0.1 ADC

When programming the ability to use a ADC (Analog to digital converter) to read a analog value on the microprocessor is of crucial impotents. In our project the ADC is use for reading values from our infrared and load sensors. On the ESP32 a total of 18 ADC channels are available with a config resolution options of 9,10,11 and 12-bits.

0.1.1 Configuring

It is impotent to ensure when using RTOS that functions are Thread safe. If not it can result in RTOS not being able to handle a task in the desired time frame. For this reason we have chosen to use libraries that are include in the ESP-IDF environment that are designed with thread safe in mind. The particular library used to facilitate ADC is called *esp_adc/adc_oneshot.h* that replaced the previous one in version 5.0.4 of ESP-IDF.

To configure a ADC unit:

```
1         adc_oneshot_unit_handle_t adc1_handle;
2         adc_oneshot_unit_init_cfg_t init_config1 = {
3             .unit_id = ADC_UNIT_1,
4             .ulp_mode = ADC_ULP_MODE_DISABLE,
5         };
6         ESP_ERROR_CHECK(adc_oneshot_new_unit(&init_config1, &adc1_handle));
7
```

Listing 1: Configuring ADC unit 1

The program utilize handles to reference the objected throughout the program. When *calling adc_oneshot_new_unit* a new instance is created with the specified configuration. In a similar way the channels are configured and created to the handle.

0.1.2 Reading

To read the raw value of the ADC unit 1 channel 0:

```
1         adc_oneshot_read(adc1_handle, ADC1_0, &adc_value);
2
```

Listing 2: Readning ADC unit 1 channel 0

This function takes 3 augments: the handle itself, the desired channel to read from and where the output should be stored. This function can acquire the raw value from any ADC on the given unit as long as they are configured.

To calculate the voltage level:

$$V_{out} = D_{out} * \frac{V_{max}}{D_max} \tag{1}$$

Where:

D_{out}:

V_{max}:

D_{max}: