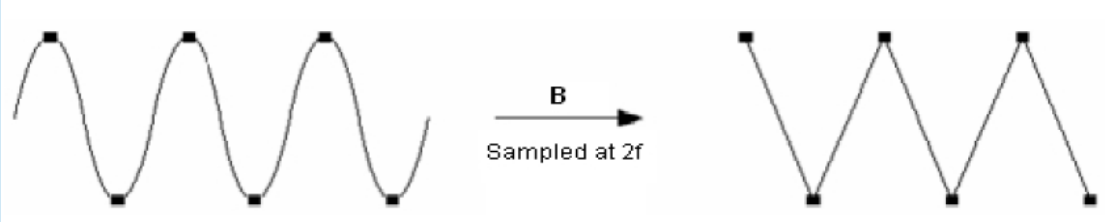
Vlastnosti ADC

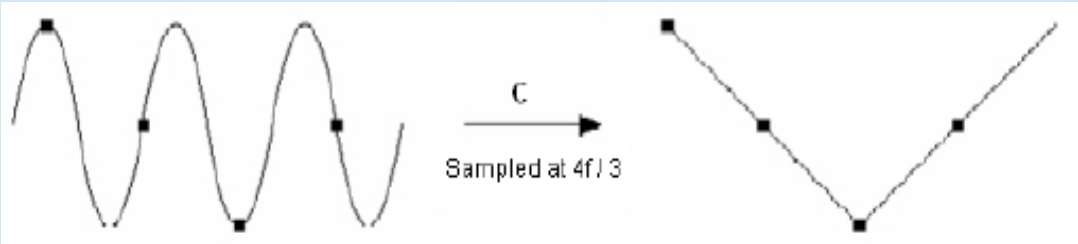
- Převod analogové veličiny (napětí) na binární hodnotu
- Rozsah vstupního napětí – referenční napětí, napájecí napětí, interní zesílení, diferenciální vstupy
- Počet vstupů (multiplex), počet AD převodníků (nezávislé, paralelní, fázově posunuté vzorkování)
- Spouštění převodu (trigger) – externí, interní časovače, volně běžící
- Sample & Hold obvod
- Rychlost vzorkování – samples/s, hodinová frekvence, počet taktů pro kompletní převod + Sample & Hold
- Rozlišení – počet bitů (8, 10, 12 … 24 …)
- Přenos dat – programově, DMA, automatické bitové zarovnání MSB/LSB

Vzorkování

- Minimální rychlost vzorkování – Nyquistovo kritérium, vzorkovací frekvence > ½ frekvence nejvyšší harmonické složky signálu



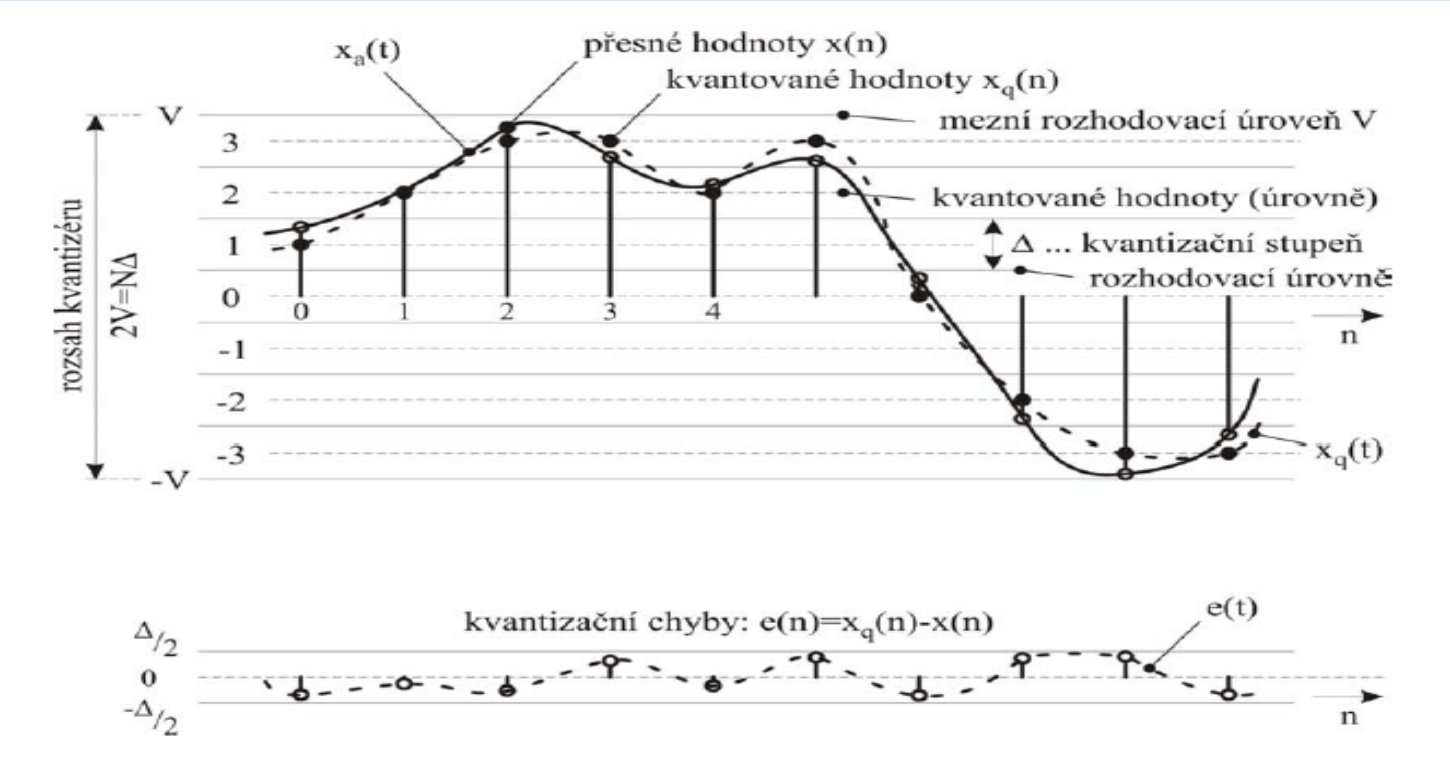
- Aliasing – znehodnocení signálu vyššími frekvencemi než odpovídají Nyq. kritériu – filtr před převodníkem, dolní propust



- Např. CD Audio – vzorkování 44.1kHz, max. Zaznamenaná sinusová frekvenční složka 22kHz;
  Telekomunikace – pásmo 0,3 .. 3,4 kHz, vzorkování 8kHz
  Digitalizace TV obrazu – pásmo jednotky až desítky MHz

Kvantování

- Rozlišení, počet možných celočíselných hodnot jako 2^n
- Kvantizační šum – odchylky celočíselné hodnoty od skutečné hodnoty signálu v čase vzorkování
- Nelinearita – vyjádřena v počtu bitů LSB (½, 1 ...)

