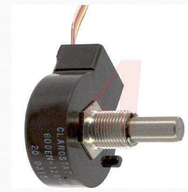
Optical Encoder

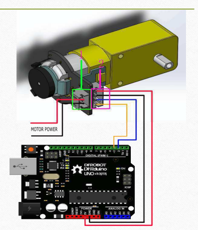
Yiran Shen

What is an encoder?



What is an encoder?

converts the angular position or motion of a shaft or axle to an analog or digital code



Absolute and incremental encoder

Absolute and incremental encoder

- Absolute encoder

  maintains position information when power is removed from the system

- Incremental encoder

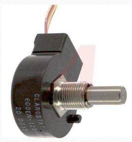
  can reports an incremental change in position of the encoder to the counting electronics

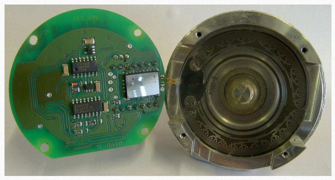
Absolute Encoder

- **Mechanical absolute encoders**
- **Optical absolute encoders**
- **Magnetic absolute encoders**
- **Capacitive absolute encoders**

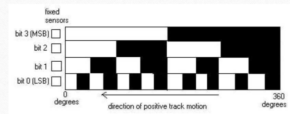
How does an encoder work?

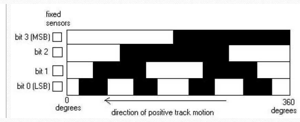
- Standard Binary Encoding
- Gray Encoding





How does an encoder work?

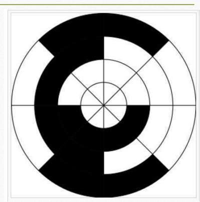




How does an encoder work?

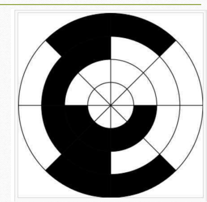
Gray Coding

Sector	Contact 1	Contact 2	Contact 3	Angle
0	off	off	off	0° to 45°
1	off	off	ON	45° to 90°
2	off	ON	ON	90° to 135°
3	off	ON	off	135° to 180°
4	ON	ON	off	180° to 225°
5	ON	ON	ON	225° to 270°
6	ON	off	ON	270° to 315°
7	ON	off	off	315° to 360°



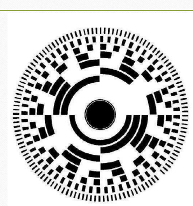
How does an encoder work?

- 3 bit
- 45 deg resolution



How does an encoder work?

- 8 bit
- 1.4 deg resolution



Questions?

References

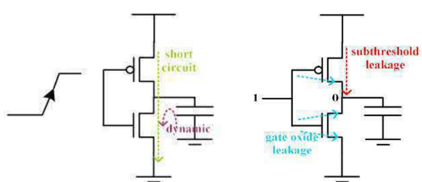
- https://www.dfrobot.com/wiki/index.php/Micro_DC_Motor_with_Encoder-SJ01_SKU:_FIT0450
- https://en.wikipedia.org/wiki/Rotary_encoder#Mechanical_absolute_encoders

Power consumption in Microprocessor

Yipeng Mou

Three source of Power Consumption

- Dynamic Power
- Static Power
- Short Circuit power



Dynamic Power Consumption: Input 1 -> 0

Energy from the power supply

$$E_{supply} = \int_0^{\infty} P(t)\cdot dt = \int_0^{\infty} V_{dd} i(t)\cdot dt = \int_0^{V_{dd}} V_{dd} C \cdot \frac{dV_O}{dt}\cdot dt = V_{dd} C \cdot \int_0^{V_{dd}} dV_O = CV_{dd}^2$$

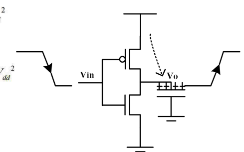
Energy stored in the capacitor

$$E_{CAP} = \int_0^{\infty} V_O \cdot i_C(t)\cdot dt = C\cdot \int_0^{\infty} V_O \cdot \frac{dV_O}{dt}\cdot dt = C\cdot \int_0^{V_{dd}} V_O \cdot dV_O = \frac{1}{2} CV_{dd}^2$$

Energy consumed by the PMOS

$$E_{PMOS} = \int_0^{\infty} P(t)\cdot dt = \int_0^{\infty} (V_{dd} - V_O)\cdot i_p(t)\cdot dt = C\cdot \int_0^{V_{dd}} (V_{dd} - V_O)\cdot dV_O = \frac{1}{2} CV_{dd}^2$$

