Digitizing Signals

- To process *continuous signals* as functions of time

- Advantages
  - free from noise (e.g. error resilient)
  - can be stored, retrieved, and manipulated by computer (e.g. speech recognition)

- Disadvantages
  - loss of information
  - higher bandwidth requirement for transmission

Data Acquisition System
Transducer

- Transducers convert analog input to electrical signals (e.g. voltages or currents)

Signal Conditioning

- Signal conditioning: electrical isolation and buffering (e.g. protect from static discharges), amplification (e.g. produce necessary voltage for ADC), and bandwidth limiting (e.g. low-pass filter to limit range of frequencies for digitization)

Analog Multiplexer

- Analog MUX: select from several analog inputs

Sample-and-Hold

- Sample-and-hold circuit holds signal steady while ADC converts it
  - High-quality capacitor and high-speed semiconductor switch
  - Close switch for very short period, let capacitor charge, then switch opened and voltage held for ADC during its conversion time
  - May be included within ADC device
**A/D Converter**

- ADC converts sampled signal to digital values; has a word-size just like other digital devices.

**Sampling Theorem & Aliasing**

- Shannon’s Theorem: sampling frequency must be at least twice the signal frequency.
- If not followed, then undersampled, which leads to aliasing and invalid signal reconstruction.
- Signal conditioning circuit may include antialiasing filter, helps here by attenuating frequencies above ½ the sample freq.
- Nyquist frequency: the maximum frequency one can sample without aliasing (Nyquist frequency = fsample/2).

---

**A/D Conversion**

- Normally interfaced to rest of system via parallel I/O interface circuit or parallel input port.
- START_CONVERT control signal asserted by CPU to begin the conversion.
- END_CONVERT informs CPU when conversion complete (use via polling or interrupt).

**A/D Converter Types**

- Successive approximation A/D:
  - Starting at MSB, each bit in reg. tested in succession with DAC output compared vs. input.
  - With each bit tried: DAC output lower than input signal ⇒ bit left set; higher ⇒ bit is reset.
  - However, N bit-times needed to set and test all the bits in succession.
**Tracking A/D Converter**

- Has an up-down counter controlled by comparator
- If input signal higher or lower than DAC output, counter counts up or down, respectively
- Converter may quickly converge to correct digital value when signal not changing quickly

**Integrating A/D Converter (Dual slope ADC)**

- Converter integrates input signal for period T1; afterwards, switch changed to minus ref. voltage and integrator discharges to zero at constant rate; time to discharge is period T2 and gives a digital value

**Parallel A/D Converter**

- An array of \(2^N - 1\) parallel comparators that quickly produces output code in prop. time of comparators plus encoder/decoder logic; fast but costly

**Two Stage Parallel A/D Converter**

- Has nearly the speed of flash ADC but less costly
**ADC Specifications**

- **Resolution:** several forms of term in use, such as # of bits in ADC, smallest input signal for which ADC will produce a digital code
  - full-scale signal $\div 2^N$
  - e.g. 8-bit ADC on 5V full-scale signal $\Rightarrow$ res. is $5V/256 = 19.5mV$ (0.4% of full scale value)

- **Accuracy:** ratio of smallest signal to measured signal; in %, describes how close measurement is to actual value
  - e.g. 8-bit ADC on 5V full-scale signal, accuracy w/ 50mV signal is $19.5/50 = 39\%$

- **Linearity:** deviation in output codes vs. line from zero to full-scale; best is $\pm \frac{1}{2}$ of LSB

- **Aperture time:** time ADC is “looking” at input signal
  - During this period, change in input signal may cause error in output code
ADC Errors

- Quantization error is fundamental in A/D conversion due to resolution of ADC
  - Can be no less than $\pm \frac{1}{2}$ LSB
- Other sources of errors include:
  - Noise (desire peak-to-peak noise to be $< \pm \frac{1}{2}$ LSB; choose appropriate ADC resolution or reduce the signal noise)
  - Aliasing (include low-pass filter to attenuate freqs. above Nyquist freq)
  - Aperture time (if signal varies during this period, sample & hold circuit achieve short aperture time))

ADC Selection

- Choose # of bits or resolution, speed or conversion time, type of digital code produced, etc
- Two ways to choose ADC resolution:
  - Find dynamic range of input signal to choose # of bits
    - $DR = \frac{V_{max}}{V_{noise}}$, where $V_{max}$ is max. input signal, $V_{noise}$ is peak-to-peak noise
    - For noise within $\pm \frac{1}{2}$ LSB, $N \geq \log_2 DR$
  - Choose based on resolution required in signal
    - $N \geq \log_2 \frac{V_{max}}{V_{min}}$, where $V_{min}$ is required resolution

