CS4101 嵌入式系統概論

Analog-to-Digital Converter

Prof. Chung-Ta King

Department of Computer Science

National Tsing Hua University, Taiwan

Materials from *MSP430 Microcontroller Basics*, John H. Davies, Newnes, 2008

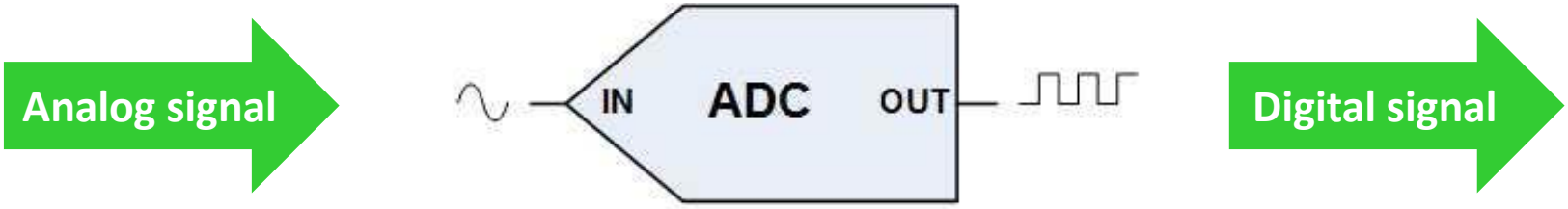
Recall the Container Thermometer

- Container thermometer: monitor the temperature of the interior of a container
  - Monitor the temperature every 5 minutes
  - Flash LED alarm at 1 Hz
  - If the temperature rises above a threshold, flash the LED alarm at 3 Hz and notify backend server
  - If the temperature drops below a threshold, return the LED alarm to normal and notify the server

Need to know the temperature!

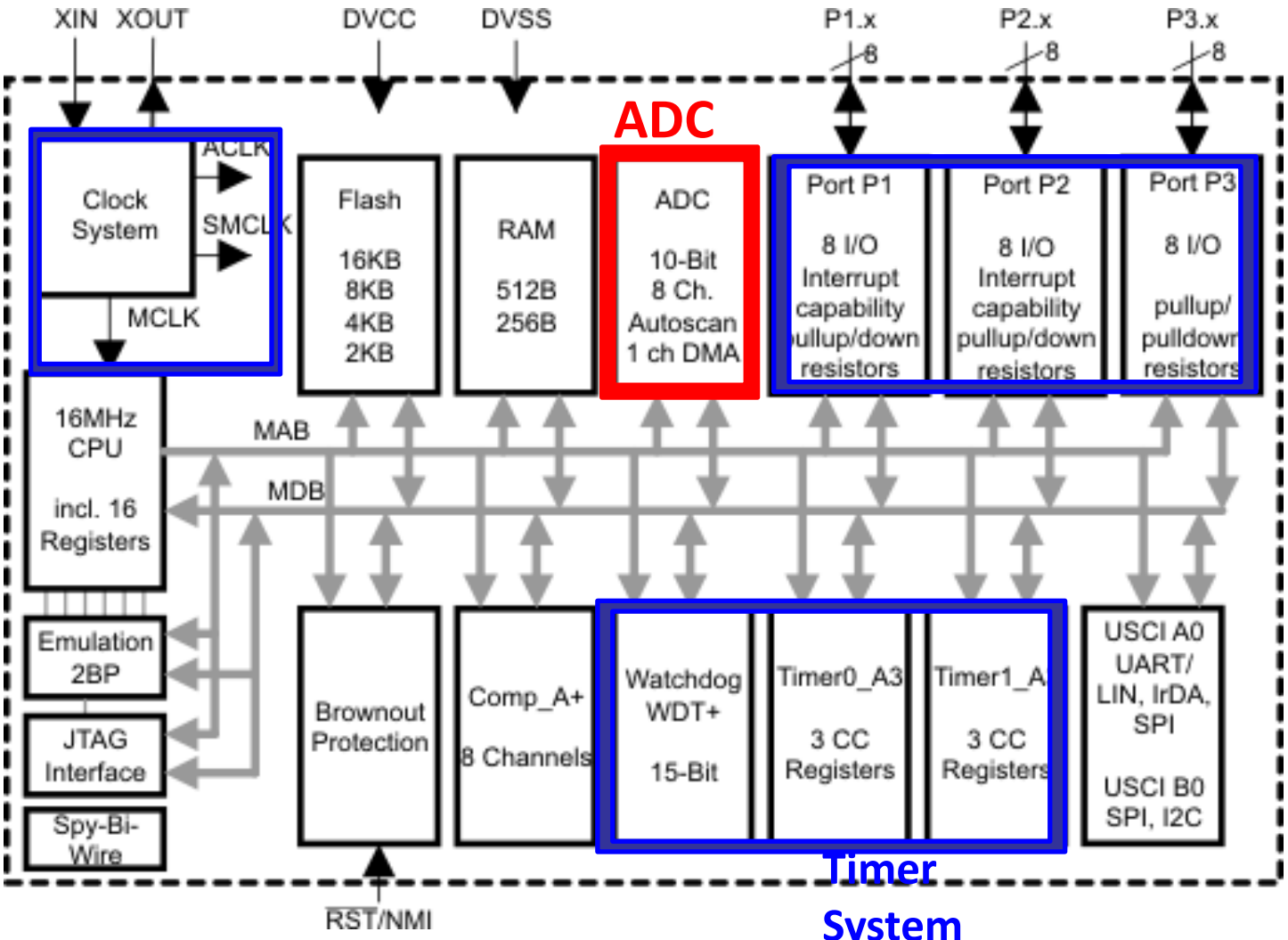
Digitizing Temperature

- Temperature is a nature phenomena, whose value vary continuously
- To make it feasible for computer to handle, we need to convert it into digital signals
- To transform an analog signal into a digital one, the *analog-to-digital converter* (ADC) samples the input at fixed interval and do the conversion



ELECTRICAL SYMBOL FOR ANALOG TO DIGITAL CONVERTER (ADC)

We Have Learned …



Clock System

IO

Timer

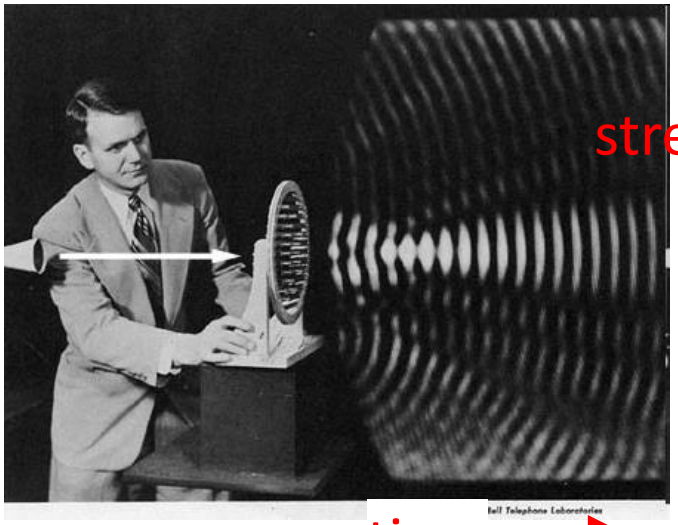
System

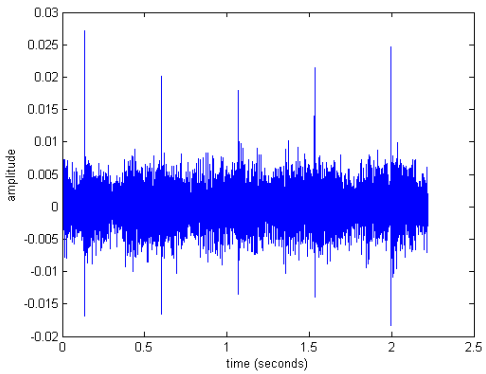
Outline

- Introduction to analog-to-digital conversion
- ADC of MSP430
- Sample code of using ADC10 in MSP430

Analog Signals

- A signal representing continuous things, e.g.
  - Fluctuations in air pressure (i.e. sound) strike the diaphragm of a microphone, which causes corresponding fluctuations in a voltage or the current in an electric circuit
  - The voltage or current is an "analog" of the sound





Analog-to-Digital Conversion

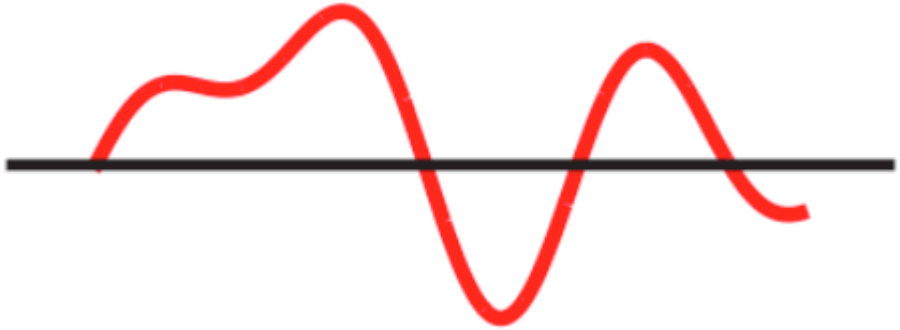
- ADC: convert an analog input, e.g., a voltage V, into a binary value that the processor can handle
  - The input V(t) is a continuous function, i.e., V can take any value within a permitted range and can change in any way as a function of time t
  - The output V[n] is a sequence of binary values. Each has a fixed number of bits and can represent only a finite number of values.
  - Typically input is sampled regularly at intervals of Ts, so the continuous nature of time has also been lost.

