1. The following example shows how to configure the PM8998 GPIO as analog input and set up the ADC read via devicetree with following settings:

GPIO - 9
Channel name - gpio9_adc
Input voltage range - 0 V to 1.875 V
Calibration type - Absolute
No interpolation to other units
Setting a delay of 100 µs is recommended for GPIO channels

2. Check whether the GPIO is disabled in the msm-pm8998.dtsi file.

NOTE: Ensure that the GPIO is not being used by other processors.

```
gpio@c800 {
reg = <0xc800 0x100>;
qcom,pin-num = <9>;
status = "ok";
qcom,master-en = <0>; /* DISABLE GPIO */
};
```

3. Set the ADC channel for GPIO in msm-pm8998.dtsi.

```
chan@13 {
label = "gpio9_adc";
reg = <0x13>; // channel for GPIO9
qcom,decimation = <2>;
qcom,pre-div-channel-scaling = <0>; //1:1 scaling
qcom,calibration-type = "absolute";
qcom,scale-function = <2>;
qcom,hw-settle-time = <2>;
qcom,fast-avg-setup = <0>;
}
```

4. In the client node, add the VADC channel A/D.

```
client_node {
qcom,test-vadc = <&pm8998_vadc>;
};
```

NOTE: To associate the client with the corresponding device, use the consumer name passed to the driver when calling the qpnp_get_vadc() function.

5. Associate the client with the corresponding device and get the device structure.

```
struct qpnp_vadc_chip *vadc_dev;
vadc_dev = qpnp_get_vadc(chip->dev, "test");
```

6. Read the VADC channel via the QPNP ADC API.

```
struct qpnp_vadc_result result;
```

err = qpnp_vadc_read(vadc_dev, P_MUX2_1_1, &result); //Read the GPIO9 VADC channel with 1:1 scaling
*adc = (int) result.physical;
*adc = *adc / 1000; /* uV to mV */
For pm660 adc channel info, pls refer Table 3-22. of doc 80-P7905-1,
for pm8998 adc chaanel info, pls refer Table 3-22 of doc 80-P1086-1