– Power: $P = f \cdot E_{PMOS} = \frac{1}{2} fCV_{dd}^2$



Dynamic Power Consumption: Input 0 -> 1

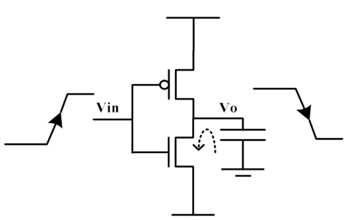
– Energy drawn from supply: 0

– Energy consumed by NMOS equals to the energy stored on the capacitance:

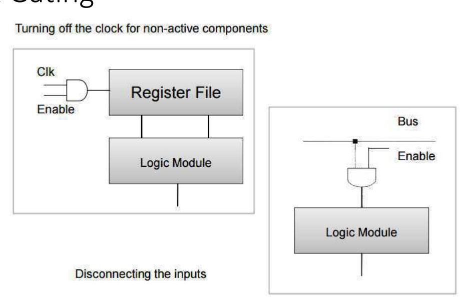
$$E_{NMOS} = \int V_O i(t)\cdot dt = \frac{1}{2} CV_{dd}^2$$

– Power:

$$P = f \cdot E_{NMOS} = \frac{1}{2} fCV_{dd}^2$$



Clock Gating



Static Power Consumption

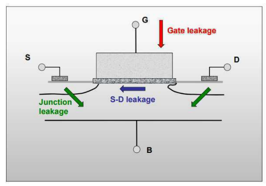
- Gate leakage
- Subthreshold leakage

Often expressed in base 10

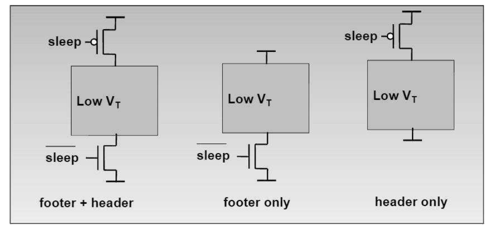
$$I_{DS} = I_S 10^{\frac{V_{GS}-V_{TH}}{S}}\left(1-10^{\frac{-nV_{DS}}{S}}\right)$$

≈ 1 for V_{DS} > 100 mV

Major MOSFET Leakage Components



Power Gating



Low Power Design

Jingyao Hu

Outline

- Why is it important?
- What limits the design?
- How to fix?

Why is low power design important?

New

Touch Bar and Touch ID
2.7GHz Processor
512GB Storage

2.7GHz quad-core Intel Core i7 processor
Turbo Boost up to 3.6GHz
16GB 2133MHz memory
512GB PCIe-based SSD[1]
Radeon Pro 455 with 2GB memory
Four Thunderbolt 3 ports
Touch Bar and Touch ID

$2,799.00

Timeline of apple products

The limit of frequency

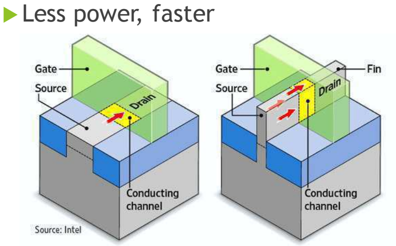
- $P(dynamic) = C*V^2*f$
- f ~ V
- In fact, $P\ (dynamic) = C*f^3$
- Enough heat to melt down

Old Technology

- Multiple cores
- Simple example:
- 70% frequency
- power = $0.7^3 = 0.34$ origin
- Two cores = 140% work with 68% power

New Cool Technology

- 2D transistor -> 3D transistor
- Less power, faster



THE BROKEN INTERNET OF THINGS

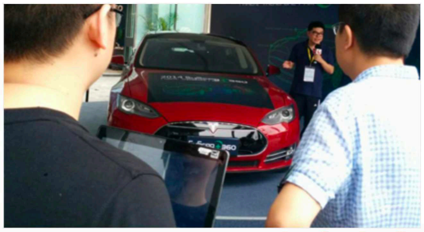
root/xc3511	root/vizxv	root/admin
admin/admin	root/888888	root/xmhdipc
root/default	root/juantech	root/123456
root/54321	support/support	root/(none)
admin/password	root/root	root/12345
user/user	admin/(none)	root/pass
admin/admin1234	root/1111	admin/smcadmin
admin/1111	root/666666	root/password
root/1234	root/klv123	Administrator/admin
service/service	supervisor/supervisor	guest/guest
guest/12345	guest/12345	admin1/password
administrator/1234	666666/666666	888888/888888
ubnt/ubnt	root/klv1234	root/Zte521
root/hi3518	root/jvbzd	root/anko

SECURITY

Security

Students hack Tesla Model S, make all its doors pop open IN MOTION

Toot the horn, too



21 Jul 2014 at 04:01, Darren Pauli

Security

TRENDnet home security cam flaw exposes thousands

Just when you thought you were alone in the bath

7 Feb 2012 at 17:01, John Leyden

TRENDnet has acknowledged a flaw that meant that live feeds from its home security cameras were accessible online without needing a password.