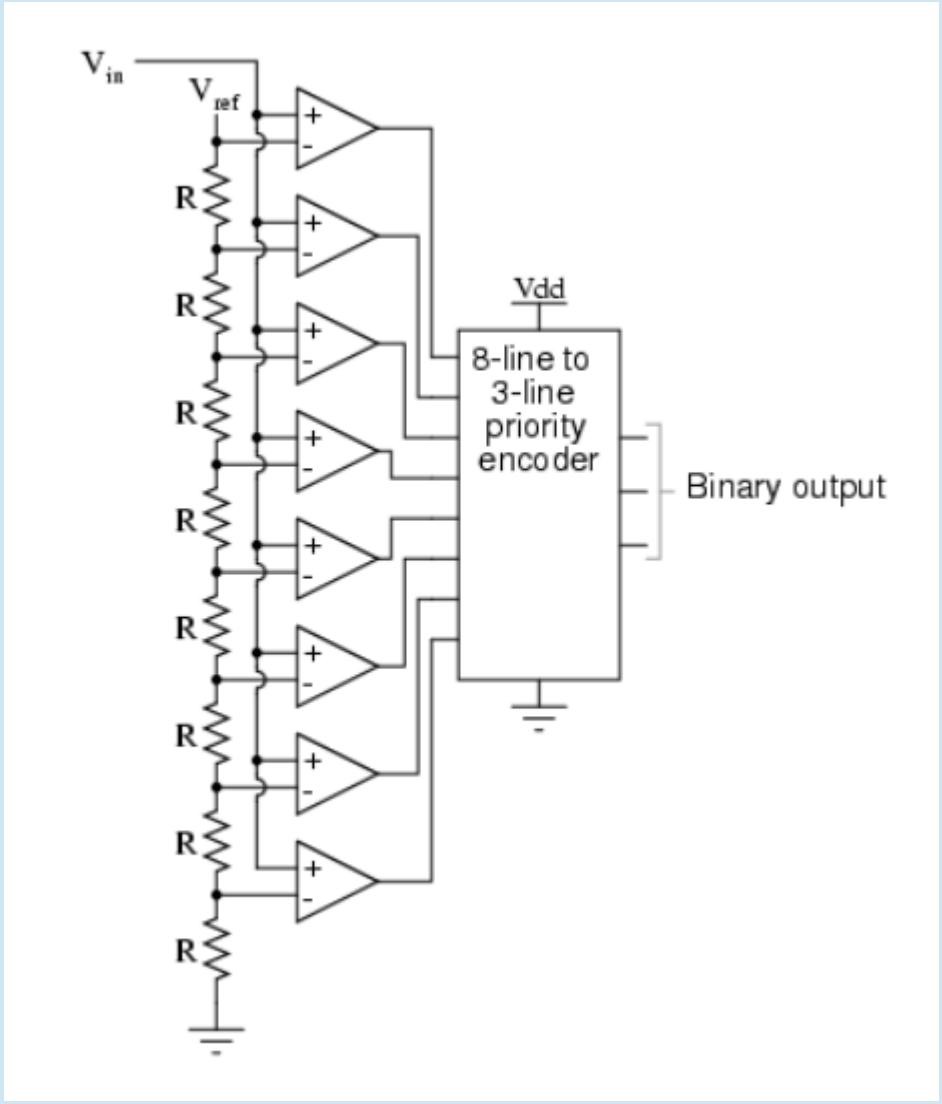


Typy AD převodníků

- Paralelní (FLASH) – nejrychlejší vzorkování
- Sigma-Delta – vysoké rozlišení, menší frekvence vzorkování
- Integrační
- Aproximační – relativně rychlé, převažující typ integrovaný v MCU

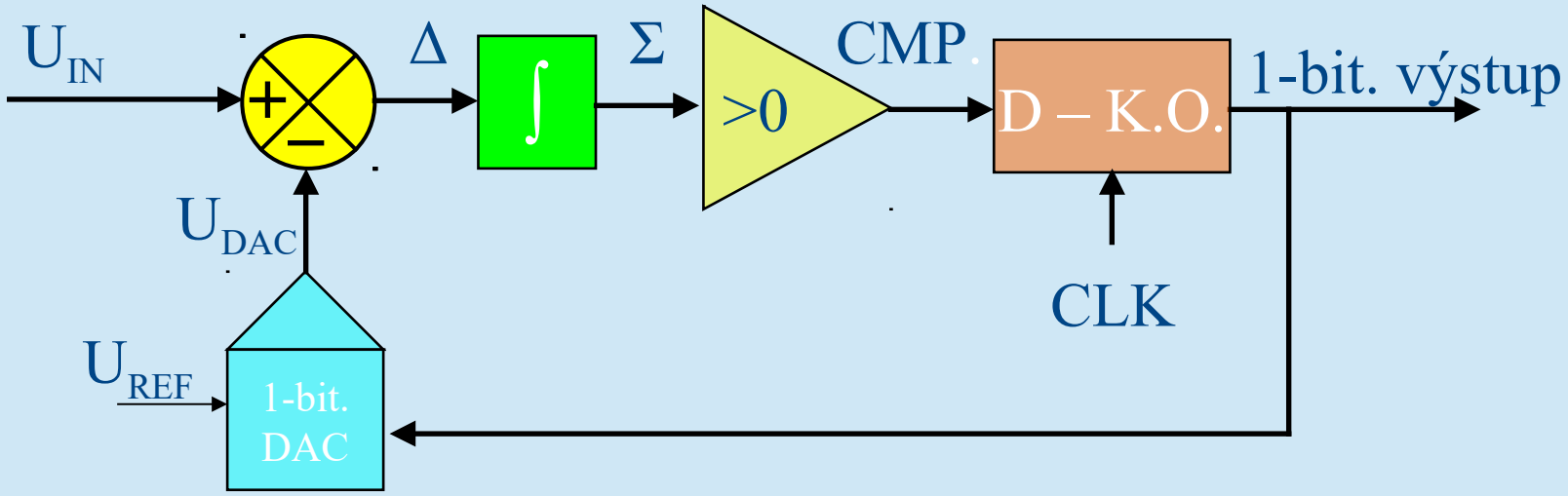
Paralelní AD převodník

Realizován jako řetězec rychlých analogových komparátorů následovaný kombinační logikou pro převod hodnoty 1/N na binární (prioritní enkoder)

