1. **Introducción**

Los protocolos de comunicación serial son una herramienta esencial a la hora de trabajar con microcontroladores, pueden ser implementados para emitir y recibir datos a otros microcontroladores, en aplicaciones como detección mediante sensores especializados (módulos de comunicación, medidores de movimiento o nivel), manejo de periféricos especializados, como el uso de pantallas que es el objetivo de este trabajo y demás aplicaciones en las cuales se requiera una traducción de un periférico de salida o entrada que deba traducir datos de la comunicación serial a los requerimientos del periférico especializado al que se encuentre unido.

Por el motivo anterior se elige la comunicación SPI, que junto con I2C y USART son los protocolos más usados para estos fines, y además una oled que funciona con el dispositivo traductor SSD1306 con el objetivo de comprender como aplicar sobre un hardware concreto la práctica del aprendizaje del manejo de la comunicación SPI.

1. **Objetivos**

*Principal:* implementar sobre el hardware mencionado la comunicación SPI con el objetivo de poder graficar sobre él y crear una interfaz cómoda para el usuario.

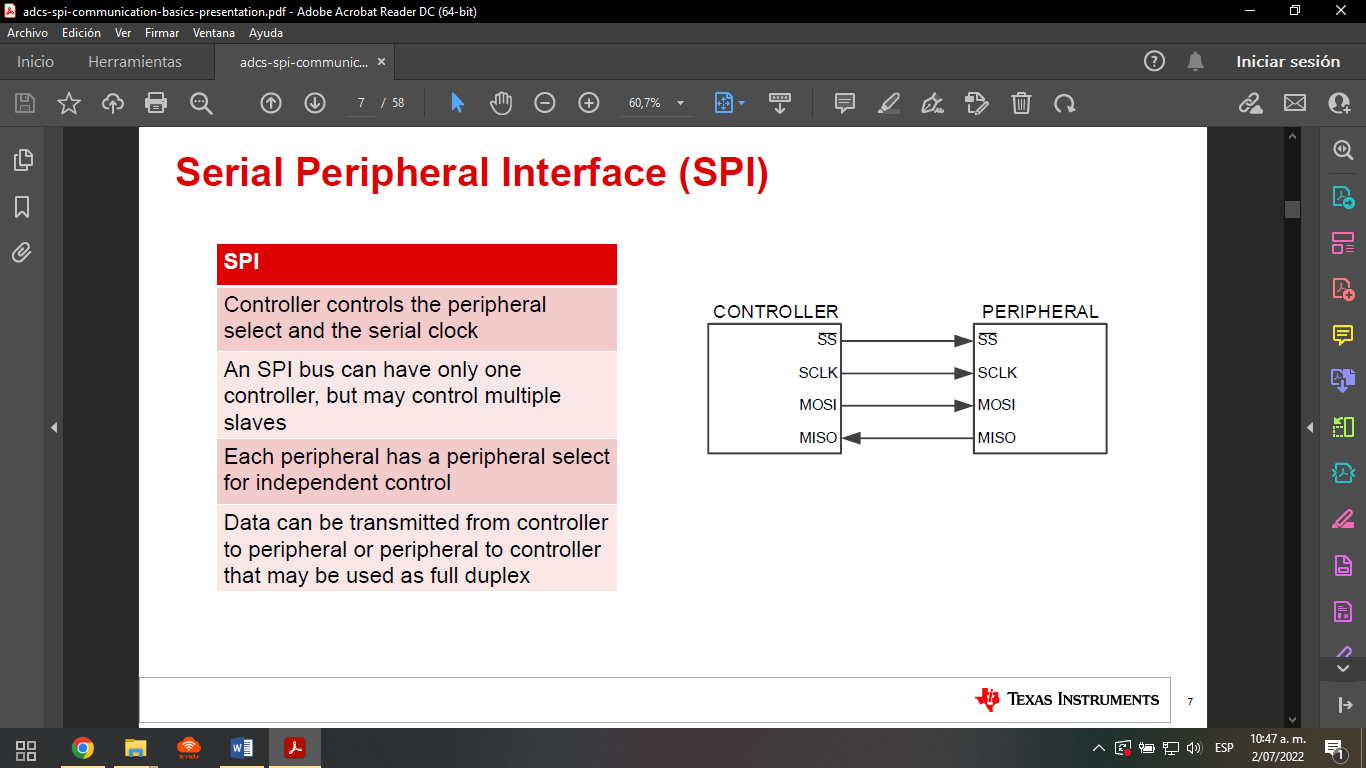
*Secundarios:*

* Comprender el uso de la comunicación serial SPI, sus parámetros e implementación para el caso concreto de la LAUNCHXL\_F28379D.
* Encontrar los requerimientos puntuales del uso del traductor SSD1306 mediante su datasheet.
* Encontrar las bases para poder comunicar otros periféricos mediante un método de reconocimiento e implementación que sirva para cualquier dispositivo que pueda comunicarse con la tarjeta mediante protocolo SPI.

1. **Marco teórico**

***Comunicación SPI, funcionamiento en general:***

*Fuente(*[*https://ieeexplore-ieee-org.crai-ustadigital.usantotomas.edu.co/servlet/opac?bknumber=9218891*](https://ieeexplore-ieee-org.crai-ustadigital.usantotomas.edu.co/servlet/opac?bknumber=9218891)

