# EECS 373
# Midterm 2 Review

Austin & Guthrie
Winter 23

# Topics

- Timers
- Serial communication
- Sampling, ADCs, DACs
- Datasheets
- Prototyping
- Memory
- PCBs
- Power consumption, temperature, and reliability
- Motors and driving them
- Analog interface circuits

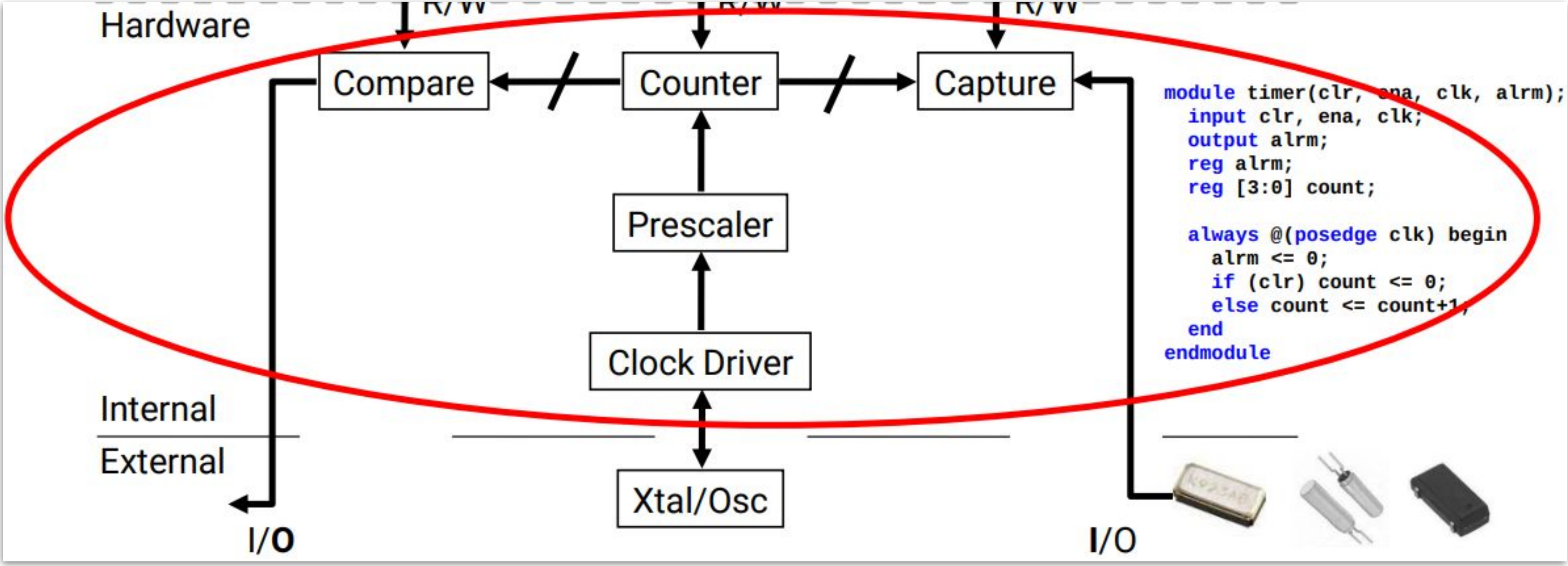
# Timers

- Better than software loop-based delays
- Allow for more precise timing, delays, and control
- Two basic uses for timers:
  - Measure how long something takes - “Capture”
  - Have something happen once or every X time period - “Compare”
- Virtual Timers
  - We can make one hardware timer look like multiple timers through software
- PWM
  - Prescaler, CCR, ARR

# Timers

- Period
  - Time between successive pulses.
- Duty Cycle
  - Percentage of time signal is “on”.
- Prescaler
  - Number that is used to divide the clock to produce counter frequency.
- ARR
  - Auto Reload Register. Used to determine period (treat is as overflow / max value).
- CCR
  - Capture & Compare Register. Used to trigger an event (e.g. duty cycle).

# Timers



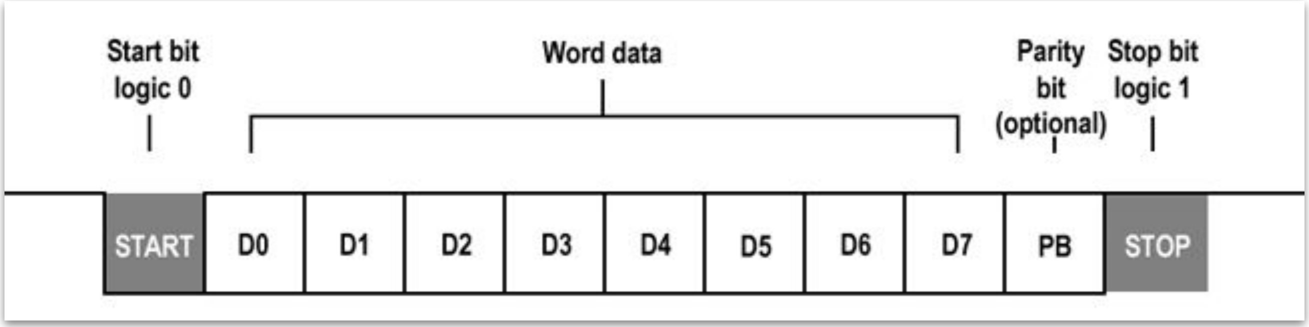
```
module timer(clr, ena, clk, alrm);
  input clr, ena, clk;
  output alrm;
  reg alrm;
  reg [3:0] count;

  always @(posedge clk) begin
    alrm <= 0;
    if (clr) count <= 0;
    else count <= count+1;
  end
endmodule
```

