In this chapter we will do:

- Tactile sensors / switches
- Shaft encoders
- A/D converters
- Infrared position sensitive devices (PSD)
- Digital cameras

General Remarks
- There are millions of different sensor types
- You have to select the right one for your application and budget and read the specs
- Note: some sensors require input from the CPU as well E.g. activation/deactivation, triggering data transfer, etc.
- Our scope here is more on interfacing sensors than on understanding the sensors themselves

### 6.1 Binary Sensors

**Examples**

- Tactile sensor
- Inclinometer
- Gyroscope
- GPS Module
- Digital Camera

**Tactile sensor / switch**

- Easy to be interfaced
- Use a resistor and link to digital input of CPU or latch

```
VCC
<table>
<thead>
<tr>
<th>input signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>R (e.g. 5kΩ)</td>
</tr>
<tr>
<td>GND</td>
</tr>
</tbody>
</table>
```

“active low”
Binary Sensors

Problem:
- Switch-bounce
  (e.g. when counting switches)

Solution:
- Wait certain time delay

6.2 Encoder

Shaft encoder
- incremental or absolute position

Encoder

Incremental encoder
- Usually requires 2 sensors to determine speed and direction
- See motor control

Technology
- Magnet + hall sensors (incremental)
- Optical sensors with black/white segments (incremental)
Encoder

- Encoder signal (2 lines) are connected to microcontroller like 2 binary sensors (digital input lines)
- Microcontrollers usually have special internal registers for pulse counting
  ⇒ This is done in parallel to normal calculations
  ⇒ Does not slow down the CPU

6.3 Analog Sensors

A number of sensors have analog output signal rather than digital signals
  ⇒ A/D converter is required to connect to CPU

Examples:
- Microphone
- Analog infrared distance sensor
- Analog compass
- Barometer sensor

6.4 A/D Converter

- Signal has to be provided at correct level, e.g. between 0 .. 5V
- If multiple channels are read: low internal resistance of signal line is important
- A/D converter translates analog voltage level into digital value
- Digital output from A/D converter can be
  - parallel
    (e.g. 8 bit, direct connection to data bus)
  - serially digital
    (provide programmed clock signal to converter to read data bit by bit)
A/D Converter

**Version 2**

- **CPU**
- **microphone**
- **serial clock**
- **CS / enable**
- **GND**
- **1 bit data to dig. input**

MAX192

**General Description**

The MAX192 is a low-cost, 10-bit data-acquisition system that contains an 8-channel multiplexed, high-accuracy track-and-hold, and serial interfaces with high conversion speed and low power consumption. The device operates with a single 5V supply. The analog inputs are software-configurable for single-ended and differential (unipolar/bipolar) operation.

The 8-bit serial interface connects directly to SPI, QSPI™, and MicroWire™ devices, without using external logic. A serial slave input allows direct connection to TI's MSP430 family digital signal processors. The MAX192 uses either the internal clock or an external serial interface clock to perform successive-approximation A/D conversions. The serial interface can operate beyond limits when the internal clock is used. The MAX192 has an internal oscillator reference with a shift of ±1ppm typical. A reference buffer amplifier provides gain trim and two-level SHDN (shutdown) function.

The MAX192 provides a hardwired SHDN pin and two software-selectable power-down modes. Accessing the serial interface automatically powers up the device, and the quick turn-on time allows the MAX192 to be shut down between conversions. By powering down between conversions, supply current can be cut to under 1µA at reduced sampling rates.

**Features**

- 8-Channel Single-Ended or 4-Channel Differential Inputs
- Single ±5V Operation
- Low Power: 1.5mA (operating)
- Internal Track/Hold, 135kHz Sampling Rate
- Internal 4.096V Reference
- 4-Wire Serial Interface is Compatible with SPI, QSPI, MicroWire, and TrueSPI™
- 20-Pin DIP, SO, SSOP Packages
- Pin-Compatible 12-Bit Upgrade (MAX198/MAX198A)

**Ordering Information**

<table>
<thead>
<tr>
<th>PART</th>
<th>TEMP RANGE</th>
<th>Pin Package</th>
<th>NAME/ORDER</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX192-10</td>
<td>0°C to +70°C</td>
<td>20-lead DIP</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-20</td>
<td>0°C to +70°C</td>
<td>20-lead SO</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-20PW</td>
<td>0°C to +70°C</td>
<td>20-lead SO-5</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-33</td>
<td>0°C to +70°C</td>
<td>20-lead SO</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-33PW</td>
<td>0°C to +70°C</td>
<td>20-lead SO-5</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-68</td>
<td>0°C to +70°C</td>
<td>20-lead SO</td>
<td>-1</td>
</tr>
<tr>
<td>MAX192-68PW</td>
<td>0°C to +70°C</td>
<td>20-lead SO-5</td>
<td>-1</td>
</tr>
</tbody>
</table>

