Daniel **Fitz**

43961229

University of Queensland

CSSE3010 – Embedded System Design

Lecture Summary

Table of Contents

1 Analog Interfacing **2**
1.1 Accuracy, Precision, Resolution 2
Accuracy 2
Precision 2
Measurement resolution 2
1.2 Sampling 2
Time Quantization 2
Amplitude Quantisation 2
1.3 Sampling Theorem 2
Nyquist Theorem 2
Aperture time 2
Signal to Noise Ratio (SNR) 2
Sample and Hold 3
Resolution and Dynamic Range 3
1.4 Conclusions 4

2 Timing Interfacing **4**
2.1 Waveform Basics 4
Waveform Time Spacing Measurement 4
Waveform Frequency/Period Measurement 4
2.2 Pulse Width Modulation (PWM) 4
PWM precision/resolution 4

3 Timer **5**

4 ADC **5**

5 Embedded Design Methodology **5**
5.1 Top Down Design 5
Valvano 6
5.2 High Level Design Overview 6
System Flow – Signal/Data Flow 6
5.3 Program Flow 7

Analog Interfacing

Accuracy, Precision, Resolution

Accuracy

Proximity of measurement results to the true value

Precision

Repeatability or reproducibility of the measurement

Measurement resolution

The smallest change in the underlying physical quantity that produces a response in the measurement

Sampling

Time Quantization

Signal value read/available only in specific times (usually at the same interval). This can cause aliasing – frequency ambiguity of signal components

Amplitude Quantisation

Amplitude of each sample can only take one of a finite number of different values. This adds **quantisation noise**: an irreversible corruption of the signal

Sampling Theorem

Nyquist Theorem

A signal having no spectral components above f_m Hz can be determined uniquely by values sampled at the rate:

$$f_s > 2f_m$$

$f_s > 2f_m$ is called the Nyquist rate

Aperture time

The time during which ADC is continuously converting the varying analog input

$$\text{Max slope} = \frac{\Delta V}{\Delta t} = \omega \times V_{peak} = 2\pi f V_{peak}$$

Example

$$\begin{aligned}
\frac{\Delta V}{\Delta t} &= 2\pi f V_{peak} && (f = (\Delta V/\Delta t)(1/2\pi V_{peak})) \\
\Delta V &= \frac{1}{4} LSB = \frac{1}{4}\left(\frac{10V}{4096}\right) = 0.6mV \\
\Delta t &= 10us \\
f &= \left(\frac{\Delta V}{\Delta t}\right)\left(\frac{1}{2\pi V_{peak}}\right) = \left(\frac{0.6mV}{10us}\right)\left(\frac{1}{2\pi 5V}\right) = 2Hz
\end{aligned}$$

Signal to Noise Ratio (SNR)

- Ratio of the maximum sine wave level to the noise level
- Maximum sine wave has an amplitude of $\pm 2^{n-1}$ which equals an RMS value of:

$$0.71 \times 2^{n-1} = 0.35 \times 2^n$$

- SNR is:

$$20\log_{10}\left(\frac{0.35 \times 2^n}{0.3}\right) = 20\log_{10}(1.2 \times 2^n) = 1.8 + 6n\, dB$$

