Design Log
Avalanche Transceiver (transmit mode only)
Philip Nevins

1/12/2023 Research and Notes (Phil N)

Frequency Generation
Avalanche transmitters are equipped with electronic components that generate a stable radio frequency signal. This signal is set to the specified carrier frequency of 457 kHz as per the EN 30071801 standard.

Crystal Oscillator
Many electronic devices, including avalanche transceivers, use crystal oscillators to generate precise and stable frequencies. A crystal oscillator provides an accurate timing reference, ensuring that the transmitter's carrier frequency remains within the specified range.

Tuning and Calibration
During the manufacturing process, avalanche transceivers undergo tuning and calibration to ensure that the actual carrier frequency aligns with the standard's requirements. This process involves adjusting the electronic components to achieve the desired frequency accuracy.

Compliance Testing
Manufacturers subject their avalanche transceivers to compliance testing to verify that the devices meet the specified standards, including the carrier frequency of 457 kHz. This testing helps ensure that the transceivers will operate effectively and reliably in real-world avalanche rescue scenarios.

Digital Signal Processing (DSP)
Avalanche transceivers often employ digital signal processing (DSP) to enhance the performance of the radio frequency components. DSP can help filter out interference, optimize signal strength, and improve the overall reliability of communication between transceivers.

Regulatory Compliance (Just FCC for transmitters)
Avalanche transceiver manufacturers adhere to international regulatory standards, such as those outlined by the European Telecommunications Standards Institute (ETSI). Compliance with these standards ensures that the devices meet the necessary technical specifications, including carrier frequency, to facilitate interoperability among different brands and models.

Microcontrollers/Microprocessors
Avalanche transceivers incorporate microcontrollers or microprocessors to handle the overall control and processing of signals. These components manage tasks such as frequency generation, signal processing, user interface control, and power management.

Radio Frequency (RF) Components
To generate and transmit the radio frequency signal, avalanche transceivers use RF components, which may include RF transceivers, crystal oscillators, and filters. These components work together to ensure the transceiver operates on the designated frequency (e.g., 457 kHz) and communicates effectively with other devices.

Digital Signal Processing (DSP):
Many modern avalanche transceivers utilize digital signal processing to enhance the accuracy and reliability of signal reception and transmission. Dedicated DSP components or processors may be integrated to handle these tasks.

Memory
Avalanche transceivers have memory components to store firmware, settings, and other necessary data. This may include Flash memory for firmware storage and RAM for temporary data storage during operation.

Power Management ICs
Efficient power management is crucial in avalanche transceivers due to the need for extended battery life. Power management ICs help regulate power distribution, optimize energy usage, and manage battery levels.

User Interface Components
Avalanche transceivers typically feature user interfaces with displays, buttons, and indicators. The components for these interfaces can include display controllers, button interfaces, and LED drivers.

Based on this information, we potentially need:
Microcontroller
Low power consumption, real-time processing capabilities, and sufficient memory for firmware storage and data handling.

RF transceiver or crystal oscillator or filter

Flash memory for firmware and bootloader

Power management ICs to optimize energy usage by microcontroller and RF output module
No UI needed since we are only concerned about transmitter mode

Will have LED display to warn the user when battery life is below compliance levels,
as seen in EN 300 718-1 - V1.2.1
(https://nhqc3s.hq.nato.int/Apps/ETSI/specs/EN_30071801v010201p.pdf)

Dedicated DSP components might be needed, determined during breadboard testing in RF chamber

Below is a more in-depth dive into each component we need or possibly need

Microprocessor / Microcontroller

1. ARM Cortex-M Series

Microcontrollers based on ARM Cortex-M cores are widely used in various embedded systems, including those requiring low power and real-time capabilities. Examples include the Cortex-M0, Cortex-M3, Cortex-M4, and Cortex-M7.