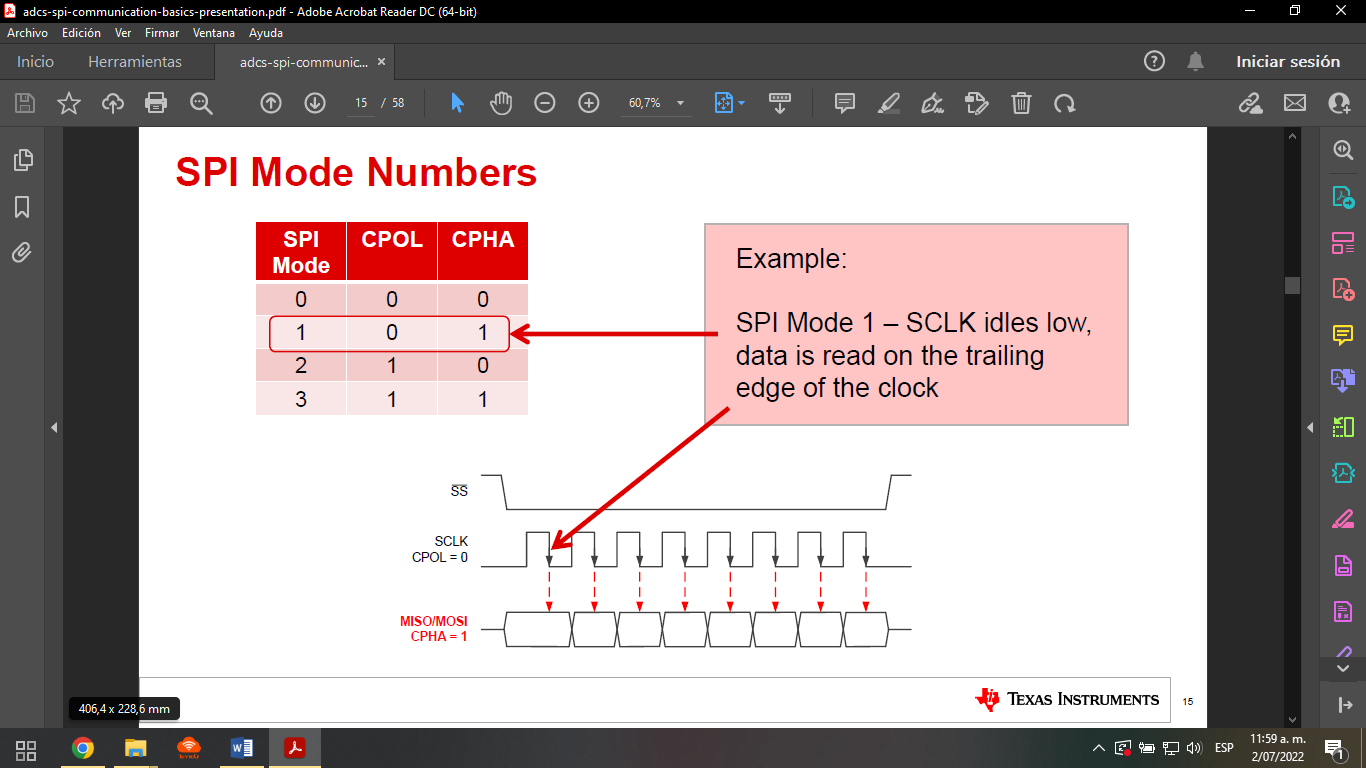


* **SCLK:** es usado para avisar cuando se ingresa un dato, por flanco de subida o bajada, también se conoce como SCK y es usado en todos los dispositivos conectados*.*
* **SS:**selecciona el dispositivo al cual nos referiremos ya que la comunicación puede hacerse con varios dispositivos, también se conoce como: SS, SSEL, CS, CS, SYNC, nSS y SS#.
* **MOSI:** Es la salida del controlador al esclavo, conocida también comoSIMO y MSTR (desde el dispositivo periférico como: SDI, DI, DIN y SI. Desde el controlador: SDO, DO, DOUT y SO). Se conectar entre todos los dispositivos al tiempo.
