KI7NNP

KI7NNP SDR Transmitter Hardware

Preliminary

Jed Marti

July 25, 2022

Contents

List of Figures

List of Tables

Chapter 1

The Minimal SDR Transmitter

We describe the low power SDR transmitter circuitry consisting of an 8051 micro-controller controlling a frequency synthesizer and power amplifier. The software is described in a companion document.

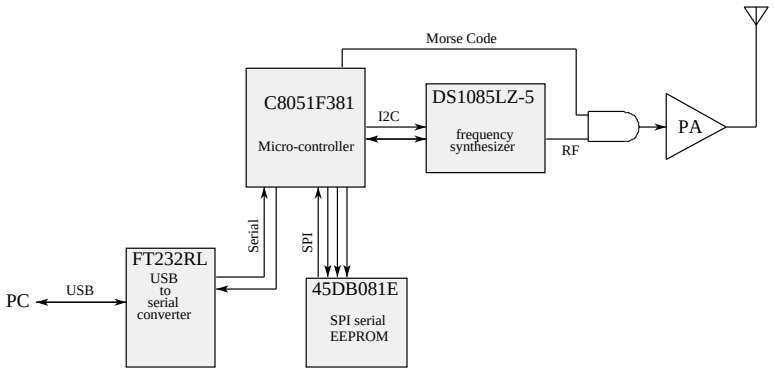

Figure 1.1: KI7NNP hardware components.

The populated 1.3" x 2.6" board is shown in Figure 1.2. Only a minimum number of readily available components[1] were required - except for the PC card and the frequency synthesizer they were all found laying about.

[1] That's today, who knows about next week

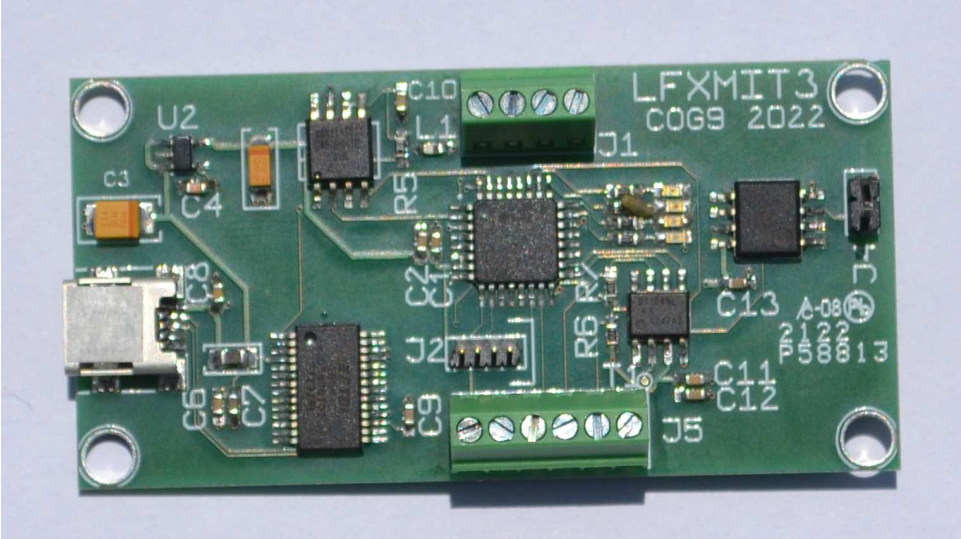

Figure 1.2: LFXMIT hardware.

.

With appropriate matching networks, the system works from 2200 to 40 meters with diminished frequency control at higher frequencies. The micro-controller and file system permit frequency modifications based on local temperature, scripts, and canned messages to be sent with both on-off keying and frequency shift.

The system was built to answer the question: “Can you build a transmitter from parts in your junk box?”. A better question would have been: “Can you build a transmitter with parts that are available?”, a more difficult problem than one would wish. The answer is a qualified, yes. My junk box includes many 8051 micro-controllers and associated peripheral, random transistors, many surface mount components and one lonesome vacuum tube left over from WWII. It also includes the software and hardware tool chain for the 8051.

It’s not an Arduino which has architectural limitations for the approach selected:

Area	ATmega	8051	Problem
Program	32k	65k	Arduino too small for all features needed, C8051F380 near the edge.
RAM	2k	4k	Atmega too small for this approach, especially file system.
Timers	3	7+	Insufficient for some applications. 4 used for LFXMIT.
Speed	16 MHz	48 MHz	ATmega kind of slow but OK, C8051 can be slowed down.
Peripherals			Both have I^2C, SPI, ADC, USART and GPIO.