The US-based manufacturer admitted the problem - which affects its SecurView Cameras bought after April 2010 - and began releasing firmware updates designed to plug the hole on Monday. It apologised to its customers for the snafu in a security bulletin (extract below) that provides links to the relevant security upgrades.

> **TRENDnet has recently gained awareness of an IP camera vulnerability common to many TRENDnet SecurView cameras. It is TRENDnet's understanding that video from select TRENDnet IP cameras may be accessed online in real time. Upon awareness of the issue, TRENDnet initiated immediate actions to correct and publish updated firmware which resolves the vulnerability.**

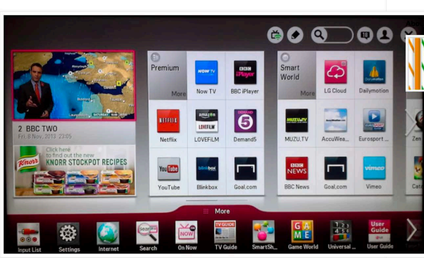
PRIVACY

https://twitter.com/localbusinessco/status/839456634745176064

Monday, 18 November 2013

LG Smart TVs logging USB filenames and viewing info to LG servers

Earlier this month I discovered that my new LG Smart TV was displaying ads on the Smart landing screen.



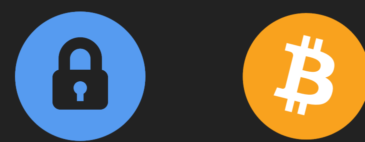
Blog Archi
2014 (
2013 (
Nov
LG S
ar

WHY?

PROBLEM:

- MANUFACTURERS' AND CONSUMERS' INCENTIVES DO NOT ALIGN
- NO INCENTIVE TO PROVIDE SUPPORT FOR PRODUCTS WITH LOW PROFIT MARGINS

SOLUTIONS



Modular Design and Smartphones

LUIS SOSA

EECS 373

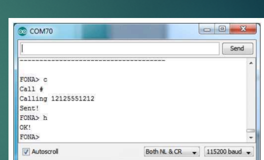
Examples of Modular Design

- Modular design is super relevant in todays world
  - Google Ara phone
  - Arduinos
  - Adafruit Raspberry Pi
- Adafruit offers several boards that make complicated devices such as a smartphone relatively simple.
  - Interfacing with cellphone towers
  - Power management
  - Protocol to send and receive messages/calls
- Most of the complications behind modular design is in the software.

Picture Source: http://www.highpants.net/wp-content/uploads/2014/03/highpants-project-ara-progresses-Ara-Phone.jpg

Adafruit FONA uFL Version

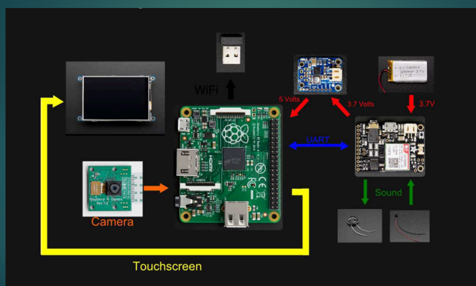
- Connects to any 2GB network
  - Provided you have a SIM card
- Make and receive phone calls
- Send and receive sms messages
- Buzzer vibrational motor
  - PWM controlled
- LiPo battery charging circuit
- Just need to know UART



Example Smartphone Design

- Raspberry Pi A+ 256MB
  - Brains of the device
  - Connects all the devices together and runs the OS
- Adafruit FONA uFL Version
  - All-in-one cellphone module
  - Interfaces with SIM card for network connection
- TYOS
  - OS that allows the user to send messages and make calls
  - Created by the author of this DIY project
  - Constantly being upgraded
- Powerboost 500 Basic
  - Really just a TPS61090 boost converter
  - Converts 1.8v+ to 5.2v

System Design Diagram



Source: https://hackaday.io/project/5083/logs

Relevance

- Modular design makes it easy to prototype projects
- Users want more flexibility in their devices
- Show and describe a project and the lay out this project uses.
  - Replace the Raspberry Pi with the Smartfusion
- Introduce the Fona Adafruit board
- Show another example of what we can do with embedded systems.

