ATmega328P - Twin Wire Interface

Narendiran S

July 27, 2020

1 Features

- 7-bit address space allows up to 128 different slave addresses
- Multi-master arbitration support
- Up to 400kHz data transfer speed
- Noise suppression circuitry rejects spikes on bus lines
- Fully programmable slave address with general call support
- Compatible with Phillips I2C

2 2-wire Serial Interface

- Suited for typical microcontroller applications
- Allows upto 128 different device.
- All devices connected must have individual address and method to resolved bus contention

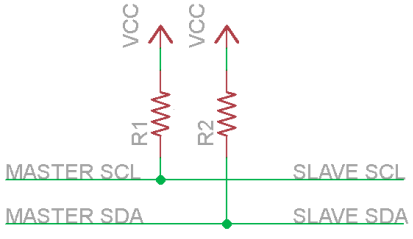
2.1 I^2C Pins

- Output driver consist of slew-rate limiter to confirm the TWI specification.
- Input stage consist of spike suppression unit to remove spikes shorter than 50ns.
- Internal pull-up can also be used.
- *SDA* - Serial Data - the actual serial data transfer pinFormat
- *SCL* - Serial Clock - driven by device in Master Mode

2.2 Terminology

Term	Descripton
Master	Device that initiates and terminates transacting and also Generates *SCL* Clock.
Slave	Device addressed by a master.
Transmitter	Device placing data on the bus.
Receiver	Device reading data on the bus.

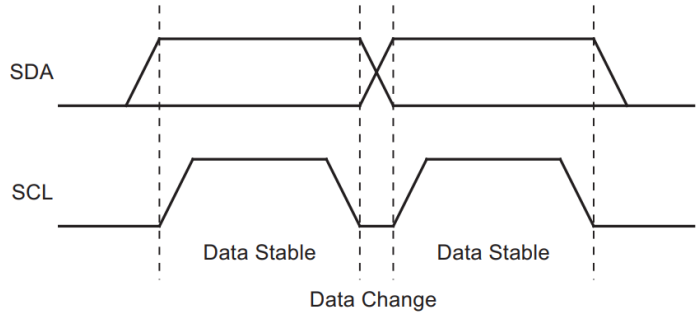
2.3 Electrical Interconnection



- both lines are connected to positive supply voltage through pull-up resistor
- bus driver are open-drain or open-collector
- no. of device also depends on the bus capacitance limit of 400pF

