Process State, Sensors, and Interfacing

Function of sensors: convert physical quantity to information.

Information will (usually) be read by computer via an interface.

Ultimately, the information, in the desired form, will be stored in a memory location.

The following steps are typical:

1: A *transducer* converts *process state* to a raw electrical quantity.

2: A *conditioning circuit* converts the raw electrical quantity into a useful electrical quantity.

3: An *analog-to-digital converter (ADC)* converts the useful electrical quantity to information.

4: A *buffer* and *interface* store, format, and present the information to a computer.

5: An *interface routine* reads the information, converts it to the desired form, and stores it in the desired place.

Process State and Process Variables

The *process state* is the current condition of the process, down to infinitesimal detail.

The *process variable* is a part or characterization of a process state, usually in terms of a common measure.

For example, consider a coffee maker.

<u>Process state:</u> amount of water in carafe, water temperature, chemical description of water in carafe, type of coffee beans, etc.

<u>Process variable:</u> temperature of water.

<u>Process variable value:</u> $70\,^{\circ}\mathrm{C}$.

Characteristics

It is impossible to know the complete process state (because of infinite detail).

It is impossible to know the exact value of a process variable.

A process variable value, however, can be determined to a high degree of precision.

Dimensions

Basics

A process variable's value is usually expressed as the product of a number and a dimension.

For example, let process variable T be the temperature of water in a coffee maker carafe.

Then a value for T might be $60\,^{\circ}\mathrm{C}$.

An equivalent value might be $T = 333.15\,\mathrm{K}$.

Notation

Dimensions will be written in Roman (upright) type. For example, mA, V, and m.

Symbols representing values (variables) will be written in italic type: T, x, and R.

Thus, $3V\,\mathrm{V}$ means "three vee volts."

Algebraic Manipulation of Dimensions

In expressions, dimensions are manipulated in the same way as numbers and variables.

For example:

$$\frac{3\,\mathrm{km}}{5\,\mathrm{MPH}} = \frac{3\,\mathrm{km\,hr}}{5\,\mathrm{Mi}} = \frac{3\,\mathrm{km\,hr}}{5\,\mathrm{Mi}}\frac{\mathrm{Mi}}{1.6\,\mathrm{km}} = \frac{3}{8}\,\mathrm{hr}.$$

Graphs of Values

Axes will be labeled with a symbol divided by a dimension.

For example, $x/\,\mathrm{V}$ or $R/\,\mathrm{k\Omega}$.

The numbers on the axis are then dimensionless.

Transducers

Transducer:
Device which converts a physical quantity from one form to another.

Usually from...
...a physical quantity which is a process variable to...
...some useful electrical quantity.
For example, a transducer might convert temperature to resistance.

Transducer Modeling

Mapping (function) from process variable to electrical quantity.

Symbol H_{t} denotes the function.

Let x be a process variable.

Then $H_{\mathrm{t}}(x)$ is the output...
...of the transducer with function H_{t}.

Example

A variable resistor can be used as a transducer.

Consider a variable resistor which consists of a slider which can move over a distance of 15 mm while resistance varies linearly from 0 to 10 kΩ.

Process variable: position of slider, x.

Mapping: $H_{\mathrm{t}}(x) = x\dfrac{10\,\mathrm{k\Omega}}{15\,\mathrm{mm}}$.

Process variable value approximated from transducer output.

Let $y = H_{\mathrm{t}}(x)$ where H_{t} and x are as above.

Quantity y is a resistance.

The position x is found by inverting H_{t}:

$$H_{\mathrm{t}}^{-1}(y) = y\frac{15\,\mathrm{mm}}{10\,\mathrm{k\Omega}}$$

(In this case the process variable is not approximated.)

The process of finding the inverse is equivalent to solving for x in the equation $y = x\dfrac{10\,\mathrm{k\Omega}}{15\,\mathrm{mm}}$.

Conditioning Circuits

Purpose

The output of a transducer is a raw electrical quantity.