2. STMicroelectronics STM32 Series

STM32 microcontrollers, based on ARM Cortex-M cores, are known for their versatility and are commonly used in applications with stringent power requirements. They offer a range of performance levels and features suitable for real-time processing.

3. Texas Instruments MSP430 Series

MSP430 microcontrollers from Texas Instruments are known for their ultra-low power consumption, making them suitable for battery-operated devices. They are often used in applications where power efficiency is critical.

4. NXP Semiconductors Kinetis Series

The Kinetis series of microcontrollers, based on ARM Cortex-M cores, are designed for applications that require low power and real-time processing. They offer a range of options with different performance levels.

5. Microchip PIC32 Series

Microchip's PIC32 microcontrollers are based on the MIPS architecture and are known for their real-time processing capabilities. They come with various peripherals and memory options.

6. Atmel AVR Series

AVR microcontrollers, often used in Arduino boards, are known for their simplicity and low power consumption. While they may be more commonly found in hobbyist and educational projects, some models can meet the requirements for certain applications.

https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus
-Standalone PWM - signal
-APM – power control
-

7. Renesas RX Series
The RX series of microcontrollers from Renesas is designed for real-time processing and features a variety of options with different power consumption profiles.

Validation and Verification research

EN 300 718-1 - V1.2.1 Notes

Date: 1/12/2024

Document: https://nhqc3s.hq.nato.int/Apps/ETSI/specs/EN_30071801v010201p.pdf

General Parameters

1. Transmitter/receiver with antenna and battery
2. On/off switch with visual indicator
3. Means of conveying information about the received signal to user
4. Commonly used battery with check feature.

**Capable of at least 200 hours of transmitting at a temperature of +10°C and subsequent receiving for 1 hour at a temperature of -10°C, in compliance with the requirements as stated in clauses 8.3 and 9.1.
**A positive check shall indicate the capability of at least 20 hours of transmitting at a temperature of +10°C

***9.1 not needed because we are not designing a reciver

5. A safety feature against involuntary or accidental leaving of the transmit mode shall be provided in the equipment.

6. The equipment shall include a carrying system that gives the possibility for easy operation and safe placing. The carrying system can be a part of the equipment or an accessory device. The carrying system shall have a joint tensile strength of at least 50 N

7. Operate at 457 kHz in transmit mode

8. Operating instructions shall be delivered with every equipment. They shall cover the following subjects:

a) a statement on avalanche danger;

b) instruction for checking the battery, transmitter and receiver performance and range;

c) instructions for turning on the transmitter and strapping the beacon to the body;

d) instructions for changing to the receive mode and the search strategy (coarse search and fine search);

e) instructions for changing back to the transmit mode, in particular in the case of secondary avalanche;

f) a statement on the temperature sensitivity of essential parts;

g) a statement on the battery lifetime;

h) device-specific measures on a tour.

9. A short form of the operating instructions shall be printed onto the case. The printing shall be clearly visible and abrasion proof. Also, the proper positioning of the batteries shall be indicated

10. The equipment shall be able to operate correctly in the temperature range from -20 to +45°C and shall be stored without damage in the temperature range from -25 to +70°C

8.3 Output field strength (H-field)

8.3.1 Definition

The H-field is measured in the direction of maximum field strength under specified conditions of measurement.

8.3.2 Method of measurement

The H-field produced by the equipment shall be measured on the axis of the transmitting antenna at distances of 10 m

on an outdoor test site (see annex A).

8.3.3 Limits

8.3.3.1 Minimum transmitted field

The minimum transmitted field strength at 457 kHz shall not be lower than -6 dBµA/m (0,5 µA/m) at a distance of

10 m.

8.3.3.2 Maximum transmitted field

The maximum transmitted field strength at 457 kHz shall not exceed 7 dBµA/m (2,23 µA/m) at a distance of 10 m

Transmitter Parameters

1. The carrier keying shall be measured by means of an oscilloscope connected to a suitable coil antenna. The measurements shall be done under normal as well as under extreme test conditions.

2. The carrier frequency shall be measured by means of a test fixture (see clause 6.2). The measurements shall be done under normal as well as under extreme test conditions

6.2 Test Fixture

A test fixture may be supplied by the applicant to enable extreme temperature measurements to be made, where applicable. The test fixture shall couple to the generated electromagnetic field from the equipment under test without disturbing the operation of the said device. The test fixture shall be provided with a 50 Ω standard connector, where the generated field can be sampled.

The test laboratory shall calibrate the test fixture by carrying out the required field measurements at normal temperatures at the prescribed test site and then by repeating the same measurements on the equipment under test using the test fixture for all identified frequency components.

The test fixture is only required for extreme temperature measurements and shall be calibrated only with the equipment under test.

8.4.2 Radiated H-field

8.4.2.1 Method of measurement (< 30 MHz)

The field strength shall be measured for frequencies below 30 MHz. The equipment under test shall be measured at a distance of 10 m on an outdoor test site. The test antenna shall be a calibrated shielded magnetic field antenna. The equipment under test and test antenna shall be arranged as stated in annex A, clause A.1. The equipment under test shall be switched on in transmit mode (see clause 8.1). The measuring receiver shall be tuned over the frequency range 9 kHz to 30 MHz, except for the frequency band ±20 kHz from the frequency on which the transmitter is intended to operate. At each frequency at which a spurious signal is detected the equipment under test and the test antenna shall be rotated until maximum field strength is indicated on the measuring receiver. This level shall be noted. The limits are quoted in dBμA or dBμA/m, so it is necessary to reduce the reading as explained in annex D for measuring equipment calibrated in dBμV or dBμV/m.

8.4.2.2 Limits

Radiated emissions below 30 MHz shall not exceed the generated H-field at 10 m given in table 2.

A graphical representation is shown in annex B, figure B.1.

8.4.3 Effective radiated power

8.4.3.1 Method of measurement (30 MHz)

On a test site, selected from annex A, the equipment shall be placed at the specified height on a non-conducting support and in the position closest to normal use as declared by the applicant. The test antenna shall be oriented for vertical polarization. The output of the test antenna shall be connected to a measuring receiver. The equipment shall be switched on in transmit mode, and the measuring receiver shall be tuned over the frequency range 30 MHz to 1 000 MHz. At each frequency at which a spurious component is detected, the test antenna shall be raised and lowered through the specified range of heights until a maximum signal level is detected on the measuring receiver.

The equipment shall then be rotated through 360° in the horizontal plane, until the maximum signal level is detected by the measuring receiver. The maximum signal level detected by the measuring receiver shall be noted. The substitution antenna shall be oriented for vertical polarization and calibrated for the frequency of the spurious component detected. The frequency of the calibrated signal generator shall be set to the frequency of the spurious component detected. The input attenuator setting of the measuring receiver shall be adjusted in order to increase the sensitivity of the measuring receiver, if necessary. The test antenna shall be raised and lowered through the specified range of heights to ensure that the maximum signal is received. When a test site according to clause A.3 is used, there is no need to vary the height of the antenna. The input signal to the substitution antenna shall be adjusted until an equal or a known related level to that detected from the transmitter is obtained on the measuring receiver. The input signal to the substitution antenna shall be recorded as a power level and corrected for any change of input attenuator setting of the measuring receiver. The measurement shall be repeated with the test antenna and the substitution antenna oriented for horizontal polarization. The measure of the effective radiated power of the spurious components is the larger of the two power levels recorded for each spurious component at the input to the substitution antenna, corrected for the gain of the substitution antenna if necessary.

8.4.3.2 Limits

The power of any radiated emission shall not exceed the values given in table 3.

10 Measurement uncertainty

The accumulated measurement uncertainties of the test system in use for the parameters to be measured should not exceed those given in table 6. This is in order to ensure that the measurements remain within an acceptable uncertainty.

2/20/2024

PCB Manufactuers: Oshpark is ideal, because we have worked with them before.
IP65 enclosure: https://www.polycase.com/wp-21 & emailed NEMA enclosures directly to see if they would work with us, doing a small order of custom enclosures. If they don't do that, polycase is perfect. We are looking at ~14.28$ per enclosure and <20$ per PCB.

**Clarify with Josh about what material would be best for the enclosure.

2/25/2024
Programming the SI5351A/B/C

We can use CBPro, write code in C like we did in ECE372 Project 2. We write to the registers the values we need to turn things on/off basically. Use an USB-to-I2C downloader to connect to the SDA, SCL pins. They have Si5351A breakout boards with the 10-MSOP. Ordered to experiment. We can also program it via a raspberry pi, using the wiringPiI2C.h library.   int si5351a_fd = wiringPiI2CSetup(SI5351A_ADDRESS);

5. Configuring the Si5351 The Si5351 is a highly flexible clock generator which is entirely configurable through its I2C interface. The device's default configuration is stored in non-volatile memory (NVM) as shown in Figure 11. The NVM is a one time programmable memory (OTP) which can store a custom user configuration at power-up. This is a useful feature for applications that need a clock present at power-up (e.g., for providing a clock to a processor). Figure 11. Si5351 Memory Configuration

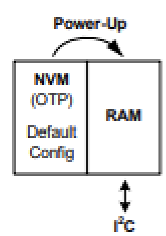


Figure 11. Si5351 Memory Configuration

During a power cycle the contents of the NVM are copied into random access memory (RAM), which sets the device configuration that will be used during normal operation. Any changes to the device configuration after power-up are made by reading and writing to registers in the RAM space through the I2C interface. A detailed register map is shown in Section "8. Register Descriptions" on page 25.

5.1. Writing a Custom Configuration to RAM To simplify device configuration, Silicon Labs has released the ClockBuilder Desktop. The software serves two purposes: to configure the Si5351 with optimal configuration based on the desired frequencies and to control the EVB when connected to a host PC. The optimal configuration can be saved from the software in text files that can be used in any system, which configures the device over I2C. ClockBuilder Desktop can be downloaded from www.silabs.com/ClockBuilder and runs on Windows XP, Windows Vista, and Windows 7. **Once the configuration file has been saved, the device can be programmed via I2C by following the steps shown in Figure 12.**

