Introduction to Communication Protocols

Radhika Ghosal and Tushar Kataria

So, what are communication protocols?

- Ever used:
  - A USB drive?
  - An Ethernet cable?
  - An SD Card?
- All of these use some communication protocol or the other!

So, what are communication protocols?

- Whenever you need to talk to a piece of hardware, you’ll need to use a communication protocol both your devices understand.
- For today, we’ll be restricting ourselves to simple protocols used in embedded systems, for talking to devices like LCD screens and sensors.

The Language of Communication

1. Serial Communication
2. Parallel Communication
3. Bit Rate vs Baud Rate vs Throughput
4. Clock Skew
5. Serial vs Parallel
6. Simplex Communication
7. Half Duplex Communication
8. Full Duplex Communication

Let’s play a game!

SPI Protocol

- What we just demonstrated, is the SPI (Serial Peripheral Interface) protocol.
- Summary:
- SPI is a full-duplex synchronous serial data transfer protocol.
- Data transfer takes place in between Master and Slave devices.
- Each Master/Slave device has an internal 8 bit shift register, which is connected to other devices so as to form a circular/ring buffer.
- At each clock pulse, data gets right shifted in the circular/ring buffer.
- After 8 clock pulses, data is completely exchanged in between devices.
- SPI bus consists of four wires/signals – MOSI, MISO, SCK and SS’.
- When we connect more than one Slave devices, then we choose them using the SS’ signal.
- CPOL and CPHA must be set so that Master and Slave devices sync properly.
- AVR ISP uses SPI to program the microcontroller.

