* **Mode 0 (the default)** − Clock is normally low (CPOL = 0), and the data is sampled on the transition from low to high (leading edge) (CPHA = 0).
* **Mode 1** − Clock is normally low (CPOL = 0), and the data is sampled on the transition from high to low (trailing edge) (CPHA = 1).
* **Mode 2** − Clock is normally high (CPOL = 1), and the data is sampled on the transition from high to low (leading edge) (CPHA = 0).
* **Mode 3** − Clock is normally high (CPOL = 1), and the data is sampled on the transition from low to high (trailing edge) (CPHA = 1).
* **SPI.attachInterrupt(handler)** − Function to be called when a slave device receives data from the master.
* (SS) : pin 10
* (MOSI) : pin 11
* (MISO) : pin 12
* (SCK) : pin 13

**#include <SPI.h>**

**void setup (void) {**

**Serial.begin(115200); //set baud rate to 115200 for usart**

**digitalWrite(SS, HIGH); // disable Slave Select**

**SPI.begin ();**

**SPI.setClockDivider(SPI\_CLOCK\_DIV8);//divide the clock by 8**

**}**

**void loop (void) {**

**char c;**

**digitalWrite(SS, LOW); // enable Slave Select**

**// send test string**

**for (const char \* p = "Hello, world!\r" ; c = \*p; p++) {**

**SPI.transfer (c);**

**Serial.print(c);**

**}**

**digitalWrite(SS, HIGH); // disable Slave Select**

**delay(2000);**

**}**

SPI as Slave

**#include <SPI.h>**

**char buff [50];**

**volatile byte indx;**

**volatile boolean process;**

**void setup (void) {**

**Serial.begin (115200);**

**pinMode(MISO, OUTPUT); // have to send on master in so it set as output**

**char c**

**if (indx < sizeof buff) {**

**buff [indx++] = c; // save data in the next index in the array buff**

**if (c == '\r') //check for the end of the word**

**process = true;**

**}**

**}**

**void loop (void) {**

**if (process) {**

**process = false; //reset the process**

**Serial.println (buff); //print the array on serial monitor**

**indx= 0; //reset button to zero**

**}**

**}**

**Kickstart Your Career**