Electronics
Bit-banging (or bit-bashing)

Terry Sturtevant

Wilfrid Laurier University

February 13, 2019

Problems

Problems

- What do you do if you want *3* SPI devices with the Raspberry Pi?

Problems

- What do you do if you want *3* SPI devices with the Raspberry Pi?
- What do you do if you want *3* PWM devices with the Raspberry Pi?

Problems

- What do you do if you want *3* SPI devices with the Raspberry Pi?
- What do you do if you want *3* PWM devices with the Raspberry Pi?
- What do you do if you want a UART sensor *and* the serial console with the Raspberry Pi?

Problems

- What do you do if you want *3* SPI devices with the Raspberry Pi?
- What do you do if you want *3* PWM devices with the Raspberry Pi?
- What do you do if you want a UART sensor *and* the serial console with the Raspberry Pi?
- What do you do if you have a sensor that has no available library?

Problems

- What do you do if you want *3* SPI devices with the Raspberry Pi?
- What do you do if you want *3* PWM devices with the Raspberry Pi?
- What do you do if you want a UART sensor *and* the serial console with the Raspberry Pi?
- What do you do if you have a sensor that has no available library?

Solution: Bit-bang more interfaces.

What’s the difference between multiple signals and an interface?

What's the difference between multiple signals and an interface?

- If there is a *library* to simplify the communication then it is an interface.

What's the difference between multiple signals and an interface?

- If there is a *library* to simplify the communication then it is an interface.
- If there is no *library* then you have to handle all of the signals yourself.

What's the difference between multiple signals and an interface?

- If there is a *library* to simplify the communication then it is an interface.
- If there is no *library* then you have to handle all of the signals yourself.

  This process of handling all of the signals yourself is often called **bit-bashing** or **bit-banging**.

- Built-in hardware ports allow complex operations to happen without ongoing software intervention.

- Built-in hardware ports allow complex operations to happen without ongoing software intervention.
- *Bit-banging* is the process of writing code to perform the necessary operations manually.

- Built-in hardware ports allow complex operations to happen without ongoing software intervention.
- *Bit-banging* is the process of writing code to perform the necessary operations manually.
- If the code can execute within whatever timing window is required, then it is an acceptable solution.

- Built-in hardware ports allow complex operations to happen without ongoing software intervention.
- *Bit-banging* is the process of writing code to perform the necessary operations manually.
- If the code can execute within whatever timing window is required, then it is an acceptable solution.

Note: Because the Raspberry Pi has an operating system running, tight timing tolerances can't be guaranteed this way.

Tips for bit-banging

Tips for bit-banging

- Use bit-banging for the slowest interfaces.

Tips for bit-banging

- Use bit-banging for the slowest interfaces.
- Use bit-banging for the least frequent tasks.

Tips for bit-banging

- Use bit-banging for the slowest interfaces.
- Use bit-banging for the least frequent tasks.
- **Avoid cumulative timing error by referencing a single event time.**

Tips for bit-banging

- Use bit-banging for the slowest interfaces.
- Use bit-banging for the least frequent tasks.
- **Avoid cumulative timing error by referencing a single event time.**
- Create functions as similar as possible to those that are built-in.

Example: bit-banging a UART (Transmitting)

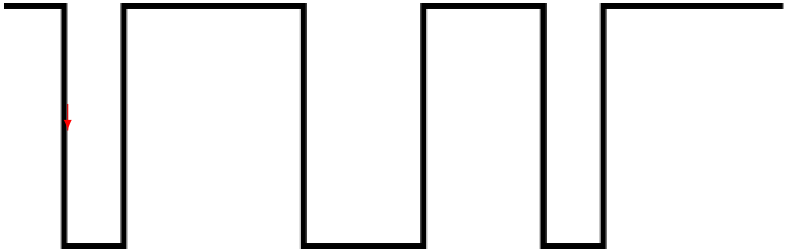
Example: bit-banging a UART (Transmitting)

- When *transmitting*, a UART basically needs to *change* a signal at fixed time intervals.

