SWIFT ACT ELECTRIC HEATER GUIDE

By : Mohamed Yousry Mohamed Rashad .

Project Documentation and Github

- Git Hub Repository Link :
- Https://github.com/rumbletech/swa_heater
- Doxygen Documentation in html format , is in the "doc" directory in the project.
- State Information and Main.c Documentation is in this presentation .

Project Documentation :

Project Documentation is divided into two parts :

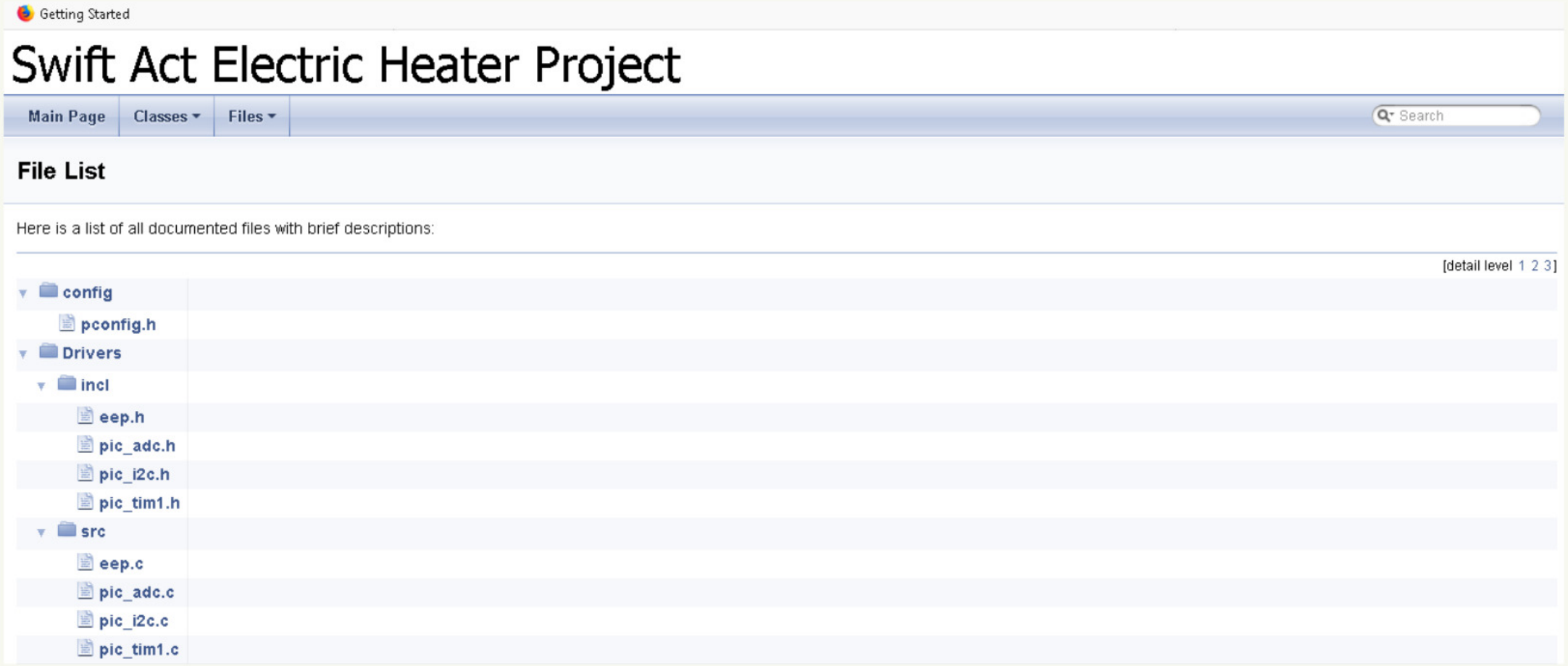
1- Drivers documentation and interfaces are documented using doxygen ,

Generated html is found inside the project folder in the "doc/html/index.html"

