Zephyr: ADC

Medical Electrical Equipment (BME590L)

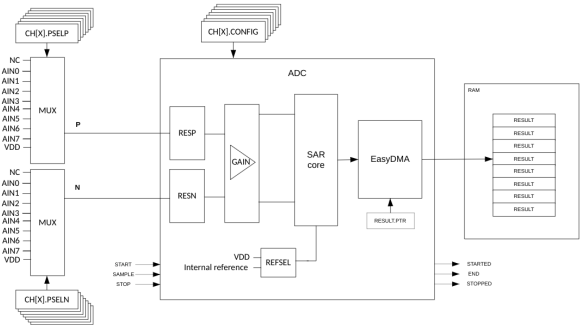
Mark L. Palmeri, M.D., Ph.D.

March 06, 2023

Duke | BME

Successive Approximation ADC (SAADC)

- The `nRF52833` uses a successive approximation ADC (SAADC).
- That 1 ADC is multiplexed across 8 potential input **channels** (GPIO pins).



Duke | BME

prj.conf

```
CONFIG_ADC=y
CONFIG_ADC_NRFX_SAADC=y   # this is no longer necessary
```

Duke | BME

nRF52833dk_nrf52833.overlay

```
/ {
    zephyr,user {
        io-channels = <&adc 0>; // need to "activate" these channels for IO usage
 };
}
```

nRF52833dk_nrf52833.overlay

```
&adc {
    #address-cells = <1>;
    #size-cells = <0>;

    vbat: channel@0 {
        reg = <0>;  // channel number
        zephyr,reference = "ADC_REF_INTERNAL";  // voltage ref for SA (0.6 V)
        zephyr,gain = "ADC_GAIN_1_5"; // gain factor to scale ref voltage
        ↪  (1/gain)
        zephyr,acquisition-time = <ADC_ACQ_TIME_DEFAULT>;
        zephyr,input-positive = <NRF_SAADC_AIN0>; // P0.02
        zephyr,resolution = <8>;  // bit depth
    };
    status = "okay"; // enable the ADC
};
```

Duke | BME

How Digital Output is Calculated

Digital output

The output result of the ADC depends on the settings in the CH[n].CONFIG and RESOLUTION registers as follows:

$$RESULT = [V(P) - V(N)] * GAIN/REFERENCE * 2^{(RESOLUTION - m)}$$

where

V(P)
: is the voltage at input P

V(N)
: is the voltage at input N

GAIN
: is the selected gain setting

REFERENCE
: is the selected reference voltage

and m=0 if CONFIG.MODE=SE, or m=1 if CONFIG.MODE=Diff.

Duke | BME

nRF52833 Reference Voltage

- Only `ADC_REF_INTERNAL` and `ADC_REF_VDD_1_4` are valid on the `nRF52833`.
- When using `VDD` as a reference, you need to also specify the nominal reference voltage in the Devicetree:
- `zephyr,vref-mv = <750>;`

6.21.2 Reference voltage and gain settings

Each SAADC channel can have individual reference and gain settings.

This is configured in registers CH[n].CONFIG (n=0..7) on page 379. Available configuration options are:

- VDD/4 or internal 0.6 V reference
- Gain ranging from 1/6 to 4

The gain setting can be used to control the effective input range of the SAADC:

```
Input range = (±0.6 V or ±VDD/4)/gain
```

For example, selecting VDD as reference, single-ended input (grounded negative input), and a gain of 1/4 will result in the following input range:

```
Input range = (VDD/4)/(1/4) = VDD
```

With internal reference, single-ended input (grounded negative input) and a gain of 1/6, the input range will be:

```
Input range = (0.6 V)/(1/6) = 3.6 V
```

Inputs AIN0 through AIN7 cannot exceed VDD or be lower than VSS.

Duke | BME

ADC parameter considerations

- Why would you choose a reference voltage different than the `HIGH` voltage?
- Why consider using `ADC_REF_VDD_1` instead of `ADC_REF_INTERNAL`?
- ADC can be directly measured or *differentially* measured (requires `input-negative`)
- If performing a single ADC measurement, you can **oversample**. But beware, oversampling averages over all activated ADC channels.

There are lots of ADC macros to help with this configuration:
`https://docs.zephyrproject.org/latest/hardware/peripherals/adc.html`

Duke | BME

Include the ADC library

```
#include <zephyr/drivers/adc.h>
```

main.c: Preamble Macros

Define some macros to use Zephyr macros to perform setup:

```
#define ADC_DT_SPEC_GET_BY_ALIAS(node_id)                       \
{                                                               \
    .dev = DEVICE_DT_GET(DT_PARENT(DT_ALIAS(node_id))),         \
    .channel_id = DT_REG_ADDR(DT_ALIAS(node_id)),               \
    ADC_CHANNEL_CFG_FROM_DT_NODE(DT_ALIAS(node_id))             \
}

#define DT_SPEC_AND_COMMA(node_id, prop, idx) \
    ADC_DT_SPEC_GET_BY_IDX(node_id, idx),
```

Duke | BME

main.c: Initialize the ADC structure

```
static const struct adc_dt_spec adc_vbat = ADC_DT_SPEC_GET_BY_ALIAS(vbat);
```

This struct is associated with all of the ADC parameters defined in the Devicetree.

https://docs.zephyrproject.org/latest/hardware/peripherals/adc.html#c.adc_dt_spec

Within main()...

- Check that the device is ready:

```
if (!device_is_ready(adc_vbat.dev)) {
    LOG_ERR("ADC controller device(s) not ready");
    return -1;
}
```

- Configure the input pin:

```
err = adc_channel_setup_dt(&adc_vbat);
if (err < 0) {
    LOG_ERR("Could not setup Vbat ADC channel (%d)", err);
    return err;
}
```

Duke | BME

Setup variables to store ADC value

```
int16_t buf;

struct adc_sequence sequence = {
    .buffer = &buf,
    .buffer_size = sizeof(buf), // bytes
};
```

Read the ADC channel

```
LOG_INF("Measuring %s (channel %d)... ", adc_vbat.dev->name,
↪  adc_vbat.channel_id);

(void)adc_sequence_init_dt(&adc_vbat, &sequence);

int ret;
ret = adc_read(adc_channel.dev, &sequence);
if (ret < 0) {
    LOG_ERR("Could not read (%d)", ret);
} else {
    LOG_DBG("Raw ADC Buffer: %d", buf);
}
```

Convert Raw ADC Value to mV

```
int32_t val_mv;
val_mv = buf;
ret = adc_raw_to_millivolts_dt(&adc_vbat, &val_mv);
if (ret < 0) {
    LOG_ERR("Buffer cannot be converted to mV; returning raw buffer value.");
} else {
    LOG_INF("ADC Value (mV): %d", val_mv);
}
```

`adc_raw_to_millivolts_dt()` uses the ADC `reference` and `gain` Devicetree parameters to convert to millivolts.

Duke | BME

References

- Zephyr: ADC
- Nordic Semiconductor SAADC
- nRF52-ADC-Examples