1. Implementación del emulador

1.1. Componentes

Para la implementación del emulador de arreglo de antenas utilizamos dos componentes: un microcontrolador y un sintetizador digital directo.

El microcontrolador elegido es un STM32F103 de ST dentro de la placa de desarrollo Blue-Pill que puede verse en la figura 1. El STM32F103 es un microcontrolador de 32-bit que trabaja a una frecuencia máxima de 72 MHz, tiene una memoria flash de 64 o 128 kB y entre otras características, cuenta con siete temporizadores e interfaces de comunicación SPI, I2C, USART, CAN y USB [**STM32F103_datasheet**].

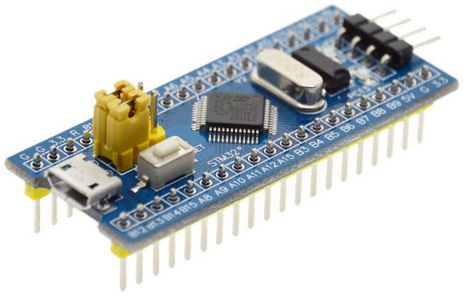

Figura 1: Placa de desarrollo Blue-Pill con un microcontrolador STM32F103.

Para el sintetizador digital directo, se cuenta con una placa de desarrollo del AD9959 de la empresa Analog Devices como la de la figura 2. El AD9959 es un sintetizador digital directo de cuatro canales que trabaja a una frecuencia máxima de reconstrucción de 500 MHz y cuenta con conversores digital-analógico de 10-bits y comunicación SPI [**ad9959_datasheet**].

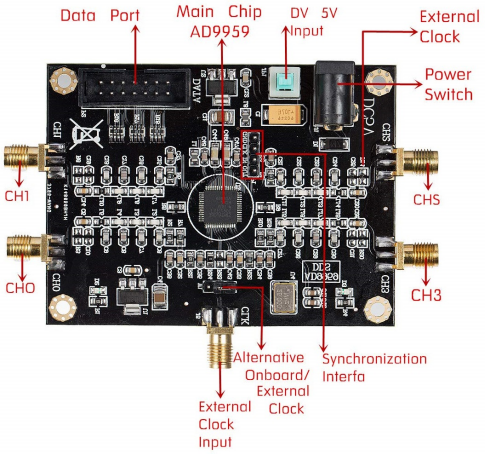

Figura 2: Placa de desarrollo del sintetizador digital AD9959.

1.2. Comunicación y conexiones

La placa de desarrollo del A9959 tiene un puerto de datos con 14 pines de entrada y salida, 4 pines para la sincronización de múltiples sintetizadores y un jumper para la selección de reloj.

El jumper controla la entrada de reloj de referencia del sintetizador. Se puede elegir entre una entrada externa de reloj o un cristal de 25 MHz que ya viene en la placa. Por simplicidad, posicionamos el jumper para elegir la última opción.

Los cuatro pines de sincronización de múltiples dispositivos son SYNC_CLK, SYNC_IN, SYNC_OUT y GND. Su funcionamiento se mencionará en el apartado 1.5.

Con respecto al puerto de datos, conectamos los pines al microcontrolador como se muestra en la figura 3.

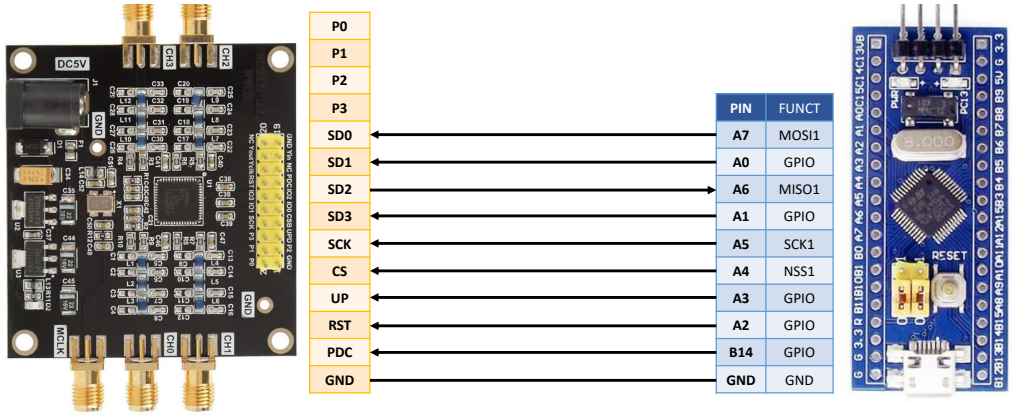

Figura 3: Conexiones entre microcontrolador y DDS.