Sample and Hold

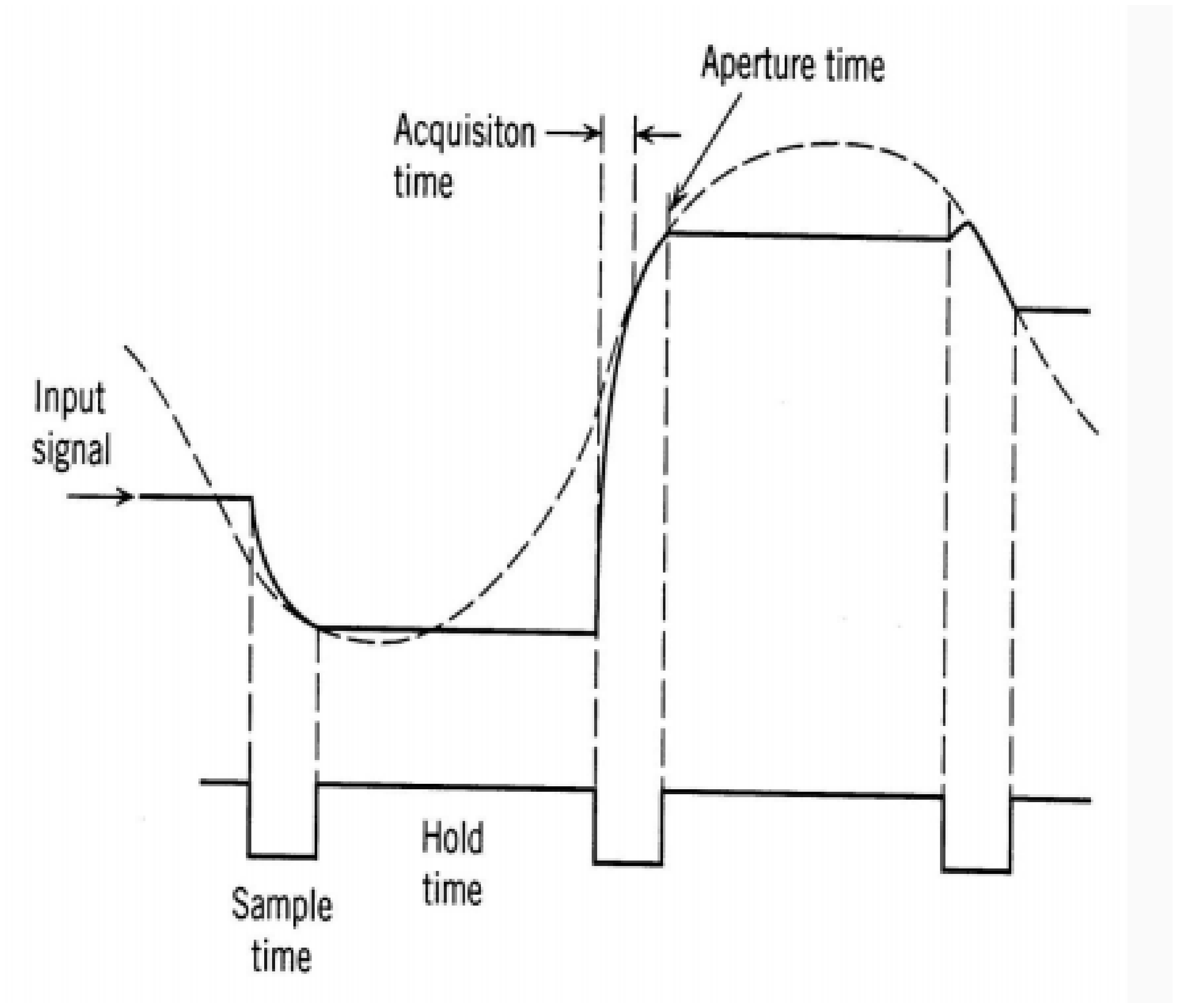

Figure 1: Sample and Hold

Resolution and Dynamic Range

Number of Binary Bits (n)	Full-Scale Decimal Value $(2^n - 1)$	LSB Weight % of Full-Scale Range	LSB Voltage for 1-V Full-Scale Range	Quantization Error Percent of Full-Scale Range	Dynamic Range (From LSB to Full Scale) (dB)
4	15	6.25	60 mV	3.12	24.08
6	63	1.56	16 mV	0.78	36.12
8	255	0.3906	3.9 mV	0.195	48.16
10	1023	0.0977	0.98 mV	0.0488	60.21
12	4095	0.0244	0.24 mV	0.0122	72.25
14	16383	0.00610	61 uV	0.00305	84.29
16	65535	0.00153	15 uV	0.00075	96.33
18	262143	0.000382	4 uV	0.0002	108.37
20	1048575	0.0000954	1 uV	0.00005	120.41

Conclusions

- Interface to analogue world requires thorough understanding and analysis of physical properties this is why it is difficult
- The A/D D/A on-chip converters on microcontrollers are average precision and would require off chip hardware to make conversions more accurate or faster
- Always start interfacing with analysis of the properties and requirements of the analogue side. Digital is always faster

Timing Interfacing

- Use of timing bistate (high or low) waveforms or 'square wave' for interfacing
- A timing waveform can 'mimic' an analog voltage
  - Note Analog voltages an be approximated with specific square waves frequencies and duty cycle
- Commonly used for Pulse Width Modulation, Waveform Frequency or Time Spacing Interfaces
- Timing Interfacing consists of three parameters:
  - Period
  - Frequency
  - Duty Cycle