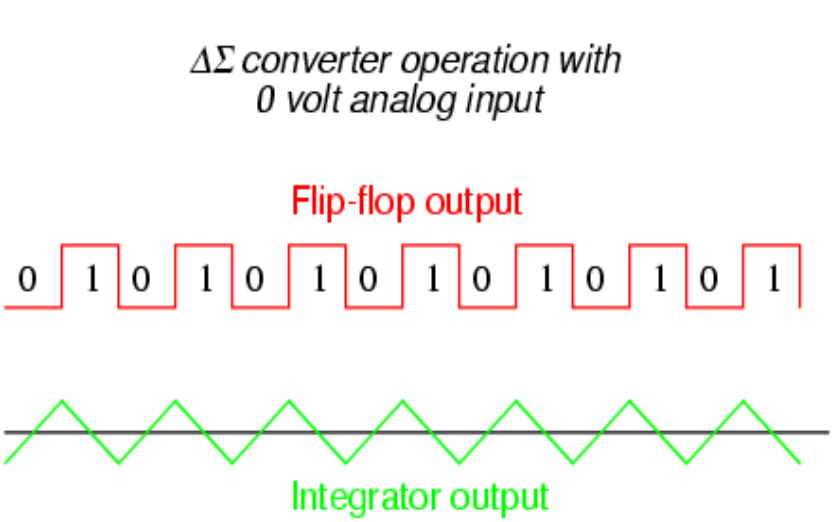


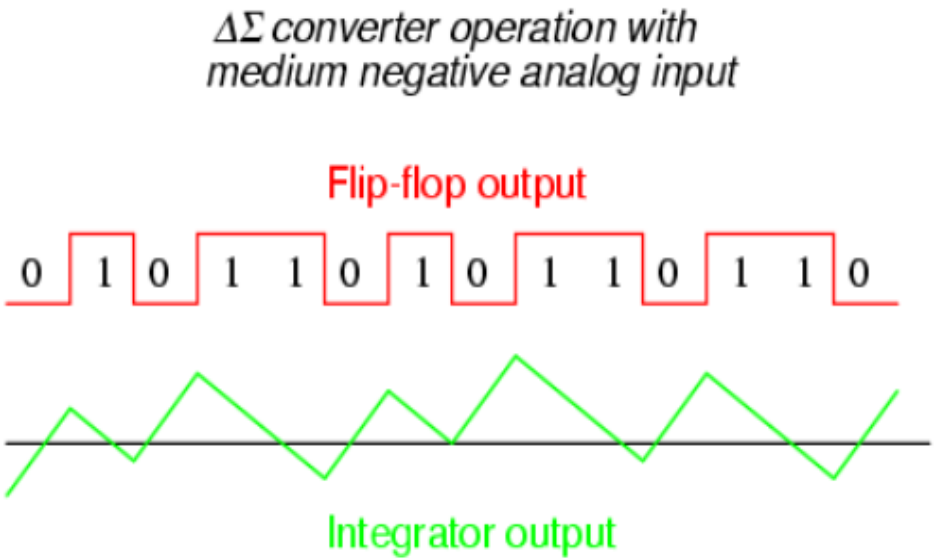
Sigma-Delta převodník

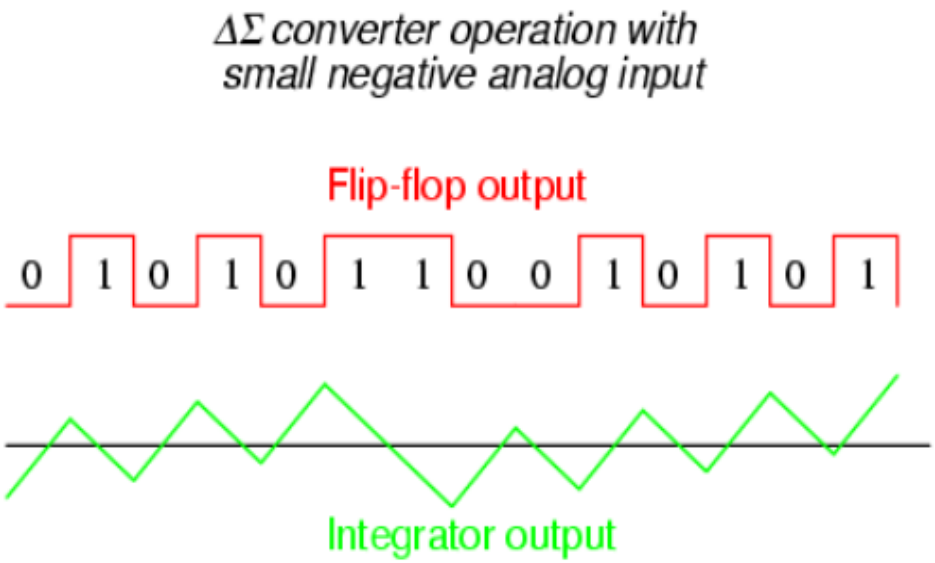
- Integrační princip, integruje součet vstupního signálu a signálu interního DA převodníku (1 bit)
- Nepotřebuje přesné nastavovaní obvodových prvků
- Hodnota vstupu úměrná poměrnému zastoupení jedniček ve výstupním bitovém proudu → velký počet hodinových pulzů pro vyšší rozlišení → relativně pomalé vzorkování