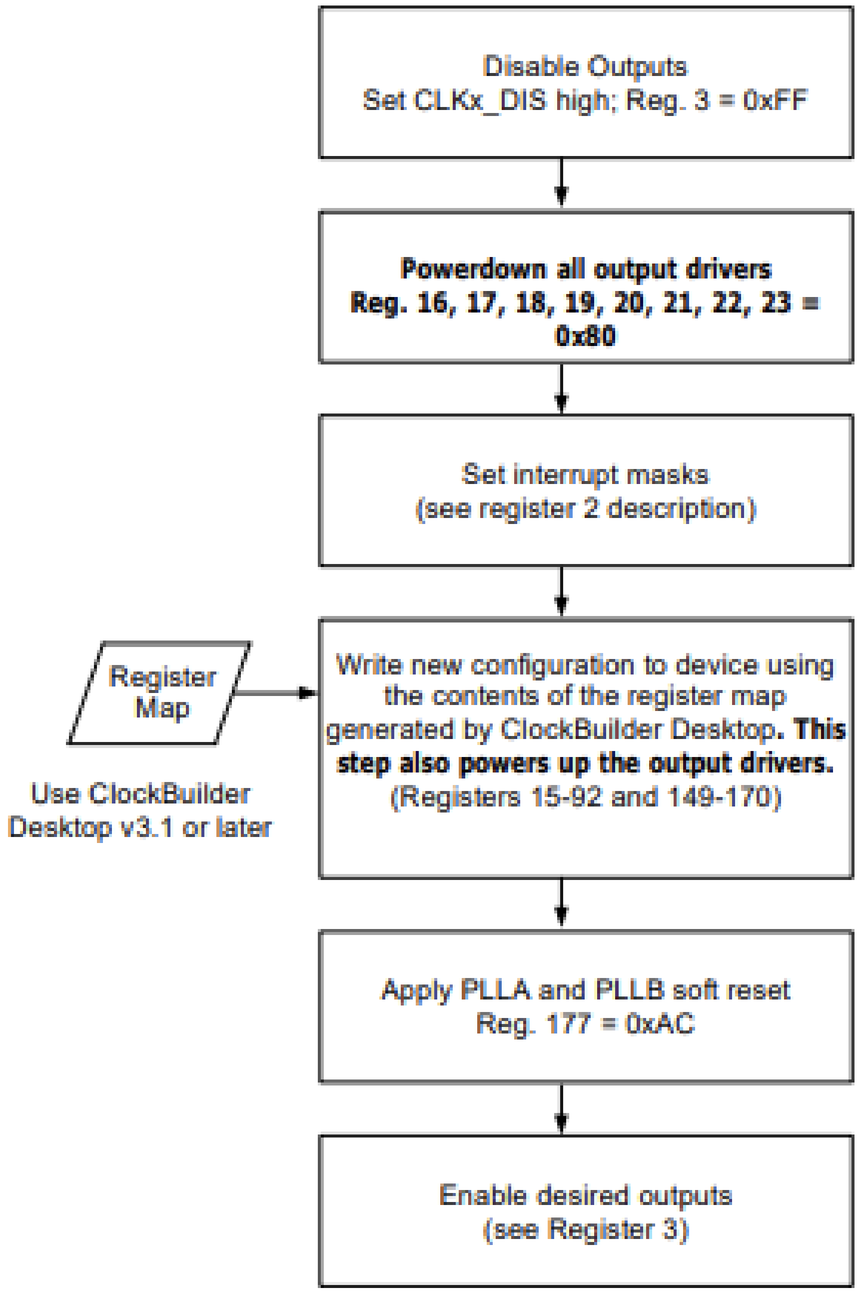


Figure 12. I^{2}C Programming Procedure

Register 2. Interrupt Status Mask

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name	SYS_INIT_MASK	LOL_B_MASK	LOL_A_MASK	LOS_MASK				
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reset value = 0000 0000

Bit	Name	Function
7	SYS_INIT_MASK	**System Initialization Status Mask.** Use this mask bit to prevent the INTR pin (Si5351C only) from going low when SYS_INIT is asserted. 0: Do not mask the SYS_INIT interrupt. 1: Mask the SYS_INIT interrupt.
6	LOL_B_MASK	**PLLB Loss Of Lock Status Mask.** Use this mask bit to prevent the INTR pin (Si5351C only) from going low when LOL_B is asserted. 0: Do not mask the LOL_B interrupt. 1: Mask the LOL_B interrupt.
5	LOL_A_MASK	**PLL A Loss Of Lock Status Mask.** Use this mask bit to prevent the INTR pin (Si5351C only) from going low when LOL_A is asserted. 0: Do not mask the LOL_A interrupt. 1: Mask the LOL_A interrupt.
4	LOS_MASK	**CLKIN Loss Of Signal Mask (Si5351C Only).** Use this mask bit to prevent the INTR pin (Si5351C only) from going low when LOS is asserted. 0: Do not mask the LOS interrupt. 1: Mask the LOS interrupt.
3:0	Reserved	Leave as default.

Register 3. Output Enable Control

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name	CLK7_OEB	CLK6_OEB	CLK5_OEB	CLK4_OEB	CLK3_OEB	CLK2_OEB	CLK1_OEB	CLK0_OEB
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reset value = 0000 0000

Bit	Name	Function
7:0	CLKx_OEB	**Output Disable for CLKx.** Where x = 0, 1, 2, 3, 4, 5, 6, 7 0: Enable CLKx output. 1: Disable CLKx output.