Waveform Basics

- Period (s) $= T_{high} + T_{low} = T_{period}$
- Frequency (Hz) $= \frac{1}{T_{period}}$
- Duty Cycle (%) $= \frac{100 \times T_{high}}{T_{high} + T_{low}} = 100 \times \frac{T_{high}}{T_{period}}$

Waveform Time Spacing Measurement

- Time spacing of a waveform used to convey information
- Useful for 'irregular' waveforms (high low times are not the same)
  - E.g. time spacing between pulses
- Implemented using Timer Input Capture interrupts
  - A timer counter value is recorded, each time a transition (rising or falling) occurs on the input line ### Frequency Measurement
- The frequency of a waveform can also convey information
- Useful for 'regular' waveforms (High and low times are the same)
- Typically used for optical or mechanical based systems

Example: Wheel Encoder The wheel encoder works by shining light through a pin wheel and detecting the frequency of the light passing through. The frequency of the light passing through is proportional to the speed of the wheel

Waveform Frequency/Period Measurement

- Implemented using Timer Input Capture interrupts
- Can measure using period/frequency by timing transitions.
  Disadvantage: Must rely on accurate timer with enough resolution/precision (e.g. 1ns resolution)
- The number of transitions or zero crossings within a time window, is proportional to the waveform frequency/period.
  Advantage: Does not need high resolution.
  Disadvantage: Only works for regular waveforms

Pulse Width Modulation (PWM)

- Pulse Width Modulation (PWM) uses duty cycle to convey information
- PWM can be used approximate analog (multi-value) waveforms
- Used for controlling mechanical systems such as motors and servo motors

PWM precision/resolution

$$period = N \times \Delta$$

$\Delta =$ resolution
$N =$ PWM precision

Example:

$$Period\,20ms, \Delta = 20us \rightarrow N = 1000 \rightarrow 10bits$$

Timer

Timers features:

- Update interrupts – cause an update interrupt (periodic or not)
- PWM – pulse width modulation (used for controlling servos)
- Timer Input Capture interrupts – cause an interrupt, when a rising or falling edge is detected on an input signal – captures value of timer
- Timer Output Compare – toggle an output pin high or low, when a compare value matches the timer value

ADC

- 3 ADCs: ADC1 (master), ADC2 and ADC3 (slaves)
- Maximum frequency of the ADC analog clock is 36MHz
- 12-bits, 10-bits, 8-bits or 6-bits configurable resolution
- ADC conversion rate with 12 bit resolution is up to:
  - 2.4 M.samples/s in single ADC mode
  - 4.5 M.samples/s in dual interleaved ADC mode
  - 7.2 M.samples/s in triple interleaved ADC mode
- Conversion range: 0 to 3.6 V
- ADC supply requirement: VDDA = 2.4V to 3.6V at full speed and down to 1.65V at lower speed
- 3 ADC1 internal channels connected to:
  - Temperature sensor
  - Internal voltage reference: Vrefint (1.2V typ)
- External trigger option for both regular and injected conversion
- Single and continuous conversion modes
- Scan mode for automatic conversion of channel 0 to channel ‘n’
- Left or right data alignment with in-built data coherency
- Channel by channel programmable sampling time
- Discontinuous mode
- Dual/Triple mode (with ADC1 and ADC2 or all 3 ADCs)
- DMA capability
- Analog Watchdog on high and low thresholds
- Interrupt generation on:
  - End of Conversion
  - End of Injected conversion
  - Analog watchdog
  - Overrun

Embedded Design Methodology

Top Down Design

- Embedded System design methodology
  - A complex system is created to meet specific design attributes
- Top down is a process in which a complex design is first organised as a top or high level view
  - The high level overview of the design is divided into sub-components
  - Each sub-componenet is a distinct section of the top level design

