Miniproyecto

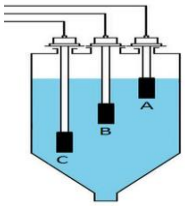
Elige una de las prácticas propuestas a continuación, y realiza la memoria con los puntos que se indican abajo:

1. Crea un canal en ThingSpeak que reciba los datos de un sensor conectado a un NodeMCU o Arduino en uno de sus campos y desarrolla un sistema de notificaciones que te avise cuando los datos cumplen una condición determinada mediante un tweet o un email. (Daniel)

2. Implementar un sistema de control de riego para plantas de interior con una placa NodeMCU, un sensor DHT11 o DHT22 para temperatura y humedad ambiente, y un sensor de humedad de suelo (disponible por unos 2 euros), enterrado en el sustrato de la planta (si no se dispone del mismo, se simulará con un potenciómetro). El sistema enviará los tres datos (temperatura y humedad ambiente, y humedad del suelo) a ThingSpeak. Se activará una salida con un LED cuando se considere que se deba activar una bomba de riego, en función de las medidas. Como ampliación, informará al usuario cuando se genere alguna alarma (temperaturas o humedad ambiente muy altas o bajas, humedad del terreno baja), para saber cuándo se han de regar las plantas.

3. Utilizando NodeMCU y la combinación Arduino UNO R3+ESP01 (o bien dos placas NodeMCU), realiza un sistema en el que se lea el dato de un sensor (luz, temperatura...) en una de ellas, y en otra, en función del estado de la primera, encienda o apague un LED cuando se supere un umbral. Para ello, una de las placas enviará los datos a ThingSpeak y la otra leerá el dato publicado para actualizar el estado del LED. Se recomienda conectar Arduino UNO R3 al sensor, pues cuenta con más entradas analógicas, lo que hará que el sistema pueda expandirse, sirviendo como base para un futuro sistema de medida. Como ampliación, la placa receptora enviará el dato por USB, pudiendo enviar máximos, mínimos...

4. Realizar un sistema para monitorizar el nivel de líquido de un depósito (agua para riego de plantas o mascota). El sistema estará formado por un depósito con tres sensores sumergibles. Los sensores se simularán mediante tres interruptores A, B y C situados según indica la figura y conectados a una placa NodeMCU. Los sensores enviarán un nivel Alto cuando estén sumergidos y Bajo en caso contrario. Realizar el programa en ArduinoBlocks que envíe mediante ThingSpeak el nivel de líquido del depósito cada 60 segundos y un email o Tweet de alerta cuando el nivel sea del 0%. Condiciones:
   A=B=C=ON->100%, A=OFF,B=ON,C=ON->75%, A=OFF,B=OFF,C=ON->25% A=B=C=OFF->0%.



5. Implementar un sistema de control de turno basado en una placa NodeMCU con tres pulsadores (Incrementar, Decrementar y Reset) y cuatro leds. Estos tres pulsadores controlarán el valor de una variable "turno" que será mostrada en binario de cuatro bits en los cuatro leds y publicada a través de ThingSpeak, Adafruit IO o Linear MQTT Dashboard. Limitar el valor de la variable de 0 a 9 y volver a comenzar la cuenta desde cero cuando se pase de 9. Para ampliar: Cambiar los leds por un display lcd-i2c conectado a la placa NodeMCU y ampliar la cuenta de 000 a 999.

6. Los padres de un joven adolescente quieren saber a qué hora llega su hijo/a a casa los fines de semana por la noche ya que en muchas ocasiones se acuestan y no llegan a saberlo. Para ello están pensando utilizar las aplicaciones IoT. De tal manera que una vez que se levantan puedan mirar el móvil o pc y recibir esa información de algunas de las plataformas IoT que hemos visto en el curso.
   Para realizar este proyecto se sabe que:

   - Los jóvenes tienen que pasar obligatoriamente por el recibidor o hall de la casa.
   - La información que reciben los padres es cuando se ha detectado que alguien ha pasado por el hall.

   Opcionalmente:

   - El dispositivo solo funciona entre la 1 y las 7 de la madrugada.

