CprE 288 – Introduction to Embedded Systems (Analog-to-Digital Converter)

Dr. Phillip Jones
Announcements

• **Exam 1:** Tuesday 10/2 (Covers HW1 – HW4 material)
• HW 4: Not Graded
• **Project Checkpoint 1:** Wed 9/26: Top 5 Project Ideas / Areas
• **Project Teams:** to be announced.
• **In class Project Activity:** Tuesday 10/9. Directions to be given.
Announcements

- HW 5, Due Wed 6/20
- **Quiz 5 (15 min)**: Wed 6/13, Textbook reading: Section 9.1, 9.2 (your one-side of 1 page of notes will be collected for Class Participation)
- Exam 2: Friday, June 22
- Lab 6: ADC (Analog to Digital Converter)
  - Textbook: Chapter 7.5: pages 446-479 (~35 pages, but quite a bit of redundancy)
Announcements

• Exam 1: Making grading adjustments
• Exam 2: Week 12 (need to finalize if it will be on Tue or Thur)
  – i.e. Week of 11/5
• Quiz 7: Thursday 10/18 (Same FULL readings as Quiz 6)
• HW 6: Sunday 10/21
• Project Proposal
  – Revision 1: Monday 10/22
    • Problem Statement: HW3 User research flow for developing Project’s problem statement (see rubric for grading criteria
    • Prototype: Sketch of usage of cybot, and worksheet mapping Application functionality to Cybot
  – Revision 2: Monday 10/29
• Servo motor lab
Week 7 Overview

• ADC general knowledge

• TM4C123G ADC programming interface
  – Textbook reading:
    • Section 7.5

• Write ADC-related functions
  – Initialize/configure ADC
  – Read from ADC
  – ADC interrupt programming
There are generally three phases of the course:
1. C Programming targeting low-level embedded concepts
2. I/O Device Programming
3. Architecture and Assembly programming

The 2\textsuperscript{nd} and 3\textsuperscript{rd} phases are much more challenging than the 1\textsuperscript{st} phase
Exam 2 will predominantly consist of questions of the form

- **Program Configure Registers to meet these specs**
  - UART, ADC, Input Capture, Output Compare, Timers, Interrupts
  - Each device has a section in the Datasheet and Textbook

- Based on a given configuration, answer questions about how the program will behave
  - E.g. How long will something take to occur?
  - E.g. How many times a second will something occur?

- Explain why a given configuration is incorrect for implementing a specified behavior

- Assuming a given configuration, write a short program to implement a specific behavior

- ADC calculation problem
ADC and DAC

• Analog-to-Digital Conversion (ADC)
• Digital-to-Analog Conversion (DAC)

• Why do we need ADC and DAC?
  – To allow our embedded programs to interact with the World
  – The World is analog, not digital

• Examples of sensors that connect to ADCs
  – Temperature, pressure, light, humidity, compass, and sound
ADC and DAC

Analog sensor: Converts a physical signal into an *analog* electrical signal

Temperature Sensor $3.95

Photoresistor $1.95 (Light Sensor)

Passive IR sensor $9.95 for motion detection

Sensor pictures and prices from http://www.parallax.com
Terminology

- **analog**: continuously valued signal, such as temperature, speed, or voltage with infinite possible values in between
  - E.g., between 1 m/s and 2 m/s you can have 1.34......2 m/s.

- **digital**: discretely valued signal, such as integers encoded in binary
  - E.g. a 2-bit integer can only have four values- 00, 01, 10, 11

- **analog-to-digital converter (ADC, A/D, or A2D)**: converts an analog input signal to a n-bit digital output signal
  - The TM4C123G has a 12-bit ADC

- **digital-to-analog converter (DAC, D/A, D2A)**: converts a n-bit digital input signal to an analog output signal
• **Span (or Range):** difference between maximum and minimum analog values (Max – Min)

• **n:** number of bits used for a digital input (DAC) or digital output (ADC) (sometimes referred to as n-bit resolution)

• **Bit Weight:** analog value corresponding to a bit position in a digital number

• **M:** Number of digital steps, either $2^{n-1}$ or $2^n$

• **Step Size (or Resolution):** smallest analog change resulting from a change of one in a digital value; also the bit weight of the Least Significant Bit (LSB)
  
  – Step Size (or Resolution) = Span / M

• **Sensitivity:** Amount sensor output changes for a change in sensor input
Analog-to-digital converter: Example

Temperature Sensor

$T_{\text{min}} = 0 \, ^\circ\text{C}$
$T_{\text{max}} = 200 \, ^\circ\text{C}$
$V_{\text{min}} = 0 \, \text{V}$
$V_{\text{max}} = 3.3 \, \text{V}$

Temperature vs. Voltage (Sensor Specification)

A/D: Analog Input vs. Digital Output
$(M = 2^n - 1 \text{ steps (or bins)}: D_{\text{max}} = V_{\text{max}})$

A/D Input (V)
$A/D_{\text{Vmax}} = 3.3 \, \text{V}$
$A/D_{\text{Vmin}} = 0 \, \text{V}$

Digital Output (D)
$D_{\text{max}} = 4095$
$D = 0$

12-bit
Sensitivity: Analog Sensor (Linear)

- Sensitivity: How much does a change in a sensor input change the sensor output

Sensor Input (Temperature (°C))

- $T_{min} = 0 \, \text{°C}$
- $T_{max} = 200 \, \text{°C}$

Analog Sensor Output (V)

- $V_{min} = 0$
- $V_{max} = 3.3\, \text{V}$

Temperature vs. Voltage

Sensor output

A/D input

Temperature Sensor
• Sensitivity: How much does a change in a sensor input change the sensor output
Sensitivity: Analog Sensor (Linear)

- **Sensitivity**: How much does a change in a sensor input change the sensor output
- **Assuming a linear sensor** (Not all sensors are linear!!)
  - It is just the slope (m) of the Input vs. Output specification
  - Remember slope = RISE/RUN = \( \Delta T / \Delta V \)
• Sensitivity: How much does a change in a sensor input change the sensor output

• For this example:

\[ m = \text{slope} = \frac{\text{RISE}}{\text{RUN}} = \frac{(T_{\text{max}} - T_{\text{min}})}{(V_{\text{max}} - V_{\text{min}})} \]

\[ = \frac{(200 - 0)}{(3.3 - 0)} = 60.61 \text{ C/V} \]
• Question: If the measured Temperature is 100°C what is the Analog output of the sensor?

• Again note, in this case the sensor is linear
  – Hint: What is the equation of a line (in y-intercept form, Yes Algebra II was actually an important course)
Analog Sensor: Compute Analog Output

• Question: If the measured Temperature is 100°C what is the Analog output of the sensor?

