Topic 18

Link

Prof Peter YK Cheung
Dyson School of Design Engineering

URL: www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/
E-mail: p.cheung@imperial.ac.uk
Linking between analogue and digital domains

- Much of the physical world is analogue in nature.
- Linking digital electronics as used in microprocessors and the physical world is three digital-to-analogue and analogue-to-digital converters. (It is common that DACs and ADCs use the American spelling of analog.)
**Sampling and Quantization**

- **Sampling Process**: convert a continuous time domain signal at discrete and uniform time intervals.
- Determines maximum bandwidth of sampled (ADC) or reconstructed (DAC) signal (Nyquist Sampling Theorem: sampling frequency must be at least twice that of maximum signal frequency)
- Frequency Domain - “Aliasing” for an ADC and “Images” for a DAC
- **Quantization Process**: convert an analogue signal with infinite resolution with a digital word having finite resolution and an analogue output which only exists in discrete levels
  - Determines Maximum Achievable Dynamic Range
  - Results in Quantization Error/Noise
A digital-to-analog converter (DAC) produces a quantized (discrete step) analogue output (voltage or current) in response to binary digital input.

- A reference quantity (either voltage or current) is accurately divided into binary and/or linear segments.
- The digital input drives transistor switches that connect an appropriate number of segments to the output.
- $2^N$ discrete values resulting in quantization errors.
- Sampling and quantization impose fundamental but predictable limitations in the system.
- The microcontroller on the Pyboard has 2 x 12 bit DAC converters.

$$V_{OUT} = \frac{X_{IN}[N-1:0]}{2^N - 1} \times V_{REF}$$
Resolution in various forms

- **Resolution** of a ADC or DAC is dependent on the number of input data bits.
- This defines the quantization step, which is expressed as LSB voltage (least significant bit).
- Resolution can also be expressed as %, parts per million (ppm), or dB relative to full scale (FS).

<table>
<thead>
<tr>
<th>Resolution, Bits (n)</th>
<th>$2^n$</th>
<th>LSB, mV (10V FS)</th>
<th>% Full Scale</th>
<th>ppm Full Scale</th>
<th>dB Full Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>256</td>
<td>39.1</td>
<td>0.391</td>
<td>3906</td>
<td>-48.0</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>9.77</td>
<td>0.098</td>
<td>977</td>
<td>-60.0</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>2.44</td>
<td>0.024</td>
<td>244</td>
<td>-72.0</td>
</tr>
<tr>
<td>14</td>
<td>16,384</td>
<td>0.610</td>
<td>0.006</td>
<td>61</td>
<td>-84.0</td>
</tr>
<tr>
<td>16</td>
<td>65,536</td>
<td>0.153</td>
<td>0.0015</td>
<td>15</td>
<td>-96.0</td>
</tr>
<tr>
<td>18</td>
<td>262,164</td>
<td>0.038</td>
<td>0.00038</td>
<td>3.8</td>
<td>-108.0</td>
</tr>
<tr>
<td>20</td>
<td>1,048,576</td>
<td>0.0095</td>
<td>0.00010</td>
<td>1.0</td>
<td>-120.0</td>
</tr>
</tbody>
</table>
Common DAC architectures

- **Simple voltage divider DAC**: dividing reference voltage into equal steps. 

  - Output is guaranteed to go up if input goes up – i.e. monotonic.

  - Number of taps increases exponentially ($2^n$) with number of bits.

- **Voltage segment DAC**: use two sets of voltage dividers, one for coarse steps and another for fine steps.

  - Number of resistors reduced from $2^n$ to $2 \times 2^{n/2}$ for two segments.

  - Could use three or more segments.

- **R-2R Ladder DAC**: same network as used in Lab 2, Task 3.

  - Each stage has a voltage (or current) half that of the previous stage.

  - Complexity is $\Omega(n)$ or $O(n)$, where $n$ is the number of input bits.
An ADC produces a digital output corresponding to the value of the signal applied to its input relative to a reference voltage.

- There are $2^N$ discrete values, therefore introducing error known as quantization noise.
- Similar effect in sampling and quantization as found in DAC.
- The microcontroller on the Pyboard has 3 x 12-bit ADC, capable of taking 2.4 Msamples per second.

\[
Y[N-1:0] = \frac{V_{IN}}{V_{REF}} \times (2^N - 1)
\]
Video on DAC converter

Digital to Analog Converter
The transmission of binary data across a link can be accomplished either in parallel mode or serial mode.
In parallel mode, multiple bits are sent with each clock period.
In serial mode, one bit is sent with each clock period.
There are 2 subclasses of serial transmission: synchronous and asynchronous.

Binary data may be organized into groups of n bits each.
By grouping, we can send data n bits at a time instead of one. This is called parallel transmission.
The advantage of parallel transmission is speed but its disadvantage is cost in interconnect resources.
In serial transmission, one bit follows another, so we need only one communicating channel (or wire) to transmit data between 2 communicating modules. The advantage of serial transmission is the reduction of interconnect resources, but it takes $N$ times longer to send the information, where $N$ is the number of bits of data. Serial transmission occurs in one of 2 ways: asynchronous or synchronous.
UART and flow control

- UART is a module found in many microcontroller that uses an asynchronous serial method for transferring data.
- The ESP32 uses UART to talk to your laptop via the USB cable.
- The UART signal waveform is shown below. (You have also seen this in Lab 1).
- Physical connections between two UARTs consist of four signals:
  - RX – data receive signal (input)
  - TX – data transmit signal (output)
  - CTS – clear to send signal (input)
  - RTS – ready to send signal (output)
- Both devices can simultaneously send and receive data. This is known as a full duplex link.