4 modes of SPI

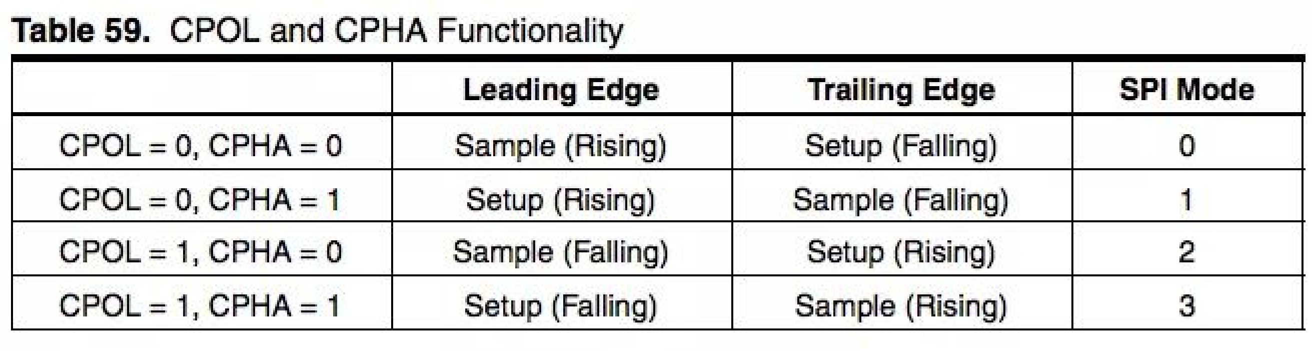


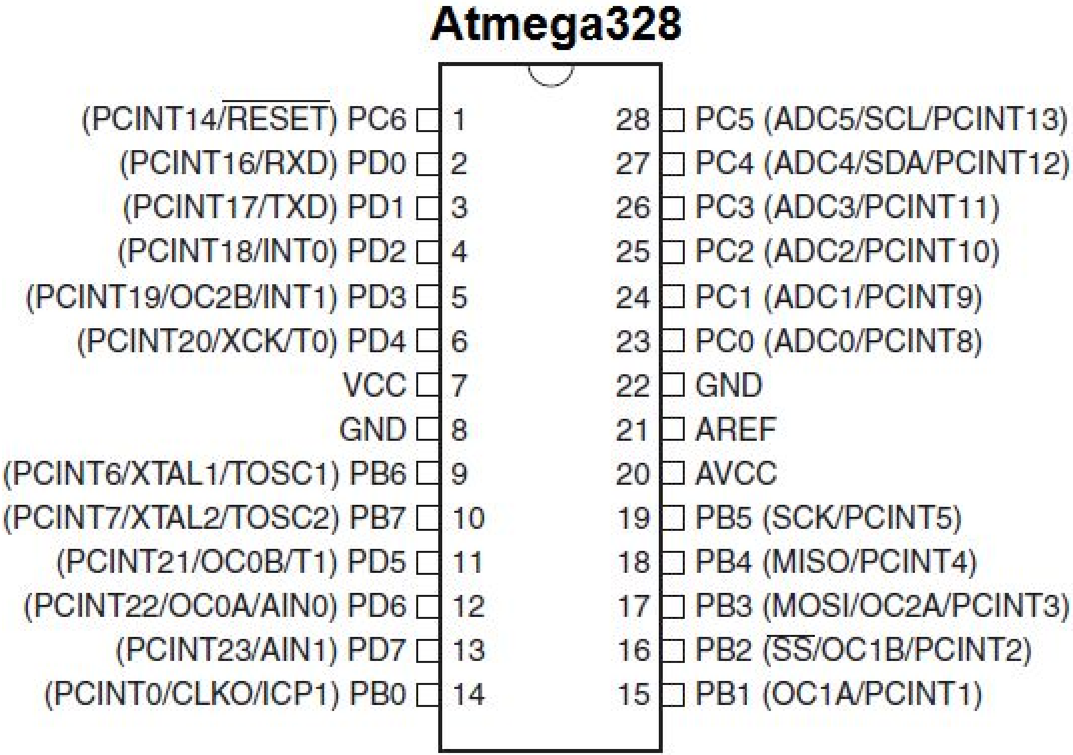
Table 59. CPOL and CPHA Functionality

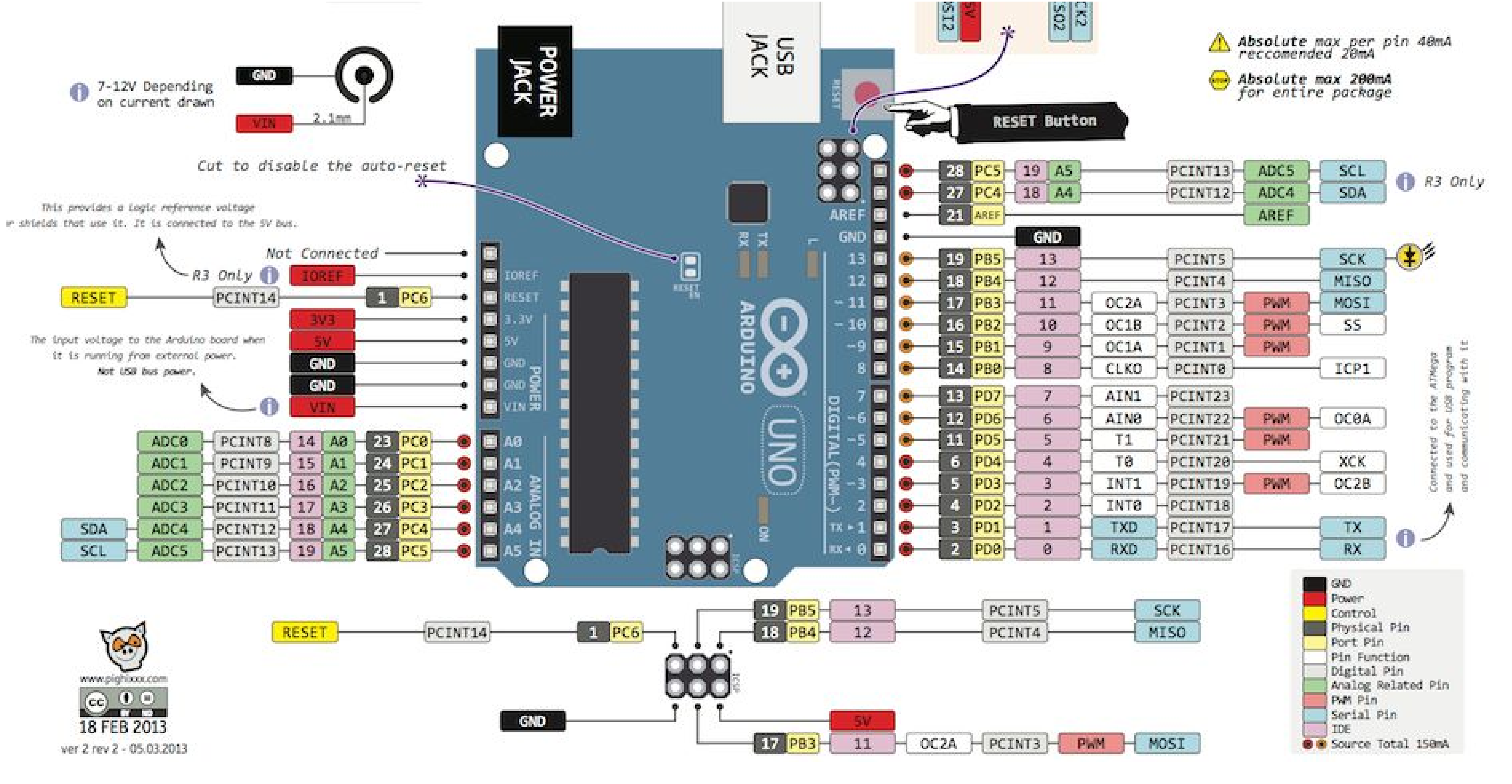
	Leading Edge	**Trailing Edge**	**SPI Mode**
CPOL = 0, CPHA = 0	Sample (Rising)	Setup (Falling)	0
CPOL = 0, CPHA = 1	Setup (Rising)	Sample (Falling)	1
CPOL = 1, CPHA = 0	Sample (Falling)	Setup (Rising)	2
CPOL = 1, CPHA = 1	Setup (Falling)	Sample (Rising)	3