Como se mencionó, el AD9959 se comunica a través de una interfaz SPI. Existen cuatro modos de comunicación programables para este dispositivo: un bit serie con dos cables, un bit serie con tres cables, dos bits serie y cuatro bits serie. Uno, dos y cuatro bits serie hacen referencia a la cantidad de bits que se transmiten en cada flanco ascendente del reloj. Para el modo de un bit serie, la transmisión y recepción puede hacerse por un canal bidireccional (dos cables) o dos canales unidireccionales (tres cables). Los modos de dos y cuatro bits usan canales bidireccionales. En nuestra implementación, usamos el modo un bit serie con tres cables por ser el estándar más utilizado para comunicaciones SPI.

Los pines SDIO_0 a SDIO_3 se usan para transmisión y recepción de datos y, si el modo de comunicación elegido deja algún pin libre, para realizar operaciones de *ramp-up* y *ramp-down* (RD/RD) y/o para sincronizar múltiples sintetizadores digitales directos. Para el modo elegido, el pin SDIO_0 funciona como entrada al sintetizador, es decir, *Master-Out Slave-In* (MOSI), y el pin SDIO_2, como salida, es decir, *Master-In Slave-Out* (MISO). Conectamos entonces el pin SDIO_0 del DDS al pin A7 del microcontrolador que corresponde a la función MOSI de la interfaz SPI1 (MOSI1) y el pin SDIO_2 al A6 (MISO1). Como para este prototipo no utilizaremos las funciones secundarias de los pines SDIO_X, los conectamos a pines GPIO configurados como salidas en 0.

El protocolo SPI también necesita dos pines adicionales: una línea de reloj que permite sincronizar la comunicación y un pin de selección para activar y desactivar la comunicación al dispositivo. Conectamos los pines de reloj SCLK y de selección $\overline{\text{CS}}$ del sintetizador a los pines A5 (SCK1) y A4 (NSS1) del microcontrolador.

El pin IO_UPDATE del AD9959 tiene como función transferir los datos recibidos de un buffer a los verdaderos registros. Esto produce que los cambios efectuados en el dispositivo se apliquen simultáneamente con el flanco ascendente del pin IO_UPDATE. Conectamos este pin a A3 como salida digital del microcontrolador.

El pin MASTER_RESET es una entrada digital que, cuando recibe un 1, reinicia los registros del sintetizador a sus valores por defecto. Conectamos este pin a A2 como salida digital del microcontrolador.

El pin PWR_DWN_CTL controla el modo de ahorro de energía. Cuando esta entrada recibe un 1, se activa el modo de ahorro de energía. Como estaremos constantemente usando el sintetizador, conectamos este pin al GPIO B14 del microcontrolador configurado como salida en 0.

Los pines P0 a P3 del AD9959 se usan para modulación. Como no utilizaremos esta funcionalidad, podemos dejar estos pines flotando.

Necesitamos finalmente que el sintetizador y el microcontrolador tengan una tierra común. Para esto, conectamos los pines GND de ambos dispositivos.

1.3. Configuración inicial de registros

El sintetizador AD9959 posee una cantidad considerable de registros que controlan las distintas funciones del dispositivo. Pero para nuestro prototipo sólo necesitaremos modificar dos registros en la configuración inicial como se explica a continuación.

El registro *Channel Selection Register* (CSR) de la dirección 0x00:

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
X	X	X	X	0	0	1	0

Los bits 7 a 4 seleccionan los canales a los que se le aplican los cambios. En la configuración inicial no es necesario seleccionar ningún canal. El bit 3 debe ser 0. Los bits 2 y 1 configuran el modo de comunicación. El valor b01 selecciona el modo de un bit serie con tres cables. El bit 0 en 0 determina que el formato de los datos es bit más significativo primero (MSB).

El registro *Function Register 1* (FR1) de la dirección 0x01:

Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
1	1	0	1	0	0	0	0
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
0	0	0	0	0	0	0	0
Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	0	0	0	0	0	0	0

Este registro nos permite, principalmente, configurar la frecuencia del sistema. Los bits 22:18 determinan la relación de división del PLL. Este valor, de 4 a 20, multiplica la frecuencia del reloj de entrada. Asignamos el valor 20 = 0b10100 para obtener la máxima frecuencia 20×25 MHz = 500 MHz.

El bit 23 debe ser 1 para frecuencias del sistema superiores a 255 MHz. Los demás bits no son de relevancia para nuestra aplicación y los dejamos en 0.

1.4. Control de las señales generadas

Una vez que hemos realizado la configuración inicial de registros, podemos comenzar con la emulación propiamente dicha.