References

- Step by step guide
  - http://www.instructables.com/id/Build-Your-Own-Smartphone/
- Project Overview
  - https://hackaday.io/project/5083-diy-smartphone
- Youtube Demonstration
  - https://www.youtube.com/watch?v=H2AY7ciEJvo
- FONA information
  - https://learn.adafruit.com/adafruit-fona-mini-gsm-gprs-cellular-phone-module/pinouts

Questions?

LCD Interfacing

Josh Liu

Relevance

- Inevitable part in almost all embedded projects
- Simple means of of adding visual appeal to embedded applications
- Usage
  - Applications: computer monitors, TVs, instrument panels, aircraft cockpit displays
  - Portable consumer devices: digital cameras, watches, calculators, smartphones
  - Consumer electronics: DVD players, video game devices, clocks
- Replaced bulky cathode ray tube (CRT) in nearly all applications
- LCD modules with integrated RS-232, I2C, and SPI serial interfaces

Introduction

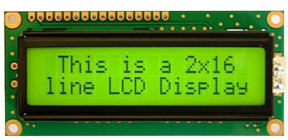
- Thin, flat consume small amount of power
- Rod-shaped tiny molecules sandwiched between a flat piece of glass and an opaque substrate
- Molecules align in two different physical positions based on electric charge applied to them
  - Apply charge: molecules align to block light entering
  - No charge: molecules become transparent

Character vs. Graphic LCDs

- Character LCD
  - Displays numbers, letters and fixed symbols
  - Used in old style industrial panel display where there's a need to display a fixed number of characters
- Graphic LCD
  - Instead of segments, has pixels in rows and columns
  - By energizing set of pixels any character can be displayed

Interfacing 16x2 LCD

- Most common configuration used due to reduced cost and small footprint
- Displays 32 characters at a time in 2 rows (16 characters per row)
  - 40 character positions, remaining 24 only displayed with "scrolling" effect



Pin Configurations

PIN NO	NAME	FUNCTION
1	VSS	Ground pin
2	VCC	Power supply pin of 5V
3	VEE	Used for adjusting the contrast commonly attached to the potentiometer.
4	RS	RS is the register select pin used to write display data to the LCD (characters), this pin has to be high when writing the data to the LCD. During the initializing sequence and other commands this pin should low.
5	R/W	Reading and writing data to the LCD for reading the data R/W pin should be high (R/W=1) to write the data to LCD R/W pin should be low (R/W=0)
6	E	Enable pin is for starting or enabling the module. A high to low pulse of about 450ns pulse is given to this pin.

Pin Configurations (cont'd)

7	DB0	
8	DB1	
9	DB2	
10	DB3	
11	DB4	DB0-DB7 Data pins for giving data(normal data like numbers characters or command data) which is meant to be displayed
12	DB5	
13	DB6	
14	DB7	
15	LED+	Back light of the LCD which should be connected to Vcc
16	LED-	Back light of LCD which should be connected to ground.

Displaying data

Follow these simple steps for displaying a character or data

- E=1; enable pin should be high
- RS=1; Register select should be high
- R/W=0; Read/Write pin should be low

Sending a command

To send a command to the LCD just follows these steps:

- E=1; enable pin should be high
- RS=0; Register select should be low
- R/W=1; Read/Write pin should be high

Commands

COMMAND	FUNCTION
0F	For switching on LCD, blinking the cursor.
1	Clearing the screen
2	Return home.
4	Decrement cursor
6	Increment cursor
E	Display on and also cursor on
80	Force cursor to beginning of the first line
C0	Force cursor to beginning of second line
38	Use two lines and 5x7 matrix
83	Cursor line 1 position 3
3C	Activate second line
0C3	Jump to second line position 3
0C1	Jump to second line position1

Sources

- http://www.eeherald.com/section/design-guide/esmod17.html
- https://www.digikey.com/en/product-highlight/n/newhaven-display/lcd-serial-displays
- http://www.electronicshub.org/interfacing-16x2-lcd-8051/
- http://embedjournal.com/interfacing-lcd-module-part-1/

Thank you

Computer Vision in Embedded Systems

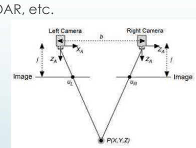
By Rishi Bhuta

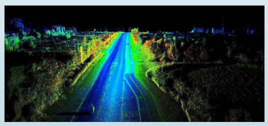
Computer Vision Introduction

- Computer Vision is an interdisciplinary field that deals with how computers can be made for gaining high-level understanding from digital images or videos.
- From a software standpoint, computer vision works to apply mathematical theory to an input image to either modify it or extract meaning from it.
- This cannot be done without the embedded systems that provide input images and processing power.