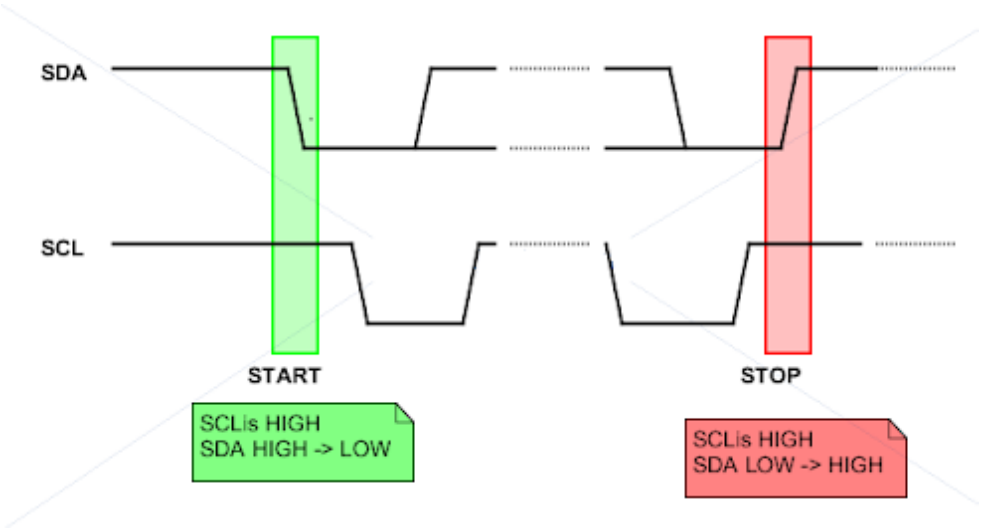
3 Data Transfer and Frame Format

3.1 Transferring Bits



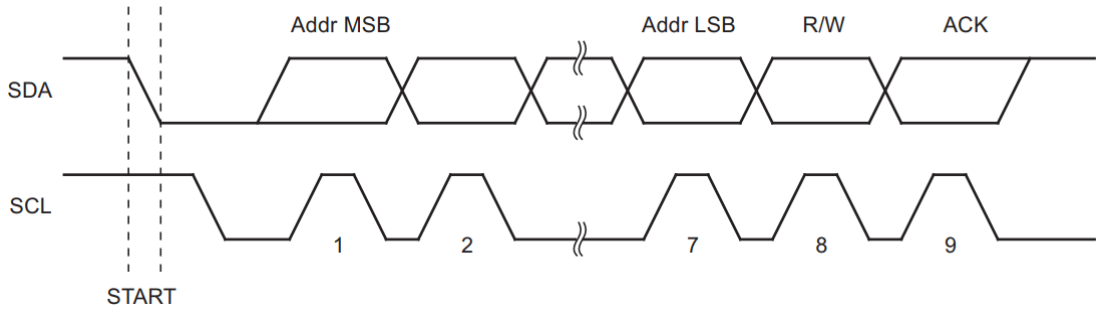
- Each data bit transferred is done by a pulse on clock line.
- Level of data line must be stable when the clock line is high.

3.2 START and STOP Conditions



- Both **START** and **STOP** conditions are done by changing *SDA* line when *SCL* is kept high.
- Master initiates transmission by issuing a **START** condition.
- Master terminates transmission by issuing a **STOP** condition.
- Between **START** and **STOP**, bus is busy and no other master should try to control bus.
- The same master however can issue **REPEATED START**(same as **START**) to initiate a new transfer.

3.3 Address Packet format



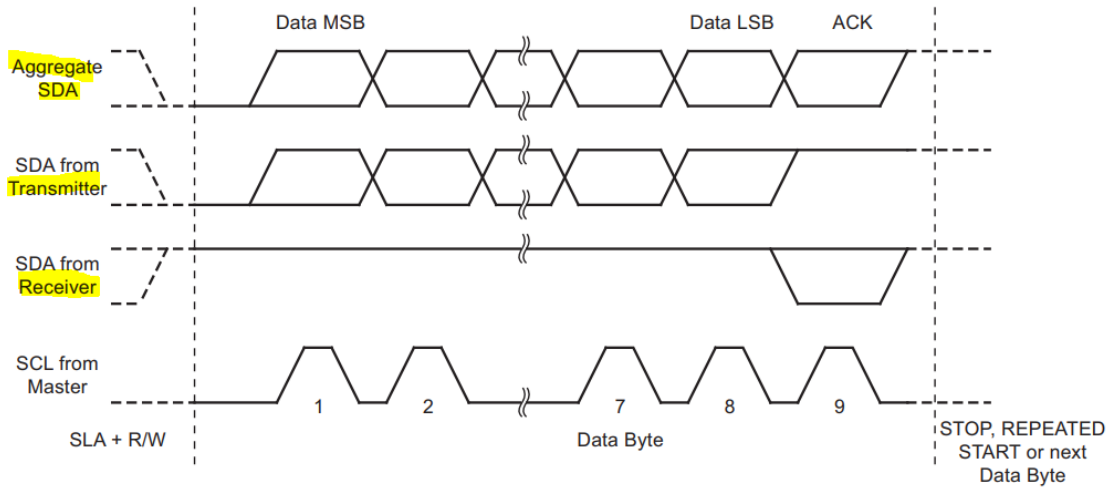
- Addresses packets(**SLA**) are 9-bit long.