• Again note, in this case the sensor is linear

\[ Y = mX + b; \] in this case the y-intercept \( b = 0 \) (This is not always the case!), so

\[ Y = mX; \] We are given the temperature \( (Y) = 100 \) and computed \( m = 60.61 \), so

\[ 100 = 60.61X; \] \( X = 100/60.61 = 1.65 \text{ V} \]
Resolution: A/D

- **Resolution**: Similar to the concept of Sensitivity for an analog sensor. For a change by 1 of the A/D digital output what size change is detected in the A/D analog input.
Resolution: A/D

- Resolution: Similar to the concept of Sensitivity for an analog sensor. For a change by 1 of the A/D digital output what size change is detected in the A/D analog input.
- A/D converters typically have a linear relationship between their Analog input and Digital output
  - Resolution is just the slope \( m \) of the Input vs. Output specification

\[ V_{\text{min}} = 0 \text{ V} \]
\[ V_{\text{max}} = 3.3 \text{ V} \]
\[ D_{\text{max}} = 4095 \]

\[ \Delta V \]
\[ \Delta D \]

A/D: Analog Input vs. Digital Output
(For \( M = 2^n - 1 \) steps (or bins), \( D_{\text{max}} = V_{\text{max}} \))
Resolution: A/D

- A/D converters typically have a linear relationship between their Analog input and Digital output
  - Resolution is just the slope of the Input vs. Output specification
- A/D specifications typically specify this in terms of the Least Significant Bit (LSB) weight

A/D: Analog Input vs. Digital Output
(For M = 2^n-1 steps (or bins), Dmax = Vmax)

- Vmin = 0 V
- Vmax = 3.3V
- D = 0
- Dmax = 4095
- Digital Output (D)
- 12-bit
- ΔV
- ΔD
- A/D: Analog Input vs. Digital Output

Sensor output → A/D input → A/D 12-bit → Digital output
Resolution: A/D

- For this example:
  Resolution = slope = RISE/RUN = (3.3 – 0 / 4095 – 0) = .000805V/bit
  LSB bit weight = .000805V/bit
A/D: Compute Digital output

- Question: If the input is 1.65 V what is the Digital output of the A/D?
- Again note, A/D converters are typically linear
  - Hint 1: What is the equation of a line
**A/D: Compute Digital output**

- **Question:** If the input is 1.65 V what is the Digital output of the A/D?
- **Again note, A/D converters are typically linear**

\[ Y = mX + b; \] in this case the y-intercept \( b = 0 \), so

\[ Y = mX; \] We are given the voltage (\( Y \)) is 1.65V and computed \( m = 0.000805 \), so

\[ 1.65 = 0.000805X; \] \( X = 1.65/0.000805 = 2049.68; \) Truncate to 2049 = 0b1000 0000 0001

For this example, a Temperature of 100 C gives a digital value of 2049
Vmin = 0 V
D = 0

Vmax = 3.3V

ΔV

ΔD

Dmax = 4095

• Question: What is the Temperature LSB weight?
A/D: Compute Digital output

• Question: What is the Temperature LSB weight?
• We know the sensor sensitivity = 60.61 C/ V
• We know the A/D resolution = .000805V/bit
• We want C/bit: So 60.61 C/V * .000805V/bit = .049 C/bit
A change of 1 in the digital output, corresponds to a change .049 degrees
ADC Bit Weight (a closer look)

LSB bit weight in the last example:
bit 0 = .000805V, this is the resolution

Each bit position is weighted with an analog value, such that a 1 in that bit position adds its analog value to the total analog value represented by the digital encoding.

For the previous example:

Decimal: 2049
Binary: 1000 0000 0001
1.648 + .000805 ~= 1.65 V
ADC Bit Weight (What if $V_{\text{min}} \neq 0$)

What if $V_{\text{max}} = 1.65$, and $V_{\text{min}} = 1.65$?

$V_{\text{max}} = 1.65V$

$V_{\text{min}} = -1.65V$

A/D: Analog Input vs. Digital Output
(For $M = 2^n - 1$ steps (or bins), $D_{\text{max}} = V_{\text{max}}$)

Digital Output ($D$)

$D_{\text{max}} = 4095$

12-bit
ADC Bit Weight (What if \( V_{\text{min}} \neq 0 \))

What if \( V_{\text{max}} = 1.65 \), and \( V_{\text{min}} = -1.65 \)?

Since the Range stayed the same (3.3V), the resolution is unchanged.

Thus, the **LSB bit weight** is still:
bit 0 = .000805V, (the resolution)

But now you must add an offset of \( V_{\text{min}} \).
(Can derive from the equation for a line)

**For modified example:**

Decimal: 2049
Binary: 1000 0000 0001

\[
1.648 + 0.000805 \\sim 1.65 \text{ V} \\
-1.65 \text{ V} \\
0 \text{ V}
\]

<table>
<thead>
<tr>
<th>Digital Bit</th>
<th>Bit Weight (V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>2048*r = 1.64864</td>
</tr>
<tr>
<td>10</td>
<td>1024*r = 0.82432</td>
</tr>
<tr>
<td>9</td>
<td>512*r = 0.41216</td>
</tr>
<tr>
<td>8</td>
<td>256*r = 0.20608</td>
</tr>
<tr>
<td>7</td>
<td>128*r = 0.10304</td>
</tr>
<tr>
<td>6</td>
<td>64*r = 0.05152</td>
</tr>
<tr>
<td>5</td>
<td>32*r = 0.02576</td>
</tr>
<tr>
<td>4</td>
<td>16*r = 0.01288</td>
</tr>
<tr>
<td>3</td>
<td>8*r = 0.00644</td>
</tr>
<tr>
<td>2</td>
<td>4*r = 0.00322</td>
</tr>
<tr>
<td>1</td>
<td>2*r = 0.00161</td>
</tr>
<tr>
<td>0</td>
<td>r = 0.000805</td>
</tr>
</tbody>
</table>

http://class.ece.iastate.edu/cpre288
Analog-to-digital converter: Example

Temperature vs. Voltage (Sensor Specification)

- \( T_{\text{min}} = 0 \) C
- \( T_{\text{max}} = 200 \) C
- \( \text{Sensor V}_{\text{min}} = 0 \) V
- \( \text{Sensor V}_{\text{max}} = 3.3 \) V

A/D: Analog Input vs. Digital Output
- \( \text{A/D V}_{\text{max}} = 3.3 \) V
- \( \text{A/D V}_{\text{min}} = 0 \) V
- \( D = 0 \) to \( D_{\text{max}} = 4095 \)

Temperature Sensor

- 100 C

Sensor output

1.65V

A/D input

A/D

Digital output = 515

Sensor output

1.65V

A/D

Digital output = 515

A/D: Analog Input vs. Digital Output
(M = \( 2^n - 1 \) steps (or bins): \( D_{\text{max}} = V_{\text{max}} \))

- Slope = Sensitive \( 60.61 \) C/V
- Slope = Resolution \( .000805 \) V/bit

12-bit
Analog-to-digital converters (Usage)

Mapping between Analog and Digital

\[ V_{\text{max}} = 7.5V \]

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Digital Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0V</td>
<td>1110</td>
</tr>
<tr>
<td>6.5V</td>
<td>1111</td>
</tr>
<tr>
<td>6.0V</td>
<td>1110</td>
</tr>
<tr>
<td>5.5V</td>
<td>1110</td>
</tr>
<tr>
<td>5.0V</td>
<td>1110</td>
</tr>
<tr>
<td>4.5V</td>
<td>1110</td>
</tr>
<tr>
<td>4.0V</td>
<td>1110</td>
</tr>
<tr>
<td>3.5V</td>
<td>1110</td>
</tr>
<tr>
<td>3.0V</td>
<td>1110</td>
</tr>
<tr>
<td>2.5V</td>
<td>1110</td>
</tr>
<tr>
<td>2.0V</td>
<td>1110</td>
</tr>
<tr>
<td>1.5V</td>
<td>1110</td>
</tr>
<tr>
<td>1.0V</td>
<td>1110</td>
</tr>
<tr>
<td>0.5V</td>
<td>1110</td>
</tr>
<tr>
<td>0V</td>
<td>1110</td>
</tr>
</tbody>
</table>

Digital sampling of an analog signal

Digital generation of an analog signal

\[ V_{\text{max}} = 7.5V \]

\[ V_{\text{min}} = 0V \]

\[ D_{\text{max}} = 15 \ (1111) \]

\[ D_{\text{min}} = 0 \ (0000) \]