Register 165. CLK0 Initial Phase Offset

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name		CLK0_PHOFF[6:0]						
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reset value = 0000 0000

Bit	Name	Function
7	Reserved	Only write 0 to this bit.
6:0	CLK0_PHOFF[6:0]	**Clock 0 Initial Phase Offset.** CLK0_PHOFF[6:0] is an unsigned integer with one LSB equivalent to a time delay of Tvco/4, where Tvco is the period of the VCO/PLL associated with this output.

Register 177. PLL Reset

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name	PLLB_RST		PLLA_RST					
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reset value = 0000 0000

Bit	Name	Function
7	PLLB_RST	**PLLB_Reset.** Writing a 1 to this bit will reset PLLB. This is a self clearing bit (Si5351A/C only).
6	Reserved	Leave as default.
5	PLLA_RST	**PLLA_Reset.** Writing a 1 to this bit will reset PLLA. This is a self clearing bit.
4:0	Reserved	Leave as default.

Register 183. Crystal Internal Load Capacitance

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Name	XTAL_CL[1:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Reset value = 11xx xxxx

Bit	Name	Function
7:6	XTAL_CL[1:0]	**Crystal Load Capacitance Selection.** These 2 bits determine the internal load capacitance value for the crystal. See "3.1.1. Crystal Inputs (XA, XB)" on page 11. 00: Reserved. Do not select this option. 01: Internal CL = 6 pF. 10: Internal CL = 8 pF. 11: Internal CL = 10 pF (default).
5:0	Reserved	Leave as default.

Si5351A 10-MSOP
Top View

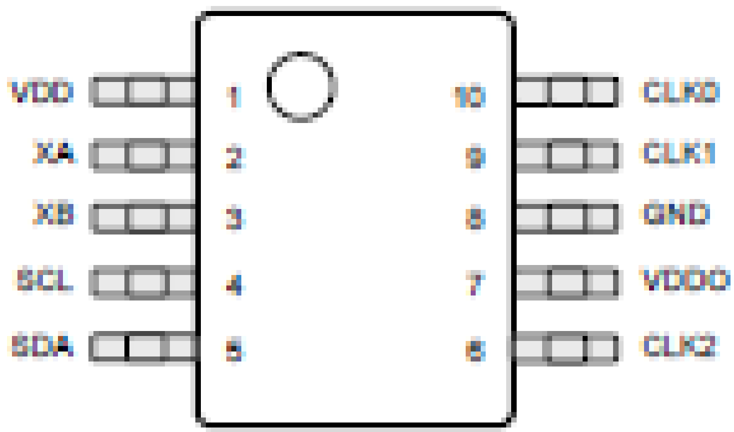


Table 13. Si5351A 10-MSOP Pin Descriptions

Pin Name	Pin Number 10-MSOP	Pin Type*	Function
XA	2	I	Input pin for external crystal.
XB	3	I	Input pin for external crystal.
CLK0	10	O	Output clock 0.
CLK1	9	O	Output clock 1.
CLK2	6	O	Output clock 2.
SCL	4	I	Serial clock input for the I^2C bus. This pin must be pulled-up using a pull-up resistor of at least 1 kΩ.
SDA	5	I/O	Serial data input for the I^2C bus. This pin must be pulled-up using a pull-up resistor of at least 1 kΩ.
VDD	1	P	Core voltage supply pin.
VDDO	7	P	Output voltage supply pin for CLK0, CLK1, and CLK2. See "6.2. Power Supply Sequencing" on page 21.
GND	8	P	Ground.