**Absolute Maximum Ratings**

<table>
<thead>
<tr>
<th>Voltage (V)</th>
<th>±0.3V to ±3V</th>
<th>±0.3V to ±3.3V</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS, I/O, AGND</td>
<td>135°C to +125°C</td>
<td>135°C to +140°C</td>
</tr>
<tr>
<td>VDD, VSS, GND</td>
<td>0.3V to ±3.3V</td>
<td>0.3V to ±3.3V</td>
</tr>
<tr>
<td>Input</td>
<td>±1.5V</td>
<td>±1.5V</td>
</tr>
<tr>
<td>Differential Input</td>
<td>±1.5V</td>
<td>±1.5V</td>
</tr>
<tr>
<td>Digital Output to GND</td>
<td>±1.5V</td>
<td>±1.5V</td>
</tr>
<tr>
<td>Operating Temperature Range</td>
<td>-40°C to +125°C</td>
<td>-40°C to +125°C</td>
</tr>
<tr>
<td>Storage Temperature Range</td>
<td>-60°C to +125°C</td>
<td>-60°C to +125°C</td>
</tr>
<tr>
<td></td>
<td>70°C to 110°C</td>
<td>70°C to 110°C</td>
</tr>
</tbody>
</table>

These devices are intended for use in Automotive/Marine/ITS Applications. They are specified under MIL-STD-883C, Method 1010C, Condition A1, and MIL-STD-883E, Method 1005, Condition A1. For detailed information on packaging, see the manufacturer's data sheet.
A/D Converter

Power consumption

<table>
<thead>
<tr>
<th>Current (mA)</th>
<th>Power (W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0.5</td>
</tr>
<tr>
<td>50</td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>0.05</td>
</tr>
<tr>
<td>1</td>
<td>0.005</td>
</tr>
</tbody>
</table>

Power-up delay

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>Delay (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.001</td>
<td>10</td>
</tr>
<tr>
<td>0.01</td>
<td>100</td>
</tr>
<tr>
<td>0.1</td>
<td>1000</td>
</tr>
<tr>
<td>1</td>
<td>10000</td>
</tr>
</tbody>
</table>

Digital Sensors

Example: Sharp GP2D02
- Available as digital or analog version (GP2D05)
- Versatile optical distance measurement sensor (requires reflective surfaces)
- Uses infrared LED and light detector
- Often called PSD (position sensitive device)
- Measurement range 6cm - 80cm
- Accuracy: about 1cm

Digital Sensors

6.5 Digital Sensors

Digital sensors are
- usually more complex than analog sensors
- often more accurate than analog sensors
- sometimes analog sensors with built-in A/D converters

Digital Sensors

Example: Sharp GP2D02
- Available as digital or analog version (GP2D05)
- Versatile optical distance measurement sensor (requires reflective surfaces)
- Uses infrared LED and light detector
- Often called PSD (position sensitive device)
- Measurement range 6cm - 80cm
- Accuracy: about 1cm

Digital Sensors

Physical dimensions

<table>
<thead>
<tr>
<th>Dimension (mm)</th>
</tr>
</thead>
<tbody>
<tr>
<td>27.5</td>
</tr>
<tr>
<td>21.5</td>
</tr>
<tr>
<td>11.5</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>

Block diagram

- Signal processing circuit
- Light detector (PSD)
- IR LED
- Block diagram
- Reflection object
Digital Sensors

Signal vs. true distance
Can you see a problem?

How do you interface this sensor to a controller?
• Hardware
• Software

6.6 PWM Sensors

We have already seen PWM for:
• Velocity control for DC motors
• Position specification for servos

Now we see PWM for:
• Sensor data
• Examples:
  – Accelerometer
  – Gyroscope
  – Inclinometer
PWM Sensors

<table>
<thead>
<tr>
<th>Gyroscope (relative)</th>
<th>Inclinometer (absolute)</th>
</tr>
</thead>
<tbody>
<tr>
<td>neg. accelerat.</td>
<td>~30 degrees</td>
</tr>
<tr>
<td>no acceleration</td>
<td>0 degrees</td>
</tr>
<tr>
<td>pos. accelerat.</td>
<td>+30 degrees</td>
</tr>
</tbody>
</table>

6.7 Sensor Calibration

Sensor output may be linear or non-linear to measured dimension:
- Before a (digitized) sensor value can be used in an embedded system, its data has to be calibrated
- Linear relationship: a few initial measurements are sufficient
- Non-linear measurements: complete measurement table is required. During runtime: usage for formula vs lookup table

Sensor Calibration

A small error during calibration can render a correctly working sensor useless!

For non-linear relationships between real dimension and sensor output:
- Use formula only if simple (consider additional computation time)
- Simpler method is using a lookup table for every possible sensor output value (if not too many different values)
- Interpolation can be used for lookup table (either for setup or during runtime)
6.8 Digital Cameras

“In 1995 cameras for embedded systems were unthinkable. That’s why I developed one!”

Today they are commonplace:
• Digital cameras
• PDAs with cameras
• Gameboy with camera
• Digital surveillance cameras with on-board processing

Digital Cameras

Technology
• CCD (charge coupled devices)
• CMOS (complementary metal oxide semiconductor)

Resolution
• 60×80 black/white up to
• several Mega-Pixels in 32bit color
• However: Embedded system has to have computing power to deal with this large amount of data!