Of course, we also have DAC (digital-to-analog converter)!

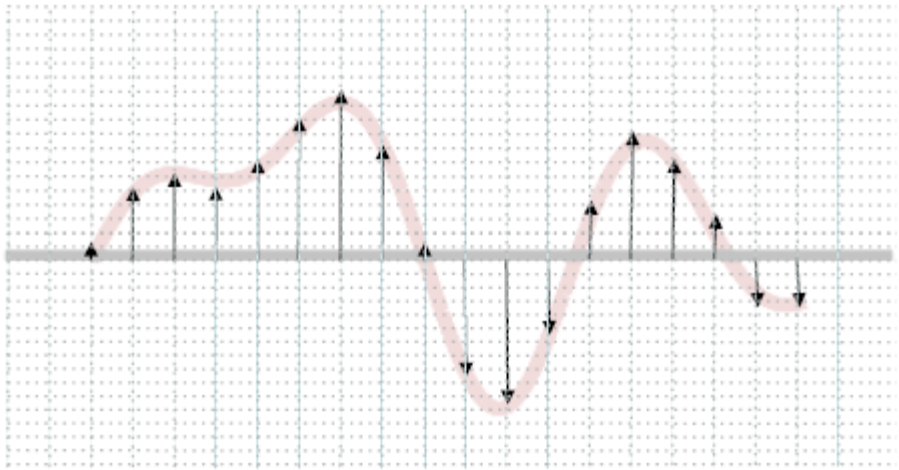
Analog-to-Digital Conversion

- Digital representations of analog waveforms

Continuous time
Continuous values

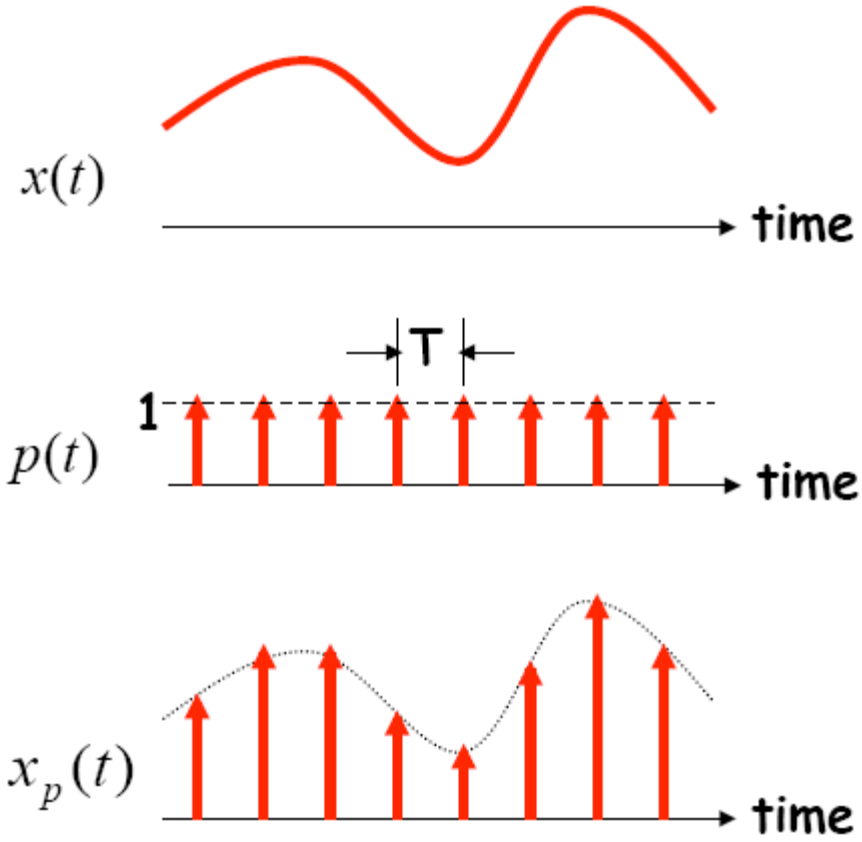


Discrete time
Discrete values



Sampling in Time

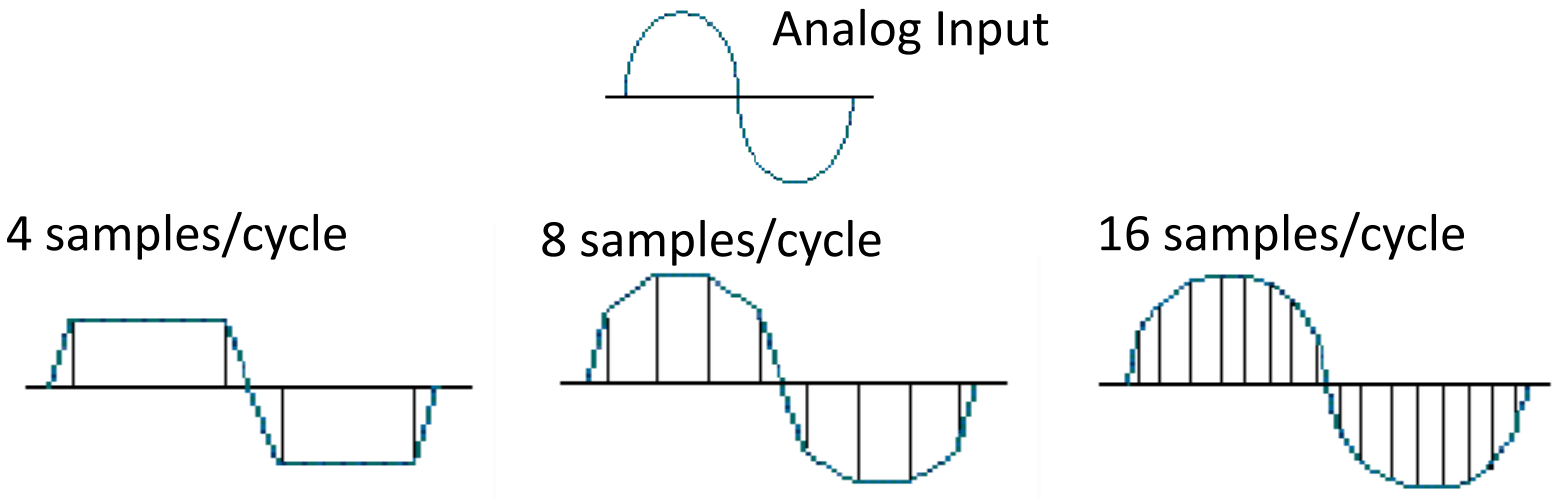
- The value of the analog signal is measured at certain intervals in time. Each measurement is referred to as a sample



A series of "snapshots"