Example: bit-banging a UART (Transmitting)

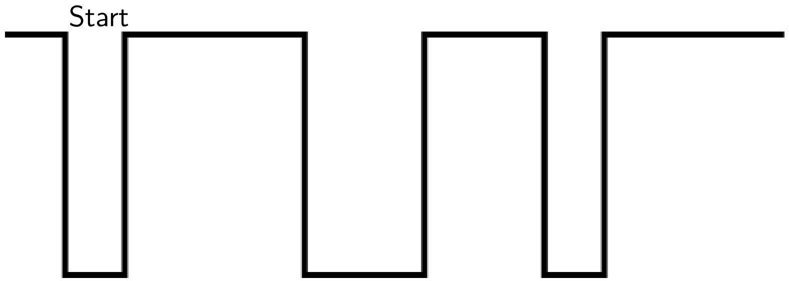
- When *transmitting*, a UART basically needs to *change* a signal at fixed time intervals.
- When *receiving*, after the detection of a START bit, a UART basically needs to *test* a signal at fixed time intervals.

Transmitting



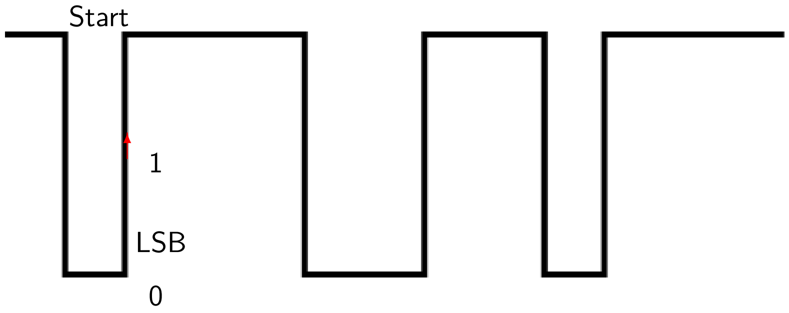
Set pin to START level

Transmitting



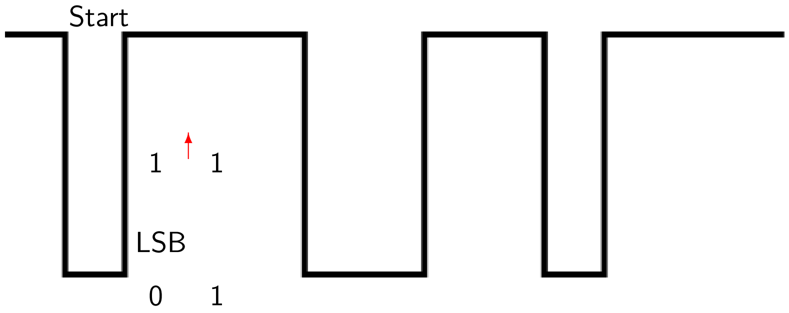
Wait one bit time before setting pin HIGH or LOW according to LSB

Transmitting



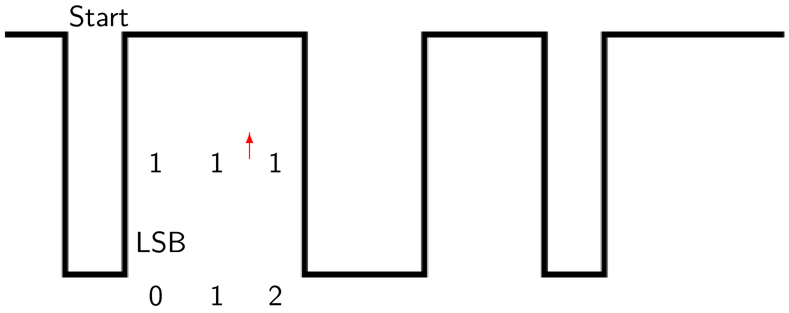
Wait one bit time before setting pin HIGH or LOW according to LSB

Transmitting



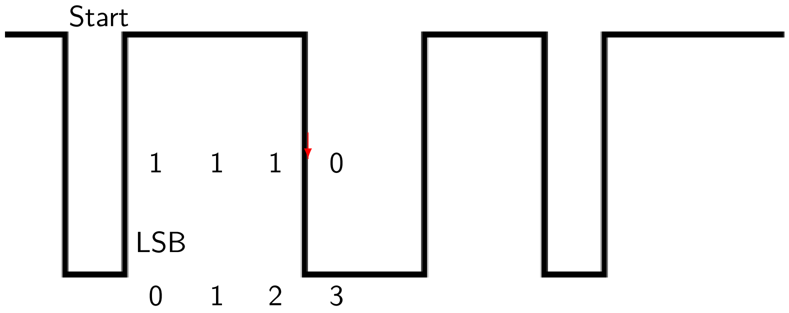
Wait one bit time before setting pin HIGH or LOW according to bit 1