- 7 address bits with MSB transmitted first
- one READ(1)/WRITE(0) control bit indicating the transmitter or receiver mode respectively
- one acknowledge bit by the Slave

- Would take 8 clock cycles by the master to send 7 address bits and one READ/WRITE control bit. **SLA+R** or **SLA+W**.
- On the 9th clock cycle, Master will leave out the control of *SDA* line (making it high due to pull-up resistor) but clocks out the 9th clock on the *SCL* line.
- On the 9th clock cycle, Slave recognizes that it is being addressed by pulling *SDA* line low making the **ACK**.
- If the Slave couldn't for some reason respond, the *SDA* line remains high in the 9th clock cycle making the **NACK**.

3.4 Data Packet format

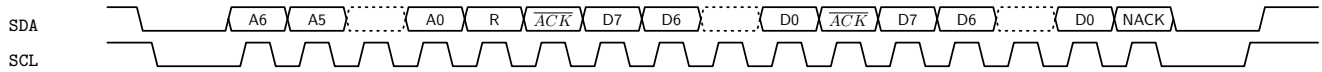
- Data packetsare 9-bit long.
  - one data byte - 8 bits with MSB first.
  - one acknowledge bit by the master or slave depending the mode.
- The transmitter(either the master or slave) send 8-bit data in 8 clock cycles.



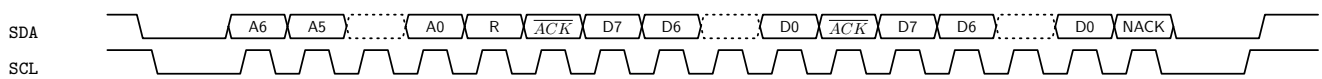
- On the 9th clock cycle, the transmitter will leave out the control of *SDA* line (making it high due to pull-up resistor).
- During the 9th clock cycle, the receiver pulls down the *SDA* line low to acknowledge the reception. – **ACK** is signaled by receiver.
- If the receiver doesn't pull down the *SDA* line for some reason, then the *SDA* line remains high. – **NACK** is signaled by receiver.
- When the receiver received the last byte or can't receive more byte, it should inform the transmitter by sending a **NACK**.