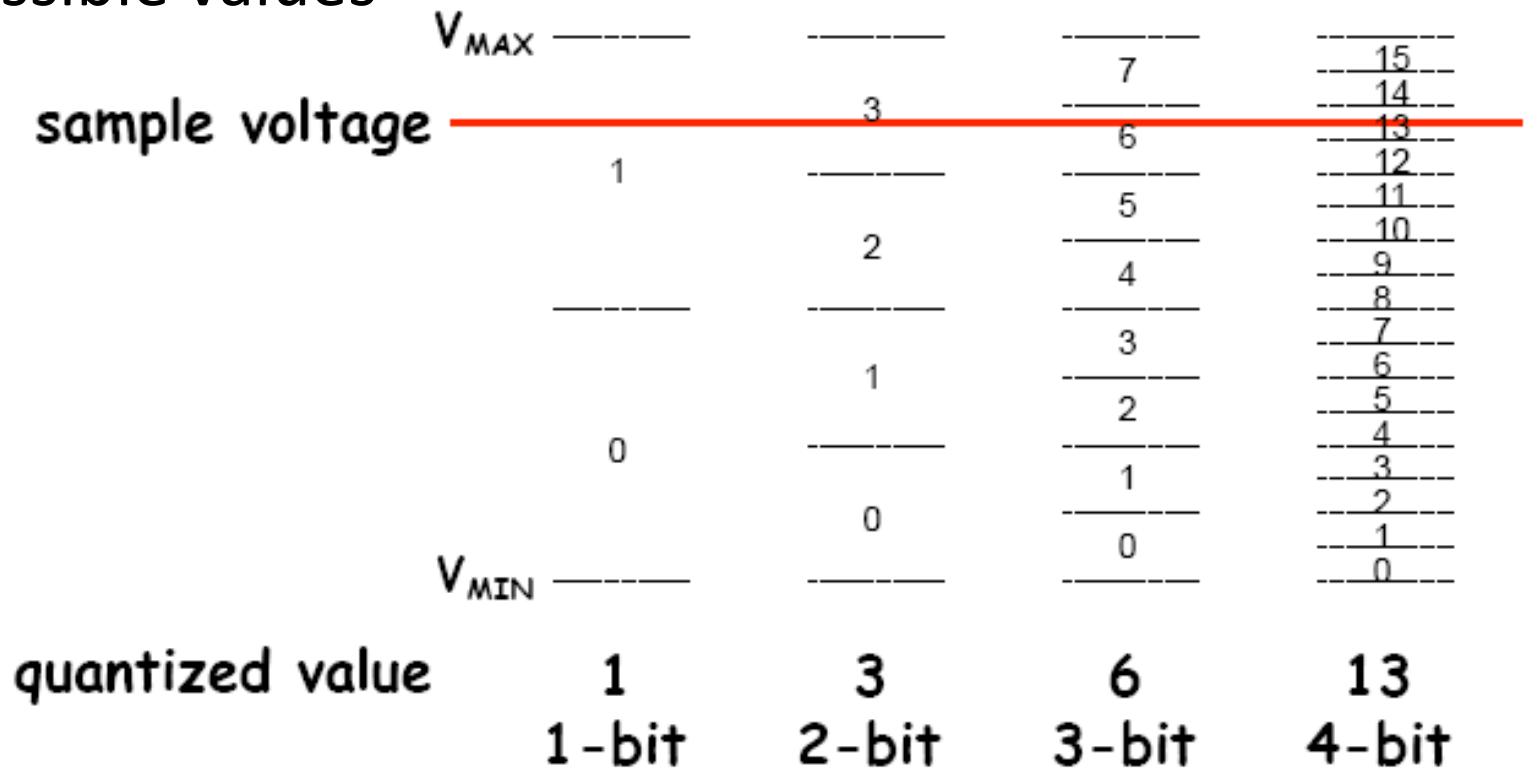
Terminologies in Sampling

- Sampling rate:
  - How often analog signal is measured (samples per second, Hz), e.g. 44,100 Hz?
- Sampling resolution:
  - Number of bits to represent each sample ("sample word length," "bit depth"), e.g. 16 bit

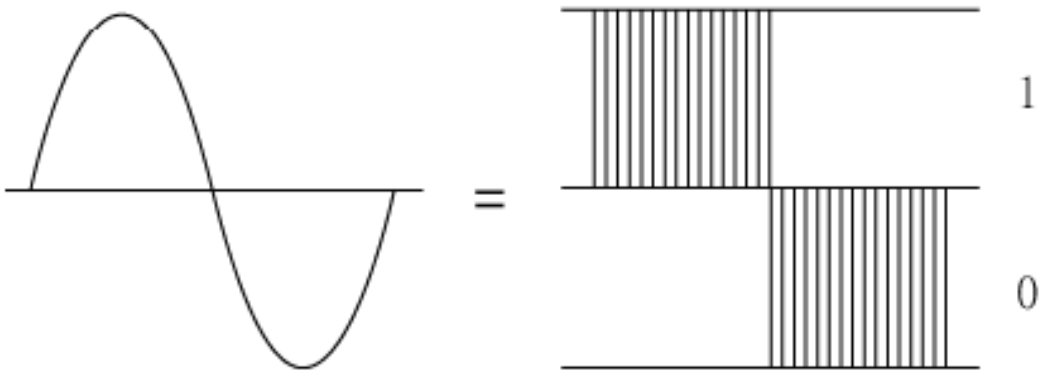


Encoding of Discrete Signals

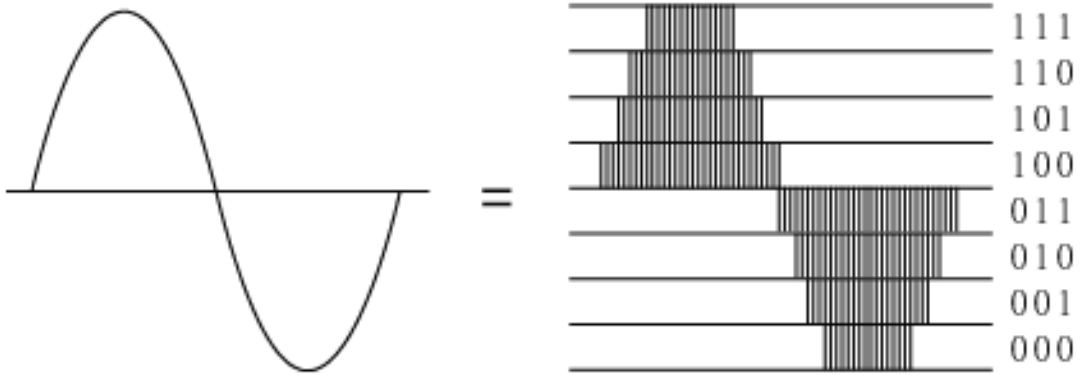
- If we use N bits to encode the magnitude of one of the discrete-time samples, we can capture 2^N possible values



Sampling Rate and Encoding Bits



1-bit



3-bit

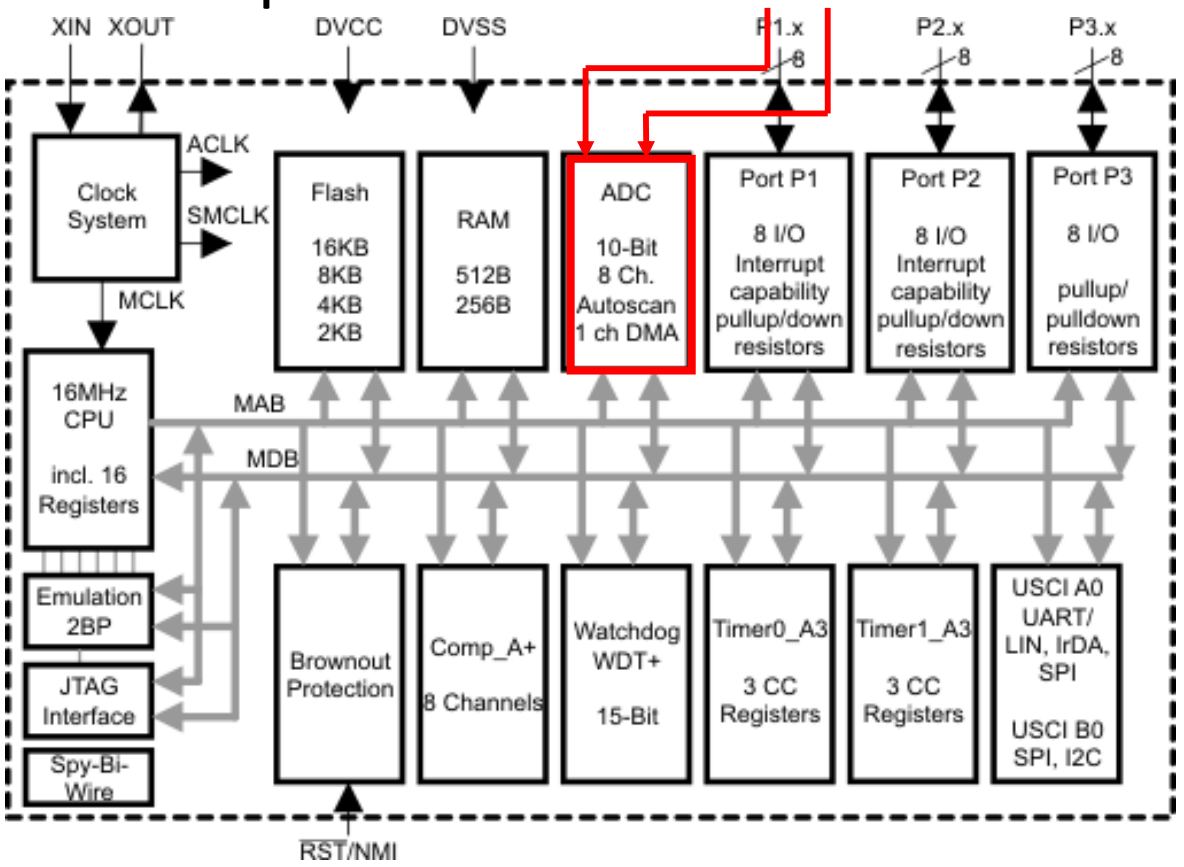
Outline

- Introduction to analog-to-digital conversion
- ADC of MSP430
- Sample code of using ADC10 in MSP430