Table 1.1: Arduino Atmega 328P vs C8051F381

But most importantly, I had several 32 pin C8051F381's left over from other projects, the Small Device C compiler setup, and the Silicon Labs flash programmer. As a professional programmer, the Arduino tool chain mostly gets in the way of efficient operation.

1.1 The Micro-controller

The Silicon Labs C8051F381 was selected for reasons previously mentioned. Connections to the chip are shown in Figure 1.3.

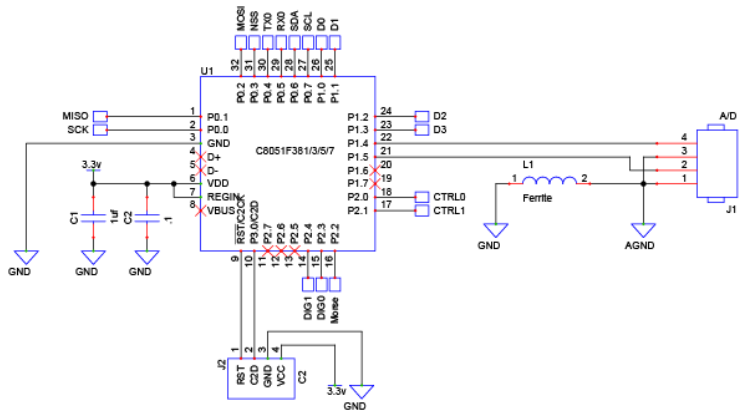

Figure 1.3: C8051F381 circuit components

Design considerations were as follows:

1. The Analog to digital converter (ADC) interface will be single ended rather than differential.

2. The ADC will use the internal 1.2v voltage reference multiplied by 2. Input will therefore be between 0 volts (0x000) and 2.4 volts (0x3FF).

3. We use the C2 interface for programming rather than JTAG. This uses only a .05" 4 pin header instead of the much larger 10 or 14 pin .1" header.

4. No reset button (may have been a mistake).

5. For layout, the analog section has its own ground plane connected to the master ground through L1.

Peripheral assignments are as follows:

Pin	Conn	Value	Description
P0_0	PP	1	SPI SCK clock output to data flash chip.
P0_1	OD	1	SPI MISO (Master In Slave Out) to data flash chip.
P0_2	PP	1	SPI MOSI (Master Out Slave In) to data flash chip.
P0_3	PP	1	Data flash chip select. Must be set/cleared manually.
P0_4	PP	1	Serial port data out.
P0_5	OD	1	Serial port data in.
P0_6	OD	1	I^2C SDA data to DS0185 and outside.
P0_7	OD	1	I^2C SCL clock to DS1085 and outside.

Table 1.2: C8051 Port 0 pin assignments.

Pin	Conn	Value	Description
P1_0	PP	0	LED0 (red). Indicates file system access.
P1_1	PP	0	LED1 (yellow). Indicates serial port, I^2C access.
P1_2	PP	0	LED2 (green). Indicates garbage collection.
P1_3	PP	0	LED3 (blue). Indicates Morse code.
P1_4	A	1	Analog input 0.
P1_5	A	1	Analog input 1.
P1_6	OD	1	No connection.
P1_7	OD	1	No connection.

Table 1.3: C8051 Port 1 pin assignments.

Pin	Conn	Value	Description
P2_0	PP	0	DS1085L control bit 0.
P2_1	PP	0	DS1085L control bit 1.
P2_2	PP	0	Morse code control bit.
P2_3	PP	0	DIG0 GPIO to outside world.
P2_4	PP	0	DIG1 GPIO to outside world.
P2_5	OD	1	No connection.
P2_6	OD	1	No connection.
P2_7	OD	1	No connection.

Table 1.4: C8051 Port 2 pin assignments.

The crossbar setup and GPIO control are all initialized in *init51.c* in the source code.

Chapter 2

USB Serial Port, Digital Power

The digital section takes its power from a USB connection which also includes a serial port conversion. We isolate the digital section so that a drain from the power amplifier doesn't affect the frequency synthesizer output.

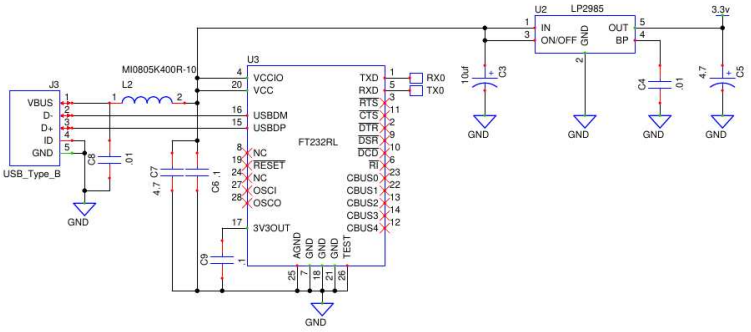

Figure 2.1: USB Serial port conversion

The FTDI FT232L USB to serial conversion chip relies on an automatic software load on the PC side. This seems to happen easily on both Windows and Linux unlike the direct interface available on the Silicon Labs C8051 chips.

