**1. a. SPI Communication protocol**

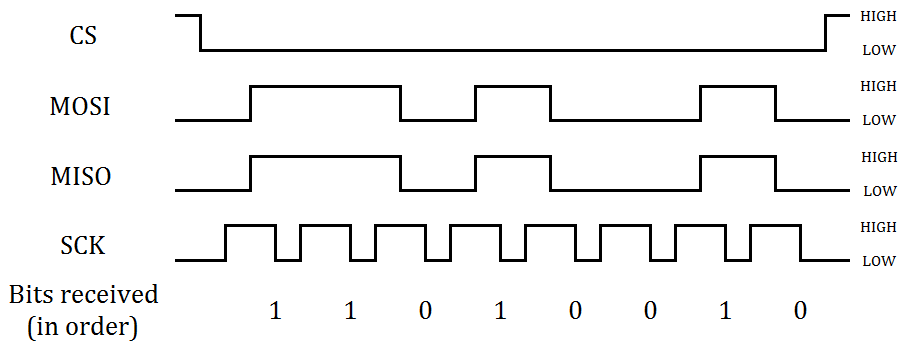


Figure 1: Timing diagram of SPI communication protocol [Source: http://blog.digilentinc.com/wp-content/uploads/2014/09/SPI-signals.png]

The SPI communication protocol is a synchronous communication protocol in which a master device is connected to one or more slave devices by shared MOSI (Master Out Slave In), MISO (Master In Slave Out), and SCK (Serial Clock) lines, and separate CS (chip select) lines.

The chip select line is active low, and is used to select a slave device for communication. It is pulled low by the master before communication. The clock is then generated by the master to trigger edge triggered shift registers in the master and slave. The shift register in the master shifts data out to the slave, while shifting in data from the slave, and vice versa. In Figure 1 (above) the shift registers are triggered on the falling edge of the clock signal. Once communication is complete, the master pulls the chip select line high again. This sequence is illustrated in Figure 1.

**1. b. Interrupts/threaded call-back**

An interrupt is a signal to the processor generated by hardware or software which alerts the processor of a high-priority event that requires immediate handling (by suspending other in-progress tasks). The processor responds by saving its current state and running a specific interrupt handler function, before returning to normal operation.

A threaded call-back is a function that is called in a new thread when a specific event occurs. The function runs immediately, in parallel with the main thread.

**1. c. ADC reading to voltage**

def adc\_to\_voltage(adc\_reading):

return round(3.3\*(adc\_reading/1023), 1)

**1. d. ADC reading to temperature**

def adc\_to\_temperature(adc\_reading):

return int((adc\_reading - 155)/3.1)

**1. e. ADC reading to light level**

def adc\_to\_light(adc\_reading):

light\_low\_calibration = 230 # ADC reading when finger over LDR

light\_high\_calibration = 780 # ADC reading when torch shining at LDR

light\_value = ((adc\_reading - light\_low\_calibration)/(light\_high\_calibration - light\_low\_calibration))\*100

if (light\_value > 100):

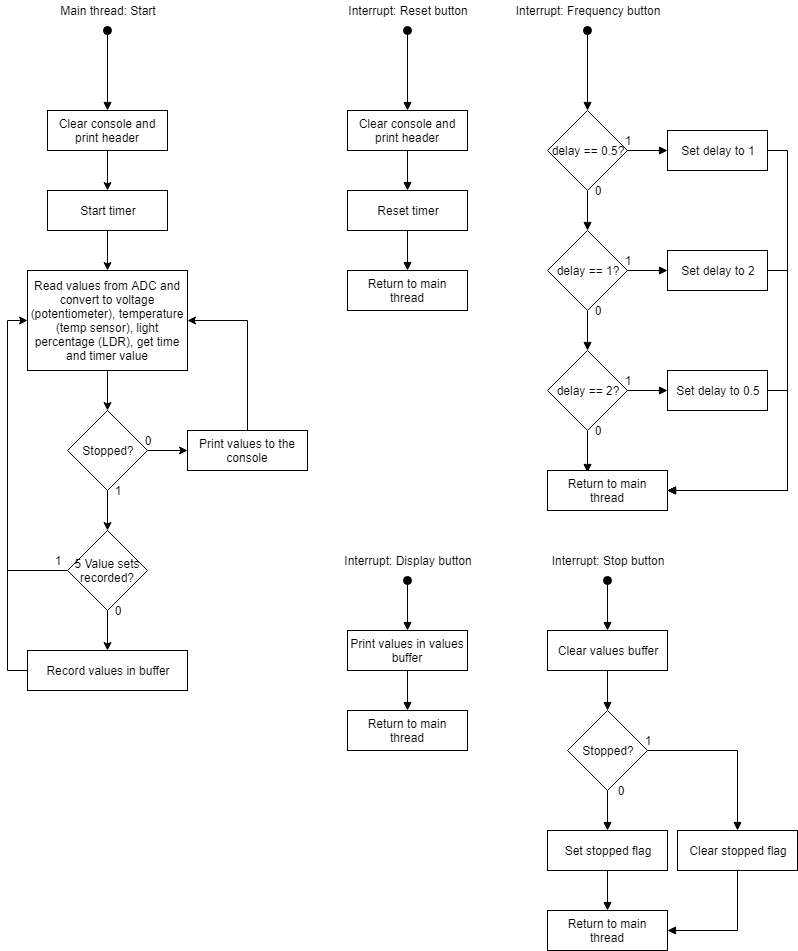
light\_value = 100

elif (light\_value < 0):

light\_value = 0

return light\_value

**1. f. Flow chart of system**



**2. Demonstration**

Functionality of the program was demonstrated to the tutors during the practical session on 18/09/2018.