1, 0, 1, 2, 2, 1, 0, 2, 3, 0, 1, 1, 2, 0, 0, 3, 3, 0, 3, 0, 1, 2,
  2, 2, 1, 0, 2, 0, 2, 0, 1, 0, 0, 3, 2, 2, 1, 1, 3, 2, 1, 3, 0, 2, 1, 1, 2, 3, 2, 0, 2, 3, 3, 3,
  2, 0, 0, 0, 0, 1, 2, 3, 0, 0, 1, 3, 0, 2, 0, 2, 2, 0, 2, 1, 3, 0, 3, 2, 1, 1, 2, 2, 0, 1, 3, 2,
  2, 2
};

const byte  FST4Wsymbols[160] = {
  0, 1, 3, 2, 1, 0, 2, 3, 0, 0,
  2, 0, 0, 0, 3, 3, 2, 3, 1, 0,
  0, 0, 1, 0, 0, 3, 0, 3, 0, 0,
  3, 2, 0, 2, 1, 1, 3, 2, 2, 3,
  1, 0, 3, 2, 0, 1, 1, 1, 2, 1,
  0, 1, 1, 3, 2, 0, 0, 3, 2, 3,
  2, 1, 3, 2, 1, 0, 0, 1, 3, 3,
  2, 1, 3, 0, 2, 3, 0, 1, 3, 2,
  1, 0, 2, 3, 2, 2, 3, 3, 0, 2,
  2, 3, 0, 3, 1, 2, 3, 2, 3, 1,
  3, 1, 3, 2, 1, 2, 0, 0, 0, 2,
  3, 2, 3, 0, 2, 3, 1, 0, 3, 2,
  0, 1, 1, 0, 1, 3, 0, 2, 1, 2,
  0, 1, 0, 0, 0, 0, 2, 1, 0, 0,
  0, 2, 0, 3, 3, 0, 1, 2, 2, 1,
  2, 2, 0, 1, 3, 2, 1, 0, 2, 3
};
```

The file to generate the data for "WSPRsymbols" is called WSPRMSG.exe and is found at http://www.knology.net/~gmarcus/>. Download this file into a convenient directory. Open this file and enter your callsign, grid locator, and power (dBm) as prompted. "WSPRsymbols" will create a file named "WSPRMSG.txt" in the same directory that "WSPRsymbols" resides. Cut and paste the symbol message data to replace the variable's data.

The file to generate the data for “FST4Wsymbols” is part of the WSJT suite of programs. Open Windows Command Prompt. Navigate to directory C:\WSJTX\bin. Enter the following command using your own callsign, grid square , and power level(dBM) enclosed in quotes as follows:

C:\WSJTX\bin> fst4sim "WZ9ZZZ FN21 30" 60 1500 0.0 0.1 1.0 10 -15 F > FST4W_message.txt

A file named “FST4W_message.txt” will be created in C:\WSJTX\bin. Open the FST4W_message.txt file into a text editor. Separate the "Channel symbols:" data with commas. Cut and paste the symbol message data to replace the variable’s data.

The modified sketch may now be downloaded into the Arduino Nano. The Arduino download website <http://arduino.cc/en/Main/Software> outlines installation instructions for the first-time Arduino user.

The sketch requires the open source library; “SSD1306Ascii” by Bill Greiman. The library is located in the Arduino IDE at; Sketch > Include Library > Manage Libraries. (Windows users will find the menu bar on top of the active window. Linux and MAC users will find the menu bar at the top of the primary display screen.) I found that other more robust ASCII/Graphics libraries were not compatible with the functions and timing complexity of this sketch.

In the unlikely event of operational problems, I suggest the builder check the Arduino, OLED display, and RTC boards individually to ensure proper operation. The Arduino board can be checked using some of the example sketches provided with the open source Arduino software. The simple “blink” example sketch will confirm that a sketch can be loaded and the Arduino board is functioning. The OLED display may be checked by running one of the AvrI2C examples found with the SSD1306Ascii library. The DS3231SN RTC board may be checked by running a time initialization sketch. A search of the internet will provide a number of methods to initialize the board. Adafruit Industries provides a very good tutorial to confirm the board is operational. Upon initialization, the time will be a few seconds slow. This is because the time set is the time the sketch was compiled, not the current time. The time may be updated using the “SET TIME” function after the project is completed. Provided a battery is used with the DS3231, the date will not require setting for a year or more.