Baud rates up to 3 MB/s are allowed, but 115.2k baud is typically the limit for most operating systems.

The linear regulator cleans up and adjusts the USB 5 volts to 3.3 to power the MCU, the serial data EEPROM and the frequency synthesizer. Their

total amperage requirement is well under the 150 mA limit.

Note that this setup allows a maximum of 6 volts input so a power only USB connection that delivers more will destroy the FT232L chip.

The two layer board has a separate ground plane for the digital domain which extends just beyond the frequency synthesizer.

Chapter 3

File System/Indicators

No computer is complete without some blinking lights so we have a few. The SPI flash storage with a rudimentary file system allows storage of large tables, program scripts, data file templates and the like.

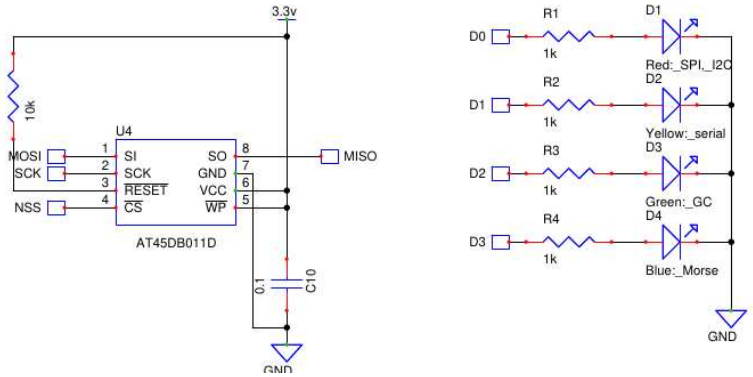

Figure 3.1: File system and blinken lights.

By default, the 4 LEDs indicate various activities. If they don't blink, either nothing is happening or the program has frozen.

File System:Red Indicates SPI activity connected to the ATD45DB081E. When formatting the chip, the LED may stay on for 30 seconds or so.

Serial Port:Yellow Indicates the system is reading or writing to the serial port. Whilst waiting for input, the light is on solid.

GC:Green Indicates a garbage collection of dynamic storage is occurring. If this is on solid, the GC has hung.

Morse:Blue Visual sending of dots and dashes or FSK.

The small file system works with up to 8 megabyte SPI EEPROMS, I had several Dialog (formerly Adesto) AT45DB081E versions with 1 megabyte. These are arranged in 264 byte sectors - we use the first 256 for data and 2 of the final 16 for the block index of the next block in a file. The circuit holds reset high, it's not needed and write protect low, also not needed.

We're running the SPI at a fairly low data rate (1 megabit/second) with the C8051 SPI hardware but doing the chip select manually. The software turns on LED D1 during all SPI access functions.

Chapter 4

Frequency Synthesizer, I^2C

The Anlog Devices DS1085L (formerly MAXIM, formerly Dallas Semiconductor) *3.3V EconOscillator Frequency Synthesizer* 8 pin SOIC. It exhibits very little frequency change vs temperature above 20 degrees centigrade (68 F) and 0.2% drop below that. Experiments are conducted to correct frequency for temperature.

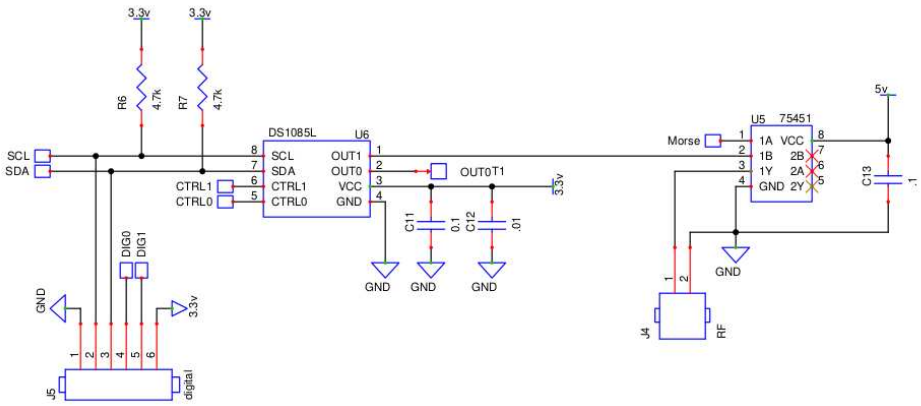

Figure 4.1: Frequency synthesizer, I^2C, GPIO

The DS1085 is controlled by the I^2C serial bus and is the only device on the bus. However, the SDA and SCL lines are brought to a connector so other devices can be connected. Two two pull up resistors R6 and R7 are sufficient for quite a few devices but will have to be lowered for more or if

the bus is to be run at 400k bps instead of 100k. At 400k, the maximum bus capacitance would be 80 *pf*.