7. Se desea controlar el paso a un garaje a distancia. Para ello hay que tener en cuenta las siguientes condiciones:
   a. la apertura y cierre de la puerta del garaje se realizará con un servomotor que gire entre 0º (puerta cerrada) y 90º (puerta completamente abierta).
   b. El movimiento del servomotor se realizará a distancia, desde un teléfono móvil con la app Linear MQTT Dashboard, donde se tenga un panel con un slider para indicar el ángulo de giro del servomotor.
   c. También se desea saber cuántos vehículos hay en el garaje en cualquier momento; para ello habrá dos detectores de obstáculos IR (que se pueden simular con dos pulsadores en su defecto), conectados a la NodeMCU, uno en cada vial de la puerta (uno para entrar y otro para salir). El número de vehículos estacionados a tiempo real aparecerá en la app Linear MQTT Dashboard.
   d. Cada vez que entre un vehículo al garaje, se activa el detector de entrada, y si un tiempo después se deja de activar el detector (activación por flanco descendente), **siempre y cuando esté la puerta abierta**, entendemos que el vehículo ha entrado y aumentará en 1 el contador de vehículos.
   e. Al salir un vehículo ocurrirá algo similar. Se activará el sensor de salida, y si se desactiva con posterioridad (activación por flanco descendente), **estando la puerta abierta,** se interpreta como que ha salido un vehículo y se debe actualizar el contador de vehículos.

f. Se considera que la puerta está abierta y permite el paso de vehículos si el servomotor está posicionado en un ángulo superior a 60º.
g. El garaje tiene una capacidad máxima de 10 vehículos (el límite inferior obviamente es cero). El contador de vehículos no puede estar fuera de ese rango.

Para detectar los flancos en los cambios de los sensores se aconseja usar dos variables booleanas para cada sensor (estado_anterior y estado_actual) y comparar sus valores entre sí, ya que los bloques de control "esperar hasta que" o "esperar mientras que" no se pueden usar porque cortarían la comunicación con internet y provocarían un fallo por activación del watchdog.

8. Se desea medir la velocidad de los vehículos que circulan por una carretera limitada a 100 km/h, y activar una cámara fotográfica si un vehículo rebasa el límite de velocidad. Para ello habrá dos sensores de presencia, separados una distancia conocida de 100 metros, para medir la velocidad (se pueden simular con dos sensores IR o dos pulsadores en caso de no disponer de esos sensores). El sistema debe funcionar de la siguiente forma:
   a. Cuando se active y posteriormente se desactive el sensor 1 el sistema se pondrá a contar el tiempo (activación por flanco descendente).
   b. Una vez que se active el sensor 2 (activación por flanco ascendente) se detendrá el tiempo de conteo y se calcula la velocidad del vehículo (espacio / tiempo).
   c. La velocidad se envía al servidor Io de Adafruit para su graficación (crear en el Dashboard un elemento de tipo "line chart").
   d. Además la tarjeta NodeMCU activará un led si la velocidad supera los 100 km/h para avisar de que se ha disparado la cámara. También enviará esta información a Adafruit donde se reflejará en un elemento del dashboard de tipo "indicator" que se pondrá en rojo. Si la velocidad es legal, el led se apagará y en el dashboard de Adafruit el "indicator" se pondrá en verde. El led y el "indicator" del dashboard permanecen en su último estado hasta que vuelva a pasar otro vehículo y vuelva a calcularse la velocidad.
   e. Se considera que la carretera tiene poco tráfico para no saturar el servidor Adafruit, y que no se va a activar el sensor 1 mientras no lo haya hecho antes el sensor 2 con el vehículo anterior (los coches circulan con una separación mínima de 100 mts entre ellos).

Se aconseja usar el cronometro de Arduinoblocks en ms para tener mayor precisión en la velocidad, y no se deben usar los bloques de tiempo "esperar" ni el bloque de control "esperar hasta que" o "esperar mientras que" para evitar que se detenga la comunicación y producir un fallo software por activación del watchdog del ESP8266. Por tanto para detectar los flancos de los sensores se aconseja usar dos variables booleanas para cada sensor (estado_anterior y estado_actual) y comparar sus valores entre sí.

Memoria (Entregar en PDF)

La memoria debe contener los siguientes puntos:
1. Propuesta del miniproyecto.
2. Materiales utilizados.
3. Explicación del funcionamiento.
4. Programa realizado con ArduinoBlock.
5. Esquema eléctrico.
6. Esquema de conexión.
7. Fotos o vídeo del proyecto (enlace).