Sensors in CV

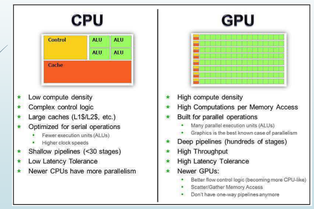
- Mono camera, Stereo camera, LIDAR, etc.

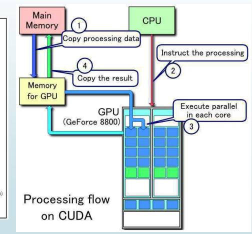




Specialized Processing for Images

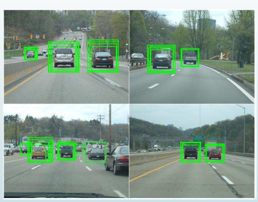
- Specialized processor or GPU for parallelizing image matrix operations.

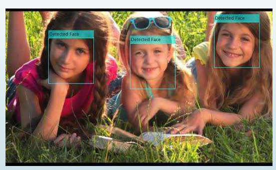




Examples of Computer Vision

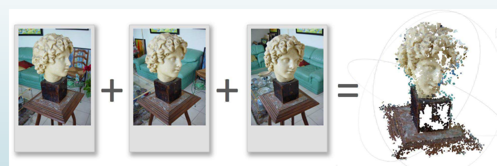
- Object Recognition (Car detection, face detection, etc.)





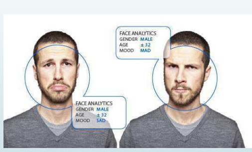
Examples of Computer Vision

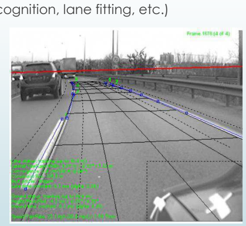
- Scene Recreation (Picture stitching, 3D reconstruction, structure from motion, etc.)



Examples of Computer Vision

- Information Extraction (Emotional recognition, lane fitting, etc.)





Using CV in Your Project

Install OpenCV

It's a well-documented library that contains a vast array of functions and useful classes.

- Built for C/C++ or Python.

Try Tutorials

Try a few examples from the OpenCV website.

- AR Tag recognition
- Face Detection
- Finding Shapes
- All you need is your laptop webcam!

Determine Applications

Find a use for CV in your project.

- Recognizing a change in environmental conditions
- Determining distances between objects

Interface with hardware

Run your CV algorithm and communicate results via a serial connection

- Output coordinates of two objects relative to a known map

References

- http://docs.opencv.org/
- https://www.embedded-vision.com/what-is-embedded-vision
- http://www.bdti.com/private/pubs/BDTI_ESC_Embedded_Vision.pdf
- http://www.embedded.com/design/system-integration/4372167/Introduction-to-embedded-vision-and-the-OpenCV-library

Batteries and Power

Denny Zhang

Embedded systems

- Embedded systems will usually require some sort of portable power source.
- The easiest way to provide this will be with a battery.
- Other options available - Energy scavenging
  - Solar panels, Motion, Heat
- Plug it in

Design Considerations

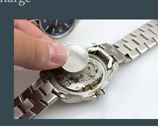
- Power Consumption
- Power Density
- Voltage requirements
- Battery size, weight
- System battery life.

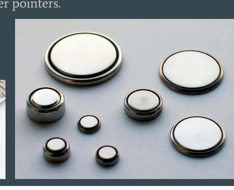
AA style

- Cheap and plentiful
- Consumer usually pays
- Alkaline AA is approx 2500mAh @ 1.5V
- Ni-MH AA is 1500-2500mAh @ 1.25V
- Used in many consumer electronics/gadgets
- Ni-MH as higher discharge
- Discharge at 0.5C - 1C

CR2032 + coin cells

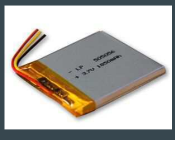
- Chemistry varies - Lithium, Alkaline, Silver Oxide
- Small size, for small electronics
- Typical uses are watches, hearing aids, laser pointers.
- CR2032 is Lithium 240mAh @ 3.0V
- An LR44 is 105mAh @ 1.5V
- Very low discharge





Lithium

- Battery shape and size varies widely
- Typically used in smartphones and newer gadgets.
- Nominal voltage is 3.7V with mAh capacity depending on size
- A typical phone contains a 2500mAh - 3000mAh battery.
- Light, and high power density
- Discharge at 2C - 10C or higher.