It might have to amplified or otherwise processed.

This is done by conditioning circuits.

Conditioning circuits might have to do one or more of the following:

- Amplify a tiny voltage.
- Convert resistance to voltage.
- Detect tiny changes in resistance (*e.g.*, 100.1 to 100.2 Ω).
- Add an offset to the transducer output.
- Correct for nonlinearities in the transducer function.
- Other functions.

Notation

The symbol H_{c} will be used for the conditioning circuit's function.

An amplifier is a simple conditioning circuit: $H_{\mathrm{c}}(v) = Av$, where A, the gain, is a dimensionless number.

For example, if x is a process variable, then $H_{\mathrm{t}}(x)$ is the transducer output and $H_{\mathrm{c}}(H_{\mathrm{t}}(x))$ is the conditioning-circuit output.

Sensors

The combination of transducer and conditioning circuit is referred to as a *sensor*.

Analog to Digital Conversion

Conditioning-circuit output is usually fed to an *ADC*.

An analog-to-digital converter converts electrical quantities to information quantities.

Input is usually a voltage, output is usually a binary number.

Symbol $H_{\mathrm{ADC}}(v)$ will be used for an ADC function.

Standard ADC Function

Since most ADCs will convert voltage to integers a standard function will be used.:

$$H_{\mathrm{ADC}(h,b)}(v) = \left\lfloor \frac{v}{h}(2^b - 1) \right\rfloor \bmod 2^b,$$

where h is a voltage and b is an integer.

This ADC would convert voltages in the range 0 to h (inclusive) to a binary number from 0 to $2^b - 1$.

For example, $H_{\mathrm{ADC}(10\,\mathrm{V},8)}(5\,\mathrm{V}) = 127$
and $H_{\mathrm{ADC}(17\,\mathrm{V},16)}(1.3\,\mathrm{V}) = 5011$.

Sampling, Buffering, and Interfacing

Buffering is the short-term storing information.

Two reasons for buffering the value of a process variable:

The value of the variable *at a particular time* is needed. (The value is buffered at that time.)

The value of a variable is only valid at certain times. (The value is buffered when it is valid.)

The buffer itself can be a simple flip-flop, a register, a RAM, etc.

Usually, the contents of the buffer will be read by a computer through an *interface*.

The interface presents the buffered data to the computer in some standard form.

The computer is running some RT program. The RT program has one or more *interface routines*.

The interface could tell the RT program that data is available by making an *interrupt request*.

...*or*...

An interface routine could read the buffer without being alerted by an external signal.

For example, it might read the buffer every millisecond.

Sampling is the process of reading a process variable at regular intervals.

Interface Routine

Here we will consider the ultimate destination of a process-variable value to be a memory location.

The following is done to transfer a value from the interface to the memory location:

1: The interface routine makes a system call to read the interface.

```
raw=readInterface();
```

2: The interface routine, or some other code, applies a function, H_{f} to the value read.

3: The result is written into the memory location.

```
theMemoryLocation=HsubF(raw);
```

The function H_{f} puts the value into the final form. It may perform one or more of the following operations:

- Convert the raw value to a floating-point quantity. (ADC output is usually an integer.)
- Correct for any nonlinearities in the transducer or conversion circuit.
- Convert the quantity to the desired dimensions. (*E.g.*, meters, microns.)

In terms of the process variable, the final value written is:

$$H_{\mathrm{f}}(H_{\mathrm{ADC}}(H_{\mathrm{c}}(H_{\mathrm{t}}(x))))$$

Once written, the value is read by the parts of the RT system that figure out what's going on.

The Conditioning Problem

Archetypical Problem:

Design a system to write variable `procVar` *with* $H(x)$, *the value of* ... *in* ..., *where process variable* x *is* ..., x *can take values in the range* $[x_{\mathrm{min}}, x_{\mathrm{max}}]$ *and where* $H(x) = \ldots$.