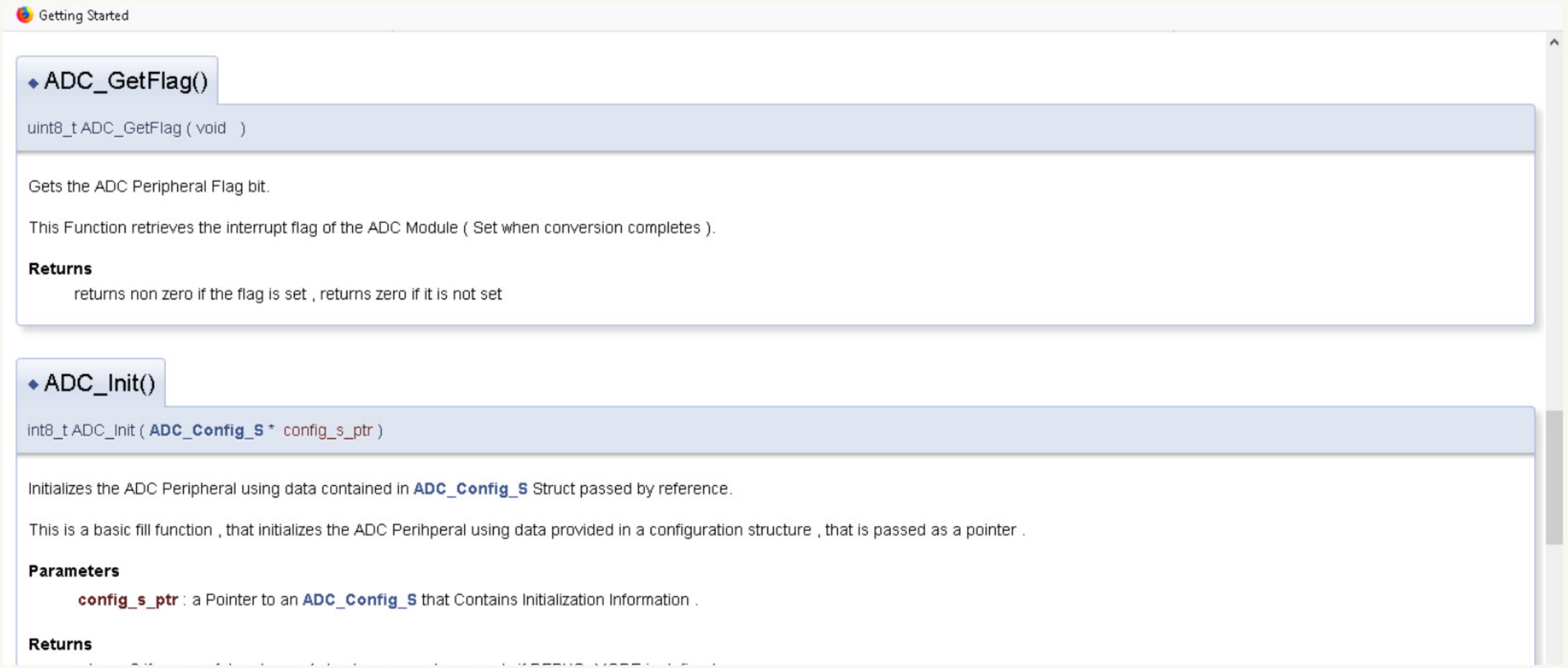
file.

2- Main program and interrupt code is documented inside the code , and explained in this presentation .

1-Inside HTML Documentation 1 :



1-Inside HTML Documentation 2 :



2- TIMER2_IRQ_Handler()

1- Timer2 Handles the scanning of the 7 segment displays , timer2's interrupt is set to fire at a rate of 120HZ .

2- Timer2_IRQ_Handler() does one thing , it checks for the display_state if it is on , it selects a digit by enabling its enable signal , and outputs its Segment data on the segment port (PORTD) , at the end it toggles the segment number so that the next interrupt , enables the other digit .

3- Since each digit takes two interrupts to get re-selected , the scan time for each digit is 8*2 = 16 ms , which gives about a scan rate of approximately 60Hz .

4- no flicker is seen as it is an appropriate rate .

2- TIMER1_IRQ_Handler()

- TIMER1 is set to fire every 32.768 ms as calculated .
- The first task handled by the interrupt is to check the push buttons to detect any press .
- If a press is detected:
  - If the button was the ON/OFF button , if the system was OFF then the next task is PWR_ON else if the system was ON the next task is PWR_OFF .
  - If the button was the Increment button , desired temp is incremented by 5 only if TEMP Setting mode is on , and it is within range .
  - If the button was the decrement button , desired temp is decremented by only if TEMP Setting mode is on and it is within range.

2-TIMER1_IRQ_Handler()(Continued)

- TIMER1 is also responsible for Starting ADC conversions , a count is kept inside the timer handler , every time the count is equal to 3 , 3*32.768 , approximately 99ms , an adc conversion is started , the result is fetched by the ADC_IRQ_Handler().
- TIMER1 is Also responsible for producing timing for other Signals , such as producing a 5 second interval , when in temperature setting mode , if this interval times out it changes the display mode back to normal , during the temp setting mode , it switches display state from off to on every second , all of these timings are produced with help of counters inside the IRQ_Handler() based on an interrupt time of 32.768 ms .