* **MISO:**  Comunicación desde el periférico especializado al microcontrolador, se convierte en alta impedancia si el dispositivo no está seleccionado, por lo mismo no hay riesgo de conectar varios dispositivos a la misma línea, también conocido como SOMI; desde el periférico como SDO, DO, DOUT,SO; desde el controlador como SDI, DI, DIN, SI.

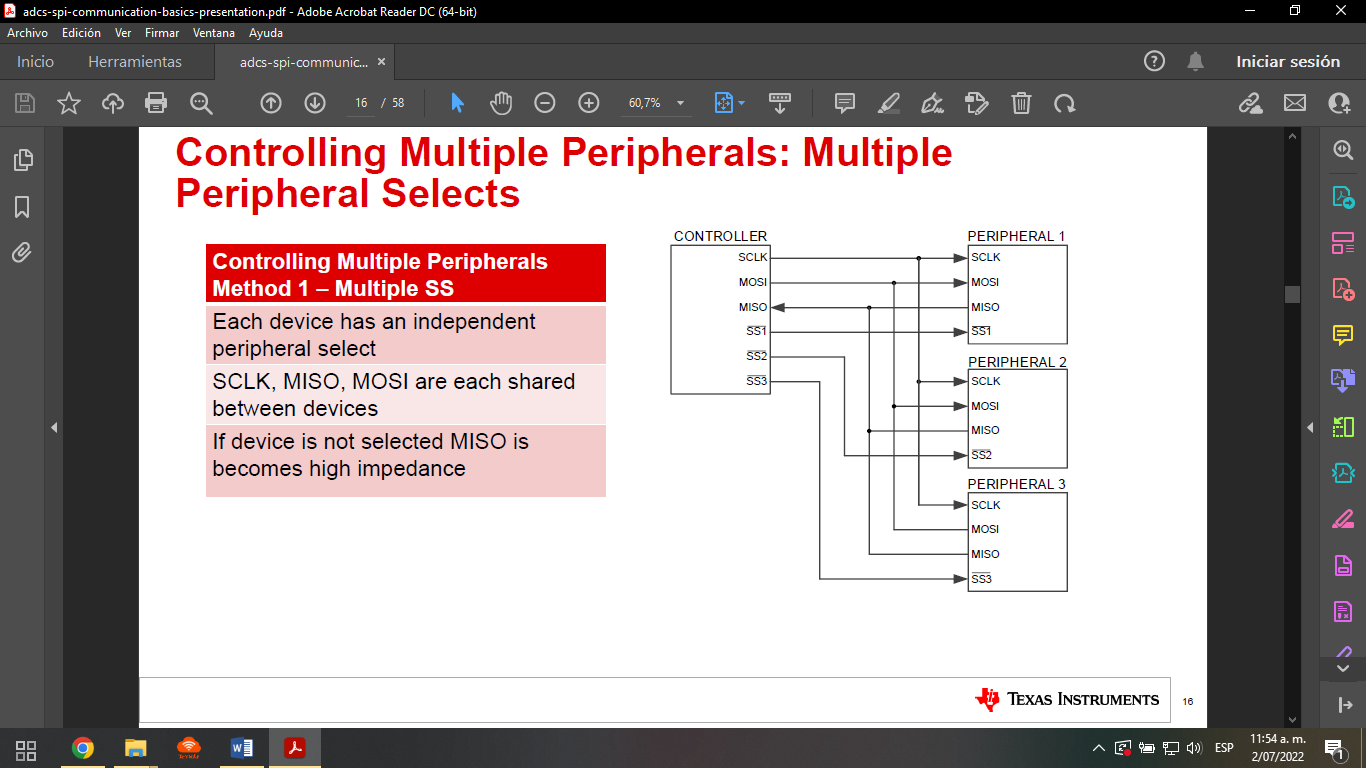
Además de las conecciónes mencionadas hay consideraciones que la misma TI nos menciona y que presumiblemente debemos implementar en la programación de la DSP:

* **CPOL:** es la polaridad del clock, si se encuentra en 0, la referencia es cero y se realiza la lectura de bit en el borde creciente, para un estado de CPOL=1, las condiciones son al revés.
* **CPHA:** Es la fase de la comunicación, si es cero, las condiciones de CPOL no cambian; si es igual a 1 la comunicación se efectúa en el flanco diferente al seleccionado en CPOL.

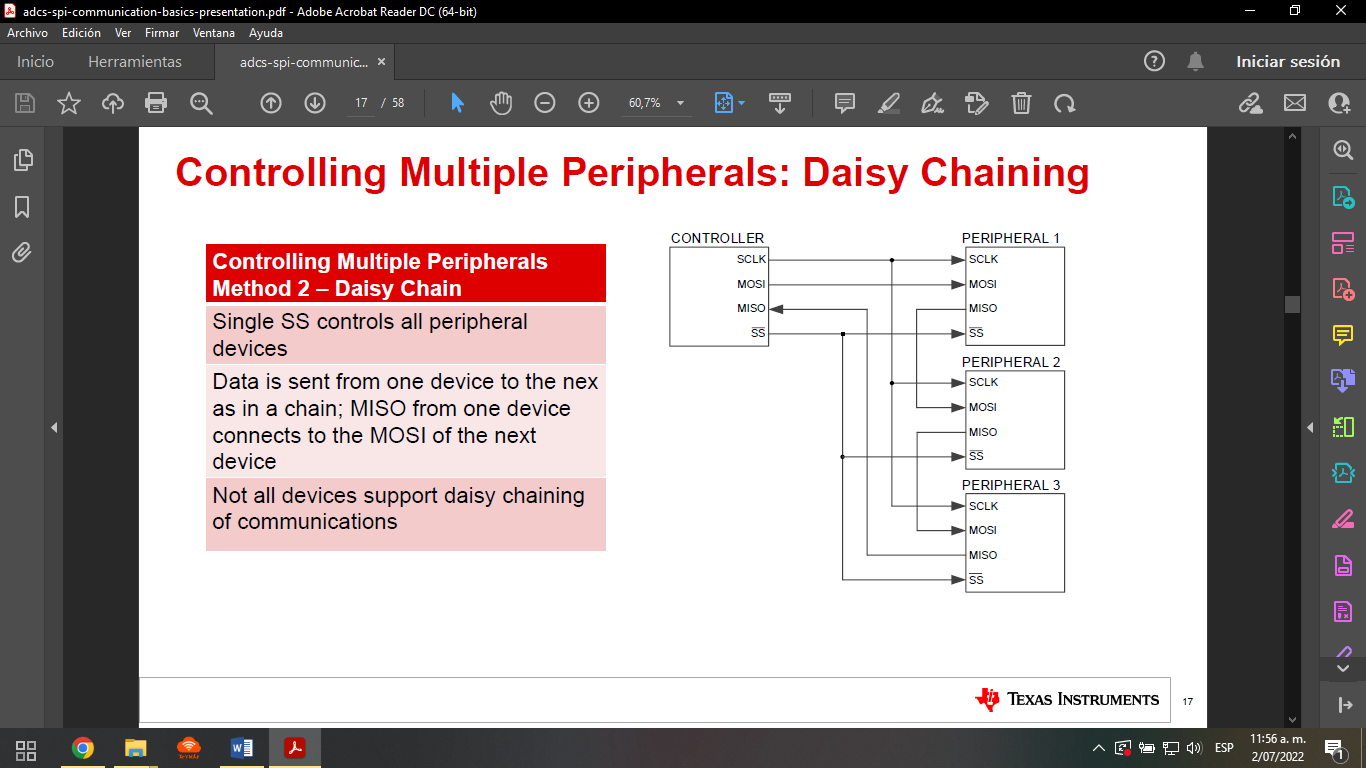
Estos bits configuran un total llamado **SPI Mode**, que se encarga de revisar la forma en que se realiza la lectura de los datos en base al CLK.