3.5 Overall Operation

Write Operation

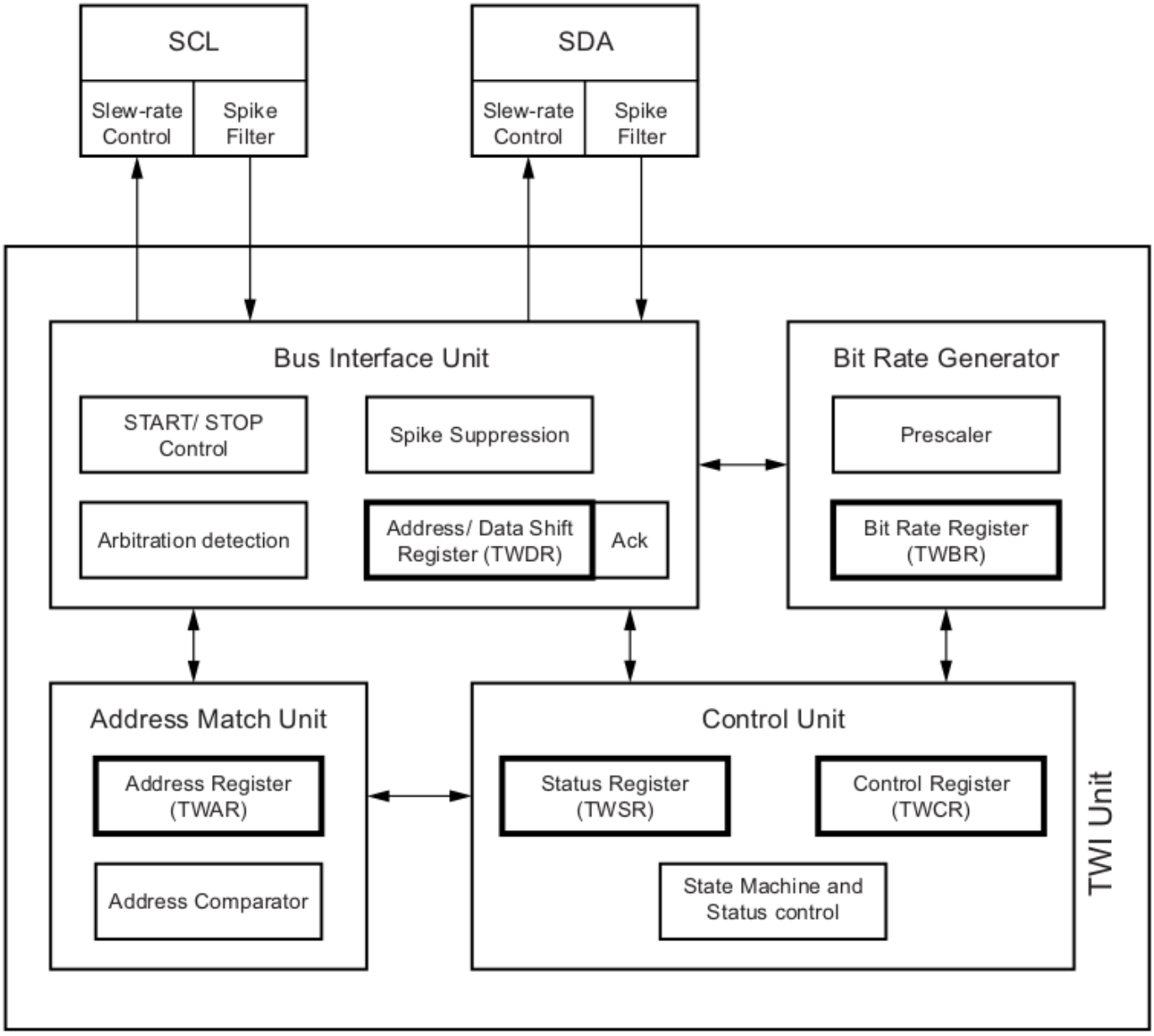


Read Operation



4 TWI Module

4.1 Block Diagram



4.2 Bit Rate Generation Unit

- controls *SCL* line when in Master mode
- ***TWBR*** (TWI Bit Rate Generator) and Prescalar Bits in TWI status register **TWSR** control *SCL*
- The *SCL* frequency can be

$$SCLfrequency = \frac{CPUClockFrequency}{16+2*TWBR*PrescalarValue}$$

Note: Slave's clock frequency must be atleast 16 times higher than SCL frequency.

4.3 Bus Interface Unit

- contains Data and adress Shift register - **TWDR** register - address or data transmitted or received
- **START/STOP** Controller - Generates and detects **START**, **STOP** and **REPEATED START**
- Register Containing the **(N)ACK** to be transmitted or received
- Arbitration detection hardware.

4.4 Address Match Unit

- Received address bytes matches the seven-bit address in **TWAR** (TWI Address register).
- address match results in informing the control unit