- Choose ADC conversion time based on highest-frequency component sampled 2x or more
- Output code options for unipolar ADC (positive signals) include unsigned binary and 1's complement; see Table 11-1
- Output code options for bipolar ADC (pos. & neg. signals) include 2’s complement, signed-magnitude, and offset binary; see Table 11-2

- Example: consider ADC for $\pm 5V$ peak-to-peak range, 5mV peak-to-peak noise, and $f_{max} = 3kHz$:
  - $DR = 10V / 5mV = 2000$
  - $N \geq \log_2 2000 \geq 10.9 \Rightarrow N = 11$ or more
  - Maximum conversion time is $1 / (2 \times 3kHz) = 167ms$
D/A Conversion

- DAC diagram
- Latch may be part of DAC or must be separately interfaced
- Signal cond. block may be used to filter and smooth the quantized output, and perhaps also isolation, voltage amplification, etc.
- Example of quantized DAC output waveform

D/A Converter Types

- Basic DAC circuit is binary-weighted register DAC
  - weighted current supplied to summing junction of amplifier

D/A Converter Types

- R-2R ladder DAC circuit using single-pole, double-throw switches between ground and reference
  - binary-weighted current supplied to summing junction

DAC Specifications

- Resolution and linearity (same as before but w.r.t. output voltage)
- Settling time: time for output voltage to settle within specified error band (e.g. ± ½ LSB)
**DAC Specifications**

- Glitches: high-speed DACs may have problems with glitches and settling time
- Glitch caused by asymmetrical switching in D/A switches (e.g. 1→0 switch faster than 0→1 switch ⇒ glitch)

![Graph showing digital input code and output voltage over time with glitches]

**DAC Specifications**

- Consider change in output on 8-bit DAC, from %10000000 to %01111111; would expect change from 1/2 full-scale to one resolution less, but asymm. switching causes transitory sequence 10000000→00000000→01111111 ⇒ glitch on output signal!
- Can be eliminated w/ sample-and-hold on DAC output, strobed to sample data after glitch and settling time

---

**EEL 4744C: Microprocessor Applications**

**Lecture 10**

**Part 2**

**M68HC12 Analog Input**

**Read Assignment**

- SHE (old version): Chapter 12
- SHE (new version): Chapter 17
- Freescale HC12 Data Sheet: Chapter 17
**Introduction**

- 8-channel, multiplexed, 8-bit, successive approx. ADC with sample-and-hold
- Linear to ±1 LSB accuracy in full temperature range with no missing codes
- Both conversion time and S&H aperture time are programmable
- Uses VRH and VRL to optimize resolution over input signal range
- VRH usually set to input signal max. (but must be ≤ 6V); VRL to min. (but must be ≥ 0V)

---

**HC12 ADC Block Diagram**

- Analog Input Pins
- Select which channels to convert
- Built-in, Programmable Sample-and-Hold Circuitry
- A/D Power-up
- A/D Interrupt Flag and Enable
- 4/8 successive conversions
- A/D Result Registers
- Select which channels to convert

---

**Introduction**

- Signal range must be VRH – VRL > 2.5V
- Resolution is (VRH – VRL)/256 – e.g. (5V-0V)/256 = 19.5mV
- ADPU bit to enable subsystem (1=enabled); delay of ~100ms needed afterwards before use
- AWAI bit to have ATD stop/continue when HC12 in wait mode (1=stop)

---

**HC12 ADC Block Diagram**

- Analog Input Pins
- Select which channels to convert
- Built-in, Programmable Sample-and-Hold Circuitry
- A/D Power-up
- A/D Interrupt Flag and Enable
- 4/8 successive conversions
- A/D Result Registers
- Select which channels to convert

---

**Introduction**

- A/D derives clock from P-clock; prescalar bits PRS4-PRS0 for 1/2 (fastest) to 1/16 (slowest) – See Table 12-1
- Limits: max. and min. conversion freqs. permitted are 2MHz and 500kHz, respectively
- Final sample time stage selectable from 2-16 clock periods (2, 4, 8, 16) via SMP1:SMP0 bits – See Table 12-2
Sampling & Conversion Timing

**ATDCTL4 register controls the A/D Timing**

- **Prescaler select bits**
- **Sample time select bits**

(The maximum analog input frequencies)

\[
\text{Nyquist Frequency} = \frac{1}{2(2^\text{total\_conversion\_time})} = 0.5 \times f_{\text{max}} \times \text{cycles}
\]