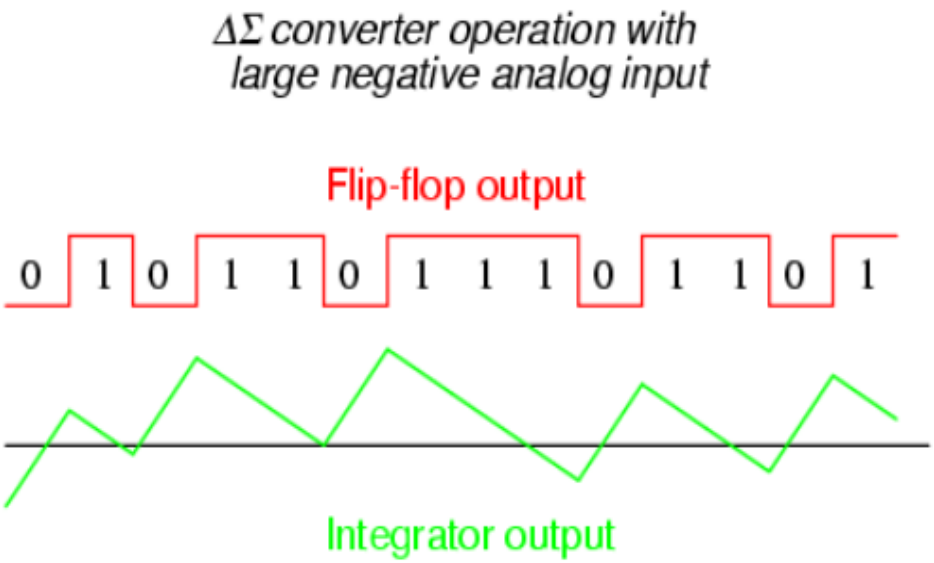


Sigma-Delta převodník



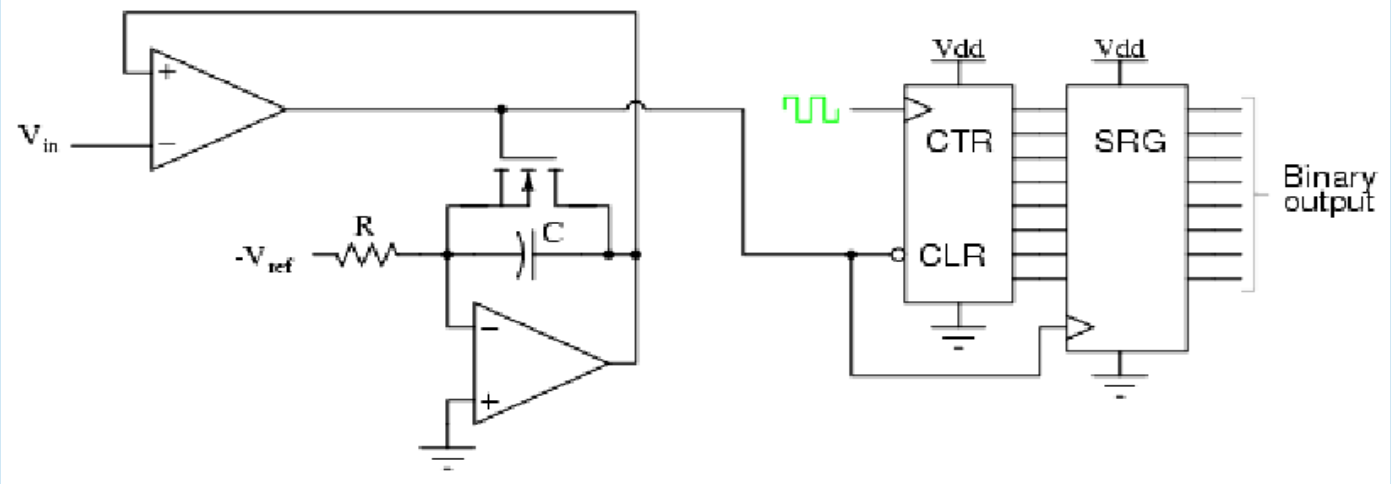






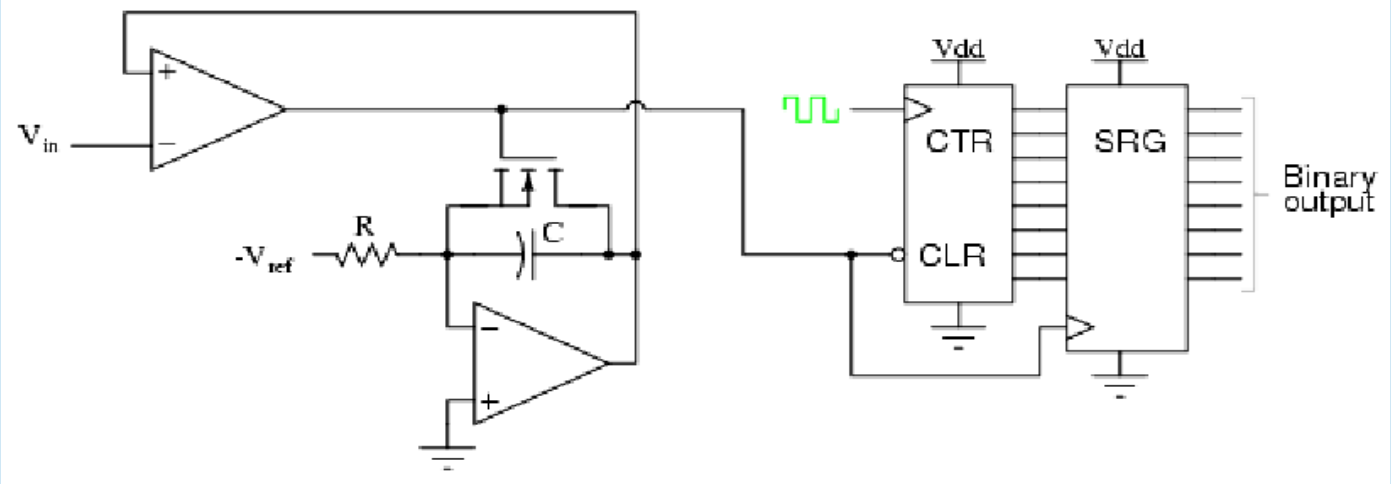
Integrační převodník

- střední rozlišení (10-18 bitů)
- nízká rychlost (< 100 S/s)
- vysoká odolnost proti šumu
- nízká cena
- jednoduché indikátory, aplikace nenáročné na rychlost, multimetry
- Vícenásobná integrace – redukce nestálosti součástek převodníku
- Vhodné časování – odstranění n*50/60Hz