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis
Analog-to-digital converters (Usage)

Mapping between Analog and Digital

\[ V_{\text{max}} = 7.5V \]

<table>
<thead>
<tr>
<th>Analog Input (V)</th>
<th>Digital Output (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5V</td>
<td>1111</td>
</tr>
<tr>
<td>7.0V</td>
<td>1110</td>
</tr>
<tr>
<td>6.5V</td>
<td>1101</td>
</tr>
<tr>
<td>6.0V</td>
<td>1100</td>
</tr>
<tr>
<td>5.5V</td>
<td>1011</td>
</tr>
<tr>
<td>5.0V</td>
<td>1010</td>
</tr>
<tr>
<td>4.5V</td>
<td>1001</td>
</tr>
<tr>
<td>4.0V</td>
<td>1000</td>
</tr>
<tr>
<td>3.5V</td>
<td>0111</td>
</tr>
<tr>
<td>3.0V</td>
<td>0110</td>
</tr>
<tr>
<td>2.5V</td>
<td>0101</td>
</tr>
<tr>
<td>2.0V</td>
<td>0100</td>
</tr>
<tr>
<td>1.5V</td>
<td>0011</td>
</tr>
<tr>
<td>1.0V</td>
<td>0010</td>
</tr>
<tr>
<td>0.5V</td>
<td>0001</td>
</tr>
<tr>
<td>0V</td>
<td>0000</td>
</tr>
</tbody>
</table>

Proportionality

Digital sampling of an analog signal

Digital generation of an analog signal

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis

4-bit

\[ V_{\text{max}} = 7.5V \]

\[ V_{\text{min}} = 0V \]
Analog-to-digital converters (Usage)

Mapping between Analog and Digital

\[ V_{\text{max}} = 7.5\text{V} \]

<table>
<thead>
<tr>
<th>Voltage (V)</th>
<th>Binary Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0V</td>
<td>1110</td>
</tr>
<tr>
<td>6.5V</td>
<td>1111</td>
</tr>
<tr>
<td>6.0V</td>
<td>1110</td>
</tr>
<tr>
<td>5.5V</td>
<td>1101</td>
</tr>
<tr>
<td>5.0V</td>
<td>1100</td>
</tr>
<tr>
<td>4.5V</td>
<td>1011</td>
</tr>
<tr>
<td>4.0V</td>
<td>1010</td>
</tr>
<tr>
<td>3.5V</td>
<td>1001</td>
</tr>
<tr>
<td>3.0V</td>
<td>1000</td>
</tr>
<tr>
<td>2.5V</td>
<td>1011</td>
</tr>
<tr>
<td>2.0V</td>
<td>1010</td>
</tr>
<tr>
<td>1.5V</td>
<td>1000</td>
</tr>
<tr>
<td>1.0V</td>
<td>1011</td>
</tr>
<tr>
<td>0.5V</td>
<td>1010</td>
</tr>
<tr>
<td>0V</td>
<td>1000</td>
</tr>
</tbody>
</table>

Proportionality

V_{\text{max}} = 7.5\text{V}

Digital sampling of an analog signal

Digital generation of an analog signal

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis
Analog-to-digital converters (Usage)

Mapping between Analog and Digital

\[ V_{\text{max}} = 7.5V \]

<table>
<thead>
<tr>
<th>Analog Input (V)</th>
<th>Digital Output (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0V</td>
<td>1111</td>
</tr>
<tr>
<td>6.5V</td>
<td>1110</td>
</tr>
<tr>
<td>6.0V</td>
<td>1101</td>
</tr>
<tr>
<td>5.5V</td>
<td>1100</td>
</tr>
<tr>
<td>5.0V</td>
<td>1011</td>
</tr>
<tr>
<td>4.5V</td>
<td>1010</td>
</tr>
<tr>
<td>4.0V</td>
<td>1001</td>
</tr>
<tr>
<td>3.5V</td>
<td>1000</td>
</tr>
<tr>
<td>3.0V</td>
<td>0111</td>
</tr>
<tr>
<td>2.5V</td>
<td>0110</td>
</tr>
<tr>
<td>2.0V</td>
<td>0101</td>
</tr>
<tr>
<td>1.5V</td>
<td>0100</td>
</tr>
<tr>
<td>1.0V</td>
<td>0011</td>
</tr>
<tr>
<td>0.5V</td>
<td>0010</td>
</tr>
<tr>
<td>0V</td>
<td>0001</td>
</tr>
</tbody>
</table>

Digital sampling of an analog signal

Digital generation of an analog signal

4-bit

\[ V_{\text{max}} = 7.5V, \quad V_{\text{min}} = 0V \]

\[ D_{\text{max}} = 15 \ (1111) \]

Embedded Systems Design: A Unified Hardware/Software Introduction, (c) 2000 Vahid/Givargis
Analog-to-digital converters (Usage)

Mapping between Analog and Digital

<table>
<thead>
<tr>
<th>Analog Input (V)</th>
<th>Digital Output (D)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.5V</td>
<td>1111</td>
</tr>
<tr>
<td>7.0V</td>
<td>1110</td>
</tr>
<tr>
<td>6.5V</td>
<td>1101</td>
</tr>
<tr>
<td>6.0V</td>
<td>1100</td>
</tr>
<tr>
<td>5.5V</td>
<td>1011</td>
</tr>
<tr>
<td>5.0V</td>
<td>1010</td>
</tr>
<tr>
<td>4.5V</td>
<td>1001</td>
</tr>
<tr>
<td>4.0V</td>
<td>1000</td>
</tr>
<tr>
<td>3.5V</td>
<td>0111</td>
</tr>
<tr>
<td>3.0V</td>
<td>0110</td>
</tr>
<tr>
<td>2.5V</td>
<td>0101</td>
</tr>
<tr>
<td>2.0V</td>
<td>0100</td>
</tr>
<tr>
<td>1.5V</td>
<td>0011</td>
</tr>
<tr>
<td>1.0V</td>
<td>0010</td>
</tr>
<tr>
<td>0.5V</td>
<td>0001</td>
</tr>
<tr>
<td>0V</td>
<td>0000</td>
</tr>
</tbody>
</table>

Proportionality

\[ V_{\text{max}} = 7.5V \]
\[ V_{\text{min}} = 0V \]
\[ D_{\text{max}} = 15 \ (1111) \]

Digital sampling of an analog signal

Digital generation of an analog signal

*Embedded Systems Design: A Unified Hardware/Software Introduction*, (c) 2000 Vahid/Givargis
Proportional Signals (Simple case)

Simple Equation

Assume $V_{\text{min}} = 0$ V.