        * The sub-components can be further broken down into elements

Valvano

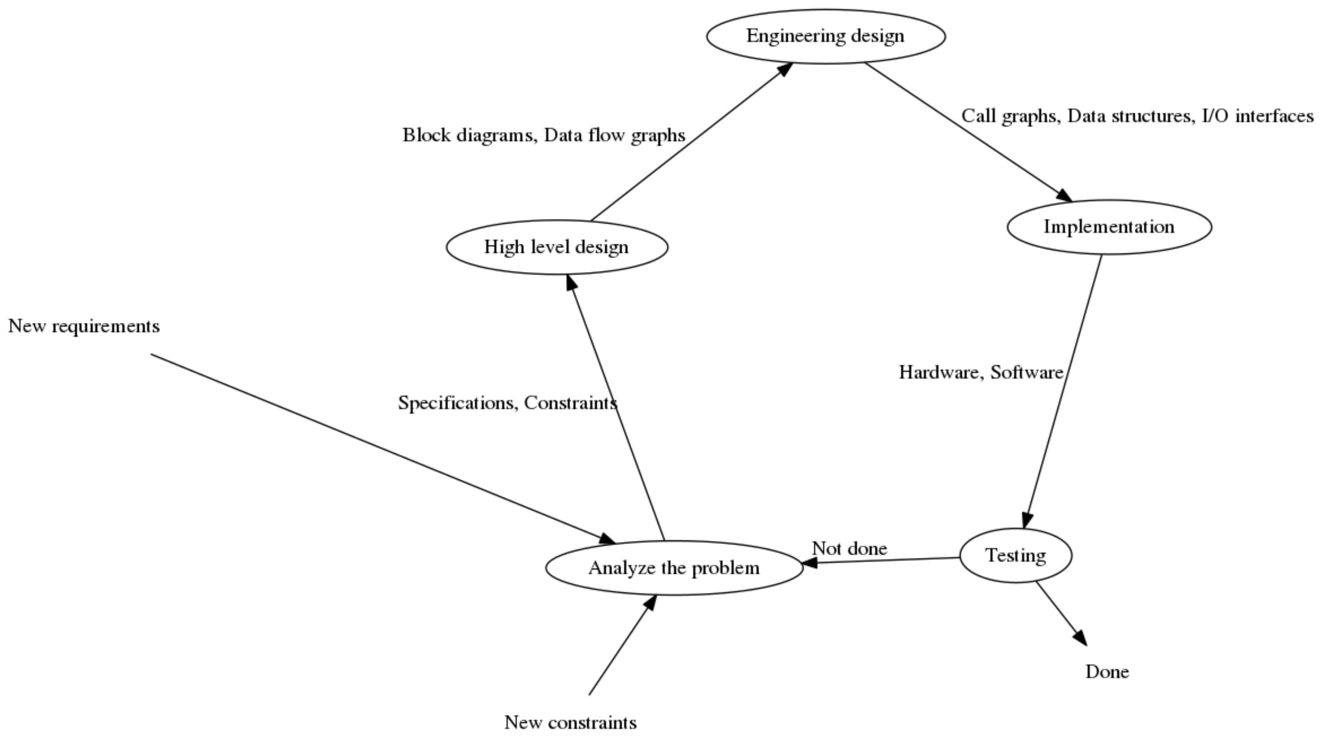

Figure 2: Top Down Valvano

High Level Design Overview

Consists of a number of concepts:

- System Flow:
  - Schematic – shows system inputs and outputs connections
  - Signal/Data Flow diagram
    - Shows the connections of the inputs, all the way to the outputs
      - Shows each stage of connecting the input to the output
- Program Flow:
  - State Diagrams
    - Embedded System Programming main loop can be abstracted as a State Controller
      - A microcontroller program must enter and exit certain states, as it executes
  - Flow Charts
    - Software abstraction of microcontroller program

System Flow – Signal/Data Flow

- Signal/Data Flow diagram
  - Shows the connections of the inputs, all the way to the outputs
- Shows each stage of connecting the input to the output
- Differs to block diagram – is not an overview of the system
- Useful for identifying which software/hardware modules to use
- Useful for debugging and identifying:
  - Break points – where your code will definitely break
  - Weak points – where your code could potentially break
  - Bottle necks – where your system's performance is limited – i.e. 'slow' to respond

