Embedded systems are product sub-systems controlled by a special-purpose computer. The computer provides software-control by using transducers called actuators and sensors.

- **Sensors** are transducers that convert physical energy to electrical energy. In general, sensors ***meter*** the environment by measuring some change in a physical parameter.
- **Actuators** are transducers that convert electrical energy to physical energy. In general, actuators ***change*** the environment by using electricity to change a physical parameter.
- **Microcontrollers** are single-chip computers that are the brain of ***most*** embedded systems.
- **Figure 1** is a basic model used to diagram embedded systems that use microcontrollers.

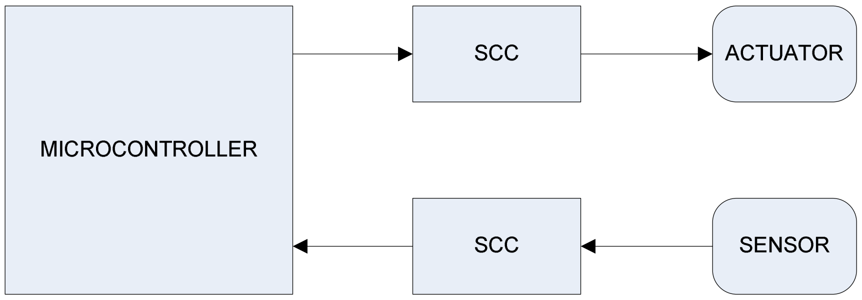

Figure 1: The Basic Embedded System Model

Analog sensors produce analog signals.

- Analog signals are continuous, real-values electrical signals.
- Example analog signals include temperature variation, speech, light variation, atmospheric communications signals, and bioelectric signals such as the ECG.
- Small analog signals may need to be signal conditioned and ranged to larger signals before presentation to the computer for conversion to binary numbers. Signal conditioning circuits are not covered in this embedded systems tech note.

Analog-to-digital converters (ADC) are common on-chip microcontroller I/O devices.

- All ADCs convert the analog voltage to an n-bit binary result.
- Microcontrollers are available with 8-bit, 10-bit, 12-bit, and 16-bit converters. The bitwidth determines the granularity of voltage conversion. Most ADCs have multiple input pins that can be converted one at a time.
- Some ADCs can convert multiple input pins simultaneously.
- Some ADCs provide built-in amplification and conversion of the differential voltage between two input pins.

Analog comparators (AC) are common on-chip microcontroller I/O devices.

- Not every design problem requires an n-bit conversion result for the analog voltage.
- Some problems just want to know if the analog voltage exceeds some other voltage (A > B).
- Analog comparators implement the A > B behavior.

- Analog comparators generally provide a hardware interrupt and a polling flag allowing both device servicing techniques to be used.
- Analog comparators generally provide a dedicated microcontroller pin that sets (1) when the A > B event is true and clears (0) when the A > B event is false.

The **ATmega32 AC I/O device** is a standard analog comparator device.

- The AC has two signals called AN0 and AN1 that share PORTB pins.
- The AC **implements** the A > B equation as AN0 > AN1 in ***normal mode operation***.
- The AC **can implement** the A > B equation as AN0 > (ADMUX selected ADC pin) in a ***special mode of operation***. This "special function" requires the user software to configure a bit called ACME in the special-function I/O register (SFIOR). This bit switches to the special mode. The default value of the bit is normal mode operation and thus, software can ignore this bit if normal mode is used.
- The AC has an output pin called AC0 that sets (1) when AN0 > AN1 in normal mode and clears (0) when AN0 < AN1 in normal mode.
- The AC provides both a polling flag and an interrupt to allow either of the event servicing techniques.

The ATmega32 AC I/O device is controlled through a **single control register** called ACSR in normal mode.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0

- ACD is the disable bit. Set this bit (1) to turn off the ACD.
- ACBG is an advanced function not used in normal mode. Clear this bit (0) for normal mode. Refer to the ATmega32 datasheet for more information about this bit.
- ACO is the current output of the AC0 pin – just in case software wants to know. It writes to the ACSR register 1 or 2 clock cycles after the actual pin voltage change.
- ACI is the AC event completion flag. This bit is set (1) if AN0 > AN1 in normal mode. This bit would be used by polling loops when using the polling service technique.
- ACIE is the AC event interrupt enable bit. Set this bit (1) to request AC hardware interrupts. Don't forget to turn on CPU interrupt processing with an sei instruction!
- ACIC is a bit used to force the time of an AC0 edge change to be captured by the ATmega32 TIMER/COUNTER1 component. This can be useful if you want to measure the time period between AC0 events. Set this bit (1) to force the time capture. Clear the bit (0) if the time capture is not wanted.
- ACIS1 and ACIS0 determine which AC0 pin voltage event causes the interrupt – in other words, the interrupt can be AN0 > AN1, AN0 < AN1, or both! The values are summarized in Table 1

Table 1: The ACI value summary

ACIS1	ACIS0	INTERRUPT
0	0	both edges of AC0
0	1	reserved – do not use
1	0	AC0 falling edge (AN0 < AN1)
1	1	AC0 rising edge (AN0 > AN1)