RPi Blinky

Connect the LED to your Windows IoT device You’ll need a few components:

• a LED (any colour you like)

• a 330 Ω resistor (Resistor Colour Code Orange, Orange, Brown, Gold)

• a breadboard

• 2 Male to Female connector wires

Wiring your Raspberry Pi 2

Connect the shorter leg of the LED to GPIO 5 (pin 29 on the expansion header) on the RPi2 or RPi3. 2. Connect the longer leg of the LED to the resistor. 3. Connect the other end of the resistor to one of the 3.3V pins on the RPi2 or RPi3. 4. Note that the polarity of the LED is important. (This configuration is commonly known as Active Low).

The code for this sample is quite simple. We use a timer, and each time the 'Tick' event is called, in this case ‘Ticking” every 500milli Seconds we flip the state of the LED from Low to High, and back again. Remember that this circuit drives the LED by switching on and off the LOW or Ground rail to the device. The LED is powered from the 3.3v supply constantly.

Timer Code Here is how you set up the timer in C#:

public MainPage() { // ...

timer = new DispatcherTimer(); timer.Interval = TimeSpan.FromMilliseconds(500); timer.Tick += Timer\_Tick; InitGPIO(); if (pin != null) { timer.Start(); } // ... } private void Timer\_Tick(object sender, object e) { if (pinValue == GpioPinValue.High) { pinValue = GpioPinValue.Low; pin.Write(pinValue); LED.Fill = redBrush; } else { pinValue = GpioPinValue.High; pin.Write(pinValue); LED.Fill = grayBrush; } }

Initialize the GPIO Pin To drive the GPIO pin, first we need to initialize it. Here is the C# code

Let's break this down a little

• First, we use GpioController.GetDefault() to get the GPIO controller.

• If the device does not have a GPIO controller, this function will return null.

• Then we attempt to open the pin by calling GpioController.OpenPin() with the LED\_PIN value.

• Once we have the pin, we set it to be off (High) by default using the GpioPin.Write() function.

• We also set the pin to run in output mode using the GpioPin.SetDriveMode() function.

using Windows.Devices.Gpio;

private void InitGPIO() { var gpio = GpioController.GetDefault();

// Show an error if there is no GPIO controller if (gpio == null) { pin = null; GpioStatus.Text = "There is no GPIO controller on this device."; return; }

pin = gpio.OpenPin(LED\_PIN); pinValue = GpioPinValue.High; pin.Write(pinValue); pin.SetDriveMode(GpioPinDriveMode.Output);

GpioStatus.Text = "GPIO pin initialized correctly."; }

Modify the state of the GPIO pin Once we have access to the GpioOutputPin instance, it's trivial to change the state of the pin to turn the LED on or off.

To turn the LED on, simply write the value GpioPinValue.Low to the pin:

pin.Write(GpioPinValue.Low);

and of course, write GpioPinValue.High to turn the LED off:

pin.Write(GpioPinValue.High);

Remember that the other end of the LED is attached to the 3.3 Volts power supply, so we need to drive the pin to a low state to have current flow through the LED, and make it glow.