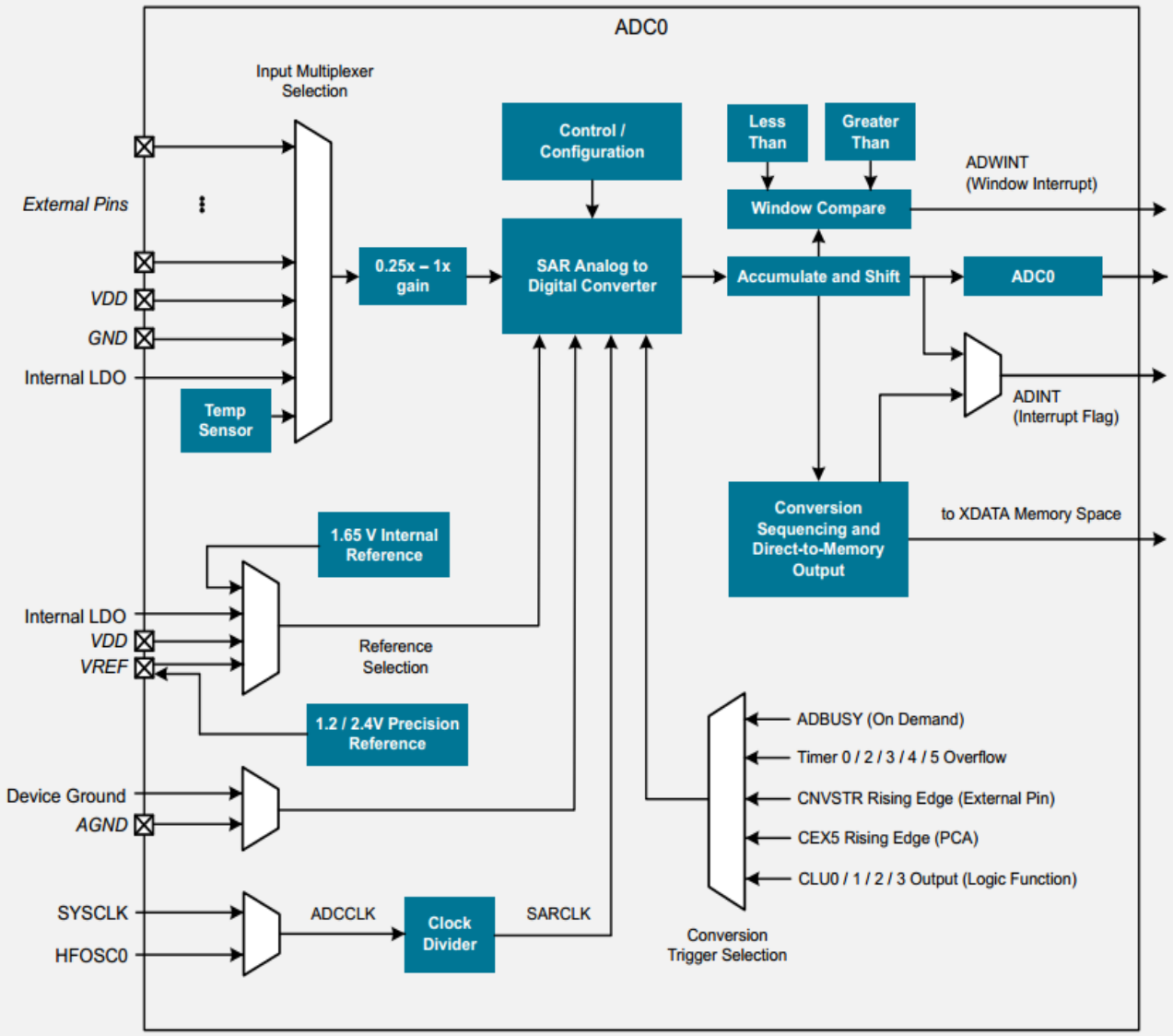
EFM8LB1 – Analog to Digital Converter (ADC)

22 SEPTEMBER 2015

Agenda

- ADC Overview
- Input Selection, Gain Setting, Reference Option
- Clock Selection, Timing, Trigger Source
- Track Time calculation
- Window detection, interrupt, accumulation, shift
- Low power feature
- Auto-scan Mode
- Configuration for autoscan mode
- Software Examples
- Calibration and additional resources

ADC Overview



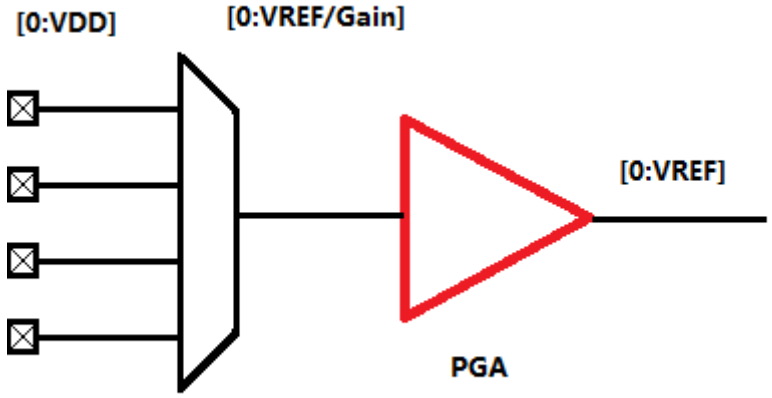
- Single ended 14-bit SAR type (LB1 only)
- Support 1 Msps speed for 12bit mode
- 20 external and 4 internal inputs selectable via input mux
- Voltage and GND reference selectable
- Accumulation and shift feature with 14/12/10bit mode
- Multi trigger sources selectable (asynchronous H/W)
- Support Window detector and interrupt
- Low power feature
- Autoscan mode without CPU intervention

Input Selection

ADC0MX setting	Signal Name	Enumeration Name	QFN32 / QFP32	QSOP24	QFN24
			Pin Name	Pin Name	Pin Name
00000	ADC0.0	ADC0P0	P0.1	P0.1	P0.1
00001	ADC0.1	ADC0P1	P0.2	P0.2	P0.2
00010	ADC0.2	ADC0P2	P0.4	P0.4	P0.4
00011	ADC0.3	ADC0P3	P0.5	P0.5	P0.5
00100	ADC0.4	ADC0P4	P0.6	P0.6	P0.6
00101	ADC0.5	ADC0P5	P0.7	P0.7	P0.7
00110	ADC0.6	ADC0P6	P1.0	P1.0	P1.0
00111	ADC0.7	ADC0P7	P1.1	P1.1	P1.1
01000	ADC0.8	ADC0P8	P1.2	P1.2	P1.2
01001	ADC0.9	ADC0P9	P1.3	P1.4	P1.4
01010	ADC0.10	ADC0P10	P1.4	P1.5	P1.5
01011	ADC0.11	ADC0P11	P1.5	P1.6	P1.6
01100	ADC0.12	ADC0P12	P1.6	P1.7	Reserved
01101	ADC0.13	ADC0P13	P1.7	Reserved	Reserved
01110	ADC0.14	ADC0P14	P2.1	Reserved	Reserved
01111	ADC0.15	ADC0P15	P2.2	Reserved	Reserved
10000	ADC0.16	ADC0P16	P2.3	Reserved	Reserved
10001	ADC0.17	ADC0P17	P2.4	Reserved	Reserved
10010	ADC0.18	ADC0P18	P2.5	Reserved	Reserved
10011	ADC0.19	ADC0P19	P2.6	Reserved	Reserved
10100	ADC0.20	TEMP	Internal Temperature Sensor		
10101	ADC0.21	LDO_OUT	Internal 1.8 V LDO Output		
10110	ADC0.22	VDD	VDD Supply Pin		
10111	ADC0.23	GND	GND Supply Pin		
11000 - 11110	ADC0.24 - ADC0.30		Reserved	Reserved	Reserved
11111	ADC0.31	NONE	No connection		

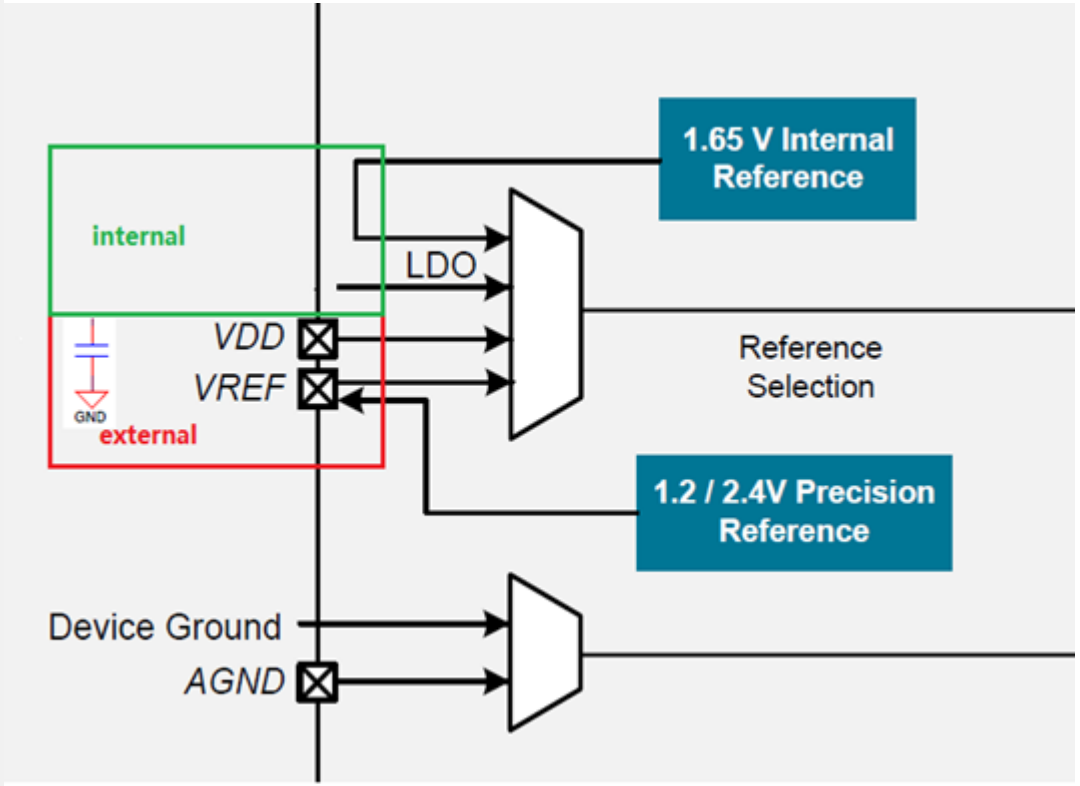
- Analog Multiplexer (ADC0MX) to select channel (single ended)
- 20 external pins for QFN32 package
- Internal LDO/VDD/GND
- Temperature sensor
- Any external port pin selected as ADC input should be configured as analog input and skipped by crossbar

Gain Setting



- 0.25x/0.5x/0.75x/1.0x 4 gain options (ADGN of register ADC0CN0[1:0])
- Higher input voltage range (>VREF) is allowed with small gain setting (attenuate the signal to fit the VREF range)
  - Small VREF (1.2V) to measure input above 1.2V voltage
  - To measure voltage between [VREF, VDD] with VREF reference
- Noted that input should not above supply rail(VDD)

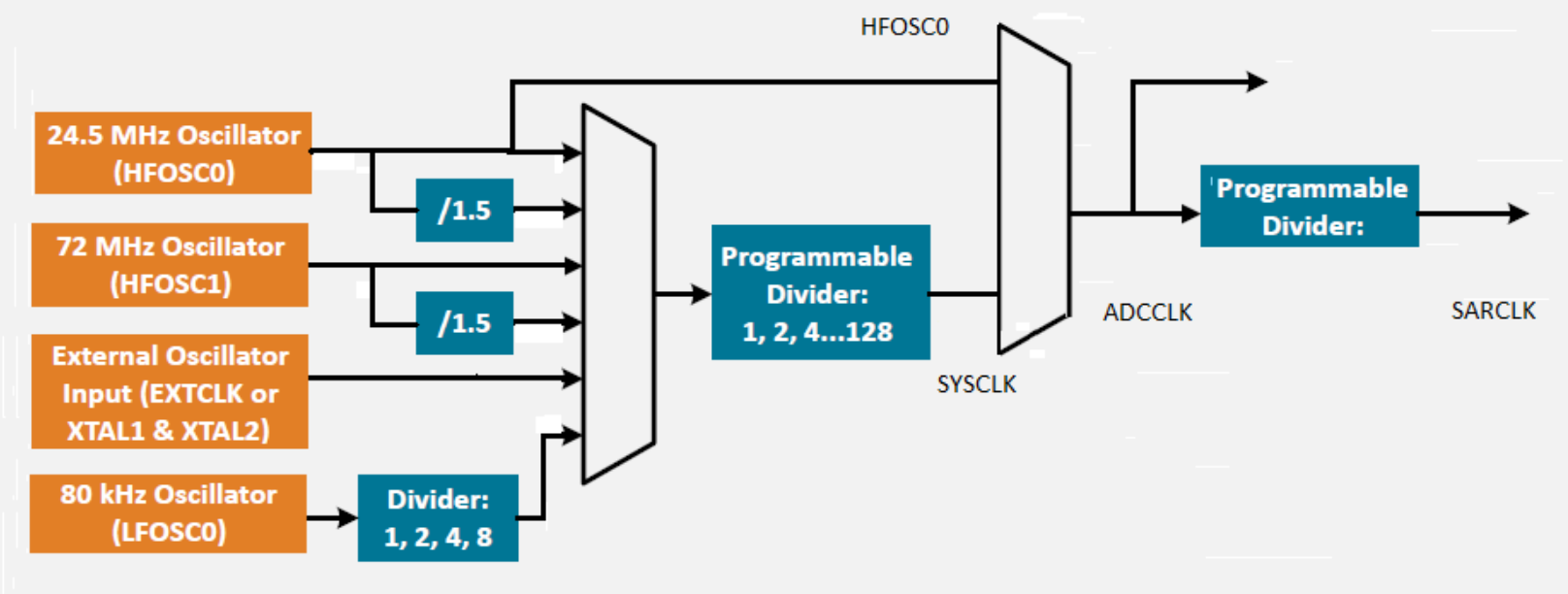
Voltage and GND reference selection



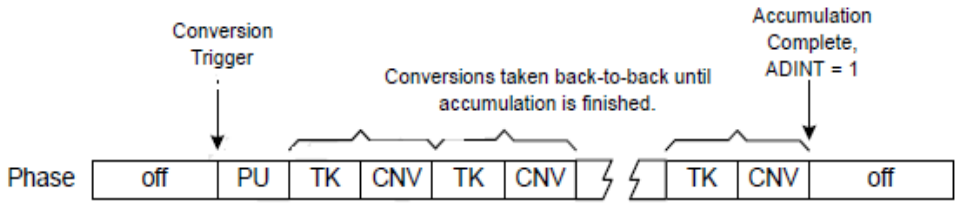
- Voltage internal (REFSL in **ADC0CF2[6:5]**)
  - 1.65V high speed (short time to become stable)
  - 1.8V LDO
- Voltage external (extra coupling component)
  - VDD supply (non-varying power supply)
  - 1.2V/2.4V internal precision reference connect to VREF pin (external capacitor needed)
  - True external reference (VREFSL in REF0CN[7:6])
  - Configure VREF pin as analog mode and skipped by crossbar
- GND reference (GNDSL in ADC0CF2[7])
  - Device GND (ground pin, internal sensor)
  - Dedicated GND pin (AGND referenced by external sensor)
- Note that some H/W determined combination for some special scenarios
  - When Internal temperature sensor or internal high speed reference is selected, device GND automatically was used

Clock Selection

- ADCCLK: SYSCLK (default) or HFOSC0 selectable by ADCLKSEL field of ADC0CF0
  - High speed ADC but slow SYSCLK, choose HFOSC0 (24.5MHz)
  - ADCCLK is used to clock register and other logic in the ADC module
- SARCLK: further divided version (ADSC) of ADCCLK
  - SARCLK is used to drive the conversion process (should be as fast as possible, up to 18 MHz)
  - SARCLK=ADCCLK/(ADSC+1), ADSC is ADC0CF0[7:3]

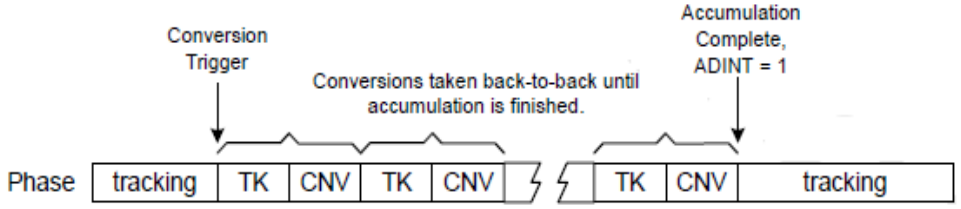


Timing



off = ADC shut down.
PU = Power-Up Phase. Timing Defined by ADPWR field.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

ADC Timing With IPOEN = 1



tracking = Converter tracking selected input any time conversion is not in progress.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

ADC Timing With IPOEN = 0

- Powerup+tracking+conversion phase.
- Power up time: (4*(ADPWR+1)+2)/ADCCLK,
  - allows time for the ADC and internal reference circuitry to power on and settle before sample and hold
  - Optional and is used only when the ADC is configured power off after conversion is complete (when IPOEN is set).
  - Upto 1.2uS in datasheet, ADPWR (ADC0CF2[0:4])
- Tracking time: to see next section
  - SARCLK as time-base, 230nS minimum @ fast mode
  - set to 278nS to match the integer number of 18MHz SARCLK (5 SARCLK)
- Conversion time: (N+1)*SARCLK
  - where N is resolution
  - 18MHz SARCLK @72MHz SYSCLK, 13/18=722nS@12bit
- 1 Msps @12bit （throughput table 4.9, page 15 in datasheet)
  - High speed mode, tracking time 278ns,
  - 1/(278nS+722ns) = 1.000MHz @12 bit mode
  - 1/(278nS+540ns) = 0.900MHz @14 bit mode
  - 1/(278nS+611ns) = 1.125MHz @10 bit mode

Initiating/Trigger source (ADCM of ADC0CN2[3:0])

Bit	7	6	5	4	3	2	1	0
Name	PACEN	Reserved			ADCM			
Access	RW	R			RW			
Reset	0	0x0			0x0			
SFR Page = 0x0, 0x30; SFR Address: 0xB3								

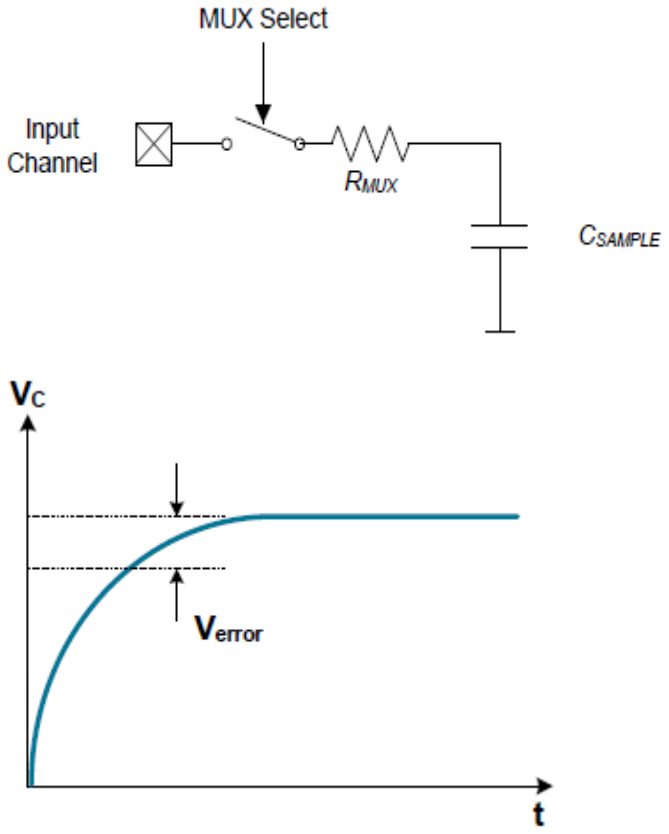
Bit	Name	Reset	Access	Description
6:4	*Reserved*	*Must write reset value.*		
3:0	ADCM	0x0	RW	**Start of Conversion Mode Select.**

Specifies the ADC0 start of conversion source. All remaining bit combinations are reserved.

Value	Name	Description
0x0	ADBUSY	ADC0 conversion initiated on write of 1 to ADBUSY.
0x1	TIMER0	ADC0 conversion initiated on overflow of Timer 0.
0x2	TIMER2	ADC0 conversion initiated on overflow of Timer 2.
0x3	TIMER3	ADC0 conversion initiated on overflow of Timer 3.
0x4	CNVSTR	ADC0 conversion initiated on rising edge of CNVSTR.
0x5	CEX5	ADC0 conversion initiated on rising edge of CEX5.
0x6	TIMER4	ADC0 conversion initiated on overflow of Timer 4.
0x7	TIMER5	ADC0 conversion initiated on overflow of Timer 5.
0x8	CLU0	ADC0 conversion initiated on CLU0 Output.
0x9	CLU1	ADC0 conversion initiated on CLU1 Output.
0xA	CLU2	ADC0 conversion initiated on CLU2 Output.
0xB	CLU3	ADC0 conversion initiated on CLU3 Output.

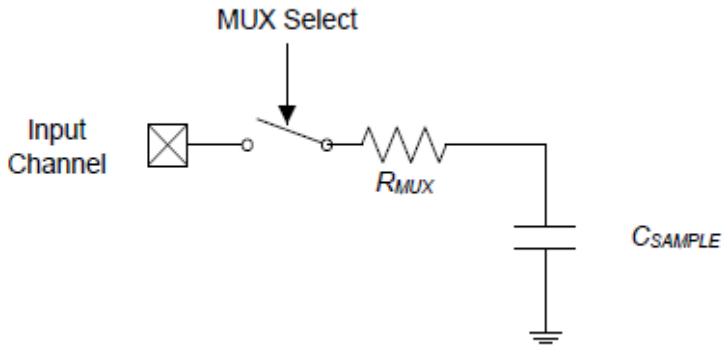
- S/W (1)
  - Writing 1 to ADBUSY
- H/W (10)
  - Timerx overflow (T0/2/3/4/5)
  - CLUx output (CLU0/1/2/3)
  - CEX5 (rising edge)
- External pin (1)
  - Rising edge of CNVSTR (rising edge)
- ADCINT flag indicate finish of conversion
- Multi trigger or single trigger in Autoscan mode

Settling time



- Each ADC conversion must be preceded by a minimum tracking time to allow the voltage on the sampling capacitor to settle, and for the converted result to be accurate.
- This diagram show how the capacitor switched SAR type ADC worked. It formed into a RC circuit. Based on this we could get the dynamic characteristics of this circuit.

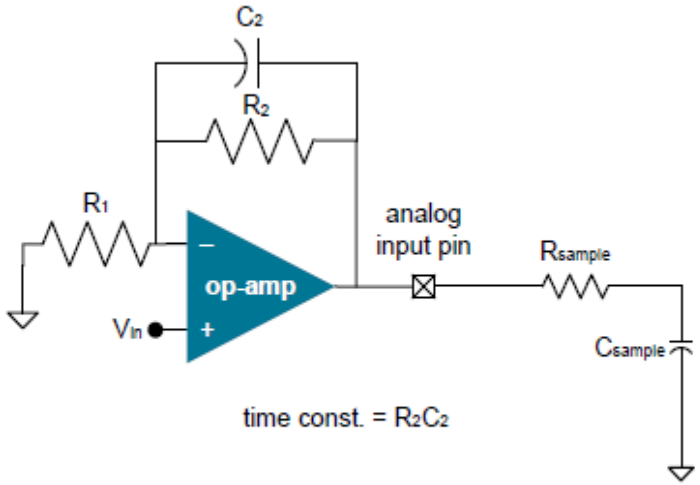
Settling time



$$t = ln\left(\frac{2^n}{SA}\right) * R_{Total} * C_{Sample}$$

- Where: SA is the settling accuracy, given as a fraction of an LSB (for example, choose 0.25 to settle within 1/4 LSB)
- t is the required settling time in seconds
- R_{TOTAL} is the sum of the ADC mux resistance and any external source resistance (550 ohm).
- C_{SAMPLE} is the size of the ADC sampling capacitor, depend on PGA gain.
- n is the ADC resolution in bits.
- Will vary based on whether the ADC is in low power mode
- Need longer if ADC input is presented with a large series impedance

Settling time



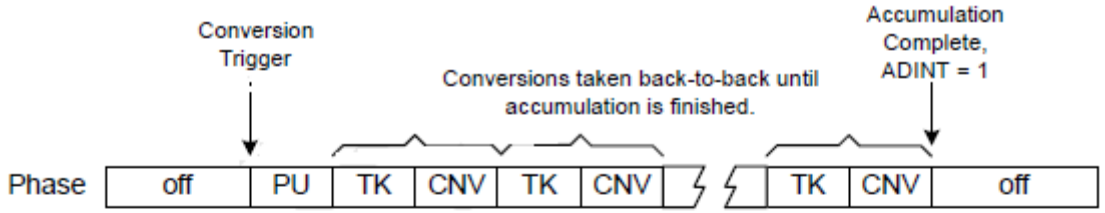
Active Anti-Alias Filter

- When the external circuitry is connected to the analog input pin, the settling time may be affected.
- Such circuitry typically includes an anti-aliasing filter used to remove higher frequency noise that will alias or fold into the signal band of interest.
- The external circuit's capacitance and output impedance will affect the settling time.
- The design of anti-alias filters should be designed to drive capacitor in the ADC input circuit.
- For example the active anti-aliasing filter like left side showed. These filter form good buffer stadge as they have higher input impedance and lower output impedance.
- But Op amplififer may introduce some noise, refer to the manufacture's datasheet for such noise information.

Track time

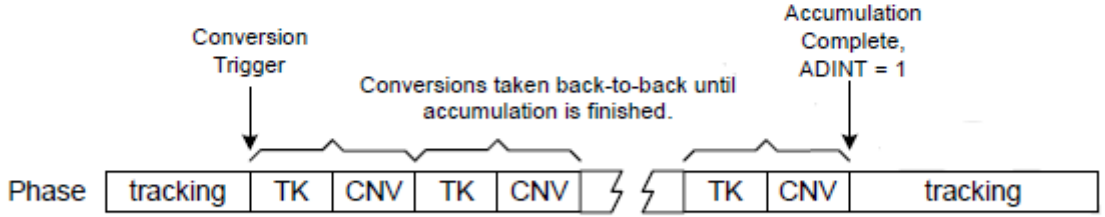
- Based on previous slide we know the track time should be set longer than settling time.
- SARCLK is used as time-base for the track process.
- Decide by ADTK (ADC0CF1[5:0])
- T_{adtk}=ADTK/SARCLK
- If SARCLK=18MHz, ADTK=13, then 13/18MHz = 230nS

Time for accumulation mode



off = ADC shut down.
PU = Power-Up Phase. Timing Defined by ADPWR field.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

ADC Timing With IPOEN = 1



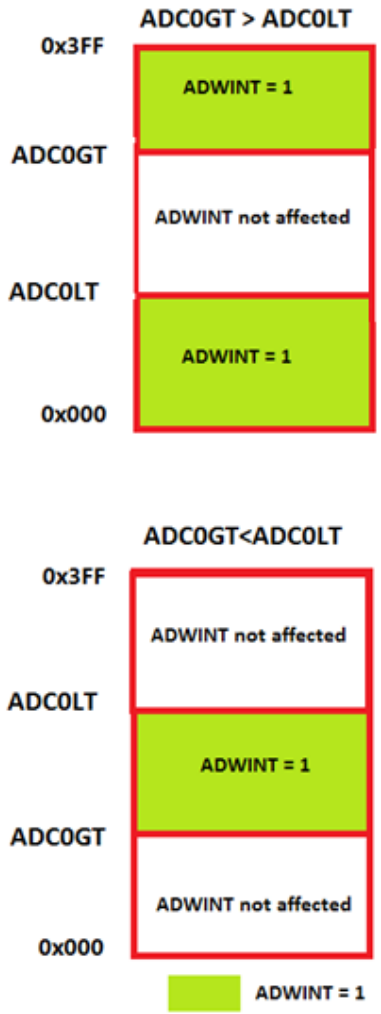
tracking = Converter tracking selected input any time conversion is not in progress.
TK = Tracking Phase. Timing Defined by ADTK field.
CNV = Conversion Phase. Timing depends on resolution and SARCLK.

ADC Timing With IPOEN = 0

$$Total\ \ Conversion\ \ Time = RPT * (ADTK + NUMBITS + 1) * T(\text{SARCLK}) + T(ADCCLK) * 4$$

- If PACEN set to 1, support more samples accumulate.
- Need only 1 trigger.

Window Comparator



- This is useful for some application use scenario, for example AGC, voltage monitor to guarantee within given range
- This is especially effective in an interrupt driven system, saving code space and CPU bandwidth while delivering faster system response times
- Related register
  - EWADC0: enable window detection
  - ADC0LT: lower than threshold
  - ADC0GT: greater than threshold
- Please pay attention how the window is configured
  - If you need trigger interrupt once the ADC value reach a range , please make sure to meet condition ADC0GT<ADC0LT. For example you need fill and unfill some amount water into a cup. Once the target amount of water is filled, the interrupt could stop the filling.
  - Otherwise, you need ADC0GT>ADC0LT, for example you need monitor the power supply voltage not dip too low. You could only trigger interrupt when the ADC value is lower than 0x100. then set ADC0GT=0x3FF and ADC0LT=0x100

Interrupt

- EWADC0/ **ADC0WC_ISR** : window detector interrupt
- EADC0/**ADC0EOC_ISR** : conversion complete interrupt
- We also could use poll mode

Resolution, Accumulation & Shift

- Data may be accumulated over multiple conversions
- The accumulated output may be shifted right by a selectable amount
- Effectively this providing an "accumulate and average" (oversampling) function
- ADRPT (ADC0CN1[2:0]): Accumulation repeat count
- SASJST(ADC0CN1[5:3]): right shift
- PACEN(ADC0CN2[7]): allowed more samples accumulation (>32).
- Need guarantee no saturation for the configuration.

Using ADSJST for Output Formatting (12-bit conversions, ADRPT = 8)

Input Voltage	ADSJST = 0 (no shift)	ADSJST = 1 (shift right 1 bit)	ADSJST = 3 (shift right 3 bits)
VREF - 1 LSB12	0x7FF8	0x3FFC	0x0FFF
VREF / 2	0x4000	0x2000	0x0800
(VREF / 2) - 1 LSB12	0x3FF8	0x1FFC	0x07FF
0	0x0000	0x0000	0x0000

Resolution, Accumulation & Shift

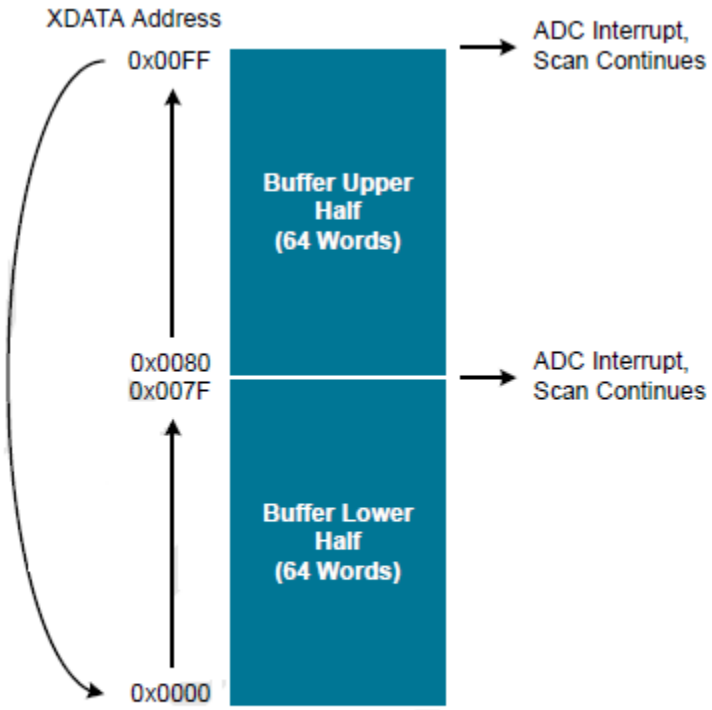
- Based on the datasheet, the 14bit ADC could provide about 12bit ENOB.
- Effectively accumulation and shift provide an oversampling function
- Can improve SNR
- For each additional bit of resolution, the signal must be oversampled by a factor of four
- There is condition for the oversampling to improve ENOB, you could read AN118 to understand how it works.
- The internal noise in the ADC module could make the oversampling works fine to get additional ENOB.

$$SNR(dB) = (6.02 * ENOB) + 1.62$$

Low power mode and Idle power off

- ADLPM(ADC0CF1[7]): decrease current consumption at the cost of additional minimum tracking time
- IPOEN(ADC0CN0[6]): This adds an additional 1.2 us power-up delay.
  - ADPWR(ADC0CF2[0:4]) is to configure the power up time.
- Autoscan mode could also save power and offload the CPU intervention.

Autoscan mode



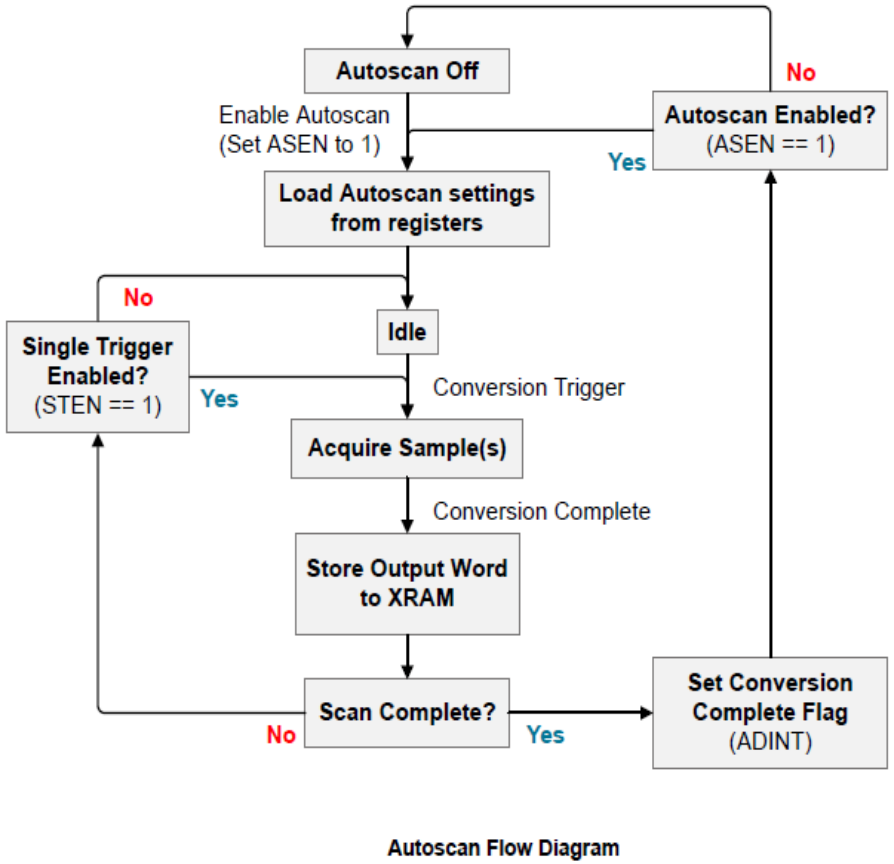
Circular Buffer Example

- Offload CPU to collection information from ADC
  - ASEN(ADC0ASCF[7]): Enable autoscan
- Support up to 4 channels
  - ADC0MX: first channel
  - NASCH(**ADC0ASCF[1:0]**): number of autoscan channel
  - ADRPT still valid.
- 64 words in XDATA space
  - ADC0ASA:start address(**ADC0ASAH**[0:3]+**ADC0ASAL[0:7]**)
  - even numbered address aligned.
  - ASCNT(ADC0ASCT[5:0]): antoscan output count
- STEN([ADC0ASCF[7]) single or multi trigger
- Shadow register allow setting for next scan when current scan is in process

Configuration

- ASEN(ADC0ASCF[7]): Enable autoscan
- Trigger source
  - STEN([ADC0ASCF[7]) single or multi trigger
- Channel selection
  - ADC0MX: first channel
  - NASCH(**ADC0ASCF[1:0]**): number of autoscan channel
  - ADRPT still valid.
- Output data configuration
  - ADC0ASA:start address(**ADC0ASAH**[0:3]+**ADC0ASAL[0:7]**)
  - even numbered address aligned.
  - ASCNT(ADC0ASCT[5:0]): antoscan output count
- Shadow register
  - Register listed above are shadow register, you could change them after conversion start but before finish.

Configuration



Autoscan Flow Diagram

- 1. Configure the autoscan in the ADC0ASAH, ADC0ASAL, AD0ASCNT and ADC0MX registers.
- 2. Write ASEN to 1. The autoscan settings are now loaded and the scan is enabled.
- 3. Write ASEN to 0. Autoscan mode will be disabled after the scan completes.
- 4. Begin ADC Conversions.
- 5. Wait for the scan to be completed as indicated by the ADINT bit in the ADC0CN0 register. If Autoscan Single Trigger is enabled
- (STEN is 1 in the ADC0ASCF register), this will take as many triggers as the scan is configured to take samples. If Single Trigger is
- disabled, only one ADC trigger is required to complete the entire scan.
- 6. Process the data.

Configuration

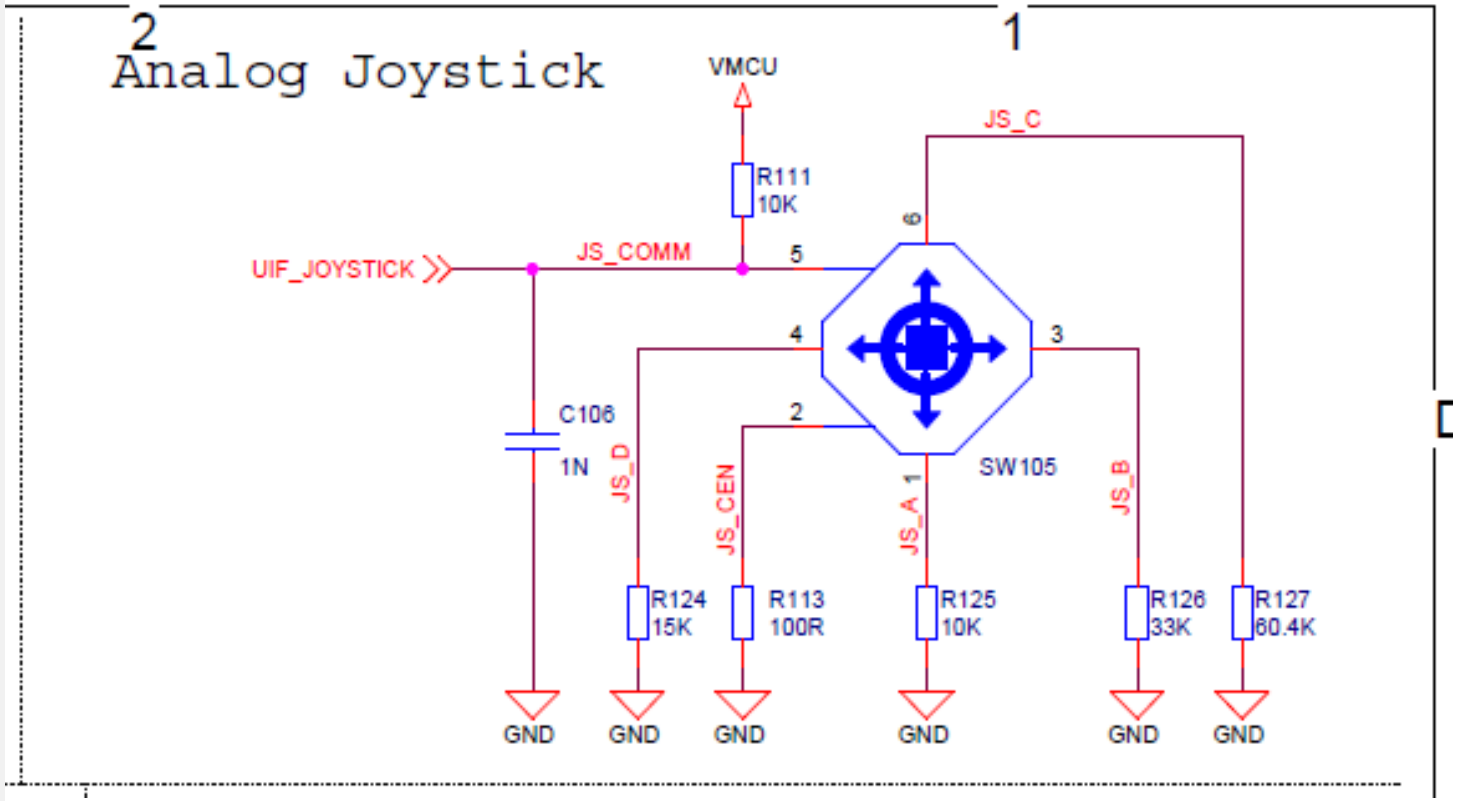
- Silabs provide below 4 samples for the autoscan mode
- Single scan
  - Collect multi samples per scan
- **Circular Buffer**
  - Like ping-pong mode DMA.
  - Shadow register
- Large buffer
  - 2048-word buffer for a single ADC channel
  - Using autoscan for this purpose reduces the requirement for CPU intervention from 2048 instances to 32 instances
- **Single Scan of Two Channels**
  - 2 channels and collect 32 32 samples per channel

Software Example

- You could get example code in below folder if you install the Simplicity Studio
  - C:\SiliconLabs\SimplicityStudio\v3\developer\sdks\si8051\v3\examples\EFM8LB1_SLSTK2030A\ADC
- Silabs also provide ADC peripheral driver library.
  - C:\SiliconLabs\SimplicityStudio\v3\developer\sdks\si8051\v3\Device\EFM8LB1\peripheral_driver
- Need include Adc_0.h

- Let us take the ExternalInput sample as a demo.
  - This example code takes and averages 2048 analog measurements from input P1.7 using ADC0, then prints the average results to a window terminal via the UART interface.
  - An analog joystick is connected to P1.7. Each joystick position corresponds to a unique voltage

Software Example



- P1.7 voltage: 3300 mV
- P1.7 voltage: 3300 mV
- P1.7 voltage: 1648 mV
- P1.7 voltage: 1648 mV
- P1.7 voltage: 1648 mV
- P1.7 voltage: 1648 mV
- P1.7 voltage: 1980 mV
- P1.7 voltage: 1980 mV

Calibration and Additional Resources

- Customer could calibrate the ADC offset and slope is needed. But LB1/BB3 don’t have register to save offset and slope for auto calibration (like F350 delta-sigma ADC).
- Offline calibrated offset and gain result could be saved in flash and be used in runtime.
- AN119: Calculating Settling Time For Switched Capacitor ADC’s
- AN118: Improving ADC Resolution by Oversampling and Averaging

Thank you!