***Note:** I = Input, O = Output, P = Power

```
#include <wiringPiI2C.h>
#include <wiringPi.h>
#include <stdio.h>

// Si5351A I2C address
#define SI5351A_ADDRESS 0x60

// Register addresses
#define PLL_A_REG_MSB 0x26
#define PLL_A_REG_LSB 0x27
#define CLK_CTRL_REG 0x0F

// Constants for PLL and clock configuration
#define PLL_DIVIDER 1
#define OUTPUT_FREQ 457000 // 457kHz

int main() {
   // Initialize WiringPi and the I2C connection
   wiringPiSetup();
   int si5351a_fd = wiringPiI2CSetup(SI5351A_ADDRESS);

   if (si5351a_fd == -1) {
      fprintf(stderr, "Error opening I2C device\n");
      return 1;
   }

   // Configure PLL A for the desired output frequency
   int pllFreq = OUTPUT_FREQ * PLL_DIVIDER;
   wiringPiI2CWriteReg8(si5351a_fd, PLL_A_REG_MSB, (pllFreq >> 8) & 0xFF);
   wiringPiI2CWriteReg8(si5351a_fd, PLL_A_REG_LSB, pllFreq & 0xFF);

   // Configure Clock 0 for the desired output frequency
   wiringPiI2CWriteReg8(si5351a_fd, CLK_CTRL_REG, 0x4C);  // CLK0 powered on, driven by
PLL A

   // Close the I2C connection
   close(si5351a_fd);

   return 0;
}
```

for raspberry pi
sudo apt-get update
sudo apt-get install wiringpi
gcc -o si5351a_init si5351a_init.c -lwiringPi
./si5351a_init

To use a Raspberry Pi to program the Si5351A, you'll need to connect the Si5351A to the Raspberry Pi using the I2C interface. Step-by-step guide:
Hardware Connections:
Connect Si5351A to Raspberry Pi:
Connect the I2C lines (SCL and SDA) from the Si5351A to the corresponding GPIO pins on the Raspberry Pi. The Si5351A may need pull-up resistors on these lines.

Power Supply:
Provide the required power supply to the Si5351A. Connect the ground (GND) pin of the Si5351A to the ground of your Raspberry Pi.

Software Setup:
Enable I2C Interface on Raspberry Pi:
Ensure that the I2C interface is enabled on your Raspberry Pi. You can enable it through the raspi-config utility or by modifying the /boot/config.txt file.

Install Necessary Software:
Install the i2c-tools package to use utilities for checking I2C devices:
sudo apt-get update
sudo apt-get install i2c-tools

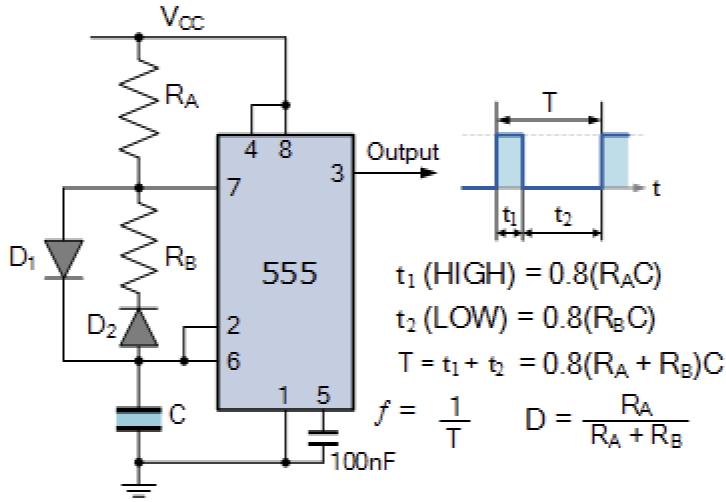
Enable I2C Kernel Module:
Add the I2C kernel modules to the /etc/modules file to ensure they are loaded at boot:
sudo echo "i2c-dev" >> /etc/modules

Reboot Raspberry Pi:
Reboot the Raspberry Pi for the changes to take effect:
sudo reboot

Install WiringPi Library:
If you haven't installed WiringPi (a GPIO access library), you can do so using:
sudo apt-get install wiringpi

Verify:
Check I2C Devices:
Use the i2cdetect command to check if the Si5351A is detected on the I2C bus:
sudo i2cdetect -y 1

We can generate a 457kHz frequency signal with the IC but then we need a 555 timer to modulate it to our required HIGH / LOW time, so the receivers can pick it up.