Source-Atmega32 Datasheet

Talk is cheap, show me the code.

SPI Protocol





7-12V Depending on current drawn

Cut to disable the auto-reset

This provides a logic reference voltage for shields that use it. It is connected to the 5V bus.

R3 Only

The input voltage to the Arduino board when it is running from external power. Not USB bus power.

RESET Button

Absolute max per pin 40mA reccomended 20mA

Absolute max 200mA for entire package

R3 Only

Connected to the ATMega and used for USB programming and communicating with it

Not Connected				
IOREF				
RESET	PCINT14	1	PC6	
3V3				
5V				
GND				
GND				
VIN				

	ADC0	PCINT8	14	A0	23	PC0
	ADC1	PCINT9	15	A1	24	PC1
	ADC2	PCINT10	16	A2	25	PC2
	ADC3	PCINT11	17	A3	26	PC3
SDA	ADC4	PCINT12	18	A4	27	PC4
SCL	ADC5	PCINT13	19	A5	28	PC5

28	PC5	19	A5		PCINT13	ADC5	SCL
27	PC4	18	A4		PCINT12	ADC4	SDA
21	AREF					AREF	
		GND					
19	PB5	13			PCINT5		SCK
18	PB4	12			PCINT4		MISO
17	PB3	11		OC2A	PCINT3	PWM	MOSI
16	PB2	10		OC1B	PCINT2	PWM	SS
15	PB1	9		OC1A	PCINT1	PWM	
14	PB0	8		CLKO	PCINT0		ICP1
13	PD7	7		AIN1	PCINT23		
12	PD6	6		AIN0	PCINT22	PWM	OC0A
11	PD5	5		T1	PCINT21	PWM	
6	PD4	4		T0	PCINT20		XCK
5	PD3	3		INT1	PCINT19	PWM	OC2B
4	PD2	2		INT0	PCINT18		
3	PD1	1		TXD	PCINT17		TX
2	PD0	0		RXD	PCINT16		RX

RESET	PCINT14	1	PC6			
19	PB5	13		PCINT5		SCK
18	PB4	12		PCINT4		MISO
GND						
		5V				
17	PB3	11	OC2A	PCINT3	PWM	MOSI

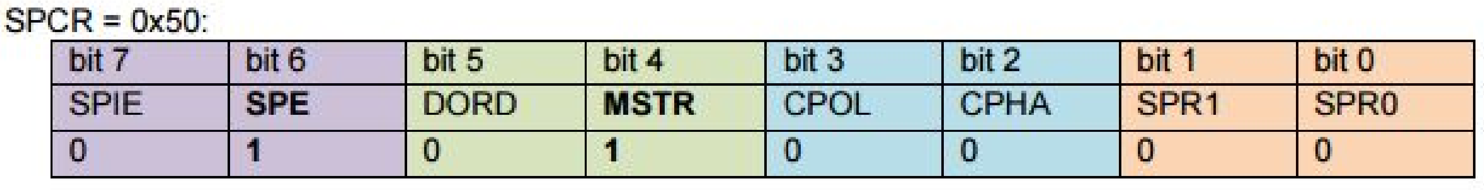
■	GND
🟥	Power
🟨	Control
■	Physical Pin
🟨	Port Pin
□	Pin Function
□	Digital Pin
🟩	Analog Related Pin
🟥	PWM Pin
🟦	Serial Pin
■	IDE
● ●	Source Total 150mA