Notes on IRQ_Handlers

- While some IRQ_Handlers are Lengthy , they are written so that they are fast , no calls to other functions are made inside the handlers , only macros are used , each IRQ_Handler only does a combination of the following :
  - Check register bits , write to registers .
  - Read registers , set flags .
  - Increment or decrement variables .

  The interrupts handlers have a fast response with respect to their firing rates , which is 100ms for ADC , 32.768 ms for TIMER1 , 8 ms For Timer2 .

2-ADC_IRQ_Handler()

- The ADC_IRQ_Handler() fires after a conversion completes that was is started every 100 ms by the TIMER1_Handler() .
- The ADC Handler fetches the conversion results adds it to an average running sum , and exits , every 10 interrupts the average is calculated and set to some variable , the average is then zeroed to be recalculated .
- It is also responsible for blinking the heater LED , with the help of a counter , after 10 ADC interrupts , it toggles the LED , if the heater state is on , if the cooler is on it turns it on and doesn't toggle , nothing is on , it turns it off.

3- PWR_ON Task

- PWR_ON Task is Set by TIMER1 when the device_state is OFF and ON/OFF Button is pressed .
- The Task is Handled in the Main loop of the program , due to the nature of the task , it enables some peripherals and reads from an eeprom memory , which makes this task quite lenghy to be handled in an interrupt routine , aslo because some of the I2C Functions used are blocking , and it isn't good to put blocking calls inside an interrupt handler .

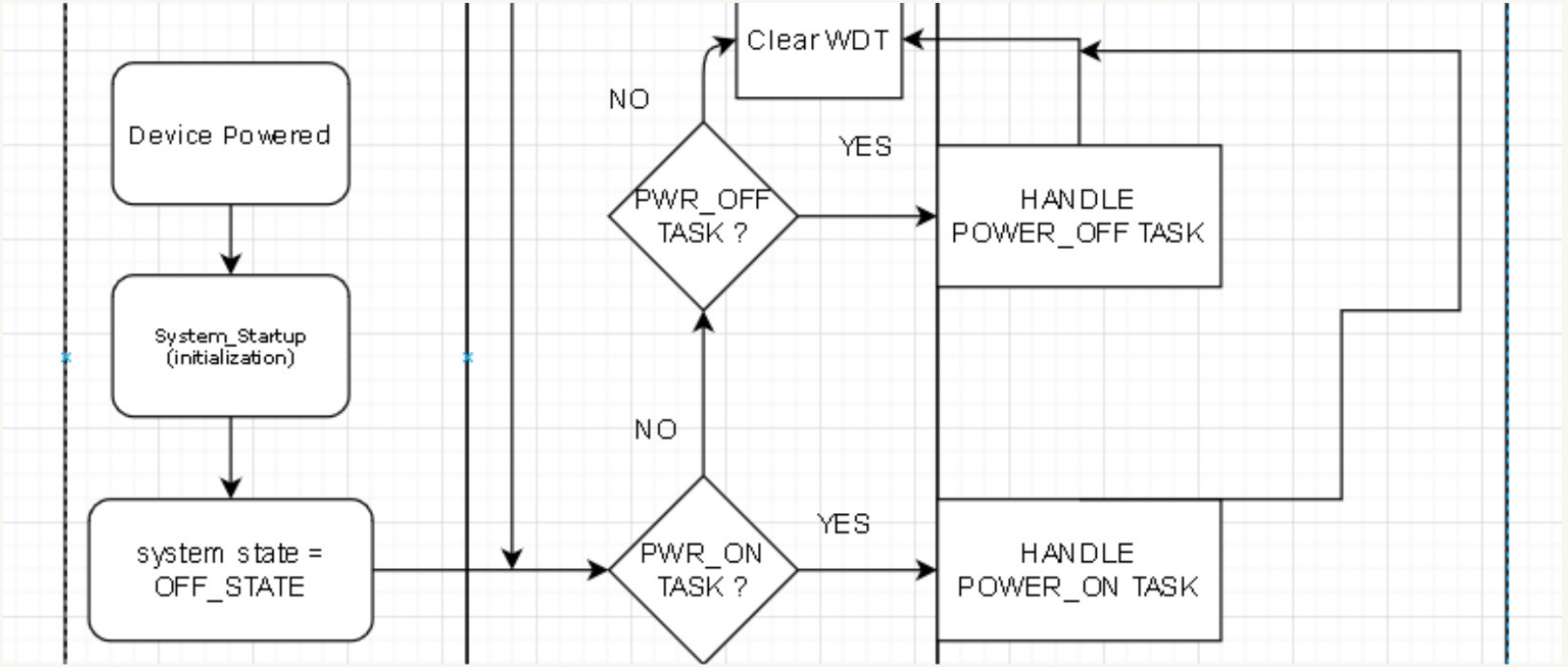
4-PWR_OFF Task

- This Task like the PWR_ON Task is handled inside the main program loop it is set when the ON/OFF button is pressed while the device_state is ON .
- The task includes turning off ADC/TIMER2/DISPLAY , and turning off Heaters and Coolers , then it proceeds to write the desired temperature into eep memory .
- This task like the previous requires calls to eep that calls other i2c functions which are blocking and lengthy so it is not handled in the interrupt routine.