Transmitting



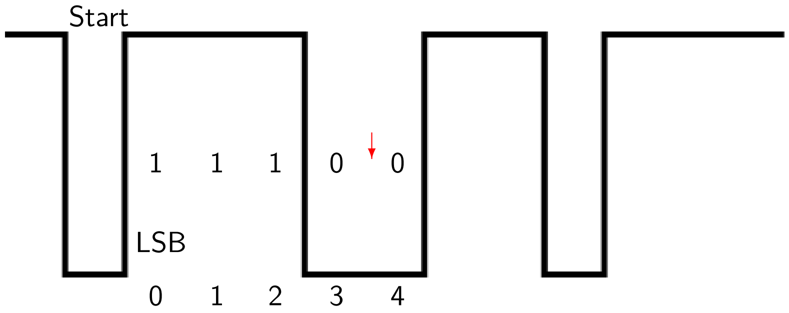
Wait one bit time before setting pin HIGH or LOW according to bit 2

Transmitting



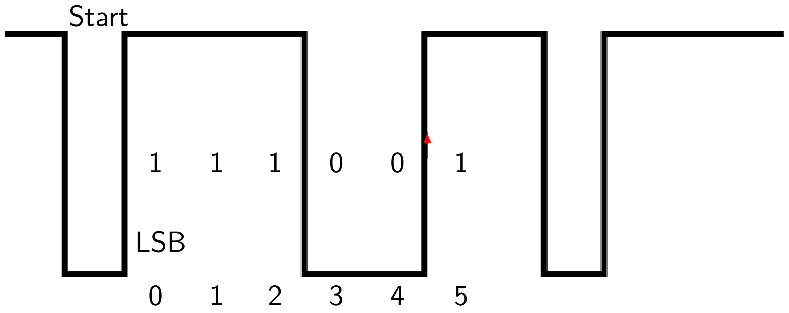
Wait one bit time before setting pin HIGH or LOW according to bit 3

Transmitting



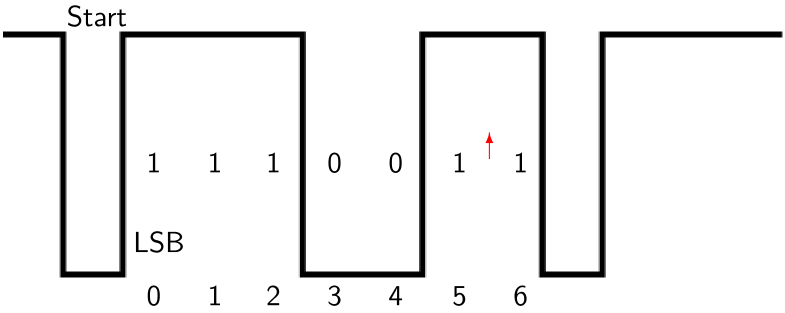
Wait one bit time before setting pin HIGH or LOW according to bit 4

Transmitting



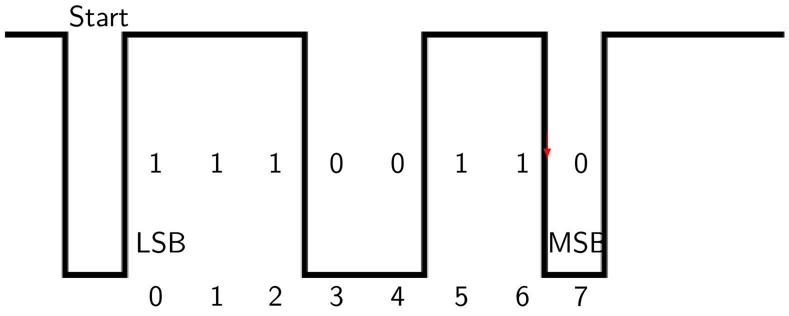
Wait one bit time before setting pin HIGH or LOW according to bit 5