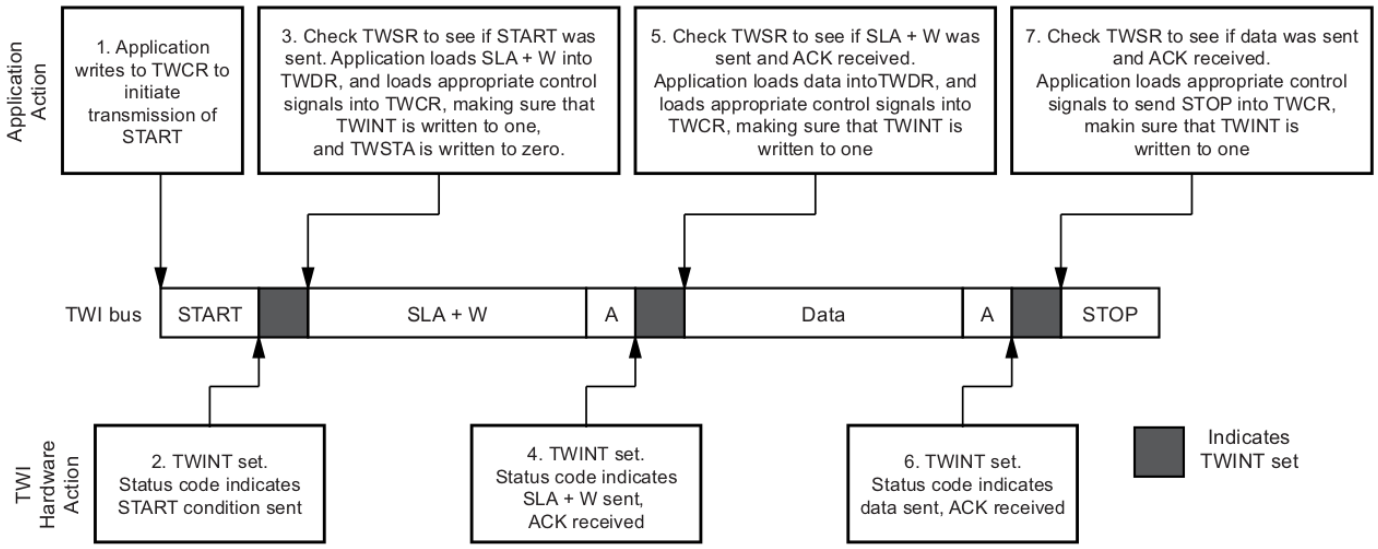
4.5 Control Unit

- Monitors TWI bus and generates responses based on TWI control register (**TWCR**).
- When a en even requires attention:
  - ***TWINT*** (TWI Interrupt flag) is set
  - **TWSR** (TWI Status Register) is updated with status code identifying the event.
  - when the ***TWINT*** is set, the *SCL* line is held low.
- ***TWINT*** flag is set If
  - TWI has transmitted a **START/REPEATED START** condition
  - TWI has transmitted **SLA+R/W**
  - TWI has transmitted an address byte
  - TWI has lost arbitration
  - TWI has been addressed by own slave address or general call
  - TWI has received a data bye
  - **STOP** or **REPEATED START** has been received
  - bus error occurred due to illegal **START** or **STOP**

5 TWI Usage

- TWI is interrupt based and so ***TWIE*** bit in **TWCR** register should be enabled; If the ***TWIE*** is diabled, then the ***TWINT*** flag must be polled.
- When the ***TWINT*** flag is asserted, TWI has finised operation and awaits for application response and **TWSR** register describes the current status of TWI bus.
- Then, application should respond by manipulating the **TWCR** and **TWDR** register.

5.1 An Example - Master transmits single data byte to slave



1. Transmission is started by writing specific value into **TWCR** register to transmit the **START** condition. The ***TWINT*** flag is cleared by writing Loging HIGH which initiate the transmission of **START** condition.
2. When the **START** condition has been transmitted, the TWINT flag in TWCR is set, and **TWSR** is updated with a status code indicating that the **START** condition has successfully been sent.

3. The application should now respond by examining the **TWSR** register value. If status code is as expected, the application loads **SLA+W** into **TWDR** and a specific value is written into **TWCR** register to transmit the **SLA+W** present in **TWDR**. The ***TWINT*** flag is cleared by writing logic HIGH which initiates the transmission of address packet.

4. When the address packet has been transmitted, the ***TWINT*** flag in **TWCR** is set, and **TWSR** is updated with a status code indicating that the address packet has successfully been sent. The status code will also reflect whether a slave acknowledged the packet or not.

5. The application should now respond by examining the **TWSR** register value and **ACK** bit is as expected. If status code is as expected, the application loads Data packet into **TWDR** and a specific value is written into **TWCR** register to transmit the Data packet present in **TWDR**. The ***TWINT*** flag is cleared by writing logic HIGH which initiates the transmission of data packet.