$V_{\text{max}}$ = maximum voltage of the analog signal

$a$ = analog value

$n$ = number of bits for digital encoding

$2^n$ = number of digital codes

$M$ = number of steps, either $2^n$ or $2^n - 1$

$d$ = digital encoding

$$a / V_{\text{max}} = d / M$$

This is derived from the equation for a line

$$a = [(V_{\text{max}} - V_{\text{min}})/ (2^n - 1 - 0)] * d + 0$$

$$Y = m * X + b$$
Proportional Signals (General case)

General Equation

Do not assume Vmin = 0 V.

\[ \text{Vmax} = \text{maximum voltage of the analog signal} \]

\[ \text{a} = \text{analog value} \]

\[ \text{n} = \text{number of bits for digital encoding} \]

\[ 2^n = \text{number of digital codes} \]

\[ \text{M} = \text{number of steps, either } 2^n \text{ or } 2^n - 1 \]

\[ \text{d} = \text{digital encoding} \]

\[ \frac{(a - \text{Vmin})}{(\text{Vmax} - \text{Vmin})} = \frac{d}{M} \]

This is derived from the equation for a line

\[ a = \left( \frac{\text{Vmax} - \text{Vmin}}{2^n - 1 - 0} \right) \cdot d + \text{Vmin} \]

\[ Y = \text{m} \cdot X + b \]
Resolution: \( M = 2^n - 1 \) vs. \( 2^n \)

Let \( n = 2 \)

\[ M = 2^n - 1 \]

3 steps on the digital scale

\[ d_0 = 0 = 0b00 \]
\[ d_{V_{\text{max}}} = 3 = 0b11 \]

\[ M = 2^n \]

4 steps on the digital scale

\[ d_0 = 0 = 0b00 \]
\[ d_{V_{\text{max}} - r} = 3 = 0b11 \] (no \( d_{V_{\text{max}}} \), it would be at 0b100=4)

\( r \), resolution: analog change resulting from a digital change of 1
Resolution: $M = 2^n - 1$ vs. $2^n$

Let $n = 2$

$M = 2^n - 1$
3 steps on the digital scale

$d_0 = 0 = 0b00$
$d_{v_{\text{max}}} = 3 = 0b11$

$M = 2^n$
4 steps on the digital scale

$d_0 = 0 = 0b00$
$d_{v_{\text{max}}} - r = 3 = 0b11$ (no $d_{v_{\text{max}}}$, it would be at $0b100=4$)

$r$, resolution: analog change resulting from a digital change of 1
Resolution: $M = 2^n - 1$ vs. $2^n$

Let $n = 2$

$M = 2^n - 1$
3 steps on the digital scale

$d_0 = 0 = 0b00$
$d_{V_{\text{max}}} = 3 = 0b11$

$V_{\text{max}} = 12V$

$r = 4V$

$a = 7V$

$V_{\text{min}} = 0V$

$M = 2^n$
4 steps on the digital scale

$d_0 = 0 = 0b00$
$d_{V_{\text{max}}} - r = 3 = 0b11$ (no $d_{V_{\text{max}}}$, it would be at $0b100=4$)

$V_{\text{max}} - r = 3 = 0b11$

$r = 3V$

$V_{\text{max}}$

$r$, resolution: analog change resulting from a digital change of 1
Resolution: \( M = 2^n - 1 \) vs. \( 2^n \)

Let \( n = 2 \)

\( M = 2^n - 1 \)
3 steps on the digital scale

\( d_0 = 0 = 0b00 \)
\( d_{V_{max}} = 3 = 0b11 \)

\( V_{max} = 12V \)
\( r = 4V \)
\( a = 7V \)
\( V_{min} = 0V \)

\( M = 2^n \)
4 steps on the digital scale

\( d_0 = 0 = 0b00 \)
\( d_{V_{max} - r} = 3 = 0b11 \) (no \( d_{V_{max}} \), it would be at \( 0b100=4 \))

\( r, \text{ resolution} \): analog change resulting from a digital change of 1
Resolution: \( M = 2^n - 1 \) vs. \( 2^n \) (Related to slope)

Let \( n = 2 \)

\[ M = 2^n - 1 \]

3 steps on the digital scale

\[ V_{\text{max}} = 12V \]

\[ r = 4V \]

\[ r_{\text{error}} \]

\[ a = 7V \]

\[ V_{\text{min}} = 0V \]

\[ M = 2^n \]

4 steps on the digital scale

\[ d_0 = 0 = 0b00 \]

\[ d_{V_{\text{max}}} = 3 = 0b11 \]

\[ V_{\text{max}} - r = 3 = 0b11 \] (no \( d_{V_{\text{max}}} \), it would be at \( 0b100=4 \))

\[ r_{\text{error}} \]

\[ V_{\text{min}} = 0V \]

\[ 0=00=0V \]

\[ 0=00 = 0V \]

\[ 3=11= 9V \]

\[ 3=11= 9V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 3=11=9V \]

\[ 3=11=9V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 3=11=9V \]

\[ 3=11=9V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 3=11=9V \]

\[ 3=11=9V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 3=11=9V \]

\[ 3=11=9V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 2=10=6V \]

\[ 2=10=6V \]

\[ 1=01 = 3V \]

\[ 1=01 = 3V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]

\[ 3=11=9V \]

\[ 3=11=9V \]

\[ 2=10=8V \]

\[ 2=10=8V \]

\[ 3=11=12V \]

\[ 3=11=12V \]

\[ 0=00 = 0V \]

\[ 0=00 = 0V \]
DAC vs. ADC

• DAC (Digital to Analog Converter)
  – n-bit digital input \( (\text{d}) \)
  – analog out between \( V_{\text{max}} \) and \( V_{\text{min}} \) \( (\text{a}) \)

• ADC (Analog to Digital Converter)
  – analog input between \( V_{\text{max}} \) and \( V_{\text{min}} \) \( (\text{a}) \)
  – n-bit digital output \( (\text{d}) \)
DAC (Digital-to-Analog Converter):

- Conceptually, given a n-bit digital input ($d$), how does the DAC generate an analog output ($a$) between $V_{\text{min}}$ to $V_{\text{max}}$?
DAC (Digital-to-Analog Converter):

- Conceptually, given a $n$-bit digital input ($d$), how does the DAC generate an analog output ($a$) between $V_{\text{min}}$ to $V_{\text{max}}$?
- What other information does the DAC need?
DAC (Digital-to-Analog Converter):

- Conceptually, given a $n$-bit digital input ($d$), how does the DAC generate an analog output ($a$) between $V_{\text{min}}$ to $V_{\text{max}}$?
DAC: Conceptual Implementation

DAC (Digital-to-Analog Converter):

- Conceptually, given a n-bit digital input ($d$), how does the DAC generate an analog output ($a$) between $V_{min}$ to $V_{max}$?
DAC (Digital-to-Analog Converter):

- Conceptually, given a n-bit digital input \( d \), how does the DAC generate an analog output \( a \) between \( V_{\text{min}} \) to \( V_{\text{max}} \)?
DAC (Digital-to-Analog Converter):

- Conceptually, given a \( n \)-bit digital input (\( d \)), how does the DAC generate an analog output (\( a \)) between \( V_{\text{min}} \) to \( V_{\text{max}} \)?
DAC (Digital-to-Analog Converter):

- Conceptually, given a n-bit digital input (\(d\)), how does the DAC generate an analog output (\(a\)) between \(V_{\text{min}}\) to \(V_{\text{max}}\)?
ADC:

Given an analog input (a), how does the ADC know what binary value to assign to digital output (d)?
ADC: Conceptual Implementation

ADC:

Given an analog input (a), how does the ADC know what binary value to assign to digital output (d)?

- Use a DAC to generate analog values for comparison with (a)
- ADC “guesses” a (d), then checks its guess by inputting (d) into the DAC and comparing the generated analog output (a') with original analog input (a)
- How does the ADC guess the correct encoding?
ADC: Digital Encoding

Guessing the encoding is similar to finding an item in a list.

1. Sequential search – counting up: start with an encoding of 0, then 1, then 2, etc. until find a match.
   - $2^n$ comparisons: Slow!

2. Binary search – successive approximation: start with an encoding for half of maximum; then compare analog result with original analog input; if result is greater (less) than the original, set the new encoding to halfway between this one and the minimum (maximum); continue dividing encoding range in half until the compared voltages are equal
   - $n$ comparisons: Faster, but more complex converter

- Each guess takes time (e.g. Assume 1us per guess)
  - 10-bit ADC, what is the time difference for 1. vs. 2. ($2^{10} \sim 1,000$)
  - For a 20-bit ADC? ($2^{20} \sim 1$ Million)
  - For a 30-bit ADC? ($2^{30} \sim 1$ Billion)
ADC: Digital Encoding

Guessing the encoding is similar to finding an item in a list.