Runtime

- Find the application's average power consumption mA or mW
- Divide capacity by power consumption mAh / mA or mWh / mW
- Ex: a 2000mAh battery with 200mA drain will last 2000/200 = 10 hours

Runtime Complications

- A 1.5V alkaline cell (2500mAh) with a 100mA drain will run about 25 hours
  However, with a 2.5A drain, will run for much less than 1 hour
- The culprit? Internal resistance
- Generally, higher drain will reduce battery capacity and battery life.

Multithreading for Dummies

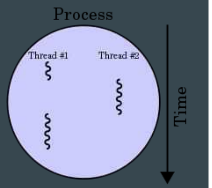
Brandon Waggoner
March 23, 2017

Motivation

- Speed up code execution
- Reduce redundancy
- Squeeze out extra "performance"

Can you sew with it?

- A thread is a stream of execution, complete with its own stack pointer, **local** variables, and context
- Threads often run out of order, and can give the appearance of running two functions simultaneously



Embedded Systems and You

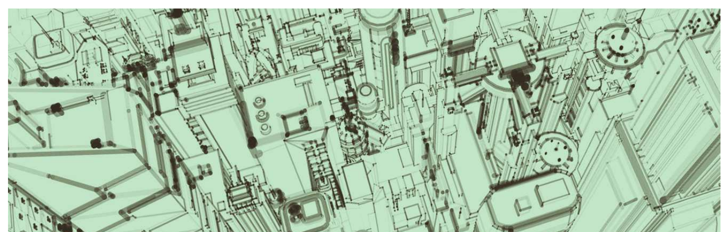
- Multithreading isn't always the best solution
  - ***"Among competing hypotheses, the one with the fewest assumptions should be selected." (Occam's Razor)***
  - Often, determinism is more valued than throughput
- Threading requires a significant effort
  - Timing and exclusivism are up in the air
  - More places for things to go wrong
  - Debugging becomes a headache

Then Why

- It's fun
- Automate remedial tasks
- Prevent busy loops

How to get started

- Pthread Library in C/thread library in C++
  - Thread create
  - Thread join/wait
  - Locks/Monitors
  - Condition Variables
- Python also has a thread library!



Smart Cities

Rohan Dasika

Agenda

- What are smart cities?
- Is there a need?
- Benefits of smart cities
- Challenges ahead
- Key players

What are smart cities?

- A city that collects and bases its decisions on vast amounts of data to best utilize their resources, improve living conditions, and manage infrastructure
- Data collection (massive sensor network)
- Connectivity (5G, IoT, etc)
- Analysis (Data analytics, machine learning, etc)
- Sustainable policy

Is there a need?

- YES!
- Motivations behind the movement
  - Climate change
  - Public health crises
  - Congested commutes
  - High costs of living
  - Fossil fuel dependency
- A new set of technologies — connectivity, real-time sensors, precise location services, autonomous systems — can collectively transform city life

Benefits of smart cities

- Mobility
  - Analyze population flow/transportation to better public transport
  - Solve the problem of never having enough parking
  - Smart roads – energy harvesting, alert cars/municipalities of their condition
- Environment
  - Green buildings
  - Smart water systems/irrigation methods
  - Smart grid
- Digital divide
  - Granting internet access to underprivileged people

Benefits of smart cities

- Manufacturing & Trade
  - More streamlined systems for production and distribution
- Government
  - Enable to be more 'in touch' with constituents
- Health
  - Monitors for dangerous levels of gases
  - Reducing pollution

Challenges ahead

- Most obvious – security
- Scaling
- Smart city model depends on the development of:
  - Cooperative government
  - Robust economic model
  - Smart and engaged citizens

Key players

- GE – Energy
- Cisco – Internet support
- AT&T/Verizon – IoT ecosystem
- Google – Data, Autonomous vehicles
- Sidewalk Labs – bridging gap between tech + policy

Logic Circuit Minimization with Espresso

Gigi Guarino

What is espresso?

- A program used to minimize logic circuits and boolean functions
- Developed by the University of California, Berkeley, in the 1990's

Why is this helpful?