**Total conversion time**

\[
\text{Min} = 2 \times 4 + 2 \times 10 = 18 \\
\text{Max} = 2 \times 4 + 16 + 10 = 32
\]

\[
\text{Nyquist Frequency} = \frac{1}{2 \times \text{cycles} \times f_{\text{ad\_clock}}} = 0.5 \times f_{\text{max}} \times \text{cycles}
\]

\[
= \frac{1}{2 \times \text{cycles} \times 2\text{MHz}} = \frac{1}{4 \times \text{cycles}} = 0.25 \times f_{\text{ad\_clock}}
\]

\[
= \frac{1}{2 \times \text{cycles} \times 2\text{MHz}} = \frac{1}{4 \times \text{cycles}} = 0.25 \times f_{\text{ad\_clock}}
\]

\[
= 250 \text{KHz} - 500 \text{KHz}
\]

**A/D Input MUX and Scanning**

- **8 input channels, selected by bits in ATDCTL5 register** (The ADC is started by writing to ATDCTL5)

- **ADC always completes sequence of either 4 or 8 conversions (chosen by S8CM bit)**

- **SCAN bit controls whether ADC converts only 1 sequence versus continuously**
  - **SCAN=0:** After the conversions are done, the A/D waits for the program to write to the ATDCTL5 again
  - **SCAN=1:** A/D starts another conversion cycle immediately

**A/D Input Synchronization**

- **MULT bit determines if 4/8 sequences done on single or successive channels**

- **Channel select bits CD-CA choose which channel (s) converted**
  - Unused ones can be used for GP input via Port AD as before

- **See Table 12-3 for details**

- **The ADC has 8, 16-bit result registers (only the high-order 8 bits are used for HC12)**

- **ADC can generate interrupts when conversion sequence complete, or user may poll flag**

- **SCF (sequence complete flag) bit set when the 4/8 conversion sequence is done**

- **Also, 8 conversion complete flags (CCF7-CCF0) associated with the A/D result registers**
  - Set when current conversion writes into associated result register

- **These flags contained in 16-bit ATDSTAT status register**
A/D Input Synchronization

- **AFFC (A/D fast flag clear) bit** controls how status flags are reset
  - AFFC=0: 2-step process: (1) read status register; (2) if CCFn then read associated result register, or if SCF then write to ATDCTL5 to start new conversion
  - AFFC=1: fast mode: CCFn flag clears by reading associated result register; SCF clears when first result register read
  - Former typically used for polling, latter for interrupts

A/D Interrupts

- Can generate interrupt when current 4/8 conversion sequence is completed
  - ASCIE bit used to enable, ASCIF is the flag that generates the interrupt
    - Cleared by reading any result register when AFFC=1
    - A/D seq. complete vector is in the vector table

A/D Programming Example

- A/D programming example: converts the data on Ch 4-7
  - P-clock is 8MHz
  - Prescalar set for P-clock + 4
  - ATD clock is 2MHz (its maximum)
  - 2-clock final sample time used $\Rightarrow 16+2 = 18$ ATD clock cycles in total conversion time
  $\Rightarrow f_{\text{conv}} = 1 / (18 \times 500\text{ns}) = 111.11 \text{ kHz}$
  $\Rightarrow f_{\text{max}} = 55.5 \text{ kHz}$ (i.e. Nyquist frequency)

A/D Programming Example (code)

- AD control registers
  - ATDCTL2: EQU $82$
  - ATDCTL4: EQU $84$
  - ATDCTL5: EQU $85$
  - ADRMH: EQU $70$
  - ADRMH: EQU $74$
  - ADRMH: EQU $78$
  - ATDSTAT: EQU $86$
  - SCF: EQU $01000000$; Seq complete flag
  - AFFC: EQU $01010000$; Fast clear
  - AHRI: EQU $00100000$; A/D wait mode
  - ASCIE: EQU $00000010$; SCF interrupt enable
  - ADPU: EQU $01000000$; A/D power up bit
  - SMP: EQU $01100000$; SMP0 and SMP1 bits
  - PRS0: EQU $00000001$
- A/D Mode: S8CM=0 4 conversion sequence
  - MULTI=4 conversions on channels 4-7
  - CD,CC,CB,CA=01xx Analog channel 4-7
  - ADMODE: EQU $00010100$
A/D Programming Example

; Power up the A/D
; A/DCTL2.ADP
; Generate a "short" delay > 100 microsec:
; delay: #200 ; 200 loops for
; delay: nop ; 800 clock cycles
; delay: s.delay

; Now set up the A/D
; Normal flag clearing, run in WAIT mode, no interrupts:
bclr A/DCTL2.AFFC|AWAI|ASCIE

; Select 2 clock sample time and divide by 4 prescaler:
bclr A/DCTL4.SMP
bset A/DCTL4.PRS0
;

; Start the conversion by writing the scan select information to A/DCTL5
loop:
ldaa #ADMODE
staa A/DCTL5

; And wait until conversion done
spin:
brclr A/DSTAT.SCF,spin

; Get the channel 4 value
clra ; set A=0
ldab ADR0H ; Channel 4 is here

Dr. Tao Li