6. When the data packet has been transmitted, the ***TWINT*** flag in **TWCR** is set, and **TWSR** is updated with a status code indicating that the data packet has successfully been sent. The status code will also reflect whether a slave acknowledged the packet or not.

7. The application should now respond by examining the **TWSR** register value and **ACK** bit is as expected. If status code is as expected, the application loads a specific value is written into **TWCR** register to transmit the **STOP** Condition. The ***TWINT*** flag is cleared by writing logic HIGH which initiates the transmission of STOP condition.

6 Transmission Modes

There are four major Modes

(i) Master Transmitter (MT)

(ii) Master Receiver (MR)

(iii) Slave Transmitter (ST)

(iv) Slave Receiver (SR)

Status Code	Meaning
S	**START** Condition
Rs	**REPEATED START** Condition
R	Read bit (high level on *SDA*)
W	Write bit (low level on *SDA*)
A	Acknowledge bit (low level on *SDA*)
$\overline{A}$	Not Acknowledge bit (high level on *SDA*)
DATA	8-bit data
P	**STOP** Condition
SLA	Slave Address

6.1 Master Transmitter Mode (MT)

- Many number of data bytes are transmitted to Slave receiver.
- For Master, **START** Condition is transmitted
- **START** condition is sent by:
  - ***TWEN*** bit is set to enable TWI.
  - ***TWSTA*** bit is set to transmit **START** condition.
  - ***TWINT*** flag is written 1 to clear to send start bit.
  - After Tranmitting **START** condition, ***TWINT*** flag is set by hardware and status code in **TWSR** register should be 0x08 - indicating successfull transmission of **START** condition.
- To enter into Master Transmitter Mode and transmit the address:
  - Write **SLA+W** into **TWDR** register.
  - ***TWINT*** flag is written 1 to clear to transmit Address and read/write status.

- After transmitting SLA+W, an acknowledgment bit will be received, the *TWINT* flag is set by hardware and status code in **TWSR** register will be 0x18 (indicating SLA+W has been transmitted and ACK has been received), 0x20 (indicating SLA+W has been transmitted and NACK has been received), 0x38 (Arbitration lose in sending SLA+W).

- Data packet is transmitted by:
  - Write DATA packet into **TWDR** register.
  - *TWINT* flag is written 1 to clear to transmit Address and read/write status.
  - After transmitting DATA packet, an acknowledgment bit will be received, the *TWINT* flag is set by hardware and status code in **TWSR** register will be 0x28 (indicating DATA packet has been transmitted and ACK has been received), 0x30 (indicating DATA packet has been transmitted and NACK has been received)
  - To send further data, the above process is repeated by sending REPEATED START.
  - To stop the transmission, the STOP condition is sent.

- STOP condition is sent by:
  - *TWSTO* bit is set to transmit STOP condition.
  - *TWINT* flag is written 1 to clear to send stop bit.

- REPEATED START condition is sent by:
  - *TWSTA* bit is set to transmit REPEATED START condition.
  - *TWINT* flag is written 1 to clear to send repated start bit.
  - After Tranmitting REPEATED START condition, *TWINT* flag is set by hardware and status code in **TWSR** register should be 0x10 - indicating successfull transmission of REPEATED START condition.