- RTS and CTS are active-low signals to indicate when a device is ready to transfer data.
- By connecting, say, the UART on the ESP32 to the laptop, one can control the flow of data between the two. This is called flow control mechanism.
One-wire link - Neopixel

Composition of 24bit data:

<table>
<thead>
<tr>
<th>G7</th>
<th>G6</th>
<th>G5</th>
<th>G4</th>
<th>G3</th>
<th>G2</th>
<th>G1</th>
<th>G0</th>
<th>R7</th>
<th>R6</th>
<th>R5</th>
<th>R4</th>
<th>R3</th>
<th>R2</th>
<th>R1</th>
<th>R0</th>
<th>B7</th>
<th>B6</th>
<th>B5</th>
<th>B4</th>
<th>B3</th>
<th>B2</th>
<th>B1</th>
<th>B0</th>
</tr>
</thead>
</table>

Note: Follow the order of GRB to sent data and the high bit sent at first.
Synchronous Serial Link

◆ In a synchronous link, the bit stream is combined into longer frames which may contain multiple bytes.
◆ Each byte is introduced onto the transmission link without a gap between it and the next one.
◆ It is the responsibility of the receiver to reconstruct the information, usually under the control of a clock signal.
◆ Without gaps and start/stop bits, timing becomes very important therefore the accuracy of the received information is completely dependent on the ability of the receiver to keep an accurate count of the bits as they come in.
I2C and SPI synchronous serial links

- I2C stands for Inter-Integrated Circuit serial protocol. Also known as I²C or Two Wire Interface (TWI).
- Originally from Philips Semiconductor, and it is now an industrial standard.
- Allows up to 127 devices to be connected, each having a unique address.
- Up to 400kHz data rate.
- SPI stands for Serial Peripheral Interface Bus.
- Both of these are common synchronous serial links for connecting to other chips in the systems, such as ADC, DAC and other sensors.
## Comparison of the three serial links

- The Pyboard has 4 UARTs, 2 each of SPI and I2C links.
- Micropython provides library functions to drive these interfaces.
- We will only use UART on this course.

<table>
<thead>
<tr>
<th></th>
<th>Synchronous</th>
<th>Asynchronous</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SPI</td>
<td>I2C</td>
</tr>
<tr>
<td><strong>Max Bit Rate</strong></td>
<td>10Mbit/s</td>
<td>1Mbit/s</td>
</tr>
<tr>
<td><strong>Max devices</strong></td>
<td>Limited by pins</td>
<td>127 devices</td>
</tr>
<tr>
<td><strong>No. of pins</strong></td>
<td>3 + n</td>
<td>2</td>
</tr>
<tr>
<td><strong>Pros</strong></td>
<td>Simple, low cost high speed</td>
<td>Low pin count; allows multiple masters</td>
</tr>
<tr>
<td><strong>Cons</strong></td>
<td>Single master, short distance</td>
<td>Slow, short distance</td>
</tr>
<tr>
<td><strong>Applications</strong></td>
<td>Connect to peripherals on PCB</td>
<td>Share bus connection with peripherals on same PCB</td>
</tr>
</tbody>
</table>
Bluetooth wireless links

- Bluetooth is a cable replacement technology.
  - 1Mb/s, range ~10m, much lower power than wifi and cheaper
- We use **Bluetooth Low Energy**, a version of Bluetooth that uses much less power, but also has much lower data rate (realistically less than 100kbit/s)
- Bluetooth standard defines a protocol stack to enable different types of devices to communicate.

- The Bluetooth stack includes protocols for the radio layer all the way up to device discovery, service discovery, etc.
A video on Bluetooth Low Energy (5 min)

Classic Bluetooth protocol
Bluetooth high speed protocol
Bluetooth Low Energy protocol
What is RFID?

- Radio Frequency Identification (RFID) is a technology that employs a microchip with an antenna that broadcasts its unique identifier and location to receivers.
- It employs a microchip called a smart tag, broadcasts unique 96-bit identifier to receiver.
- The receiver relays the data to a computer.
- A RFID Tag contains two main parts:
  - Silicon chips
  - Antennas
- These components enable tags to receive and respond to radio frequencies queries from RFID transceivers.
- Two types of RFIDs:
  - Passive
  - Active
RFID Tags types

◆ Passive
  • Have no internal power supply
  • Electrical current induced in antenna by the incoming signal provides power for integrated circuit in tag to power up and transmit response
  • Very Small, Limited Range, Unlimited Life

◆ Active
  • Have their own internal power source
  • Many operate at fixed intervals
  • Also called beacons (broadcast own signal)
  • Large (coin), Much larger memories, Longer range

---

**Four main frequencies:**

<table>
<thead>
<tr>
<th></th>
<th>Frequency</th>
<th>Distance</th>
<th>Example Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF</td>
<td>125khz</td>
<td>Few cm</td>
<td>Auto-Immobilizer</td>
</tr>
<tr>
<td>HF</td>
<td>13.56Mhz</td>
<td>1m</td>
<td>Building Access</td>
</tr>
<tr>
<td>UHF</td>
<td>900Mhz</td>
<td>~7m</td>
<td>Supply Chain</td>
</tr>
<tr>
<td>μwave</td>
<td>2.4Ghz</td>
<td>10m</td>
<td>Traffic Toll</td>
</tr>
</tbody>
</table>
A video about RFID (4 min)