Transmitting



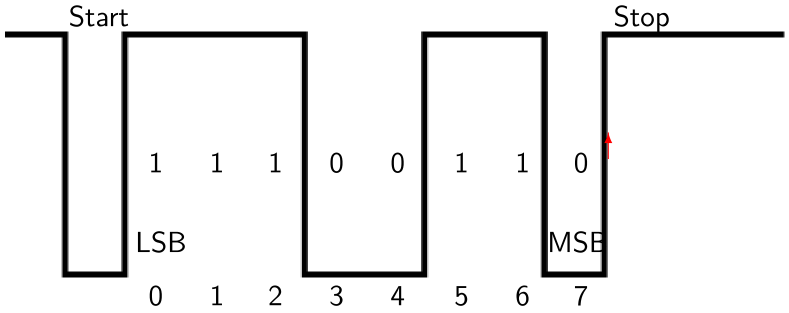
Wait one bit time before setting pin HIGH or LOW according to bit 6

Transmitting



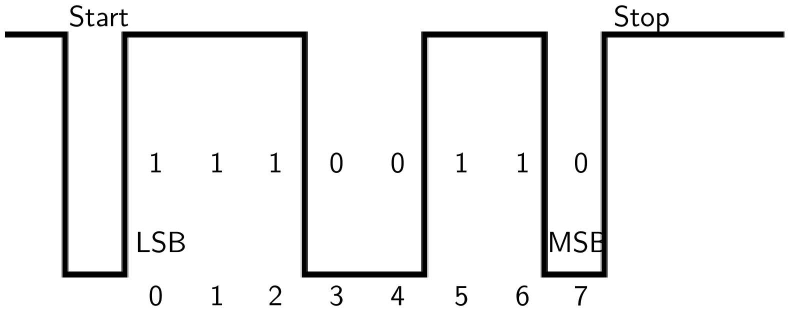
Wait one bit time before setting pin HIGH or LOW according to MSB

Transmitting



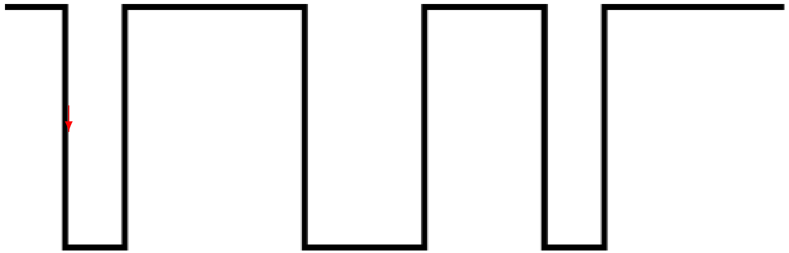
Wait one bit time before setting pin to STOP level

Transmitting



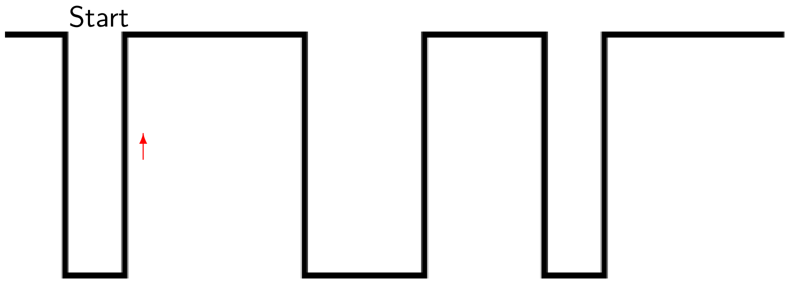
Wait 1 bit time (if 1 STOP bit) before next START bit

Receiving



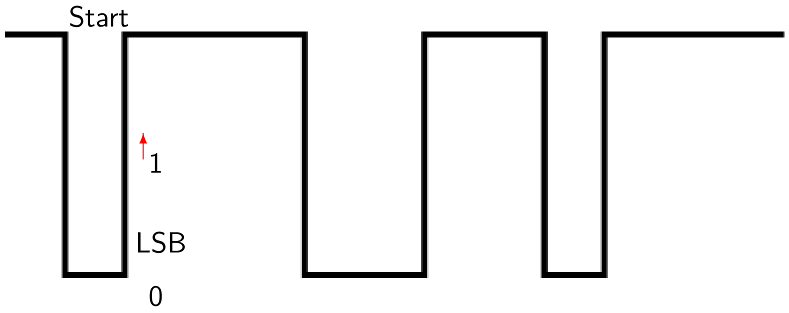
Poll for START level

Receiving



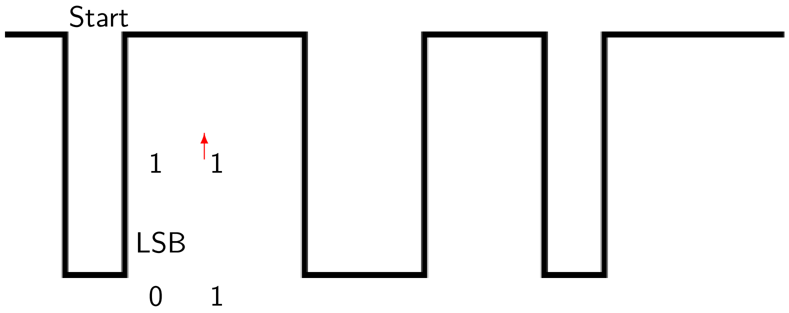
Wait one *and a half* bit times before testing pin for LSB

Receiving



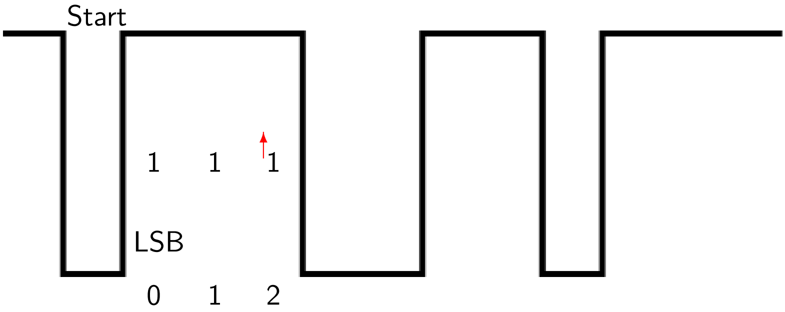
Wait one *and a half* bit times before testing pin for LSB

Receiving



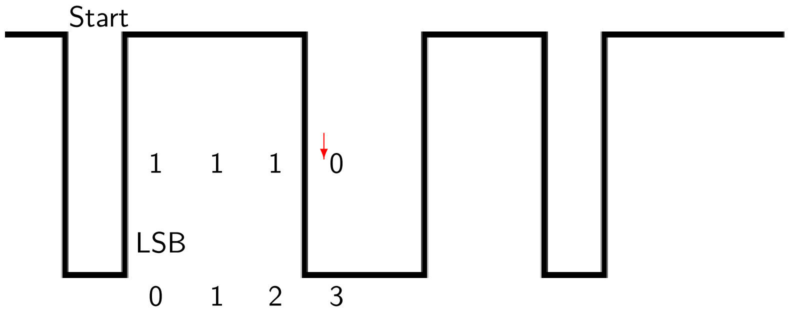
Wait one bit time before testing pin for bit 1

Receiving



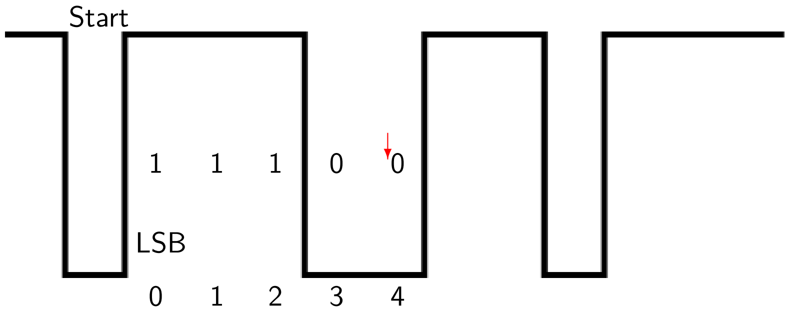
Wait one bit time before testing pin for bit 2

Receiving



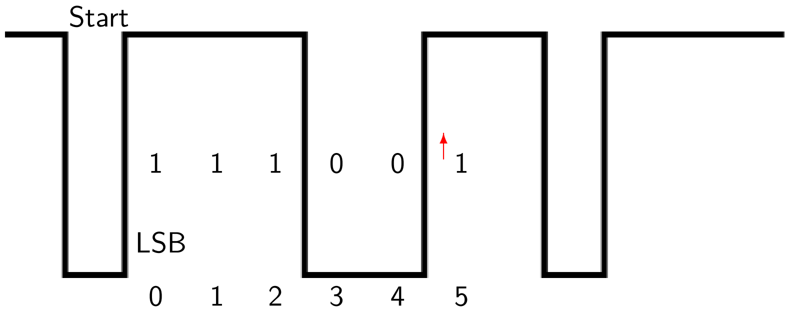
Wait one bit time before testing pin for bit 3