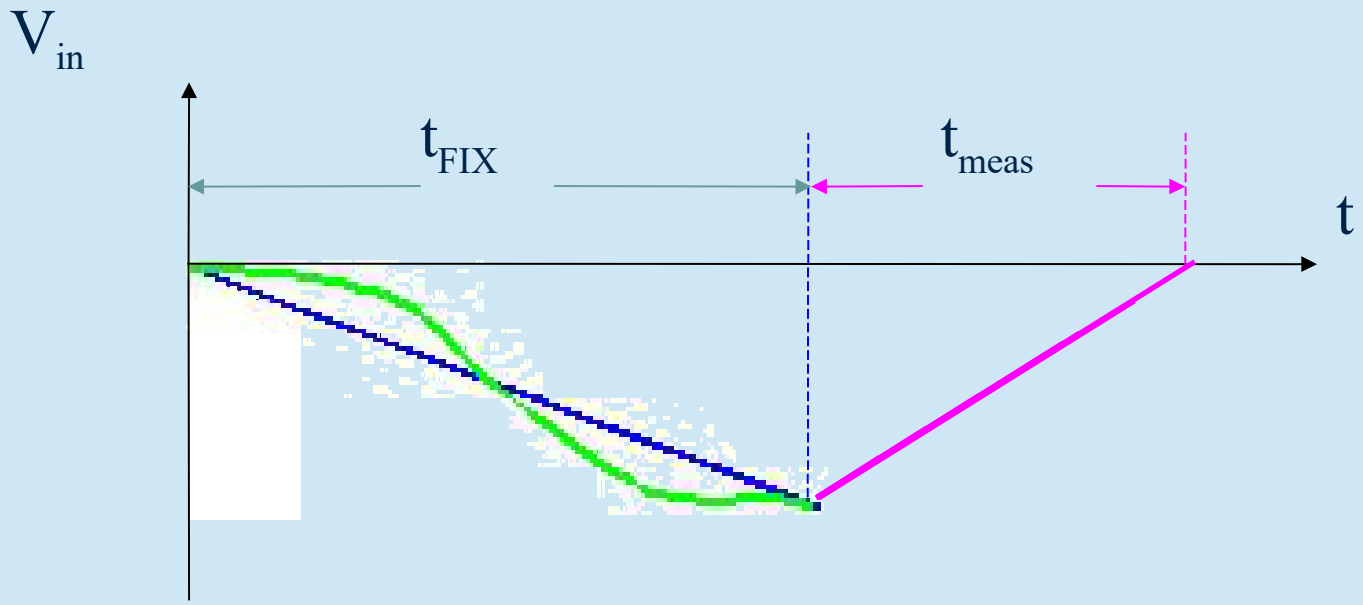
Integrační převodník

- střední rozlišení (10-18 bitů)
- nízká rychlost (< 100 S/s)
- vysoká odolnost proti šumu
- nízká cena
- jednoduché indikátory, aplikace nenáročné na rychlost, multimetry
- Vícenásobná integrace – redukce nestálosti součástek převodníku
- Vhodné časování – odstranění n*50/60Hz



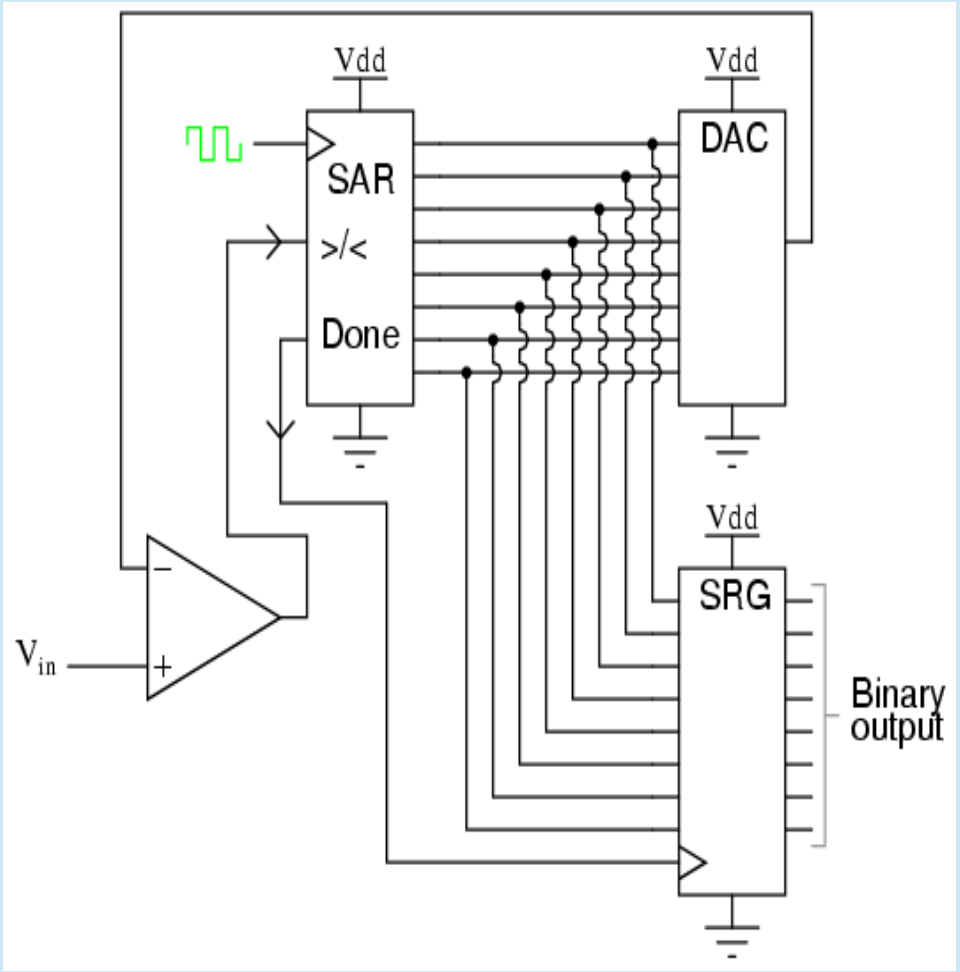
Integrační převodník (s dvojitou integrací)

- Pevný časový interval pro integraci vstupního signálu
- Integrace referenčního napětí a měření časového intervalu do dosažení výchozí hodnoty integrátoru (0)
- Delší doba integrace → vyšší rozlišení, pomalejší vzorkování