Digital Cameras

Interfacing to CPU
• Completely depends on sensor chip specs
• Many sensors provide several different interfacing protocols
⇒ versatile in hardware design
⇒ software gets very complicated
• Typically: 8 bit parallel (or 4, 16, serial)
• Numerous control signals required

Interfacing to CPU
• Digital camera sensors are very complex units. In many respects they are themselves similar to an embedded controller chip.
• Some sensors buffer camera data and allow slow reading via handshake (ideal for slow microprocessors)
• Most sensors send full image as a stream after start signal (CPU must be fast enough to read or use hardware buffer or DMA)
• We will not go into further details in this course. However, we consider camera access routines
Digital Cameras

OV6620 SINGLE-CHIP CMOS CIF COLOR DIGITAL CAMERA
OV16120 SINGLE-CHIP CMOS CIF & B/W DIGITAL CAMERA

Features
- 161,792 pixels, 14" lens, 0.67"/0.7" format
- Progressive scan read out
- Data format - YC/CbCr 4:2:2, 4:2:0, 4:1:1, 4:1:2, 4:1:4, 4:2:2, 4:2:0

General Description
The OV6620 (color) and OV16120 (black and white) CMOS image sensors are single-chip, high-speed imaging camera devices designed to provide a high level of functionality in a single, small-footprint package. Both devices incorporate a 32x240 image array capable of capturing up to 60 frames per second. They offer optional image capture and proprietary color technology utilizing advanced algorithms to correct for low light levels. The OV6620 also includes an internal memory to store data, which reduces the need for external components. Applications include digital cameras, camcorders, and video conferencing systems.

Figure 1. OV6620/OV16120 CMOS Image Sensor Block Diagram

Figure 2. Zoom Video Port Timing

Array Elements (CE)
- 20H x 17V (342 x 294)

Pixel Size
- 2.8 x 2.8 μm

Max Frames/sec
- 30 fps (600 frames per second)

Electronic Exposure
- 1/300 s, 1/1000 s

Scan Mode
- Progressive, interlaced

Camera Correction
- 0 ± 20 μm

Min. Illumination (0.1 lux)
- OV6620 = 3 lux, TTY0 = 0.1 lux

S/N Ratio (Digital Camera Out)
- 48 dB (48 dB; 48 dB)

FPF
- < 0.05% max

Dark Current
- < 0.02 A/mm²

Power Supply
- 5 VDC (±5%)

Power Requirements
- < 30 mA (max)

Package
- 64-pin SOIC
Problem
- Every pixel from the camera causes an interrupt
- Interrupt service routines take long, since they need to store register contents on the stack
- Everything is slowed down

Solution
- Use RAM buffer for image and read full image with single interrupt

Idea
- Use FIFO as image data buffer
- FIFO is similar to dual-ported RAM, required since there is no synchronization between camera and CPU
- When FIFO is half full, interrupt is generated
- Interrupt service routine then reads FIFO until empty (Assume delay is small enough to avoid FIFO overrun)
6.9 Image Data Formats

Grayscale Camera Chip: 160x120 Pixels

1 Byte per pixel

Color Camera Chip: 160x120 “Pixels”

1 Byte per “square”
4 squares are 1 pixel

“Bayer Pattern”
green, red, green, red, …
blue, green, blue, green, …

Same chip as grayscale version,
just a layer of color added!
**Image Data Formats**

Color Camera Chip: 160x120 “Pixels”

What is this?

- Color Camera Chip: 160x120 “Pixels”

```
5 200 10 196 248 248 
8  4  0  7 179 249 
7 192 11 247 248 248 
9  3  2 12 247 250 
244 245 243 250 251 239 
241 243 242 238 246 239 
241 249 252 237 193 199 
241 284 235 224 201 204 
```

**Image Data Formats**

```
"De-Mosaic"
  green, red, green, red, …
  blue, green, blue, green, …

Double the resolution in x and y
```
Image Data Formats

Bayer Pattern

- Output format of most digital cameras
- Note: 2×2 pattern is not spatially located in a single point!
- Can be simply converted to RGB (drop one green byte) 160×120 Bayer → 80×60 RGB
- Can be better converted using “demosaicing” technique 160×120 Bayer → 160×120 RGB

typedef BYTE image[imagerows][imagecolumns];
typedef BYTE colimage[imagerows][imagecolumns][3];

int CAMGetColFrame ( colimage *buf, int convert);
Input: (buf) a pointer to a color image (convert) flag if image should be reduced to 4 bit on the fly
0 = get 24bit color image
1 = get 4bit grayscale image
Output: NONE
Semantics: Read an image from color cam and reduce it eventually to 8 bit gray scale. The color image can be reduced to grayscale afterwards by using IPColor2Grey(...) HINT: buf should be a pointer to an 'image' if conversion is enabled, like this:
image buffer;
CAMGetColFrame((colimage*)&buffer, 1);