Recordemos que dichas comunicaciones solo se dan si se selecciona el periférico mediante SS, que generalmente está negado (se activa con un cero). Hay dos formas de comunicar varios dispositivos, la primera es mediante un selector individual por cada uno:



O un solo selector y la conexión entre los dispositivos periféricos, llamada **Daisy Chaining**:



Por el momento no ahondaremos en este apartado ya que solo queremos comunicar un solo periférico.

***Consideraciones para trabajar sobre la LAUNCHXL\_F28379D en base a la documentación de la familia C2000:***

* Se maneja un registro para la comunicación llamado SPIDAT que es donde se enviarán todos los datos.
* Los datos de entrada de los dispositivos se guardan en el registro SPIBUF. La CPU debe recibir un dato en este registro para poder continuar la operación, de lo contrario puede ocurrir un error de desbordamiento.

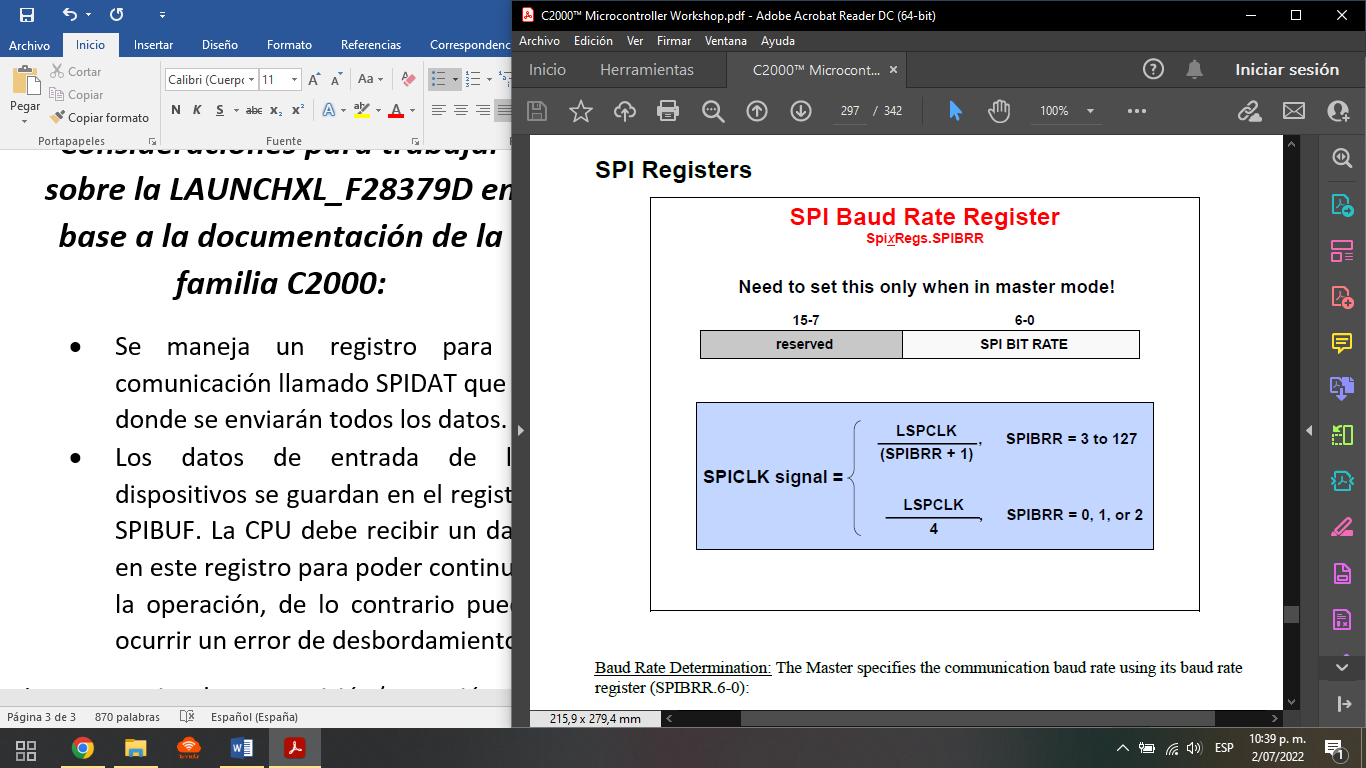
La secuencia de transmisión/recepción se debe dar de la siguiente manera según la fuente citada:

1. El esclavoescribe datos a ser enviados a su registro de intercambio (SPIDAT).
2. El maestro escribe datos a ser enviados a su registro de intercambio (SPIDAT o SPITXBUF).
3. Completando el paso 2 automáticamente la señal del maestro SPICLK se activa.