The I^2C bus's speed is limited by the driving chip capability and system capacitance. The maximum pull up resistance R_{max} is:

$$R_{max} = \frac{t_r}{0.8473 C_b} \tag{4.1}$$

where:

t_r The maximum I^2C rise time in seconds. For 100 KHz this is 1 microsecond and for 400 KHz 300 nanoseconds.

C_b The bus capacitance in farads. Typical capacitance values are 10 pico-farads for must slave chips and a few pico farads for the bus line.

For 100 KHz and 20 *pf* bus capacitance, the maximum resistance would be 59k ohms, for 400k this drops to 18k. The minimum resistance is a function of current draw - 3 mA for 100 KHz and 400 KHz). The minimum resistance R_{min} would be:

$$R_{min} = \frac{V_{DD} - V_{OL}}{I_{OL}} \tag{4.2}$$

where:

V_{DD} The supply voltage (3.3 volts in our design).

V_{0L} The maximum low voltage for a 0. Typically 0.4 volts.

I_{0L} The low level current draw, typically 3 mA.

For our application this is slightly less than 1k ohms.

The synthesizer is connected to a two input, open drain peripheral driver, the 75451. One input is the RF square wave, the other Morse code output. The voltage to drive the class D power amplifier is provided by it's board. Alternatively, the low power output can directly drive a filter network.

This is simple but has some chirp during on and off as a partial RF cycle may appear.

Chapter 5

Temperature Sensor

We expose the circuit boards to different temperatures, compare the MCU measured temperature to a meter temperature and then fit the data to raw sensor.

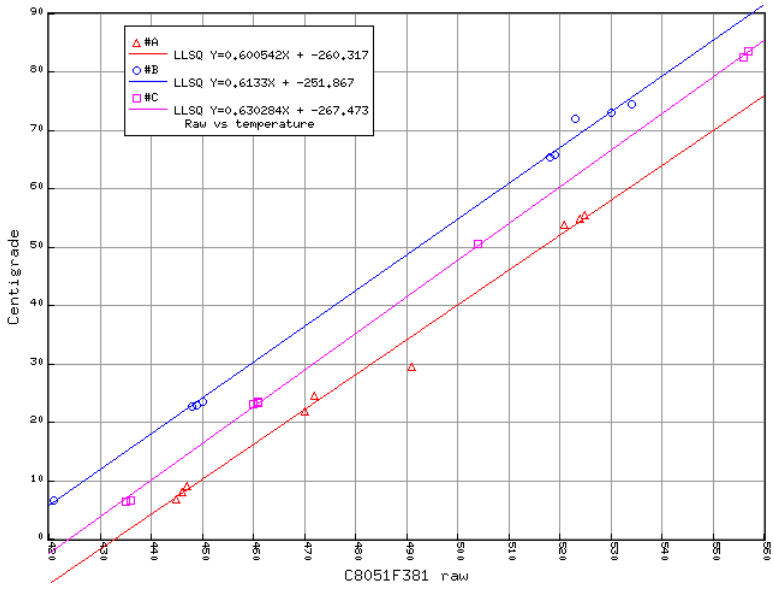

Figure 5.1: Raw vs Temperature C, 3 different MCUs

Each data point was 256 raw temperature samples averaged - the X axis. A thermocouple was taped to the MCU and 3 different temperature groups

were gathered. The thermocouple was attached to a Klein Tools MM 1000 set to read degrees centigrade. Three different boards were tested.

Ice Water Inside a sandwich bag the boards were placed in a bowel of salt, ice, and water, 3 separate readings were taken and recorded.

Room Temperature 3 separate readings taken on each board and recorded.

Oven Somewhere between 65 C and 85 C with multiple readings.

The ratio of raw reading to temperature remains nearly the same for the 3 boards tested, the value beining about 0.615. However, the base reading, the predicted temperature of a zero A2D reading occurs varies considerably. This indicates each board must be calibrated individually.

Index