Receiving



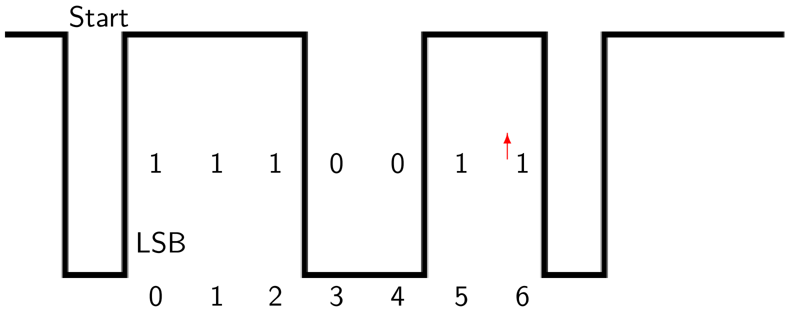
Wait one bit time before testing pin for bit 4

Receiving



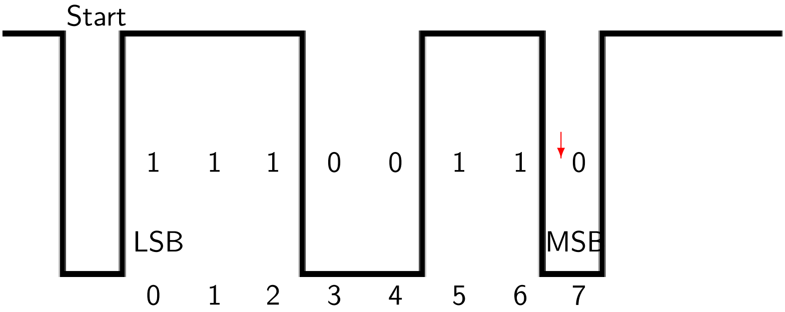
Wait one bit time before testing pin for bit 5

Receiving



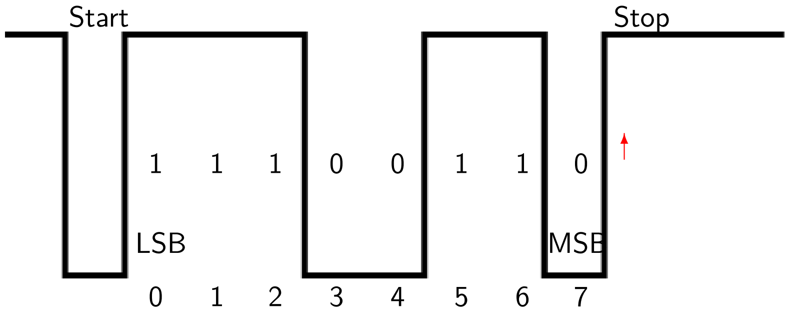
Wait one bit time before testing pin for bit 6

Receiving



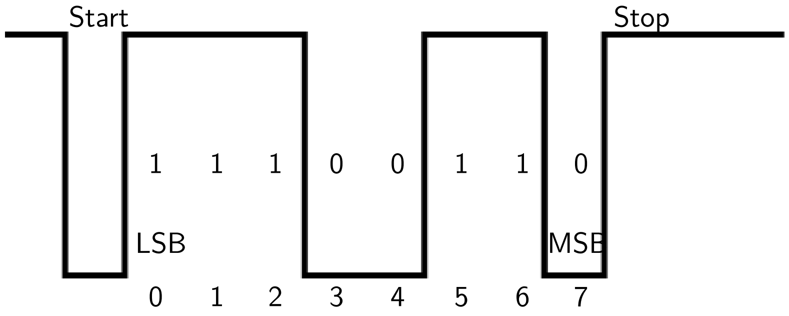
Wait one bit time before testing pin for MSB

Receiving



Wait one bit time before testing pin for STOP level

Receiving



Poll for next START bit