4. El MSB del registro (SPIDAT) del master se envía, y el LSB del registro (SPIDAT) del esclavo es cargado
5. El anterior paso se repite hasta que el número especificado de bits se halla transmitido
6. El registro SPIDAT es copiado al registro SPIRXBUF.
7. La bandera SPI INT se coloca en 1.
8. Se afirma la interrupción si el bit SPI INT ENA se establece en 1.
9. Si hay datos en SPITXBUF (de maestro o esclavo), se carga en SPIDAT y la transmisión vuelve a empezar tan pronto como se cargue el registro SPIDAT del maestro.

En el texto se enuncia que para el envío de datos de menos de 16 bits se debe justificar en los registros los bits que no se deben usar implementando mascaras para los datos a usar, no se presenta de forma clara la información sobre este tema.

A continuación, se enuncian los registros SPI a implementar:

* El registro de frecuencia de baudios:



Hasta el momento no entiendo qué es y LSPCLK y SPICLK (página 297).

SPIBRR es precisamente el registro de frecuencia de baudios.

SPICLK se refiere a el pin de pulsos serial

LSPCLK es la velocidad (rate) de los pulsos de reloj.

Recuerda que falta la tabla de los registros

Nueva fuente (<https://www.ti.com/lit/ug/sprug72/sprug72.pdf?ts=1657246330857&ref_url=https%253A%252F%252Fwww.google.com%252F>):

‘The maximum transmission rate in both slave mode and master mode is now LSPCLK/4’