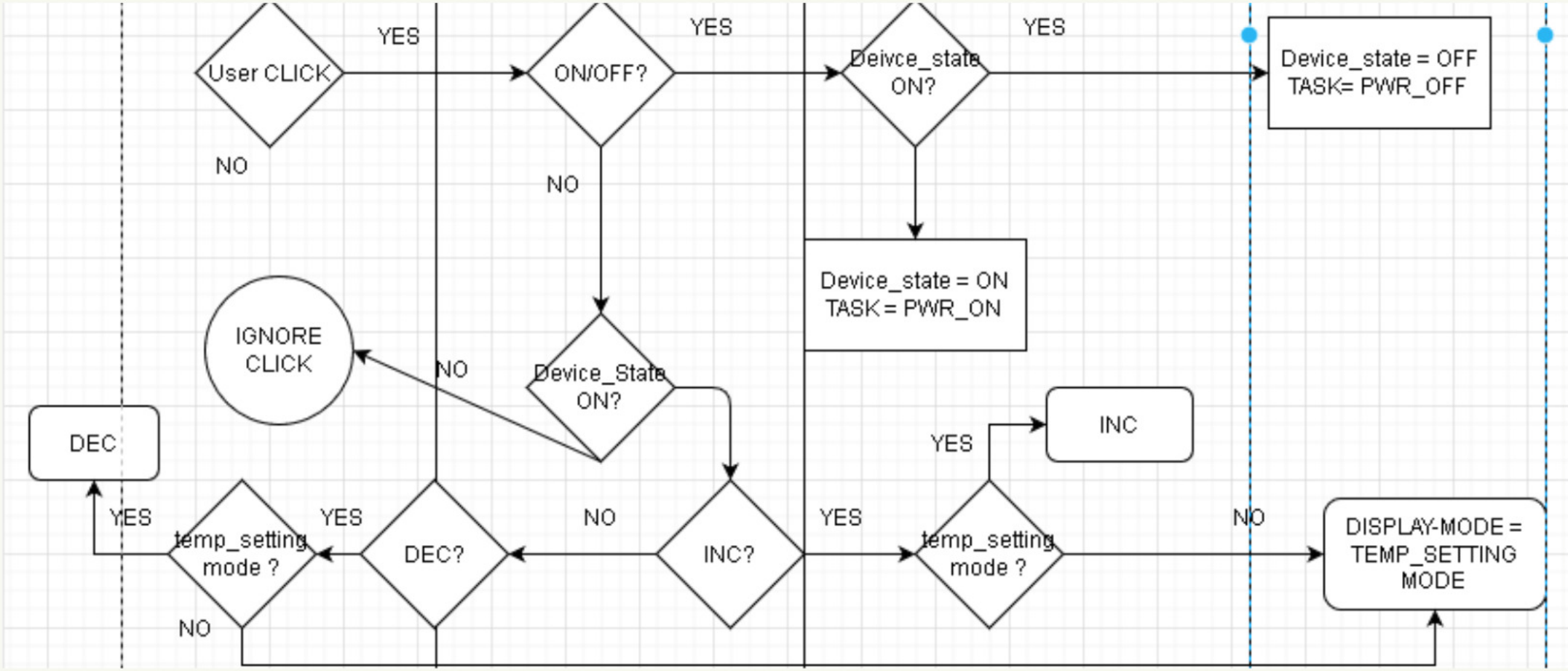
5-Watch Dog Timer

- This application i have decided to enable the watchdog timer , that resets the microcontroller if it hangs (no CLRWDT Instruction is exceeded , WDT timed out) .
- The Watchdog timer Timeout value is set at 500ms , if the application fails to clear the timer before that , the application resets .
- The Watchdog is cleared at the end of the main program loop.

State Diagram/Flow Chart for main .



State Diagram/Flow Chart for TIM1 Handler



Notes on diagram

- I have neglected timing signals in the diagrams as they are not important , they have logic such that if a count reaches some value it takes some action such as starting adc conversion or blinking leds and so on , also TIMER2 and ADC Handlers are very basic and do simple tasks .