Configuramos el Timer1 del microcontrolador para realizar una interrupción cada milisegundo, es decir, con una velocidad de 1 kHz. En cada interrupción se calcula la posición del satélite y en base a ello, la frecuencia, amplitud y fase de las señales recibidas por los elementos del arreglo de antena como se describió en el apartado 1. Luego, se escriben los registros necesarios para obtener estas señales en los canales del sintetizador como explicaremos a continuación.

Primero, debemos seleccionar el canal que queremos modificar. Recordemos que para ello se deben escribir los bits 7 a 4 del registro CSR, dirección 0x00. Un 1 activa el canal y un 0 lo desactiva. El bit 7 controla el canal 3, el bit 6, el canal 2, el bit 5, el canal 1 y el bit 4, el canal 0. Al escribir este registro, debemos tener cuidado en respetar los valores del resto de los bits que asignamos para la configuración deseada.

Por un lado, para asignar la frecuencia, escribimos el registro de 4-bytes *Channel Frequency Tuning Word 0* (CFTW0), dirección 0x04. El valor de la palabra de 32-bits a asignar, *Frequency Tuning Word* (FTW), está dada por

$$FTW = \frac{f_{out}}{f_s} \times 2^{32}, \tag{1}$$

siendo f_{out} [Hz] la frecuencia de salida deseada y f_s [Hz], la frecuencia del sistema (teniendo en cuenta la división del PLL). Como sabemos por la teoría de Nyquist, $f_{out} \leq f_s/2$.

Por otro lado, para asignar la fase, escribimos el registro de 2-bytes *Channel Phase Offset Word 0* (CPOW0), dirección 0x05. El valor de la palabra de 14-bits, *Phase Offset Word* (POW), a asignar en los bits 13 a 0, está dada por

$$POW = \frac{\phi_{out}}{360} \times 2^{14}, \tag{2}$$

siendo $0° \leq \phi_{out} < 360°$ la fase de salida deseada. Los bits 15 y 14 pueden tomar cualquier valor.

Por último, para asignar la amplitud, escribimos el registro de 3-bytes *Amplitude Control Register* (ACR), dirección 0x06. El valor de la palabra de 10-bits, *Amplitude Scale Factor* (ASC), a asignar en los bits 9 a 0, está dada por

$$ASC = a_{out} \times \left(2^{10} - 1\right), \tag{3}$$

siendo $0 \leq a_{out} \leq 1$ la amplitud de salida normalizada. Los bits 23 a 10 definen otros parámetros de la amplitud que no utilizaremos para nuestro prototipo, por lo que los dejamos en 0.

1.5. Posibilidad de extensión del emulador

El AD9959 permite sincronizar múltiples sintetizadores del mismo modelo fácilmente [**ad9959_datasheet**]. Esto es una característica especialmente interesante si queremos extender el sistema propuesto para un arreglo plano de antenas.

Existe un modo automático y dos manuales para la sincronización de los sintetizadores. A continuación explicaremos, brevemente, el modo automático por ser el de más fácil implementación.

Un AD9959 actuará como maestro y el resto, como esclavo. En lo que respecta a las conexiones, el pin SYNC_OUT del maestro se conecta al pin SYNC_IN de los esclavos. Un circuito interno de autosincronización se encarga de que las máquinas de estados de los esclavos se igualen a las del maestro. El pin SYNC_CLK no se utiliza para el modo automático, sí para los modos manuales. Se debe además interconectar los pines GND para unificar la tierra de los dispositivos.

En lo que respecta a la configuración de registros, se debe escribir el bit 7 del registro *Function Register 2* (FR2), dirección 0x02, en 1 para activar el modo de sincronización automático. La asignación de maestro y esclavos se realiza mediante el bit 6 del registro FR2. Un 1 determina que el dispositivo es maestro, un 0, esclavo. El bit 5 del FR2 indica que hubo una falla en la sincronización, pero no la detiene.

Lo descripto anteriormente sincroniza las máquinas de estado de los dispositivos, pero se debe tener además especial cuidado con el pin IO_UPDATE. La salida del microcontrolador que controla este pin debe llegar a todos los sintetizadores simultáneamente. Si hubiese algún retraso, se producirá un desfasaje indeseado de las señales.

Por último, la comunicación SPI entre microcontrolador y sintetizadores debe respetar la configuración de esclavos independientes. Cabe aclarar que, ahora, con maestro nos referimos al microcontrolador y con esclavos, a todos los sintetizadores (independientemente de cuál es el sintetizador maestro o esclavo en la sincronización). En esta configuración de esclavos independientes, las líneas SCLK, MOSI y MISO son compartida entre todos los esclavos y las líneas de $\overline{\text{CS}}$ son individuales, es decir que hay una para cada esclavo, y controlan con cuál dispositivo se quiere comunicar el maestro.