Hola LeoFabiani,

He estado trabajando en el firmware de tu proyecto donde se han conseguido los siguientes puntos:

- Options to set setpoint temperature manually and 5 user programmable setpoints with default standard setpoints that can be easily changed at compile time

- PID values should be easily changed at compile time.

- Maximum setpoint limit value easily changeable at compile time

- Error states with output power shutdown for thermocouple failure.

- Red LED flashes in relation to output power being applied to triac

- Green blink at 1hz to show the firmware is running and not locked up.

- Display brightness adjustable

- Display unit changeable from F to C.

- Se lee la temperatura del microcontrolador cada 5 minutos, si dicha temperatura se encuentra por encima de una temperatura critica (En este caso 70°C) el microcontrolador desactiva el control PID y apaga la salida de poder.

- Presets selected by one button press. Up or down through the 5 presets

- Enter Menu system by pressing both buttons

- Unit must be able to operate properly if started from cold state and if already warmed up and is unplugged and plugged back in.

Aún estamos trabajando en la implementación de:

* PID temperature control. With robust runaway detection protection (for example if power is being applied to heater for more than 10 or 15 seconds with no increase in temperature or unreasonable change it should shut down output power) runaway parameters shoul be easily changeable at compile time
* Error states for runaway detection with output power shutdown upon detection.
* ¿Te parece bien si dejamos que el led azul brille cuando existe algún error en el sistema?

Por otra parte, surgen algunas limitaciones:

1. PID auto tune would be great to have if possible and doesn’t add too much development cost (Esto es peligroso debido que genera un sobre impulso alto en el calentador)
2. LED brightness adjustable. (Estos led solo pueden tener un estado encendido o pagado, esto se debe a que el microcontrolador no cuenta con suficientes puertos PWM para regular la potencia de los leds)
3. Temperature display offset. Note; this is to show estimated nozzle temp due as opposed to actual heater temp (Esto puede provocar inestabilidad en el Sistema de control debido a que no contamos con un sensor que capture la temperatura de la boquilla)
4. An optional stand by mode with reduced temp/output power after no detection of load change for a certain amount of time if possible, user setable in 15 minute increments from 30min - 3 hrs defaults easily changeable at compile time (Esto es inviable debido a que podemos alterar el lazo de control y provocar un comportamiento no deseado del sistema)
5. Sleep timer user changeable from 30 minutes - 3hrs in 15 minute increments. Default

time easily changeable at compile time. ( Con esto te refieres a que quieres suspender la salida de poder después de cierto tiempo tiempo sin poner a dormir el micro controlador? Te pregunto esto porque podemos hacer de que el micro controlador se duerma después de cierto tiempo y se despierte al pulsar uno de los botones)

Hi LeoFabiani,

I have been working on the firmware of your project where the following points have been achieved:

- Options to set setpoint temperature manually and 5 user programmable setpoints with default standard setpoints that can be easily changed at compile time

- PID values ​​should be easily changed at compile time.

- Maximum setpoint limit value easily changeable at compile time

- Error states with output power shutdown for thermocouple failure.

- Red LED flashes in relation to output power being applied to triac

- Green blink at 1hz to show the firmware is running and not locked up.

- Display brightness adjustable

- Display unit changeable from F to C.

- The temperature of the microcontroller is read every 5 minutes, if said temperature is above a critical temperature (In this case 70 ° C) the microcontroller deactivates the PID control and turns off the power output.

- Presets selected by one button press. Up or down through the 5 presets

- Enter Menu system by pressing both buttons

- Unit must be able to operate properly if started from cold state and if already warmed up and is unplugged and plugged back in.

We are still working on the implementation of:

- PID temperature control. With robust runaway detection protection (for example if power is being applied to heater for more than 10 or 15 seconds with no increase in temperature or unreasonable change it should shut down output power) runaway parameters shoul be easily changeable at compile time

- Error states for runaway detection with output power shutdown upon detection.

- Is it okay if we let the blue led shine when there is an error in the system?

On the other hand, some limitations arise:

1. PID auto tune would be great to have if possible and doesn’t add too much development cost (This is dangerous because it generates a high boost in the heater)

2. LED brightness adjustable. (These LEDs can only have an on or paid state, this is because the microcontroller does not have enough PWM ports to regulate the power of the LEDs)

3. Temperature display offset. Note; this is to show estimated nozzle temp due as opposed to actual heater temp. (This can cause instability in the Control System because we do not have a sensor that captures the temperature of the nozzle)

4. An optional stand by mode with reduced temp / output power after no detection of load change for a certain amount of time if possible, user setable in 15 minute increments from 30min - 3 hrs defaults easily changeable at compile time. (This is not feasible because we can alter the control loop and cause unwanted behavior of the system)

5. Sleep timer user changeable from 30 minutes - 3hrs in 15 minute increments. Default

time easily changeable at compile time. (By this you mean you want to suspend the power output after a certain time without putting the micro controller to sleep? I ask you this because we can make the micro controller go to sleep after a certain time and wake up by pressing one of the buttons)