Operation

When the unit is turned on, the auto-calibration algorithm begins to calculate the correction factor for the Si5351A’s 25 MHz clock. This takes 40 seconds to complete. If a RTC is used, the internal DS3231 temperature compensation algorithm concurrently calculates its correction factor every 64 seconds. The correction factors are continuously updated, therefore you may see small frequency jumps for the first few minutes until the unit stabilizes.

GPS sketch:
When the GPS receiver obtains valid data, the OLED screen will display all the operational parameters. The transmitter arming (ON/OFF) status is displayed in the upper right hand corner. Depress pushbutton 2 to toggle the transmitter arming status on and off.

Depress the **MENU** pushbutton to change bands, transmit interval, mode of operation, transmit offset frequency, or internal band hopping ON/OFF. Depress pushbutton 3 to scroll through the menu options. Depress pushbutton 2 to select the desired option.

Once selected, all functions are controlled by pushbuttons. An outline of pushbutton operation follows:

	EXIT	TX INTERVAL	SET OFFSET	TX FREQ HOP
PB1	N/A	Increase interval	Increase offset	“Yes”
PB2	Exit	N/A	N/A	N/A
PB3	N/A	Save and exit	Save and exit	Save and exit
PB4	N/A	Decrease interval	Decrease offset	“No”

	SELECT MODE	SELECT BAND
PB1	Scroll thru modes	Scroll thru bands
PB2	N/A	N/A
PB3	Save and exit	Save and exit
PB4	Scroll thru modes	Scroll thru bands

RTC sketch:
Depress pushbutton 2 to toggle the transmitter arming status on and off.

Depress the **MENU** pushbutton to change bands, transmit interval, mode of operation, transmit offset frequency, internal band hopping ON/OFF, set time, or set date. Depress pushbutton 3 to scroll through the menu options. Depress pushbutton 2 to select the desired option.

Once selected, all functions are controlled by pushbuttons. An outline of pushbutton operation follows:

	EXIT	SET TIME	SET DATE
PB1	N/A	Time sync / Set Hour*	Set Day*
PB2	ON / OFF	Set Minute*	Set Month*
PB3	Exit from menu	Save and exit	Set Year* / Save and exit
PB4	N/A	*Hold to change time	*Hold to change date

	TX INTERVAL	SELECT MODE	SELECT BAND
PB1	Increase TX interval	Depress to select mode	Increase band
PB2	N/A	N/A	N/A
PB3	Save and exit	Save and exit	Save and exit
PB4	Decrease TX interval	Depress to select mode	Decrease band

	SET OFFSET	TX FREQ HOP
PB1	Increase offset	"YES"
PB2	N/A	N/A
PB3	Save and exit	Save and exit
PB4	Decrease offset	"NO"

The time is set by selecting the “SET TIME” function. Hold down pushbutton 4 while depressing pushbutton 1 to set hours or pushbutton 2 to set minutes. Release pushbutton 4 at the top of the minute. Depress pushbutton 3 to save and exit.

* A shortcut to synchronize the time is available provided the displayed time is correct to within +/- 30 seconds. Simply depress pushbutton 1 at the top of the minute while in the “CLOCK SET” function to synchronize the time. Depress pushbutton 3 to save and exit.

Multi Mode Scheduling

If the “MULTI MODE” mode of operation is selected, the unit will begin transmitting each mode of operation using the following schedule:

Time	Mode
h:00	FST4W 1800
h:30	FST4W 900
h:45	FST4W 300
h:50	FST4W 120

h:52 WSPR
h:54 FST4W 120
h:56 WSPR
h:58 FST4W 120

The schedule may be changed within the sketch with the following start time limitations:

FST4W 1800: 0 or 30 minutes past top of hour
FST4W 900: 0, 15, 30, 45 minutes past top of hour
FST4W 300: 0, 5, 10……45, 50, 55 minutes past top of hour
WSPR : any even numbered minute

Ensure that a scheduled mode transmit time does not overlap the succeeding time interval.