1. **Sequential search** – counting up: start with an encoding of 0, then 1, then 2, etc. until find a match.
   - $2^n$ comparisons: Slow!

2. **Binary search** – successive approximation: start with an encoding for half of maximum; then compare analog result with original analog input; if result is greater (less) than the original, set the new encoding to halfway between this one and the minimum (maximum); continue dividing encoding range in half until the compared voltages are equal
   - n comparisons: Faster, but more complex converter

• Each guess takes time (e.g. Assume 1us per guess)
  - 10-bit ADC, what is the time difference for 1. vs. 2. ($2^{10} \sim 1,000$): 1ms vs. 10us
  - For a 20-bit ADC? ($2^{20} \sim 1$ Million)
  - For a 30-bit ADC? ($2^{30} \sim 1$ Billion)
ADC: Digital Encoding

Guessing the encoding is similar to finding an item in a list.

1. Sequential search – counting up: start with an encoding of 0, then 1, then 2, etc. until find a match.
   • $2^n$ comparisons: Slow!

2. Binary search – successive approximation: start with an encoding for half of maximum; then compare analog result with original analog input; if result is greater (less) than the original, set the new encoding to halfway between this one and the minimum (maximum); continue dividing encoding range in half until the compared voltages are equal
   • $n$ comparisons: Faster, but more complex converter

• Each guess takes time (e.g. Assume 1us per guess)
  – 10-bit ADC, what is the time difference for 1. vs. 2. ($2^{10} \sim 1,000$): 1ms vs. 10us
  – For a 20-bit ADC? ($2^{20} \sim 1$ Million): 1s vs. 20us
  – For a 30-bit ADC? ($2^{30} \sim 1$ Billion)
ADC: Digital Encoding

Guessing the encoding is similar to finding an item in a list.

1. Sequential search – counting up: start with an encoding of 0, then 1, then 2, etc. until find a match.
   - $2^n$ comparisons: Slow!

2. Binary search – successive approximation: start with an encoding for half of maximum; then compare analog result with original analog input; if result is greater (less) than the original, set the new encoding to halfway between this one and the minimum (maximum); continue dividing encoding range in half until the compared voltages are equal
   - $n$ comparisons: Faster, but more complex converter

- Each guess takes time (e.g. Assume 1us per guess)
  - 10-bit ADC, what is the time difference for 1. vs. 2. ($2^{10} \sim 1,000$): 1ms vs. 10us
  - For a 20-bit ADC? ($2^{20} \sim 1$ Million): 1s vs. 20us
  - For a 30-bit ADC? ($2^{30} \sim 1$ Billion): 1000s = 16min vs. 30us
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

DAC

Comparator

Vmax

Vmin

Guess

SAR

Digital output

State machine

Timing control

SAR BUF

SAR: Successive approximation register

1 OR 0

n

Analog Input

Vmax

Vmin

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

V_{\text{max}} = 16\text{V} \\
V_{\text{min}} = 0\text{V}

Let \( M = 2^n \)

\begin{align*}
\text{Digital output} \\
\text{SAR: Successive approximation register}
\end{align*}

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Let $M = 2^n$

V_{\text{max}} = 16V

V_{\text{min}} = 0V

 DAC

Comparator

Analog Input

a = 9.5V

State machine

Digital output

SAR: Successive approximation register

Timing control

http://class.ece.iastate.edu/cpre288

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a $\geq$ Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Analog Input
a=9.5V

DAC

Let M = \(2^n\)

vmax=16V

vmmin=0V

Comparator

1 OR 0

SAR

n=4

SAR: Successive approximation register

State machine

Timing control

Digital output

Table:

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a &gt;= Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0bxxxx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0bxxxx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0bxxxx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0bxxxx</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

\[ V_{\text{max}} = 16V \]
\[ V_{\text{min}} = 0V \]

Analog Input
\[ a = 9.5V \]

Let \( M = 2^n \)
\[ n = 4 \]

Digital output

State machine

Timing control

SAR: Successive approximation register

Guess = 8V

\[
\begin{array}{c|c|c|c|c}
\text{Step} & \text{Range} & \text{Mid (digital)} & \text{Mid (voltage)} & \text{Is a } \geq \text{ Guess (voltage)?} \\
0 & 0bxxxx & 0b1000 & 8 Volts & \\
1 & & & & \\
2 & & & & \\
3 & & & & \\
4 & & & & \\
\end{array}
\]
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

- Analog Input: a = 9.5V
- DAC
- Let M = $2^n$
- n = 4
- V_{max} = 16V
- V_{min} = 0V
- Guess = 8V
- Comparator

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a &gt;= Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SAR: Successive approximation register

Digital output

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Analog Input

\[ V_{\text{max}} = 16V \]

\[ V_{\text{min}} = 0V \]

Let \( M = 2^n \)

\[ n = 4 \]

\[ \text{Guess} = 8V \]

State machine

Timing control

SAR BUF

Digital output

SAR: Successive approximation register

DAQ

Comparator

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a ( \geq ) Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Analog Input
a = 9.5V

Digital output

Vmax = 16V
Vmin = 0V

Let M = 2^n

DAC

Guess = 8V

Comparator

State machine

Timing control

SAR

n = 4

SAR BUF

SAR: Successive approximation register

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a &gt;= Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

- Analog Input: \( V_{\text{max}} = 16V \), \( V_{\text{min}} = 0V \)
- Let \( M = 2^n \) with \( n = 4 \)
- Guess = 12V

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a ( \geq ) Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
It’s built upon a DAC

DAC

Vmax=16V
Vmin=0V

Comparator

Let M = 2^n

n=4

Let a = 9.5V

Analog Input

State machine

Timing control

Digital output

SAR: Successive approximation register

Guess = 12V

0bxxxx

0b1000

8 Volts

Yes

0b1xxx

0b1100

12 Volts

No

Step | Range  | Mid (digital) | Mid (voltage) | Is a >= Guess (voltage)?
-----|--------|--------------|---------------|-------------------
0    | 0bxxxx| 0b1000       | 8 Volts       | Yes               |
1    | 0b1xxx| 0b1100       | 12 Volts      | No                |
2    |        |              |               |                   |
3    |        |              |               |                   |
4    |        |              |               |                   |

http://class.ece.iastate.edu/cpre288
constructing the adc (successive approximation)

it’s built upon a dac

vmax=16v
vmin=0v

analog input
a=9.5v

let m = 2^n
n=4

guess = 12v

0b1xxx 0b1100 12 volts no

step range mid (digital) mid (voltage) is a >= guess (voltage)?
0 0bxxxx 0b1000 8 volts yes
1 0b1xxx 0b1100 12 volts no
2 0b10xx
3
4

http://class.ece.iastate.edu/cpre288

67
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

- Analog Input
  - a = 9.5V

- Let $M = 2^n$
  - $n = 4$

- DAC
  - $V_{max} = 16V$
  - $V_{min} = 0V$

- Guess = 12V

- SAR: Successive approximation register

- SAR BUF

- State machine

- Timing control

- Digital output

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a $\geq$ Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>0b10xx</td>
<td>0b1010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