Example Problem:

Design a system to write variable `waterLevel` *with* $H(x)$, *an integer giving the water level in meters, where process variable* x *is the water level in room 2147 CEBA,* x *can take on values in the range* $[0\,\mathrm{m}, 1\,\mathrm{m}]$ *and where* $H(x) = \frac{5x}{\mathrm{m}}$.

Solution Overview

1: Choose a transducer.
A variable resistor connected to a float with cables.

2: Choose an ADC.
Suppose an ADC with function $H_{\mathrm{ADC}(5\,\mathrm{V},8)}$ *is available.*

3: Design a conversion circuit.
This will convert resistance to voltage.

4: Design the buffer and interface.
Details for this part will be skipped here.

5: Write the function for computing the final value.
Easy, but tedious.

The Transducer

Use a variable resistor with the function $H_{\mathrm{vr}}(\theta) = \frac{\theta}{6} 100\,\mathrm{k}\Omega$, where $\theta = [0, 6]$ is angle (in radians) of the shaft.

Use floats, guides, and cables to convert water level to shaft rotation.

$H_{\mathrm{t}}(x) = \frac{x}{\mathrm{m}} 100\,\mathrm{k}\Omega$.

The Conversion Circuit

The output of the conversion circuit will be:

$$H_c(H_t(x)) = H_c(\frac{x}{\mathrm{m}} 100\,\mathrm{k\Omega}).$$

For correct operation, the input to the ADC must be in the range of 0 to 5 V.

A variety of conversion circuits could be used.

The simplest is a linear conversion from resistance to voltage.

$$H_c(R) = \frac{R}{100\,\mathrm{k\Omega}} 5\,\mathrm{V}.$$

$$H_c(H_t(x)) = \frac{(x/\,\mathrm{m})100\,\mathrm{k\Omega}}{100\,\mathrm{k\Omega}} 5\,\mathrm{V}.$$

$$H_c(H_t(x)) = (x/\,\mathrm{m})5\,\mathrm{V}.$$

ADC, Buffering, and Final Processing

The ADC function is fixed at

$$H_{\mathrm{ADC}(5\,\mathrm{V},8)}(v) = \left\lfloor \frac{v}{5\,\mathrm{V}}(2^8 - 1) \right\rfloor \bmod 2^8 = \left\lfloor \frac{v}{5\,\mathrm{V}} 255 \right\rfloor \bmod 256.$$

$$H_{\mathrm{ADC}(5\,\mathrm{V},8)}(H_c(H_t(x))) = \left\lfloor \frac{(x/\,\mathrm{m})5\,\mathrm{V}}{5\,\mathrm{V}} 255 \right\rfloor \bmod 256 = \left\lfloor \frac{x}{\mathrm{m}} 255 \right\rfloor \bmod 256.$$

The ADC output is clocked into a buffer and then transfered to the computer through an interface.

Details of these parts will be covered later in the semester.

Finally, the interface routine converts the raw form into the desired form: $H(x) = \frac{5x}{\mathrm{m}}$.

$$H_f(H_{\mathrm{ADC}(5\,\mathrm{V},8)}(H_c(H_t(x)))) = H(x)$$
$$H_f(\lfloor (x/\,\mathrm{m})255 \rfloor \bmod 256) = H(x)$$

Define $y = g(x) = \lfloor (x/\,\mathrm{m})255 \rfloor \bmod 256$.

Then $x = g^{-1}(y) \approx y\frac{\mathrm{m}}{255}$ for $x \in [0\,\mathrm{m}, 1\,\mathrm{m}]$.

Then

$$\begin{aligned} H_f(g(x)) &= H(x) \\ H_f(y) &= H(g^{-1}(y)) \\ &= H(y\frac{\mathrm{m}}{255}) \\ &= \frac{5}{\mathrm{m}} y \frac{\mathrm{m}}{255} \\ &= \frac{y}{51}. \end{aligned}$$

The code fragment in the RT program is then:

```
int raw;

double waterLevel;

raw=readInterface();

waterLevel=raw/51.0;
```

. . . and we're done!