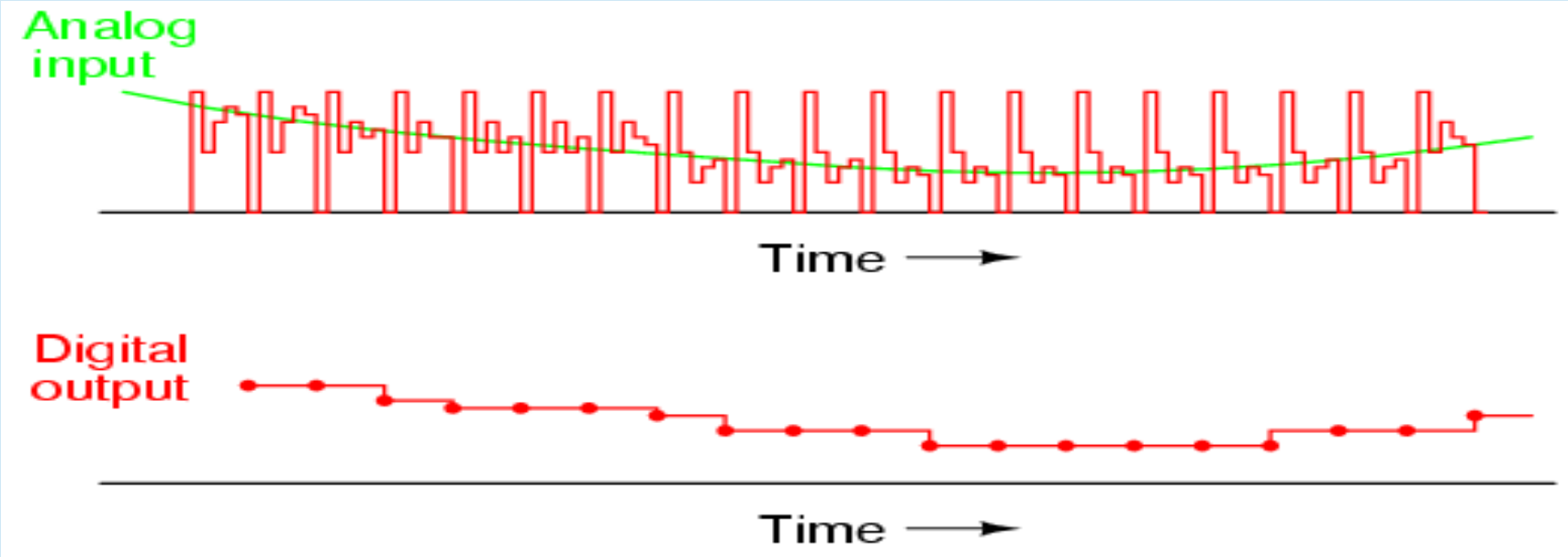


Aproximační převodník (SAR)

- Porovnání vstupního napětí s výstupem interního DA převodníku
- Na principu půlení intervalu
- Během převodu musí být vstupní signál konstantní (Sample & Hold obvod)

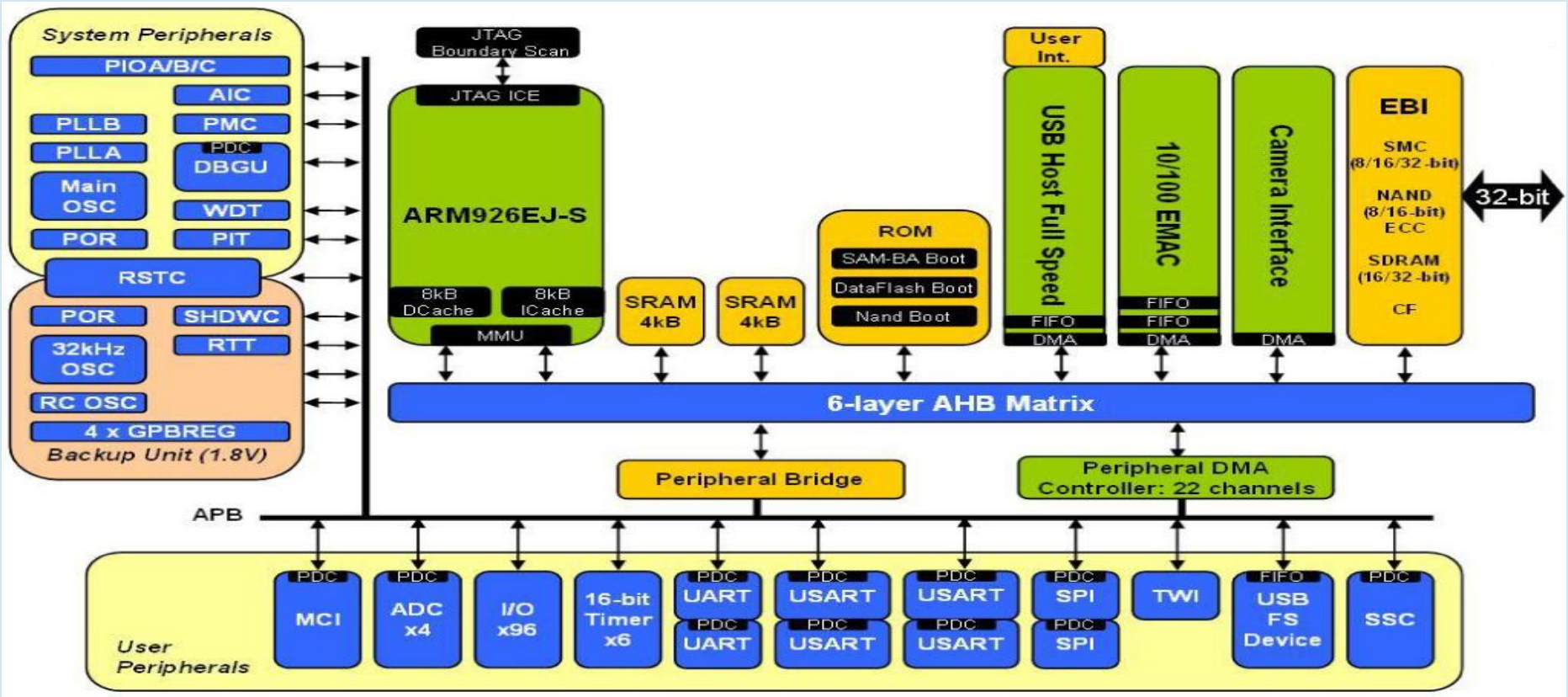


Aproximační převodník (SAR)