DAC

Comparator

Let M = 2^n

Vmax=16V
Vmin=0V

Analog Input
a=9.5V

Digital output

State machine

Timing control

n=4

SAR

BUF

ADC: Successive approximation register

Step | Range | Mid (digital) | Mid (voltage) | Is a >= Guess (voltage)?
---|---|---|---|---
0 | 0bxxxx | 0b1000 | 8 Volts | Yes
1 | 0b1xxx | 0b1100 | 12 Volts | No
2 | 0b10xx | 0b1010 | 10 Volts | |
3 |
4 |
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Analog Input

DAC

Comparator

Let $M = 2^n$

$V_{\text{max}} = 16V$

$V_{\text{min}} = 0V$

Guess = 10V

$V_{\text{max}} = 16V$

$V_{\text{min}} = 0V$

SAR: Successive approximation register

Timing control

Digital output

State machine

$\text{SAR BUF}$

Step | Range | Mid (digital) | Mid (voltage) | Is a $\geq$ Guess (voltage)?
--- | --- | --- | --- | ---
0 | 0bxxxx | 0b1000 | 8 Volts | Yes
1 | 0b1xxx | 0b1100 | 12 Volts | No
2 | 0b10xx | 0b1010 | 10 Volts | No
3 | 0b1010 | 0b1010 | 10 Volts | No
4 | 0b1011 | 0b1011 | 12 Volts | No

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

- V_{max} = 16V
- V_{min} = 0V

Let M = 2^n

Guess = 10V

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a &gt;= Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>0b10x</td>
<td>0b1010</td>
<td>10 Volts</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>0b100x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Digital output

SAR: Successive approximation register

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

V_{\text{max}}=16\text{V}
V_{\text{min}}=0\text{V}

Scheme:

\text{DAC} \rightarrow \text{Comparator} \rightarrow \text{SAR (Successive Approximation Register)} \rightarrow \text{State Machine} \rightarrow \text{SAR BUF}

\text{Analog Input} \ a=9.5\text{V}

\text{Let } M = 2^n
n=4

\text{Guess} = 10\text{V}

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a &gt;= Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>0b10xx</td>
<td>0b1010</td>
<td>10 Volts</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>0b100x</td>
<td>0b1001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\text{http://class.ece.iastate.edu/cpre288}
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

- **Vmax=16V**
- **Vmin=0V**

Let $M = 2^n = 4$

**Guess = 9V**

**Analog Input $a=9.5V$**

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a $\geq$ Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0xxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>0b10x</td>
<td>0b1010</td>
<td>10 Volts</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>0b100</td>
<td>0b1001</td>
<td>9 Volts</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Comparison:

- **0 (No)**

Digital output

DAC: Digital to Analog Converter

State machine

Timing control

SAR: Successive approximation register

http://class.ece.iastate.edu/cpre288
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Let $M = 2^n$

$V_{\text{max}} = 16V$

$V_{\text{min}} = 0V$

Let $a = 9.5V$

Comparitor

Digital output

State machine

Timing control

SAR: Successive approximation register

<table>
<thead>
<tr>
<th>Step</th>
<th>Range</th>
<th>Mid (digital)</th>
<th>Mid (voltage)</th>
<th>Is a $\geq$ Guess (voltage)?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0bxxxx</td>
<td>0b1000</td>
<td>8 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>1</td>
<td>0b1xxx</td>
<td>0b1100</td>
<td>12 Volts</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>0b10xx</td>
<td>0b1010</td>
<td>10 Volts</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>0b100x</td>
<td>0b1001</td>
<td>9 Volts</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>0b1001</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Constructing the ADC (Successive Approximation)

It’s built upon a DAC

Vmax=16V
Vmin=0V

Let M = 2^n

Guess = 9V

DAC

Comparator

Analog Input
a=9.5V

Let M = 2^n

n=4

SAR

SAR: Successive approximation register

Timing control

Digital output

Step | Range  | Mid (digital) | Mid (voltage) | Is a >= Guess (voltage)?
-----|--------|--------------|---------------|-------------------
0    | 0bxxxx | 0b1000       | 8 Volts       | Yes               
1    | 0b1xxx | 0b1100       | 12 Volts      | No                
2    | 0b10xx | 0b1010       | 10 Volts      | No                
3    | 0b100x | 0b1001       | 9 Volts       | Yes               
4    | 0b1001 |              |               |                   

http://class.ece.iastate.edu/cpre288
Example: 0V-16V range, 9.5V input, 4-bit resolution.

\[
\frac{a - V_{\text{min}}}{V_{\text{max}} - V_{\text{min}}} = \frac{d}{2^n}
\]

\[
\frac{9.5}{16 - 0} = \frac{d}{16}
\]

\[
d = \frac{9.5 \times 16}{16} = 9.5
\]

\[d = 0b1001 = 9 \text{ (Why not just use the above equation?)}\]
ADC Using Successive Approximation

- Example: 0V-16V range, 9.5V input, 4-bit resolution.

\[
\frac{a - V\text{min}}{V\text{max} - V\text{min}} = \frac{d}{2^n}
\]

\[
\frac{9.5}{16 - 0} = \frac{d}{16}
\]

\[
d = \frac{9.5 \times 16}{16} = 9.5
\]

\[d = 0b1001 = 9\] (Why not just use the above equation?)

(i.e. Why is Successive Approximation needed?)
Example
- 2-bits of resolution
- $V_{\text{min}} = 0$ Volts
- $V_{\text{max}} = 12$ Volts

$M = 2^n$

$2^2 = 4$ buckets (or ranges) for the analog signal to fall

**r, resolution**: analog change resulting from a digital change of 1
Practice Problem 1 (Linear equation)

• Question:
  – Given:
    • \( n = 2 \) bit resolution
    • \( V_{\text{min}} = 0 \) volts
    • \( V_{\text{max}} = 12 \) volts
    • \( a = 5 \) volts
    • \( M = 2^n \) bins
  – What is \( d \)?
Practice Problem 1 (Linear equation)

• Question:
  – Given:
    • \( n = 2 \) bit resolution
    • \( V_{\text{min}} = 0 \) volts
    • \( V_{\text{max}} = 12 \) volts
    • \( a = 5 \) volts
    • \( M = 2^n \) bins
  – What is \( d \)?

• Answer:
  – Analog and Digital signals are proportional
    \[
    \frac{a - V_{\text{min}}}{V_{\text{max}} - V_{\text{min}}} = \frac{d}{2^n}
    \]
  – \( d \) is 0b01

http://class.ece.iastate.edu/cpre288
Practice Problem 2 (Successive Approximation)

• Question:
  – Given:
    • \( n = 4 \) bit resolution
    • \( V_{\text{min}} = 0 \) volts
    • \( V_{\text{max}} = 12 \) volts
    • \( a = 5 \) volts
    • \( M = 2^n \) bins
  – What is \( d \)?
Practice Problem 2 (Successive Approximation)

- **Question:**
  - **Given:**
    - \( n = 4 \) bit resolution
    - \( V_{\text{min}} = 0 \) volts
    - \( V_{\text{max}} = 12 \) volts
    - \( a = 5 \) volts
    - \( M = 2^n \) bins
  - What is \( d \)?

- **Answer:**
  - Use successive approximation
  - \( d = 0b??? \)
    - Midpoint is \( 0b1000 \) at 6 volts
    - If \( a \) is greater than 6 volts, record a 1
    - If \( a \) is less than 6 volts, record a 0

http://class.ece.iastate.edu/cpre288
Practice Problem 2 (Successive Approximation)

• Question:
  – Given:
    • $n = 4$ bit resolution
    • $V_{\text{min}} = 0$ volts
    • $V_{\text{max}} = 12$ volts
    • $a = 5$ volts
    • $M = 2^n$ bins
  – What is $d$?