6.2 Master Receiver Mode (MR)

- Many number of data bytes can be received from Slave transmitter.
- For Master, START Condition is transmitted
- START condition is sent by:
  - *TWEN* bit is set to enable TWI.
  - *TWSTA* bit is set to transmit START condition.
  - *TWINT* flag is written 1 to clear to send start bit.
  - After Tranmitting START condition, *TWINT* flag is set by hardware and status code in **TWSR** register should be 0x08 - indicating successfull transmission of START condition.
- To enter into Master Receiver Mode and transmit the address:
  - Write SLA+R into **TWDR** register.
  - *TWINT* flag is written 1 to clear to transmit Address and read/write status.
  - After transmitting SLA+R, an acknowledgment bit will be received, the *TWINT* flag is set by hardware and status code in **TWSR** register will be 0x40 (indicating SLA+R has been transmitted and ACK has been received), 0x48 (indicating SLA+R has been transmitted and NACK has been received), 0x38 (Arbitration lose in sending SLA+R).
- Data packet is received by:
  - Reading the DATA packet from **TWDR** register if *TWINT* flag is logic HIGH.
  - *TWINT* flag is written 1 to clear.
  - After receiving DATA packet, an acknowledgment bit will be returned, the *TWINT* flag is set by hardware and status code in **TWSR** register will be 0x58 (indicating DATA packet has been recieved and ACK has been returned), 0x50 (indicating DATA packet has been recieved and NACK has been returened)
  - To receive further data, the above process is repeated by sending REPEATED START.
  - To stop the reception, the STOP condition is sent.
- STOP condition is sent by:

  - ***TWSTO*** bit is set to transmit **STOP** condition.
  - ***TWINT*** flag is written 1 to clear to send stop bit.
- **REPEATED START** condition is sent by:
  - ***TWSTA*** bit is set to transmit **REPEATED START** condition.
  - ***TWINT*** flag is written 1 to clear to send repated start bit.
  - After Tranmitting **REPEATED START** condition, ***TWINT*** flag is set by hardware and status code in **TWSR** register should be 0x10 - indicating successfull transmission of **REPEATED START** condition.

6.3 Slave Receiver Mode (SR)

- Many number of data bytes are received from Master transmitter.
- To initiate the Slave Mode:
  - ***TWA[5:0]*** bits from **TWAR** regFormat is loaded with our slave address.
  - ***TWSTA*** and ***TWSTO*** are set to 0
  - ***TWEN*** bit is set to enable the TWI.
- TWI waits until it is addressed by its own slave address followed by data direction bit.
- After receiving own Slave address and Write bit, the ***TWINT*** flag is set and valid status code is available in **TWSR** register.
- If the status code is 0x60 (Own **SLA+W** has been recieved and **ACK** has been returned)
- Now, data can be read by wiring Logic HIGH on ***TWINT*** flag to clear and read from **TWDR** register.
- ***TWEA*** bit is set to acknowledge and recieve further data or ***TWEA*** bit is cleared and last byte is received.
- Now, ***TWINT*** flag is set and status code is avalaible in **TWSR**.
- If status code is 0x80 (Previously addressed with own **SLA+W** and data has been received and **ACK** has been returned) - to receive further data.
- If status code is 0x88 (Previously addressed with own **SLA+W** and data has been received and **NACK** has been returned) - last data is recieved.
- If status code is 0xA0 - A **STOP** condition is recieved. has been received

6.4 Slave Transmitter Mode (ST)

- Many number of data bytes are transmitted to Master recive.
- To initiate the Slave Mode:
  - ***TWA[5:0]*** bits from **TWAR** regFormat is loaded with our slave address.
  - ***TWSTA*** and ***TWSTO*** are set to 0
  - ***TWEN*** bit is set to enable the TWI.
- TWI waits until it is addressed by its own slave address followed by data direction bit.
- After receiving own Slave address and Read bit, the ***TWINT*** flag is set and valid status code is available in **TWSR** register.
- If the status code is 0xA8 (Own **SLA+R** has been recieved and **ACK** has been returned)
- Now, data to be sent is set on the **TWDR** register.
- ***TWINT*** flag is written 1 to clear to transmit the data.
- Now, ***TWINT*** flag is set and status code is avalaible in **TWSR**.
- If status code is 0xB8 -(Data byte in **TWDR** has been transmitted and **ACK** has been received) - can send further data.
- If status code is 0xC8 -(Data byte in **TWDR** has been transmitted and **NACK** has been received) - last byte send and dont' send further.