For our project:

- Minimize the combinational logic in our Verilog

In RTL design:

- Minimize the amount of logic gates

Input

A .pla file

Specify number of inputs with .i

Specify number of outputs with .o

Truth table for only outputs that result in a 1

Signify end of file with .e

example.pla

```
.i 3
.o 1

000 0
001 1
010 1
011 0
100 1
101 0
110 0
111 0

.e
```

Output

To run espresso from command line...

With no flag, outputs a minimized .pla file in sum of products form

```
$ espresso.exe input.pla
```

With flag -epos, outputs a minimized .pla file in product of sums form

```
$ espresso.exe -epos input.pla > output.pla
```

SOP vs. POS

Sum of products example:

```
out = a'b'c + a'bc' + ab'c'
```

Product of sums example:

```
out =(a + b + c)(a + b' + c')
(a' + b + c')(a' + b' + c)
(a' + b' + c')
```

a	b	c	out
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Example:

a	b	c	d	out
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

To the left is a truth table for our desired logic circuit

To the right is the .pla file we are going to input to espresso
It has 4 inputs
It has 1 output

The truth table only contains the input combinations that result in a 1, the 0's could be included but are unnecessary

```
.i 4
.o 1

0001 1
0010 1
0011 1
0110 1
0111 1
1000 1
1001 1
1010 1
1100 1
1101 1
1110 1

.e
```

Logic function before

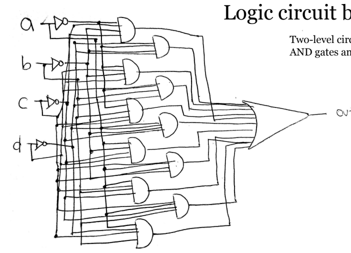
SOP:

```
f(a,b,c,d) = a'b'c'd + a'b'cd' + a'b'cd + a'bcd' + a'bcd +
ab'c'd' + ab'c'd + ab'cd' + abc'd' + abc'd + abcd'
```

POS:

```
f(a,b,c,d) = (a + b + c + d)(a + b' + c + d)
(a + b' + c + d')(a' + b + c' + d')(a' + b' + c' + d')
```

Logic circuit before (SOP)



Two-level circuit with 11 4-input AND gates and 1 11-input OR gate

Output .pla

```
$ espresso.exe abcd.pla

.i 4
.o 1
.p 4
-001 1
--10 1
1-0- 1
0-1- 1
```

```
$ espresso.exe -epos abcd.pla

.i 4
.o 1
.#phase 0
.p 3
1-11 1
0-00 1
010- 1
.e
```

Minimized functions

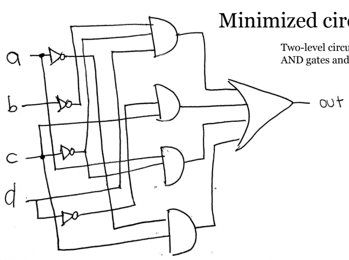
SOP:

```
f(a,b,c,d) = b'c'd + cd' + ac' + a'c
```

POS:

```
f(a,b,c,d) = (a + c + d)(a' + c' + d')(a' + b + c')
```

Minimized circuit (SOP)



Two-level circuit with 4 4-input AND gates and 1 4-input OR gate

Questions?

Sensors used in Self-Driving Vehicles

Presented by Saad Shaik

Introduction

- Self-driving vehicles need to read a variety of information about their surroundings
- Different sensors have their own unique advantages and disadvantages
- Measuring all necessary information requires an array of specialized sensors
- The three main sensing techniques are LIDAR, Radar, and Cameras

LIDAR

- Stands for Light Detection and Ranging
- Scans the surroundings by rotating a laser and measuring the reflected intensity
- Provides information on the distance, shapes, and speed of nearby objects
- Range of 100m with resolution of ~10cm
- Pros:
  - Mid-range, high precision object detection
  - Generates 3D maps to detect hills
  - Accurately detects stationary objects

Radar

- Stands for Radio Detection and Ranging
- Scans the surroundings with radio waves and measures the reflected intensity
- Provides information on the distance and velocity of near to mid-range objects
- Pros:
  - Works in all weather conditions; rain, snow, fog
  - Radar can see behind obstacles and two cars ahead
  - Accurate for close range object detection, useful for parking and lane-changing

Camera

- Uses a camera to capture visual and color data of the surroundings
- Provides information on visual cues such as traffic lights, cones, signs, lane markers
- Range of up to 250m
- Pros:
  - Only sensor that can detect color and text, can differentiate objects based on color
  - Best sensor for scene interpretation
  - Cheap enough to have multiple on one car