Requirements of MSP430 for ADC

- Provide continuous sampling of multiple analog inputs and store sampled data

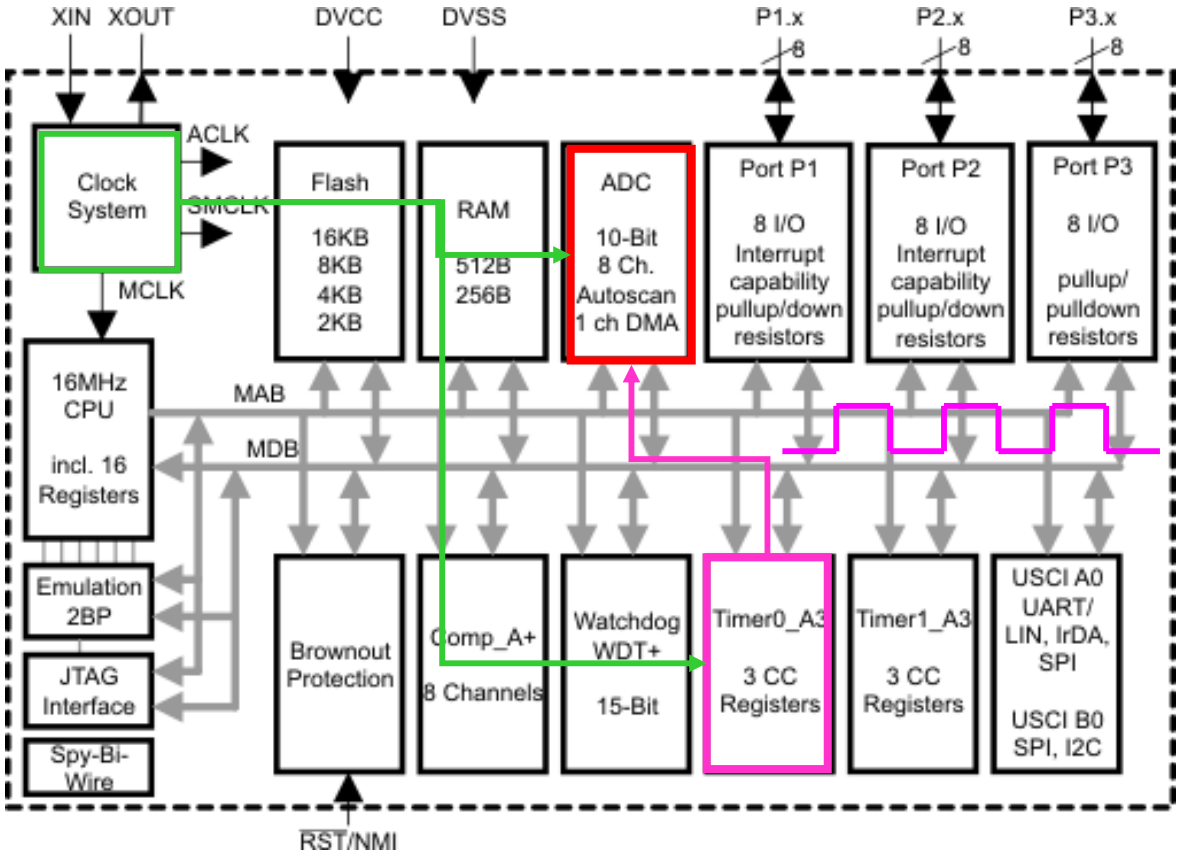
  - ADC10AE0
  - INCH in ADC10CTL1



Requirements of MSP430 for ADC

- Provide continuous sampling of multiple analog inputs and store sampled data

- SHS, ADC10SSEL, CONSEQ in ADC10CTL1

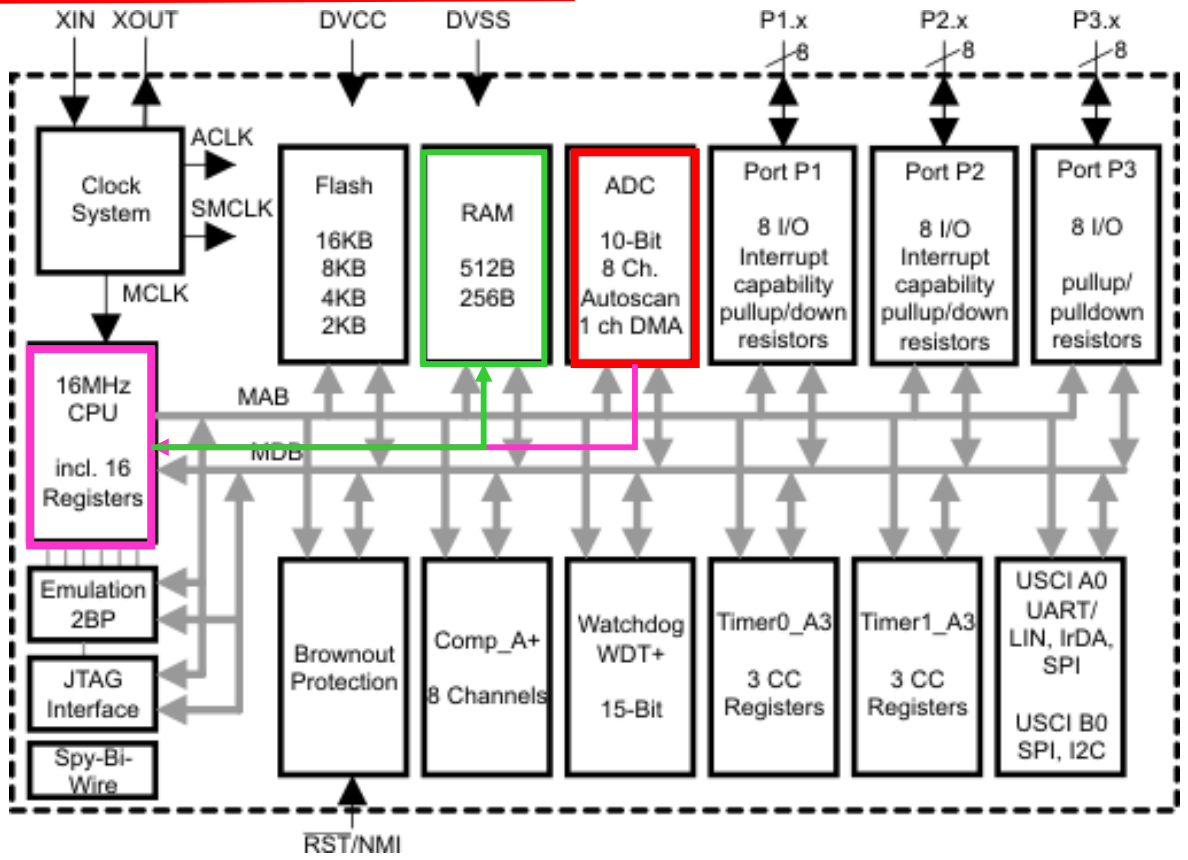


Requirements of MSP430 for ADC

- Provide continuous sampling of multiple analog inputs and store sampled data

Interrupt CPU on every word sampled from ADC

Use software (ISR run by the CPU) to move data from ADC (ADC10MEM) to memory

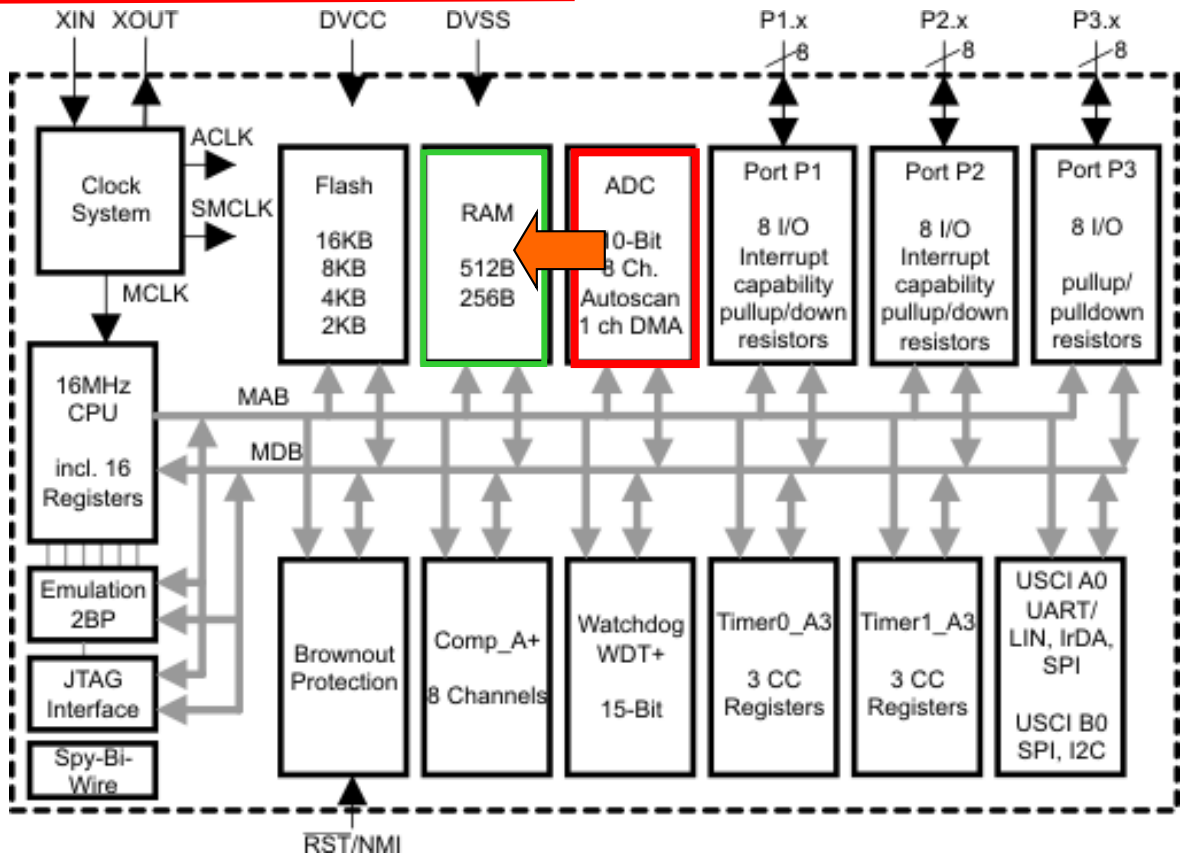


Requirements of MSP430 for ADC

- Provide continuous sampling of multiple analog inputs and store sampled data

Use hardware (Data Transfer Controller, DTC) to move data from ADC to memory

Interrupt CPU when block transfer is completed

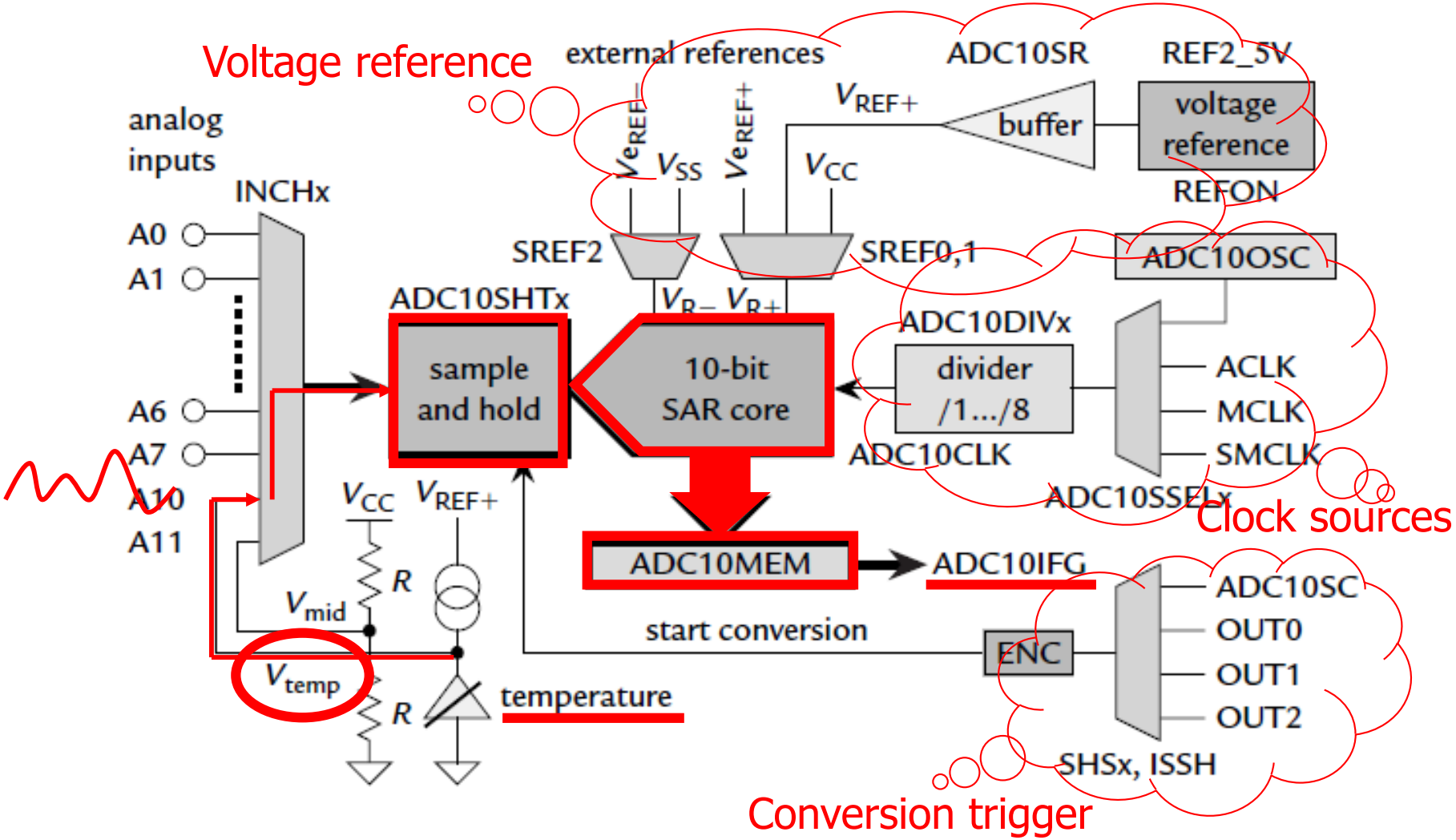


ADC in MSP430

MSP430 may contain one or more converters:

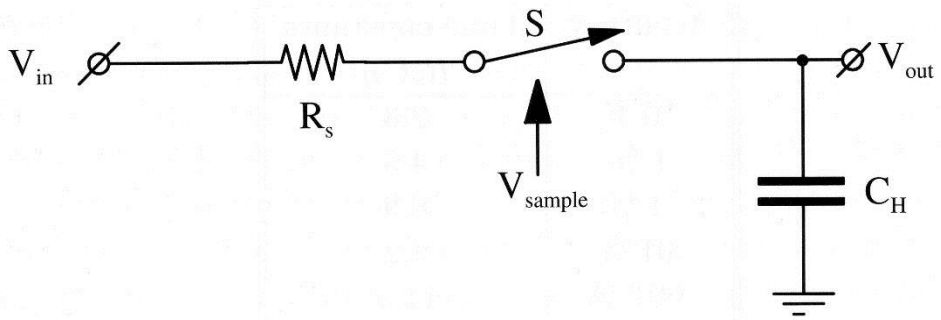
- Comparator:
  - Compare the voltages on its two input terminals and return 0 or 1, e.g., Comparator_A+
- Successive-approximation ADC:
  - Use binary search to determine the closest digital representation of the input signal, e.g., ADC10 and ADC12 to give 10 and 12 bits of output
- Sigma-delta ADC:
  - A more complicated ADC that gives higher resolution (more bits) but at a slower speed, e.g., SD16 and SD16_A, both of which give a 16-bit output

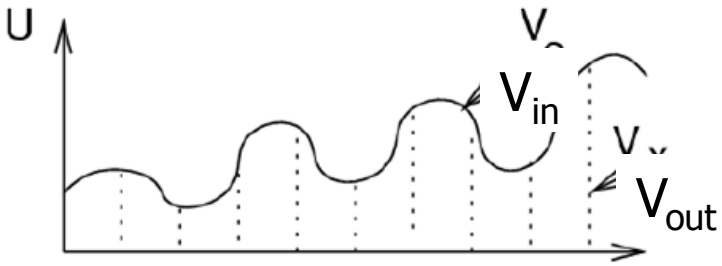
Simplified Block Diagram of ADC10



Main Components of ADC10

- Sample-and-Hold circuit:
  - $V_{out} = V_{in}$ when $V_{sample} = 1$

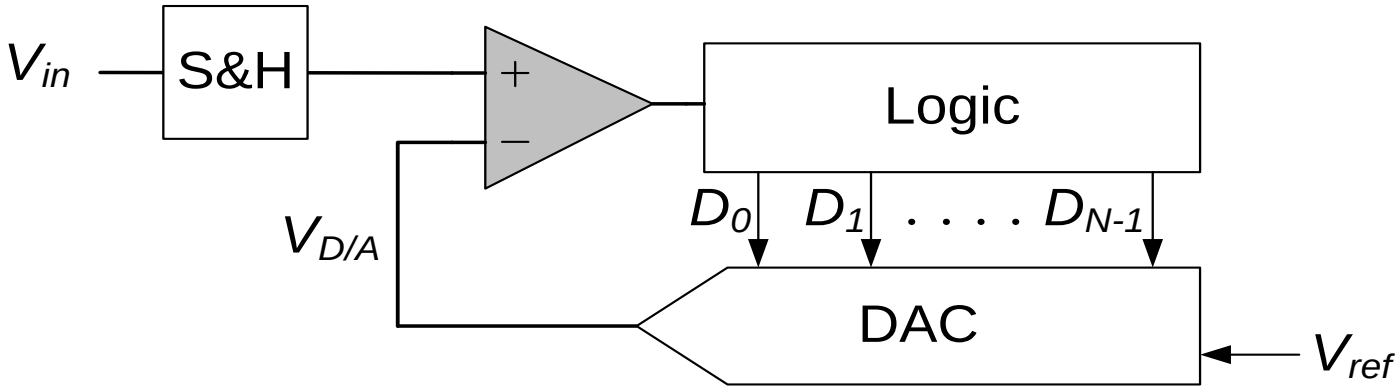




- SAR (Successive-Approximation Register):
  - 10-bit
  - Result written to ADC10MEM and raising ADC10IFG

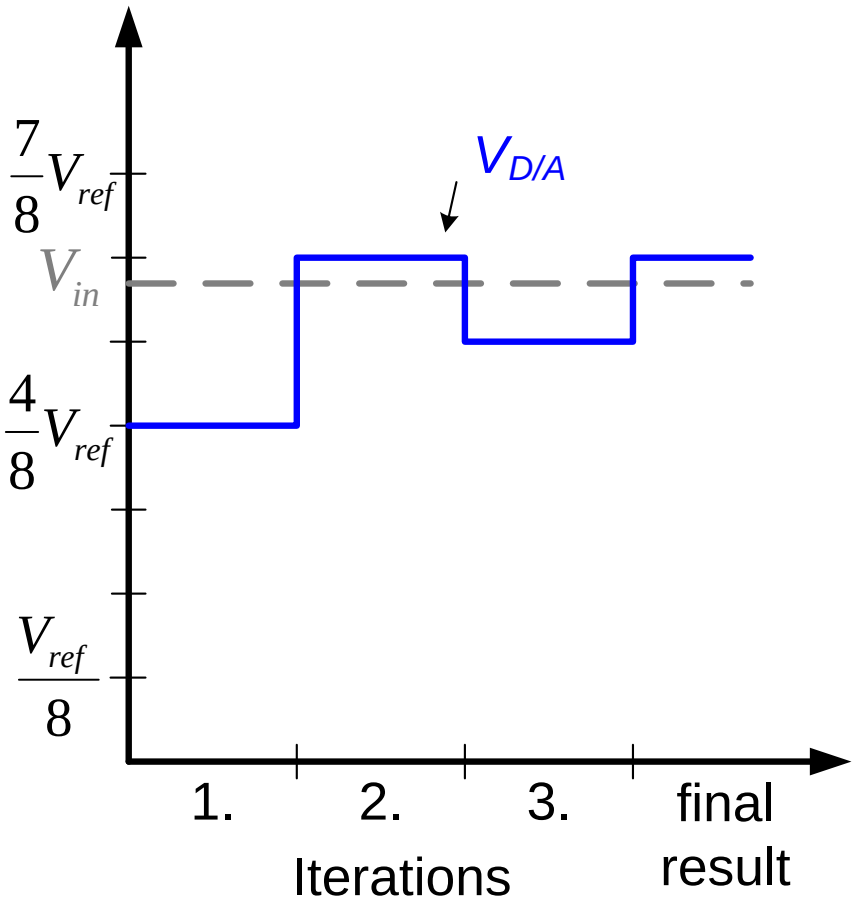
Successive-Approximation ADC

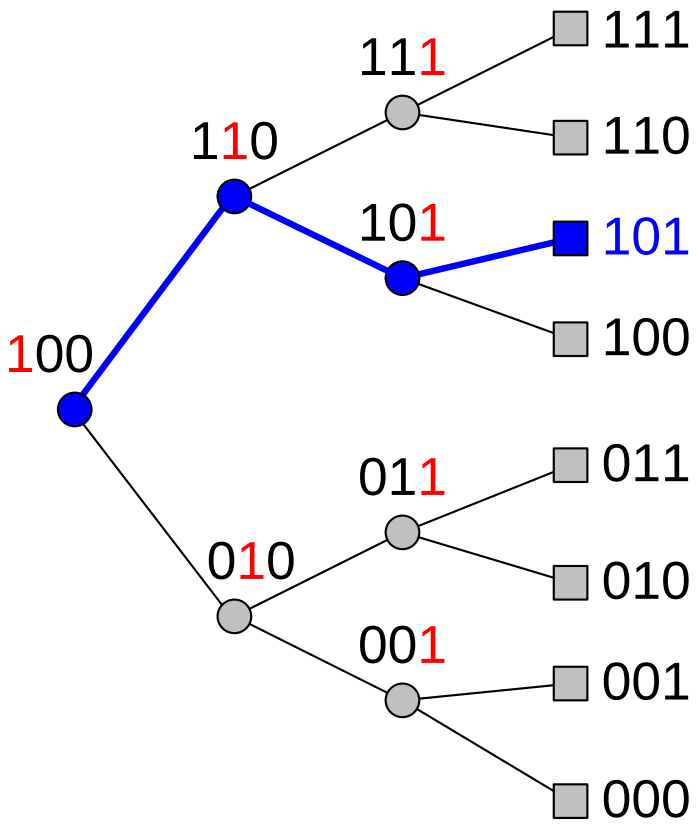
- Generate internal analog signal $V_{D/A}$ by DAC
- Compare $V_{D/A}$ with input signal V_{in}
- Modify $V_{D/A}$ by $D_0D_1D_2…D_{N-1}$ until closest possible value to V_{in} is reached



Dr.-Ing. Frank Sill, Department of Electrical Engineering, Federal University of Minas Gerais, Brazil

Successive-Approximation ADC





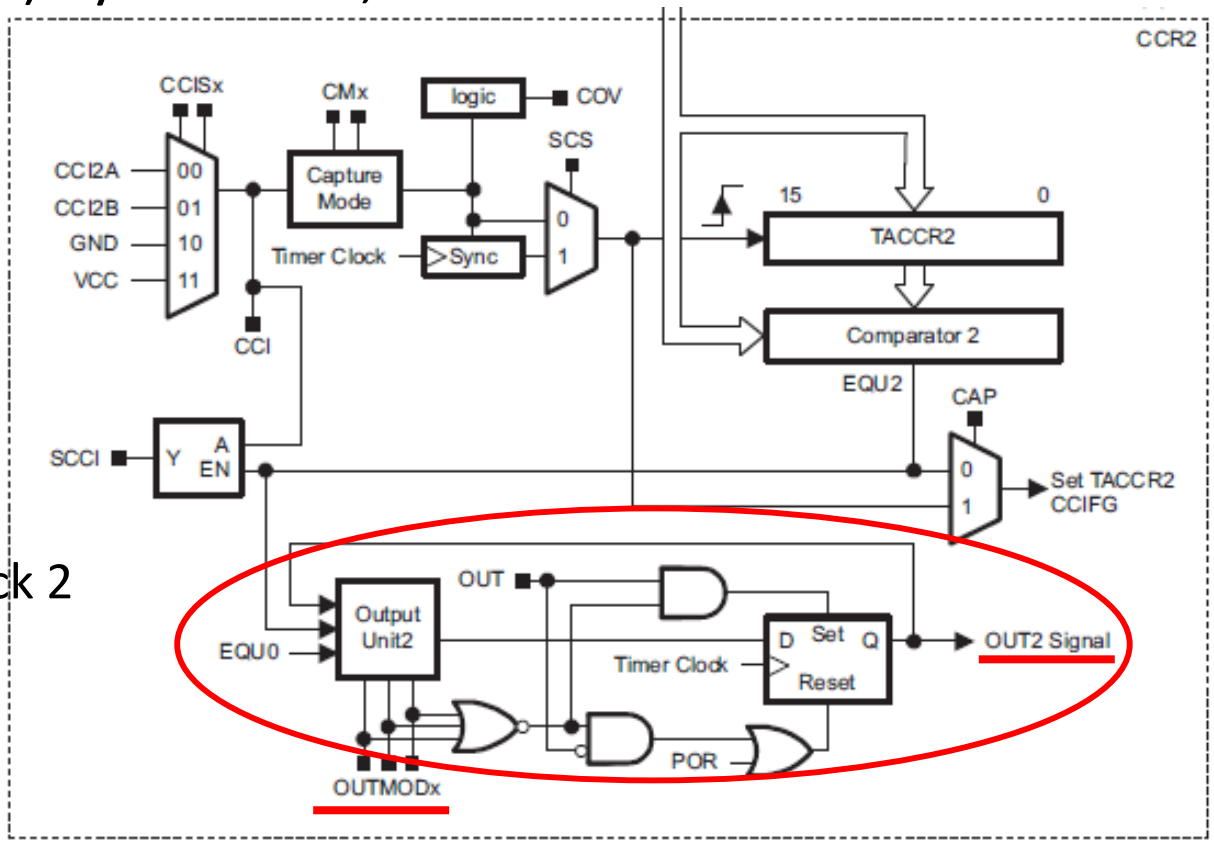
P. Fischer, *VLSI-Design - ADC und DAC*, Uni Mannheim, 2005

Main Components of ADC10

- Built-in voltage reference:
  - Two selectable voltage levels, 2.5 V and 1.5 V
  - Setting REFON in ADC10CTL0 register to 1 enables the internal reference
  - Setting REF2_5V in ADC10CTL0 to 1 selects 2.5 V as the internal reference, otherwise 1.5 V
  - After voltage reference is turned on, we must wait about 30μs for it to settle

Main Components of ADC10

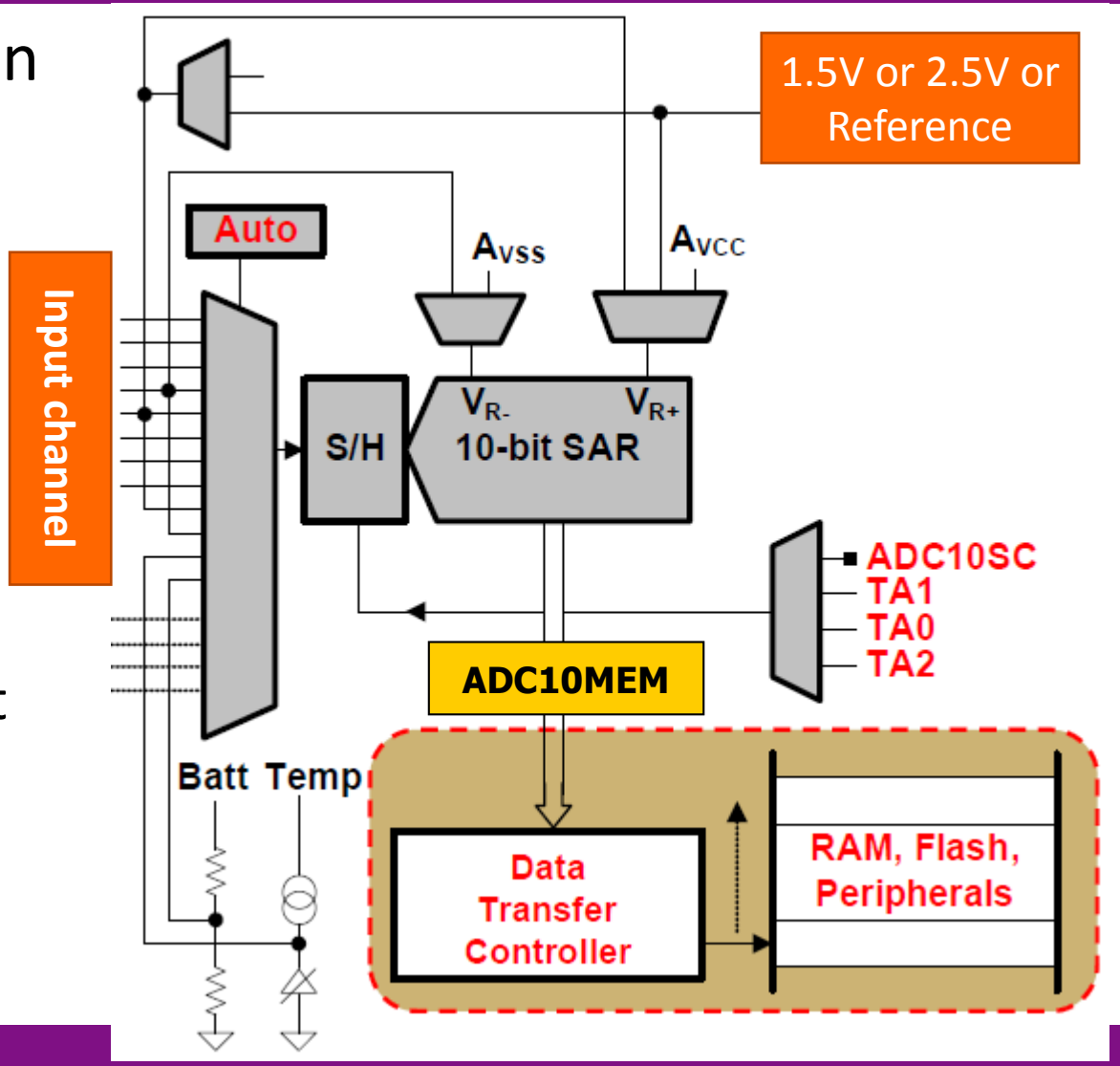
- Sources of sample-and-hold circuit:
  - ADC10SC bit in ADC10CTL0 register, which can be set (and is thus triggered) by software, or
  - OUTx from Timer_A: for periodic sampling



Capture/Compare Block 2 of Timer_A

Data Transfer Controller (DTC)

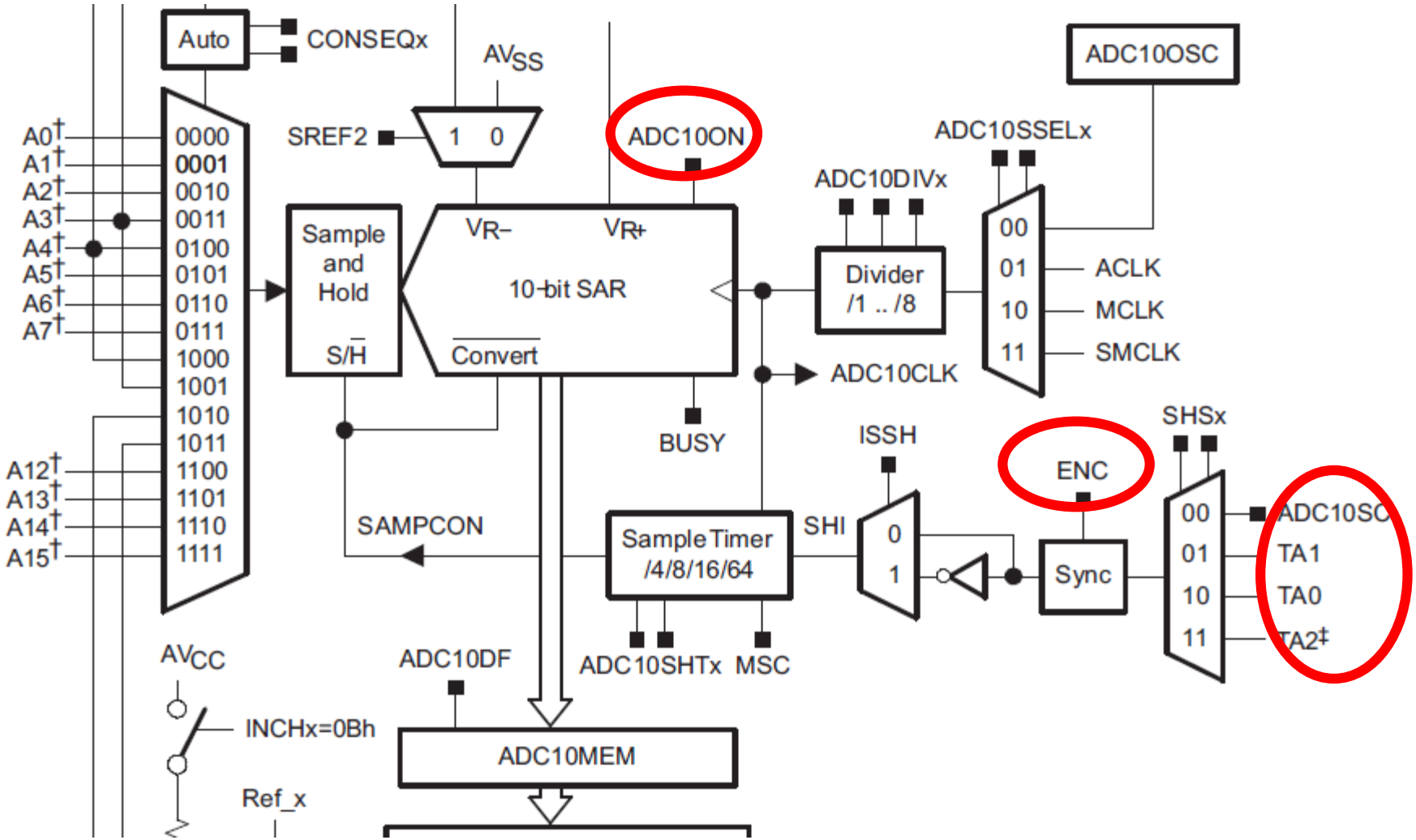
- Transfer conversion results from ADC10MEM to other on-chip memory locations
  - Each load of ADC10MEM triggers a data transfer until a set amount
  - During each DTC transfer, CPU is halted



ADC10 Interrupts

- One interrupt and one interrupt vector
  - When DTC is not used (ADC10DTC1 = 0), ADC10IFG is set when conversion results are loaded into ADC10MEM
  - When DTC is used (ADC10DTC1 > 0), ADC10IFG is set when a block transfer completes
- If both ADC10IE and GIE bits are set, then ADC10IFG generates an interrupt request
  - ADC10IFG is automatically reset when interrupt request is serviced, or it may be reset by software

Enabling Sampling and Conversion



Steps for Single Conversion

(1) Configure ADC10, including the ADC10ON bit to enable the module.

The ENC bit must be clear so that most bits in ADC10CTL0 and ADC10CTL1 can be changed.

(2) Set the ENC bit to enable a conversion.

This cannot be done while the module is being configured in the previous step.

(3) Trigger the conversion.

This is done either by setting the ADC10SC bit or by an edge from Timer_A.

- ADC10ON, ENC, ADC10SC are all in control register ADC10CTL0

ADC10 Registers

Register	Short Form	Register Type	Addr.	Initial State
ADC10 input enable register 0	ADC10AE0	Read/write	04Ah	Reset with POR
ADC10 input enable register 1	ADC10AE1	Read/write	04Bh	Reset with POR
ADC10 control register 0	**ADC10CTL0**	Read/write	01B0h	Reset with POR
ADC10 control register 1	**ADC10CTL1**	Read/write	01B2h	Reset with POR
ADC10 memory	**ADC10MEM**	Read	01B4h	Unchanged
ADC10 data transfer control register 0		Read/write	048h	Reset with POR
ADC10 data transfer control register 1	ADC10DTC1	Read/write	049h	Reset with POR
ADC10 data transfer start address	ADC10SA	Read/write	01BCh	0200h with POR

Where the data is saved

ADC10CTL0

15	14	13	12	11	10	9	8
SREFx			ADC10SHTx		ADC10SR	REFOUT	REFBURST
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
MSC	REF2_5V	REFON	ADC10ON	ADC10IE	ADC10IFG	ENC	ADC10SC
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Can be modified only when ENC = 0

SREFx Bits 15-13 Select reference

000	$V_{R+} = V_{CC}$ and $V_{R-} = V_{SS}$
001	$V_{R+} = V_{REF+}$ and $V_{R-} = V_{SS}$
010	$V_{R+} = Ve_{REF+}$ and $V_{R-} = V_{SS}$
011	V_{R+} = Buffered Ve_{REF+} and $V_{R-} = V_{SS}$
100	$V_{R+} = V_{CC}$ and $V_{R-} = V_{REF-}/ V_{eREF-}$
101	$V_{R+} = V_{REF+}$ and $V_{R-} = V_{REF-}/ V_{eREF-}$
110	$V_{R+} = Ve_{REF+}$ and $V_{R-} = V_{REF-}/ V_{eREF-}$
111	V_{R+} = Buffered Ve_{REF+} and $V_{R-} = V_{REF-}/ V_{eREF-}$

ideal for the temperature sensor

ADC10SHTx Bits 12-11 ADC10 sample-and-hold time

00	4 × ADC10CLKs
01	8 × ADC10CLKs
10	16 × ADC10CLKs
11	64 × ADC10CLKs

ideal for the temperature sensor

ADC10CTL0 cont'd

REFON	Bit 5	Reference generator on
		0 Reference off
		1 Reference on
ADC10ON	Bit 4	ADC10 on
		0 ADC10 off
		1 ADC10 on
ADC10IE	Bit 3	ADC10 interrupt enable
		0 Interrupt disabled
		1 Interrupt enabled
ENC	Bit 1	Enable conversion
		0 ADC10 disabled
		1 ADC10 enabled
ADC10SC	Bit 0	Start conversion. Software-controlled sample-and-conversion start. ADC10SC and ENC may be set together with one instruction. ADC10SC is reset automatically.
		0 No sample-and-conversion start
		1 Start sample-and-conversion

```
ADC10CTL0 = SREF_2 + ADC10SHT_1;  // Reference range & SH time
```

ADC10CTL1

15	14	13	12	11	10	9	8
INCHx				SHSx		ADC10DF	ISSH
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
ADC10DIVx			ADC10SSELx		CONSEQx		ADC10BUSY
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r-0

Can be modified only when ENC = 0

INCHx Bits 15-12 Input channel select. These bits select the channel for a single-conversion or the highest channel for a sequence of conversions.

0000	A0
0001	A1
0010	A2
0011	A3
0100	A4
0101	A5
0110	A6
0111	A7
1000	V_{eREF+}
1001	V_{REF-}/V_{eREF-}
1010	Temperature sensor
1011	$(V_{CC} - V_{SS}) / 2$
1100	$(V_{CC} - V_{SS}) / 2$, A12 on MSP430x22xx devices
1101	$(V_{CC} - V_{SS}) / 2$, A13 on MSP430x22xx devices
1110	$(V_{CC} - V_{SS}) / 2$, A14 on MSP430x22xx devices
1111	$(V_{CC} - V_{SS}) / 2$, A15 on MSP430x22xx devices

ADC10CTL1 cont'd

SHSx	Bits 11-10	Sample-and-hold source select
	00	ADC10SC bit
	01	Timer_A.OUT1
	10	Timer_A.OUT0
	11	Timer_A.OUT2 (Timer_A.OUT1 on MSP430x20x2 devices)
ADC10DF	Bit 9	ADC10 data format
	0	Straight binary
	1	2s complement
ISSH	Bit 8	Invert signal sample-and-hold
	0	The sample-input signal is not inverted.
	1	The sample-input signal is inverted.
ADC10DIVx	Bits 7-5	ADC10 clock divider
	000	/1
	001	/2
	010	/3
	011	/4
	100	/5
	101	/6
	110	/7
	111	/8
ADC10SSELx	Bits 4-3	ADC10 clock source select
	00	ADC10OSC
	01	ACLK
	10	MCLK
	11	SMCLK

CONSEQx	Bits 2-1	Conversion sequence mode select
	00	Single-channel-single-conversion
	01	Sequence-of-channels
	10	Repeat-single-channel
	11	Repeat-sequence-of-channels

```
ADC10CTL1 = INCH_10 + ADC10DIV_0; // Temp Sensor ADC10CLK
```

Outline

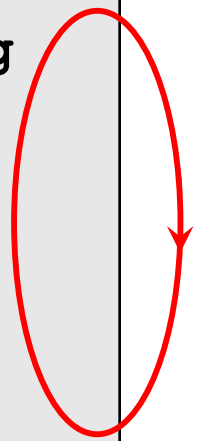
- Introduction to analog-to-digital conversion
- ADC of MSP430
- Sample code of using ADC10 in MSP430

Sample Code 1 for ADC10

- Repetitive single conversion:
  - A single sample is made on A1 with reference to Vcc
  - If A1 > 0.5*Vcc, P1.0 set, else reset.
  - Software sets ADC10SC to start sample and conversion. ADC10SC automatically cleared at end of conversion.
  - Use ADC10 internal oscillator to time the sample and conversion.

Sample Code 1 for ADC10

```
#include "msp430.h"
void main(void) {
  WDTCTL = WDTPW + WDTHOLD;     // Stop WDT
  // H&S time 16x, interrupt enabled
  ADC10CTL0 = ADC10SHT_2 + ADC10ON + ADC10IE;
  ADC10CTL1 = INCH_1;     // Input A1
  ADC10AE0 |= 0x02; // Enable pin A1 for analog in
  P1DIR |= 0x01;     // Set P1.0 to output
  for (;;)   {
    ADC10CTL0 |= ENC + ADC10SC; // Start sampling
    __bis_SR_register(CPUOFF + GIE); // Sleep
    if (ADC10MEM < 0x1FF) // 0x1FF = 511
      P1OUT &= ~0x01;  // Clear P1.0 LED off
    else
      P1OUT |= 0x01;   // Set P1.0 LED on  }
}
```



Sample Code 1 for ADC10

```
// ADC10 interrupt service routine
#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR(void)
{
  __bic_SR_register_on_exit(CPUOFF);
  // Clear CPUOFF bit from 0(SR)
}
```

Sample Code 2 for ADC10

- Continuous sampling driven by Timer_A
  - A1 is sampled 16/second (ACLK/2048) with reference to 1.5V, where ACLK runs at 32 KHz driven by an external crystal.
  - If A1 > 0.5Vcc, P1.0 is set, else reset.
  - Timer_A is run in up mode and its CCR1 is used to automatically trigger ADC10 conversion, while CCR0 defines the sampling period
  - Use internal oscillator times sample (16x) and conversion (13x).

Sample Code 2 for ADC10

```
#include  "msp430.h"
void main(void) {
  WDTCTL = WDTPW + WDTHOLD; // Stop WDT
  // TA1 trigger sample start
  ADC10CTL1 = SHS_1 + CONSEQ_2 + INCH_1;
  ADC10CTL0 = SREF_1 + ADC10SHT_2 + REFON +
              ADC10ON + ADC10IE;
  __enable_interrupt();  // Enable interrupts.
  TACCR0 = 30;      // Delay for Volt Ref to settle
  TACCTL0 |= CCIE; // Compare-mode interrupt.
  TACTL = TASSEL_2 + MC_1; // SMCLK, Up mode.
  LPM0;                    // Wait for settle.
  TACCTL0 &= ~CCIE;      // Disable timer Interrupt
  __disable_interrupt();
```

Sample Code 2 for ADC10

```
  ADC10CTL0 |= ENC;      // ADC10 Enable
  ADC10AE0 |= 0x02;      // P1.1 ADC10 option select
  P1DIR |= 0x01;         // Set P1.0 output
  TACCR0 = 2048-1;       // Sampling period
  TACCTL1 = OUTMOD_3;    // TACCR1 set/reset
  TACCR1 = 2046;         // TACCR1 OUT1 on time
  TACTL = TASSEL_1 + MC_1;     // ACLK, up mode

  // Enter LPM3 w/ interrupts
  __bis_SR_register(LPM3_bits + GIE);
}
```

Timer_A CCR1 out mode 3: The output (OUT1) is set when the timer *counts* to the TACCR1 value. It is reset when the timer *counts* to the TACCR0 value.

Sample Code 2 for ADC10

```
  // ADC10 interrupt service routine
#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR(void){
  if (ADC10MEM < 0x155) // ADC10MEM = A1 > 0.5V?
    P1OUT &= ~0x01;     // Clear P1.0 LED off
  else
    P1OUT |= 0x01;      // Set P1.0 LED on
}

#pragma vector=TIMERA0_VECTOR
__interrupt void ta0_isr(void){
  TACTL = 0;
  LPM0_EXIT;            // Exit LPM0 on return
}
```

Summary

- ADC: analog-to-digital conversion
  DAC: digital-to-analog conversion
  - Conversions will necessarily introduce errors. Important to understand constraints and limitations
- ADC10 in MSP430
  - Convert an analog signal into 10-bit digitals
  - Registers associated with ADC10
- Sample program of ADC10
  - Single conversion
  - Continuous conversion driven by Timer_A