www.pighixxx.com

18 FEB 2013

ver 2 rev 2 - 05.03.2013

SPI Protocol



SPCR = 0x50:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
SPIE	**SPE**	DORD	**MSTR**	CPOL	CPHA	SPR1	SPR0
0	**1**	0	**1**	0	0	0	0

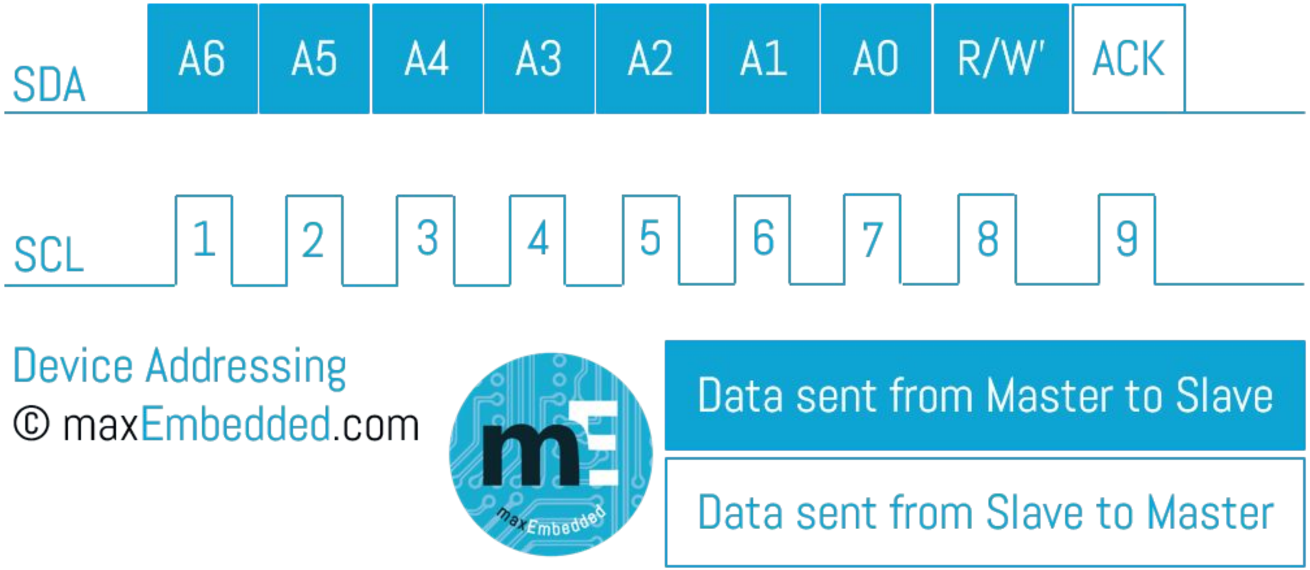
- Bit 7: SPIE - SPI Interrupt Enable - We don’t need no interrupts, so **0**
- Bit 6: SPE - SPI Enable - Yes, so **1**
- Bit 5: DORD - Data Order - **0** when MSB sent & received first
- Bit 4: MSTR - Master/Slave Select - Master, so **1**
- Bit 3: CPOL - Clock Polarity - **0**, clock starts LOW
- Bit 2: CPHA - Clock Phase - **0**, read on posedge
- Bit 1, Bit 0: SPR1, SPR0 - SPI Clock Rate Select - **00**, osc/4 = 16MHz/4 = 4MHz

Yes, we really like games.