• Answer:
  – Use successive approximation
  – $d = 0b0???
    • Midpoint is $0b0100$ at 3 volts
    • If $a$ is greater than 3 volts, record a 1
    • If $a$ is less than 3 volts, record a 0

http://class.ece.iastate.edu/cpre288
Practice Problem 2 (Successive Approximation)

• **Question:**
  
  – **Given:**
    
    • \( n = 4 \) bit resolution
    • \( V_{\text{min}} = 0 \) volts
    • \( V_{\text{max}} = 12 \) volts
    • \( a = 5 \) volts
    • \( M = 2^n \text{bins} \)
  
  – **What is \( d \)?**

• **Answer:**
  
  – Use successive approximation
  – \( d = 0b01?? \)
    
    • Midpoint is \( 0b0110 \) at 4.5 volts
    • If \( a \) is greater than 4.5 volts, record a 1
    • If \( a \) is less than 4.5 volts, record a 0
• **Question:**
  - Given:
    • \( n = 4 \) bit resolution
    • \( V_{\text{min}} = 0 \) volts
    • \( V_{\text{max}} = 12 \) volts
    • \( a = 5 \) volts
    • \( M = 2^n \text{bins} \)
  - What is \( d \)?

• **Answer:**
  - Use successive approximation
  - \( d = 0b011? \)
    • Midpoint is \( 0b0111 \) at 5.25 volts
    • If \( a \) is greater than 5.25 volts, record a 1
    • If \( a \) is less than 5.25 volts, record a 0
**Practice Problem 2 (Successive Approximation)**

- **Question:**
  - **Given:**
    - $n = 4$ bit resolution
    - $V_{\text{min}} = 0$ volts
    - $V_{\text{max}} = 12$ volts
    - $a = 5$ volts
    - $M = 2^n\text{bins}$
  - **What is $d$?**

- **Answer:**
  - Use successive approximation
  - $d = 0b0110$
    - Midpoint is $0b0111$ at 5.25 volts
    - If $a$ is greater than 5.25 volts, record a 1
    - If $a$ is less than 5.25 volts, record a 0
ADC ON TM4C123G
2 12-bit ADCs
12 shared input channels
Analog input on one of ADC0-ADC7 pins
Up to 1M samples per second
0 – Vcc or 0 – 2.56V ADC input voltage range

Which pins on the TM4C123G are used for the ADC?
– Alternative I/O Functions (See next slides)
TM4C123G I/O Ports and **Alternative Functions**

Ports A-F, each pin can be configured as **General Purpose Digital I/O Pin**:
- GPIODIR decides if a pin is for input or output *
- GPIODATA is for writing/reading from pins
(* There is a special tri-state configuration)

**Alternatively**, those pins can be used as I/O pins for internal I/O devices
Most pins have Alternative Functions: USART, ADC, input capture, output compare, and others

UART3 uses port C
– PC5: Transmit Pin
– PC4: Receive Pin

UART1 uses port B
– PB1: Transmit Pin
– PB0: Receive Pin
TM4C123G I/O Pins (Alternative Functions)

From TM4C123G Data Sheet starting on pg 799
The ADC uses Port B, D, and E:

- There are 12 input channels
- In lab we will use AIN10 on pin PB4
- See table 13-1 on pg 801 of datasheet for all channels
Programming Interface Registers
ADC GPIO Registers

**RCGCGPIO**  General-Purpose Input/Output Run Mode Clock Gating Control

**GPIOAFSEL**  GPIO Alternate Function Select

**GPIODIR**  GPIO Direction

**GPIODEN**  GPIO Digital Enable

**GPIOAMSEL**  GPIO Analog Mode Select

**GPIOADCCTL**  GPIO ADC Control
RCGCGPIO – GPIO run mode clock gating control

5 R5 – 0 disable port F, 1 provide clock to port F
4 R4 – 0 disable port E, 1 provide clock to port E
3 R3 – 0 disable port D, 1 provide clock to port D
2 R2 – 0 disable port C, 1 provide clock to port C
1 R1 – 0 disable port B, 1 provide clock to port B
0 R0 – 0 disable port A, 1 provide clock to port A

In lab we will provide clock to port B
GPIOAFSEL – Alternate function select

7:0 AFSEL – 0 pin functions as normal GPIO, 1 pin works with alternate function

In lab we will set PB4 to alternate function
GPIODIR – GPIO direction

7:0 DIR – 0 pin is input, 1 pin is output

In lab we will set PB4 to input
7:0 DEN – 0 digital function disabled, 1 digital function enabled

In lab we will disable PB4 digital function
GPIOAMSEL – Analog mode select

7:0 GPIOAMSEL – 0 analog function disabled, 1 analog function enabled

In lab we will enable PB4 analog function
GPIOADCCTL – ADC control

7:0 ADCEN – 0 = pin not used to trigger ADC, 1 = pin is used to trigger ADC

In lab we will not be using any pins to trigger ADC conversion, so this can be written 0x00, which should be the default value
ADC Registers

RCGCADC  Analog-to-Digital Converter Run Mode Clock Gating Control
ADCACTSS  ADC Active Sample Sequencer
ADCEMUX  ADC Event Multiplexer Select
ADCSSCTL1  ADC Sample Sequence Control 1
ADCPSSI  ADC Processor Sample Sequence Initiate
ADCRIS  ADC Raw Interrupt Status
ADCSSFIFO1  ADC Sample Sequence Result FIFO one
ADCISC  ADC Interrupt Status and Clear
ADCSAC  ADC Sample Averaging Control

Pg 819 of datasheet
RCGCADC – ADC run mode clock gating control

1 R1 – 1 provide clock to ADC1, 0 disable ADC1
0 R0 – 1 provide clock to ADC0, 0 disable ADC0
3 ASEN3 – 0 disable SS3, 1 enable SS3
2 ASEN2 – 0 disable SS2, 1 enable SS2
1 ASEN1 – 0 disable SS1, 1 enable SS1
0 ASEN0 – 0 disable SS0, 1 enable SS0
3:0 EM0 – SS0 trigger select
7:4 EM1 – SS1 trigger select
11:8 EM2 – SS2 trigger select
15:12 EM3 – SS3 trigger select

In lab we will be using SS1. EM1 will be set to 0x0 using the default trigger, which means your C program must write to a memory mapped register to start a conversion.
ADCSSCTRLn – Sample Sequence Control

3 TS0 – 1 if temp sensor 1st sample differential input select, 0 if ADCSSMUXn is read
2 IE0 – 1 if 1st sample raw interrupt enable, else 0
1 END0 – 1 if 1st sample is end of sequence, else 0
0 D0 – 1st sample input select

*3, *2, *1, refer to 4th, 3rd, and 2nd samples but function the same

In lab we will only need to sample once so we will end at the first sample.
We will also check the raw interrupt.