Summary

- Each sensor has its own advantages and specialized use cases
- LIDAR is used for precision object detection and 3D mapping
- Radar is used for measuring velocities and validating LIDAR in all weather conditions
- Cameras are used for visual cues and color detection
- The best self-driving system will have a multi-sensor network to include each sensor's unique advantages

Works Cited

1. "Self-driving vehicles -- are we nearly there yet?" Rudy Ramos http://www.embedded.com/electronics-blogs/say-what-/4442823/Self-driving-vehicles---are-we-nearly-there-yet-
2. "Autonomous Cars' Pick: Camera, Radar, Lidar?" Davide Santo http://www.eetimes.com/author.asp?section_id=36&doc_id=1330069
3. "Self-driving cars will bristle with sensors" Stephen Shankland https://www.cnet.com/news/self-driving-cars-will-bristle-with-sensors/
4. "Self-Driving Cars' Spinning-Laser Problem" Tom Simonite https://www.technologyreview.com/s/603885/autonomous-cars-lidar-sensors/
5. "The Autonomous Car: A Diverse Array of Sensors Drives Navigation, Driving, and Performance" Bill Schweber http://www.mouser.com/applications/autonomous-car-sensors-drive-performance/

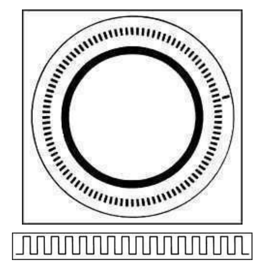
Quadrature Decoding

Christopher Schmotzer
March 23, 2017

Motivation

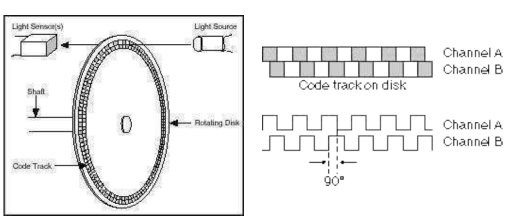
- Position and Velocity Measurements for Motor Control
  - Tachometer
  - Potentiometer
  - Optical Encoder
    - Absolute Encoder
    - **Incremental Encoder**

Incremental Encoder



http://www.ni.com/white-paper/14805/en/

Quadrature Encoder

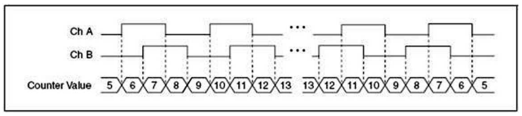


http://www.ni.com/tutorial/7109/en/
http://www.ni.com/white-paper/4763/en/

Direction of Rotation

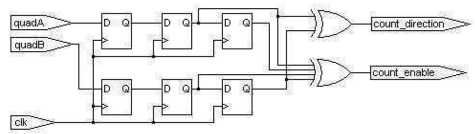
- *Phase Difference* corresponds to Direction of Rotation
  - Channel A leads Channel B implies Increment
  - Channel B leads Channel A implies Decrement

Quadrature Decoder



http://www.ni.com/tutorial/7109/en/

Block Diagram



http://www.fpga4fun.com/QuadratureDecoder.html

Implementation

```
module quad(clk, quadA, quadB, count);
input clk, quadA, quadB;
output [7:0] count;

reg [2:0] quadA_delayed, quadB_delayed;
always @(posedge clk) quadA_delayed <= {quadA_delayed[1:0], quadA};
always @(posedge clk) quadB_delayed <= {quadB_delayed[1:0], quadB};

wire count_enable = quadA_delayed[1] ^ quadA_delayed[2] ^ quadB_delayed[1] ^ quadB_delayed[2];
wire count_direction = quadA_delayed[1] ^ quadB_delayed[2];

reg [7:0] count;
always @(posedge clk)
begin
  if(count_enable)
  begin
    if(count_direction) count<=count+1; else count<=count-1;
  end
end

endmodule
```

http://www.fpga4fun.com/QuadratureDecoder.html

APPENDIX: Types of Decoding

- 1X
  - Counter is incremented/decremented by rising edges of one channel only
  - Cannot determine direction
- 2X
  - Counter is incremented/decremented by rising AND falling edges of one channel only
  - Cannot determine direction
- 4X
  - Counter is incremented/decremented by rising and falling edges of Channels A and B
  - Can determine direction

APPENDIX: Definitions

- Cycles Per Revolution (CPR)
  - Number of full quadrature cycles per full shaft revolution (360 mechanical degrees)
  - 200 CPR encoder provides 200, 400, or 800 distinct positions in 1x, 2x, or 4x modes respectively
- Quadrature
  - Phase difference of 90° between two waves at the same frequency