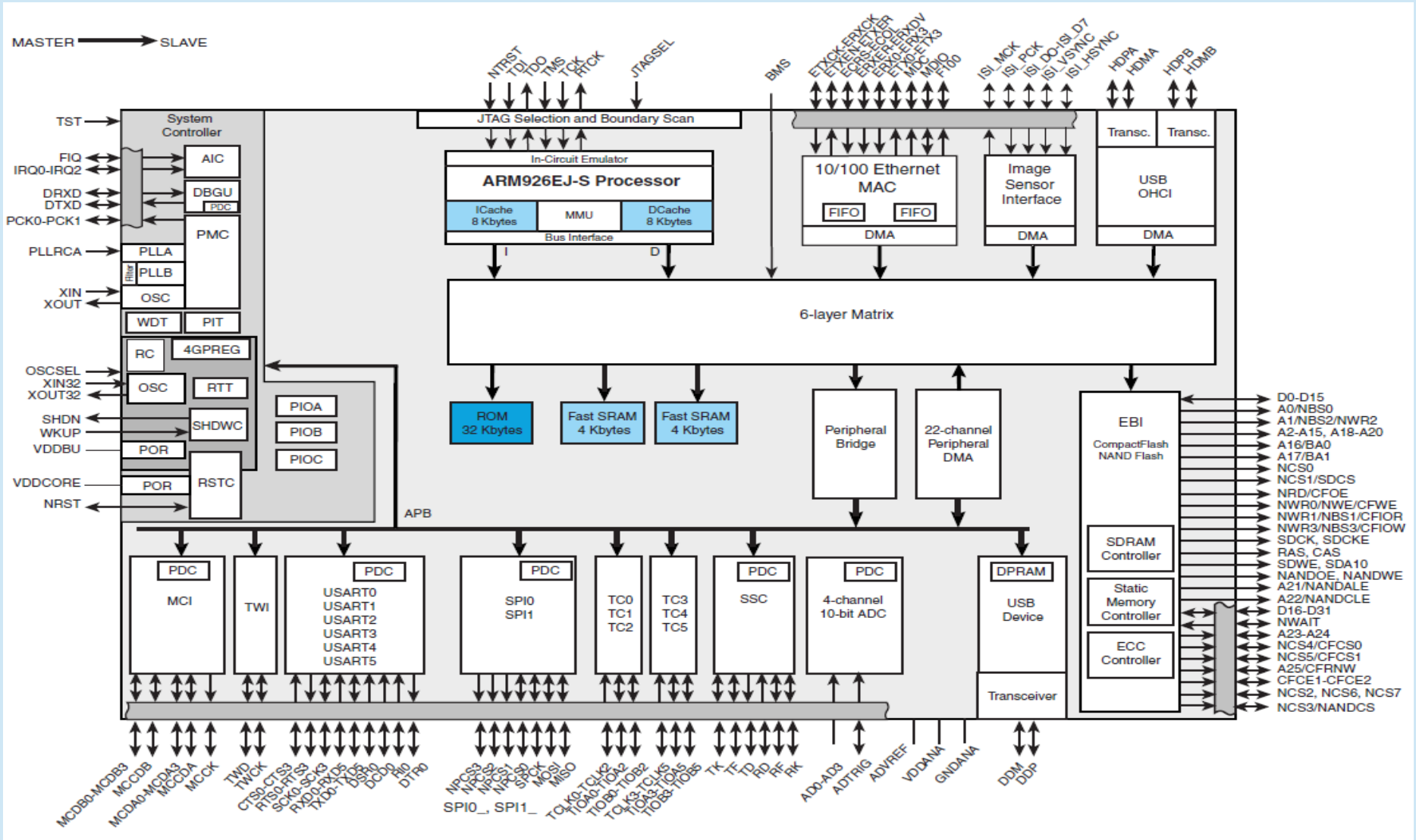


- Jednoduchá realizace, relativně rychlé (až cca 5 MS/s)
- Střední rozlišení a přesnost (8-16b, nejčastěji 10/12b)
- Použit ve většině MCU – vyhovuje pro většinu běžných měření
- Možnost sériového výstupu dat během převodu