***Note this specific register is for SS1 though others will be nearly identical check the datasheet
31 GSYNC – global synchronize, 1 will start all initialized and syncwait ADCs, 0 cleared once initiated

27 SYNCWAIT – synchronize wait, 1 allows for gsync to initiate all initialized ADCs

3 SS3 – SS3 initiate, 1 begins sampling SS3

2 SS2 – SS2 initiate, 1 begins sampling SS2

1 SS1 – SS1 initiate, 1 begins sampling SS1

0 SS0 – SS0 initiate, 1 begins sampling SS0

This is used to trigger the start of a conversion
ADCRIS – Raw interrupt status

3 INR3 – SS3 raw interrupt status, 1 means conversion complete on sample sequencer
2 INR2 – SS2 raw interrupt status, 1 means conversion complete on sample sequencer
1 INR1 – SS1 raw interrupt status, 1 means conversion complete on sample sequencer
0 INR0 – SS0 raw interrupt status, 1 means conversion complete on sample sequencer

In lab we can use this to check when conversions are complete
ADCSSFIFO - Sample sequence result FIFO

11:0 DATA – contains the conversion result for SSn
ADCISC – Interrupt status and clear

19:16 DCINSSn – digital comparator interrupt status on SSn
3:0 INn – Interrupt Status and clear if read 1 ADCRIS and MASKn set, write 1 to clear interrupt

In lab we will only be using INn registers
2:0 AVG –

0x0 = No oversample
0x1 = 2x hardware oversample
0x2 = 4x hardware oversample
0x3 = 8x hardware oversample
0x4 = 16x hardware oversample
0x5 = 32x hardware oversample
0x6 = 64x hardware oversample
0x7 = reserved
Enable ADC0 using SS0:

//enable ADC 0 module on port D
SYSCTL_RCGCGPIO_R |= SYSCTL_RCGCGPIO_R3;

//enable clock for ADC
SYSCTL_RCGCADC_R |= 0x1;

//enable port D pin 0 to work as alternate functions
GPIO_PORTD_AFSEL_R |= 0x01;

//set pin to input - 0
GPIO_PORTD_DEN_R &= 0b11111110;

//disable analog isolation for the pin
GPIO_PORTD_AMSEL_R |= 0x01;

//initialize the port trigger source as processor (default)
GPIO_PORTD_ADCCTL_R = 0x00;
Enable ADC0 using SS0 continued:

//disable SS0 sample sequencer to configure it
ADC0_ACTSS_R &= ~ADC_ACTSS_ASEN0;

//initialize the ADC trigger source as processor (default)
ADC0_EMUX_R = ADC_EMUX_EM0_PROCESSOR;

//set 1st sample to use the AIN10 ADC pin
ADC0_SSMUX0_R |= 0x000A;

//enable raw interrupt status
ADC0_SSCTL0_R |= (ADC_SSCTL0_IE0 | ADC_SSCTL0_END0);

//enable oversampling to average
ADC0_SAC_R |= ADC_SAC_AVG_64X;

//re-enable ADC0 SS0
ADC0_ACTSS_R |= ADC_ACTSS_ASEN0;
Use a different coding style: using defines

\[
\text{ADC0\_ACTSS\_R} |\!|= (\text{ADC\_ACTSS\_ASEN1} \mid \text{ADC\_ACTSS\_ASEN0});
\]

Is the same as

\[
\text{ADC0\_ACTSS\_R} |\!|= (0x00000002 \mid 0x00000001);
\]

ADC\_ACTSS\_ASENn is defined in tm4c123gh6pm.h

This style allows better portability of your code if you code may be run on a family of microcontrollers
Assume ADC has been configured appropriately and in one-shot mode without an interrupt handler.

Write code to (1) start ADC0, (2) wait for the conversion to complete, and (3) read the output.

```c
// initiate SS1 conversion
ADC0_PSSI_R = ADC_PSSI_SS1;

// wait for ADC conversion to be complete
while ((ADC0_RIS_R & ADC_RIS_INR1) == 0){
    // wait
}

// grab result
int value = ADC0_SSFIFO1_R;
```
Enable ADC interrupt handler

//clear interrupt flags
ADC0_ISC_R |= ADC_ISC_IN0;

//enable ADC0SS0 interrupt
ADC0_IM_R |= ADC_IM_MASK0;

//enable interrupt for IRQ 14 set bit 14
NVIC_EN0_R |= 0x00004000;

//tell cpu to use ISR handler for ADC0SS0
IntRegister(INT_ADC0SS0, ADC0SS0_Handler);

//enable global interrupts
IntMasterEnable();
Manually check raw ADC interrupt (polling)

//initiate SS0 conversion
ADC0_PSSI_R=ADC_PSSI_SS0;

//wait for ADC conversion to be complete
while((ADC0_RIS_R & ADC_RIS_INR0) == 0){
    //wait
}

//clear interrupt
ADC0_ISC_R=ADC_ISC_IN0;
unsigned ADC_read(char channel){
  //disable ADC0SS0 sample sequencer to configure it
  ADC0_ACTSS_R &= ~ADC_ACTSS_ASEN0;

  //set 1st sample to use the channel ADC pin
  ADC0_SSMUX0_R |= channel;

  //re-enable ADC0 SS0
  ADC0_ACTSS_R |= ADC_ACTSS_ASEN0;

  //initiate SS0 conversion
  ADC0_PSSI_R=ADC_PSSI_SS0;

  //wait for ADC conversion to be complete
  while((ADC0_RIS_R & ADC_RIS_INR0) == 0){}

  //clear interrupt
  ADC0_ISC_R=ADC_ISC_IN0;

  return ADC0_SSFIFO0_R;
}
### Sampling Dynamics

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value 1</th>
<th>Value 2</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_{ADC}$</td>
<td>ADC conversion clock frequency</td>
<td>-</td>
<td>16</td>
<td>MHz</td>
</tr>
<tr>
<td>$F_{CONV}$</td>
<td>ADC conversion rate</td>
<td>-</td>
<td>1</td>
<td>Msps</td>
</tr>
<tr>
<td>$T_S$</td>
<td>ADC sample time</td>
<td>-</td>
<td>250</td>
<td>ns</td>
</tr>
<tr>
<td>$T_C$</td>
<td>ADC conversion time</td>
<td>-</td>
<td>1</td>
<td>μs</td>
</tr>
<tr>
<td>$T_{LT}$</td>
<td>Latency from trigger to start of conversion</td>
<td>-</td>
<td>2</td>
<td>ADC clocks</td>
</tr>
</tbody>
</table>

Pg 1389 of datasheet
Measuring Distance with the IR Sensor

- The IR sensor emits an IR beam, and sets a voltage based on the distance of an object
From the IR Sensor Datasheet

• The voltage from the IR sensor depends on the distance

• As the distance increases, the voltage decreases (see graph)
Getting a distance from the IR sensor involves the following process:

1. The IR sensor measures a distance and sets the voltage on the wire leading to GPIO PIN D0
2. The ADC converts this voltage into a digital value between 0 and 4095 and stores it in the register ADCSSFIFO
3. Your program reads value and converts the value into a distance... but how?!?
How To Measure Distance with the IR Sensor

• Two methods to calibrate your distance
• Measure 50 points, create a table for comparing
  – Create a table that has the value of ADCSSFIFO when an object is X centimeters away
  – Use this table to lookup the distance when a similar ADCSSFIFO result is returned

• Measure 5 points, use Excel to get a trend line
ADC Summary

• ADC general knowledge
  – Applications, sampling and quantization
  – ADC conversion formulas
  – ADC design: Successive approximation
  – Terminology, Performance and other issues

• TM4C123G ADC programming interface
  – GPIO initialize
  – ADC initialize
  – Reading ADC (polling vs interrupts)

• API functions you will create in lab
  – ADC_init()
  – ADC_read()
Take 5 minutes to think about and respond to the following questions:

• “Describe your experience during the in class UART activity and what you took away from the activity about engineering practice.”

Upload your response to these two questions to BlackBoard by midnight today (i.e. first day of class).

Final Product: [https://www.youtube.com/watch?v=ulidN0rs1NA](https://www.youtube.com/watch?v=ulidN0rs1NA)
Microcontroller / System-on-Chip (SoC)

Microcontroller

- Program Memory (C-code)
- CPU
- Data Memory (variables)

Outside World

Interrupt Controller

Devices
- UART
- ADC
- Timers

GPIO