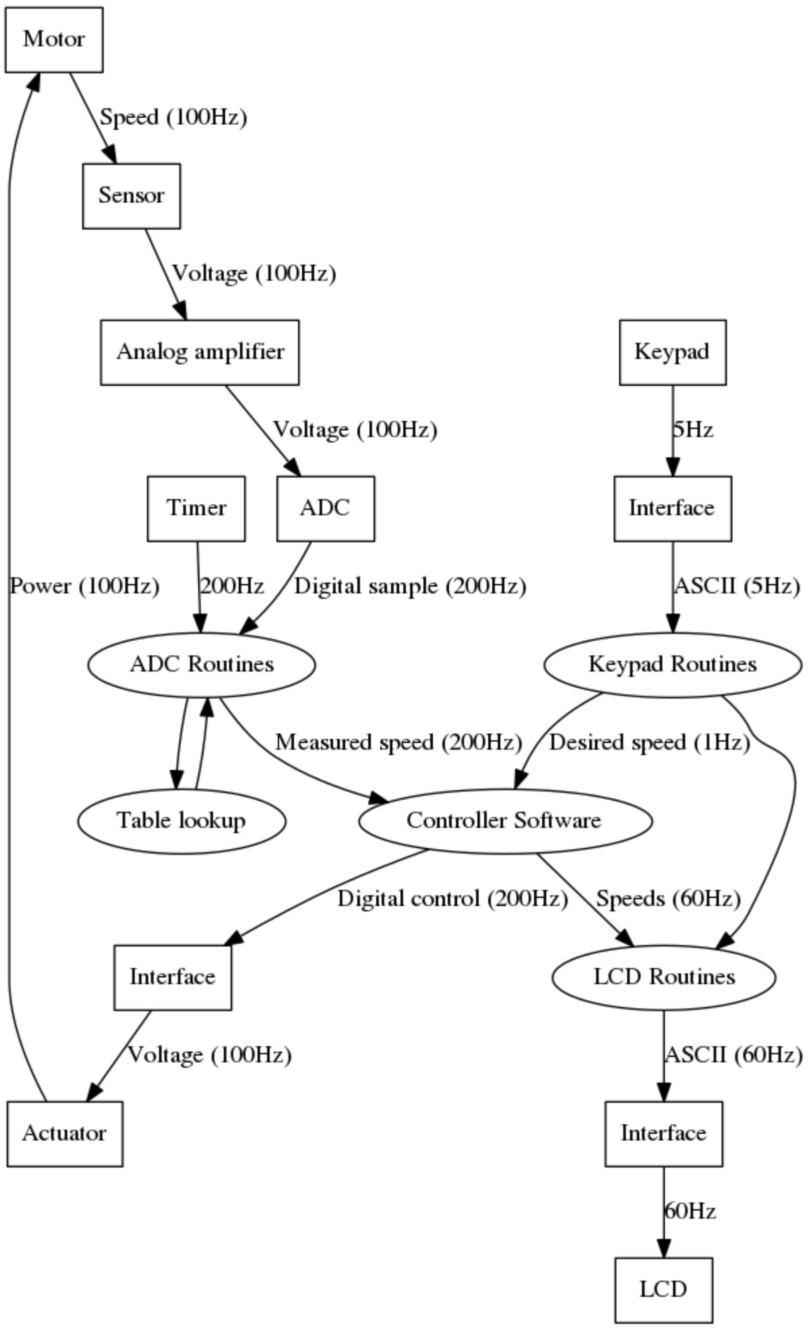

Figure 3: Motor controller Signal/Data Flow diagram

Program Flow

Your program flow should consist of:

- Main loop
- Hardware Initialisation function
- Functions – callable block of code
- Subroutine (not a function but a unit of code)
- Interrupt Service Routines

Program flow is described as:

- State Diagrams
  - main loop

- Flow Charts
  - subroutines

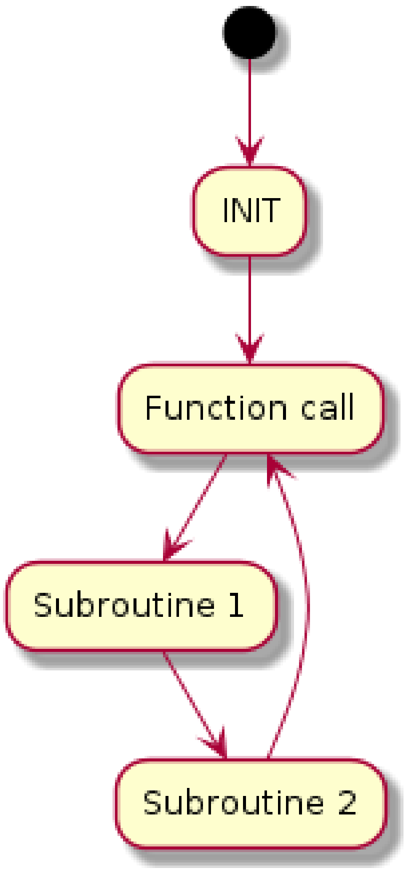

Figure 4: Program Flow – Outline

ISRs have a lightning symbol