AT91SAM9260

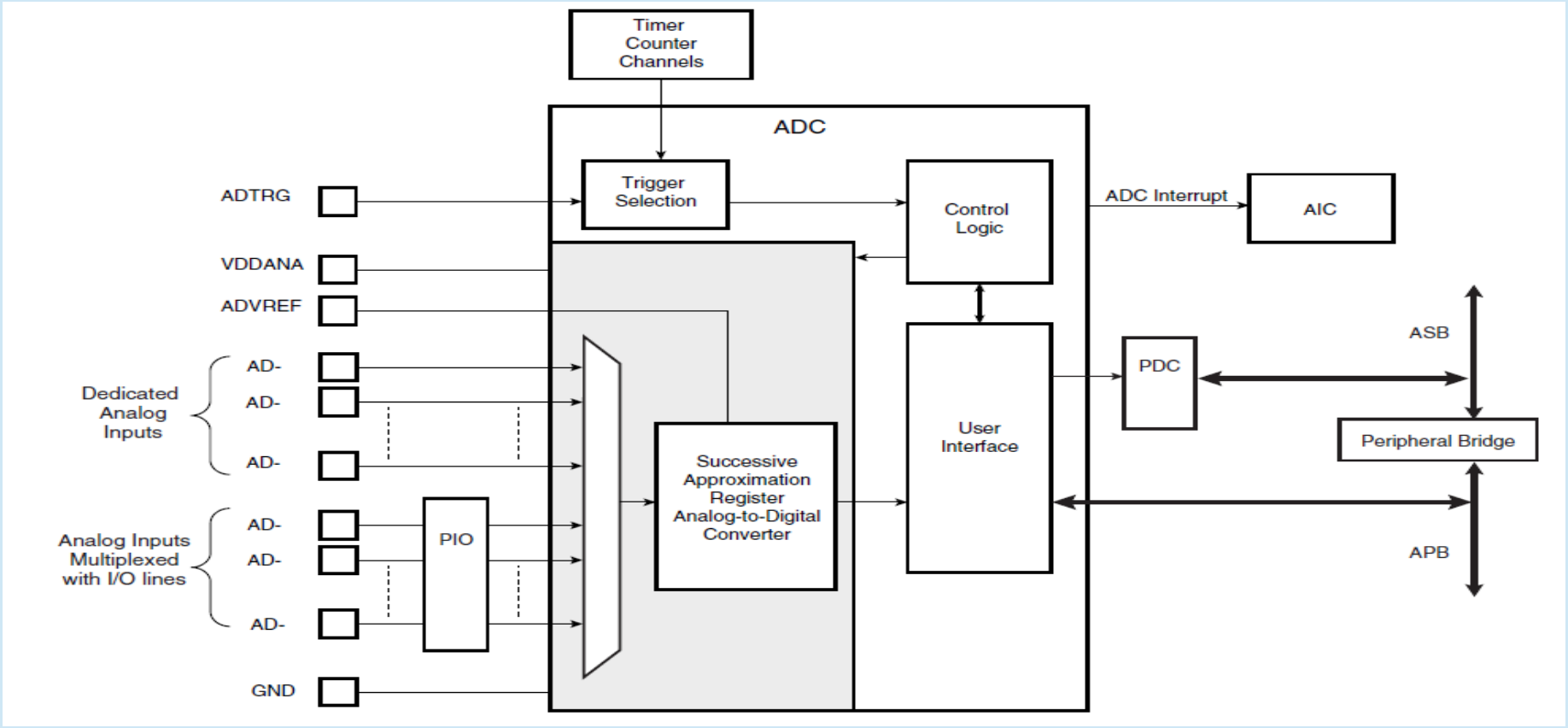


Blokové schéma AT91SAM9260



AD převodník v AT91SAM9260

- 10/8 bitů
- 2 multiplexované kanály (4 u BGA verze)
- Spouštění převodu softwarově, časovačem nebo externím signálem



Registry AD převodníku

Offset	Register	Name	Access	Reset
0x00	Control Register	ADC_CR	Write-only	–
0x04	Mode Register	ADC_MR	Read-write	0x00000000
0x08	Reserved	–	–	–
0x0C	Reserved	–	–	–
0x10	Channel Enable Register	ADC_CHER	Write-only	–
0x14	Channel Disable Register	ADC_CHDR	Write-only	–
0x18	Channel Status Register	ADC_CHSR	Read-only	0x00000000
0x1C	Status Register	ADC_SR	Read-only	0x000C0000
0x20	Last Converted Data Register	ADC_LCDR	Read-only	0x00000000
0x24	Interrupt Enable Register	ADC_IER	Write-only	–
0x28	Interrupt Disable Register	ADC_IDR	Write-only	–
0x2C	Interrupt Mask Register	ADC_IMR	Read-only	0x00000000
0x30	Channel Data Register 0	ADC_CDR0	Read-only	0x00000000
0x34	Channel Data Register 1	ADC_CDR1	Read-only	0x00000000
...	...	...	...	...
0x40	Channel Data Register 3	ADC_CDR3	Read-only	0x00000000
0x44 - 0xFC	Reserved	–	–	–

Programové ovládání ADC

Výrobce poskytuje modul adc.c a adc.h s funkcemi pro inicializaci a obsluhu AD převodníku

```
//------------------------------------------------------------------------------
//         Exported functions
//------------------------------------------------------------------------------
extern void ADC_Initialize (AT91S_ADC *pAdc,
                            unsigned char idAdc,
                            unsigned char trgEn,
                            unsigned char trgSel,
                            unsigned char sleepMode,
                            unsigned char resolution,
                            unsigned int mckClock,
                            unsigned int adcClock,
                            unsigned int startupTime,
                            unsigned int sampleAndHoldTime);
extern unsigned int ADC_GetModeReg(AT91S_ADC *pAdc);
extern void ADC_EnableChannel(AT91S_ADC *pAdc, unsigned int channel);
extern void ADC_DisableChannel (AT91S_ADC *pAdc, unsigned int channel);
extern unsigned int ADC_GetChannelStatus(AT91S_ADC *pAdc);
extern void ADC_StartConversion(AT91S_ADC *pAdc);
extern void ADC_SoftReset(AT91S_ADC *pAdc);
extern unsigned int ADC_GetLastConvertedData(AT91S_ADC *pAdc);
extern unsigned int ADC_GetConvertedData(AT91S_ADC *pAdc, unsigned int channel);
extern void ADC_EnableIt(AT91S_ADC *pAdc, unsigned int flag);
extern void ADC_EnableDataReadyIt(AT91S_ADC *pAdc);
extern void ADC_DisableIt(AT91S_ADC *pAdc, unsigned int flag);
extern unsigned int ADC_GetStatus(AT91S_ADC *pAdc);
extern unsigned int ADC_GetInterruptMaskStatus(AT91S_ADC *pAdc);
extern unsigned int ADC_IsInterruptMasked(AT91S_ADC *pAdc, unsigned int flag);
extern unsigned int ADC_IsStatusSet(AT91S_ADC *pAdc, unsigned int flag);
extern unsigned char ADC_IsChannelInterruptStatusSet(unsigned int adc_sr,
                                                     unsigned int channel);
```

Programové ovládání ADC

Příklad použití v programu:

```
static const Pin pinsADC[] = {PINS_ADC};
```

```
PIO_Configure(pinsADC, PIO_LISTSIZE(pinsADC));

ADC_Initialize( AT91C_BASE_ADC,
                AT91C_ID_ADC,
                AT91C_ADC_TRGEN_DIS,
                0,
                AT91C_ADC_SLEEP_NORMAL_MODE,
                AT91C_ADC_LOWRES_10_BIT,
                BOARD_MCK,
                ADC_MAX_CK_10BIT,
                10,
                1200);
```

```
ADC_EnableChannel(AT91C_BASE_ADC, ADC_CHANNEL_0);

for (;;)
{
    ADC_StartConversion(AT91C_BASE_ADC);

    while (!ADC_IsStatusSet(AT91C_BASE_ADC, AT91C_ADC_EOC0));

    int adc = ADC_GetConvertedData(AT91C_BASE_ADC, ADC_CHANNEL_0);
    printf ("ADC = %04X, %d\n", adc, adc);
    uint32_t tck = GetTickCount();
    while (GetTickCount() - tck < 200);
}
```