I^2C Protocol

- Two-wire protocol, uses only 2 lines, SDA (Serial Data Line) and SCL (Serial Clock Line)
- Synchronous, half-duplex protocol, adheres to clock signal
- Best of both worlds: fewer wires, maintains speed

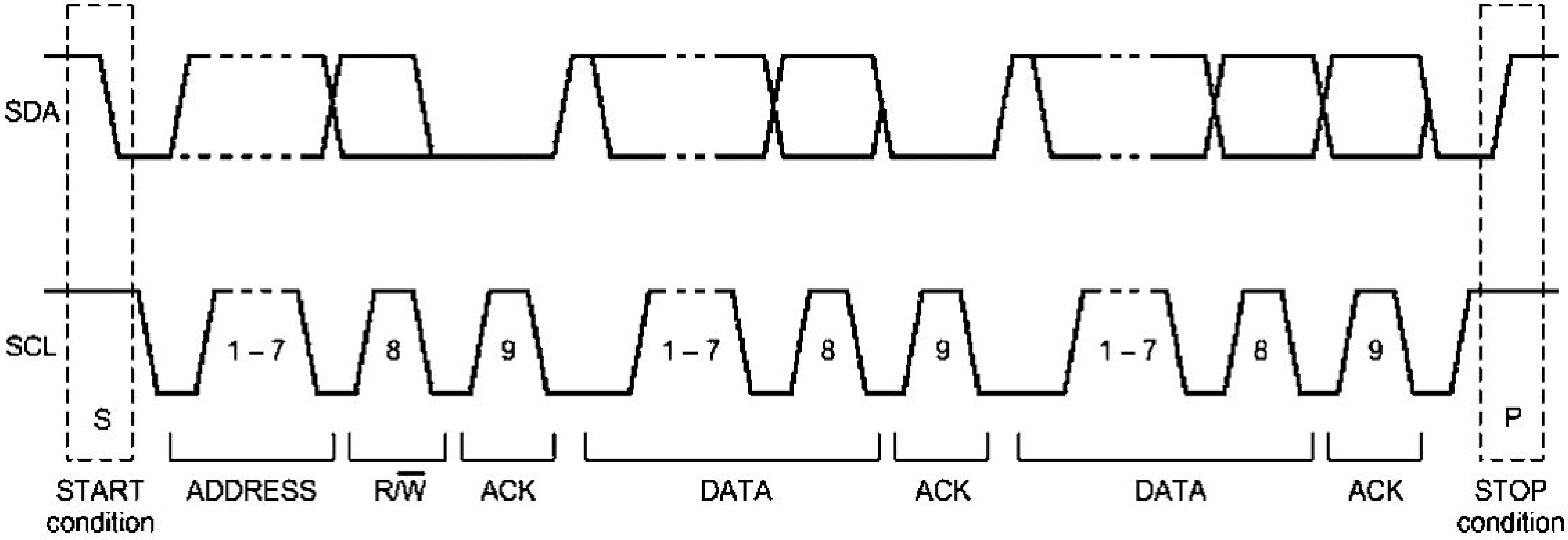
I^{2}C Protocol


SDA
A6
A5
A4
A3
A2
A1
A0
R/W'
ACK
SCL
1
2
3
4
5
6
7
8
9
Data sent from Master to Slave
Data sent from Slave to Master
maxEmbedded

Device Addressing

© maxEmbedded.com

I²C Protocol



Arduino Version

```
// Master writer
#include <Wire.h>

void setup() {
  Wire.begin(); // join i2c bus (address optional for master)
}

byte x = 0;

void loop() {
  Wire.beginTransmission(8); // transmit to device #8
  Wire.write("x is ");        // sends five bytes
  Wire.write(x);              // sends one byte
  Wire.endTransmission();    // stop transmitting

  x++;
  delay(500);
}
```

Arduino Version

```
#include <Wire.h>

void setup() {
  Wire.begin(8);                // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
  Serial.begin(9600);           // start serial for output
}

void loop() {
  delay(100);
}
void receiveEvent(int howMany) {
  while (1 < Wire.available()) { // loop through all but the last
    char c = Wire.read(); // receive byte as a character
    Serial.print(c);         // print the character
  }
  int x = Wire.read();    // receive byte as an integer
  Serial.println(x);         // print the integer
}
```

Today’s Task

- Write your name onto the given LCD display in AVR C
    - No libraries allowed! (except basic AVR libraries)
    - Hint: Use the SPI Protocol
    - Tip: Check the datasheet

https://www.sparkfun.com/datasheets/LCD/HD44780.pdf