T1 => 70ms
T2 => 400ms
T = 1000 +/- 300 ms

T1 = 200ms
T2 = 800ms
T = 1000ms

3/3/24 Arduino premade v PCB design (RT)

Arduino Nano + Si5351 Breakout

More money, easier, user friendly, not scalable

- may still need small pcb, or solderable breadboard for the extra stuff

AtmegaIC + Si5351 IC + PCB design

Cheaper, less user friendly, manufacturable.

Arduino
https://www.adafruit.com/product/2010

Adafruit Si5351 Breakout
https://www.adafruit.com/product/2045

Protoboard
https://www.adafruit.com/product/4786

4xAA Batt Holder
https://www.adafruit.com/product/830

Ferrite Rod

Wire

SMA edge launch (need male side still)
https://www.adafruit.com/product/1865

3/4/24 Ebay Avalanche Transmitter disassembly:

Ferrite rod is: 99.5 mm long, 9.65mm diameter, 21 windings, wire 0.71mm diameter. (22 AWG?)

4/7/2023

I believe I have successfully programmed the Si5351. Monday, I will go into the lab to confirm. **We have changed to an easier route, that will ensure people building our open source product will have an easier time**. Here are my notes from the date:

Once the Si5351 is powered off and then powered back on, it will lose its programmed configuration and will not automatically resume producing the previously programmed frequency. The Si5351 does not have non-volatile memory to retain its configuration across power cycles. Therefore, it will need to be reprogrammed each time it is powered on to output the desired frequency.

When you power on the Si5351, it will start in its default state, and you will need to initialize it and set the desired frequency again using your control program (e.g., from an Arduino, Raspberry Pi, or any other microcontroller or microprocessor).

To ensure the Si5351 consistently outputs the desired frequency upon power-up, you should:

Implement Initialization in Startup Code: Ensure that your system's startup code (running on the microcontroller or microprocessor that controls the Si5351) initializes the Si5351 and sets it to the desired frequency as part of its boot process.

Use a Persistent Control Source: If the device controlling the Si5351 (like an Arduino or Raspberry Pi) retains its program memory across power cycles, it can automatically reconfigure the Si5351 each time it powers up.

Consider Power Management: If the Si5351 needs to maintain its output frequency continuously, consider designing your system's power management to prevent the Si5351 from losing power. For example, you could use a backup power source or uninterruptible power supply (UPS) for critical components.

In summary, you will need to programmatically ensure that the Si5351 is configured each time the system is powered up to maintain the desired frequency output.

Meaning we just initialize the Si5351 inside the setup of the microcontroller we use. The person building this will need to download the Arduino IDE, download our code, connect the microcontroller to their PC, upload the code, then they should be good to go for the lifetime of the product, even if its powered off. The Arduino board will house the code. We could also set this up with pre-loaded Arduinos.

ARDUINO IDE V1.8.19, Library Etherkit Si5351, Mega2560 microcontroller
Code

```
#include <si5351.h>

#include <Wire.h>



Si5351 si5351;



void setup() {

  Wire.begin();

  si5351.init(SI5351_CRYSTAL_LOAD_8PF, 0, 0);



  // Set up Clock 0 for 457 kHz

  si5351.set_freq((457000ULL * 100ULL), SI5351_CLK0);

  si5351.output_enable(SI5351_CLK0, true);

}



void loop() {

  // Nothing to do in the loop

}
```