# Serial Communication

- Send data one bit at a time (serially).
- Asynchronous vs synchronous communication protocols
  - Asynchronous protocols (e.g. UART, CAN) do not use a clock signal to synchronize data signals.
  - Synchronous protocols (e.g. I2C, SPI) use a clock signal to synchronize data signals.
- Methods of dealing with multiple bus drivers
  - Tri-State Buffers
  - MUX
  - Open collector (pull-up, wired OR)

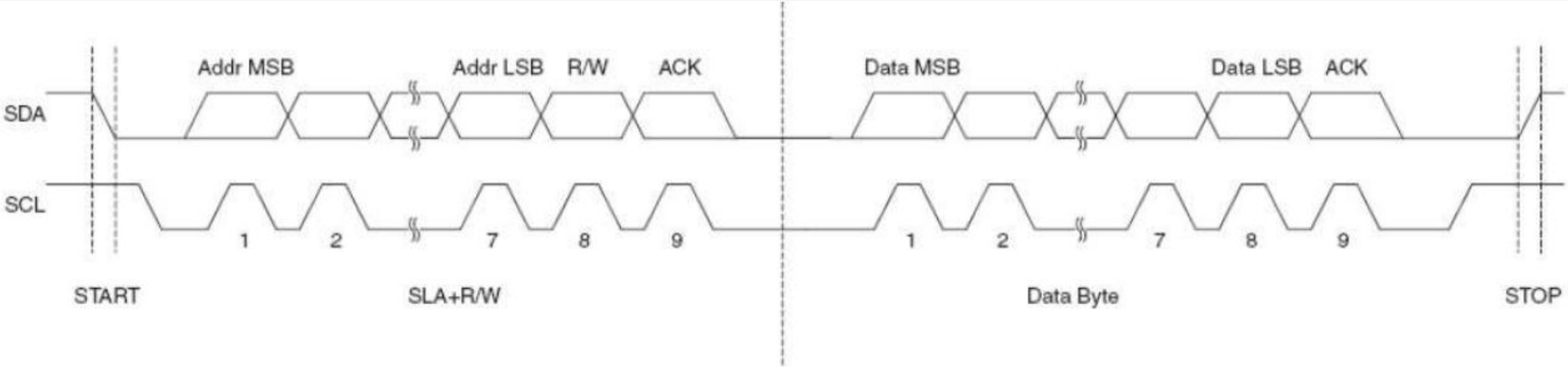
# Serial Communication - UART

- Wires are RXD and TXD. Sometimes RTS and CTS for flow control (e.g., RS-232).
- UART is push-pull type (not open-drain). Default high. Asynchronous.
- Communication between two devices only. Need to agree on parity and baud rate.
- Transmission is the following: Logic low start bit, data bits, parity bit, logic high stop bit.



# Serial Communication - I2C

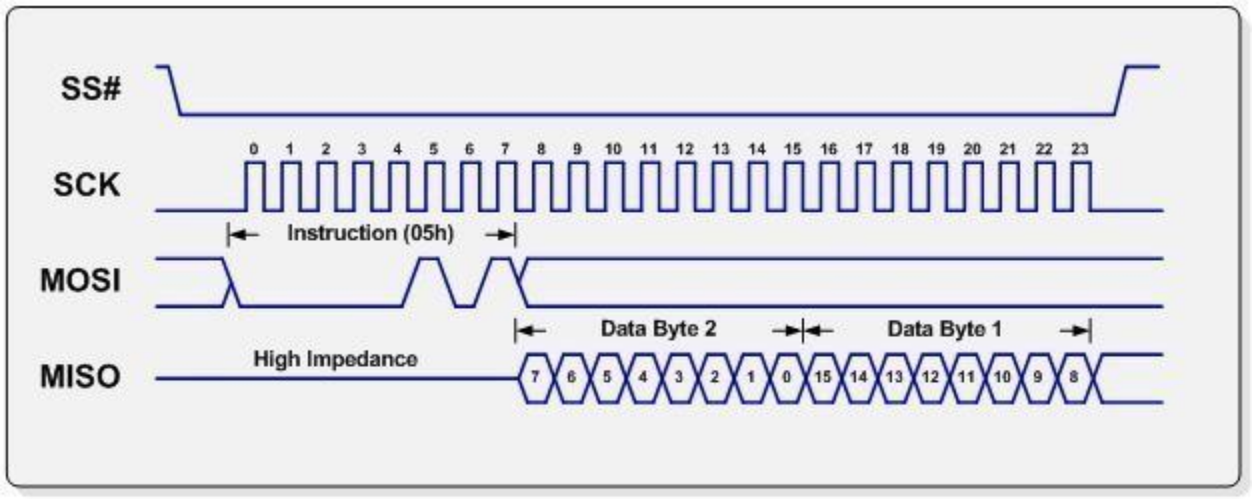
- Wires are SDA and SCL (2).
- I2C is open-collector. Default high. Synchronous.
- Multi-leader, multi-follower. Addressing is used. 1 is read. 0 is ack.
- Transmission is the following: Leader initiates, start, address, data, ack, stop.



Source: ATMega8 Handbook

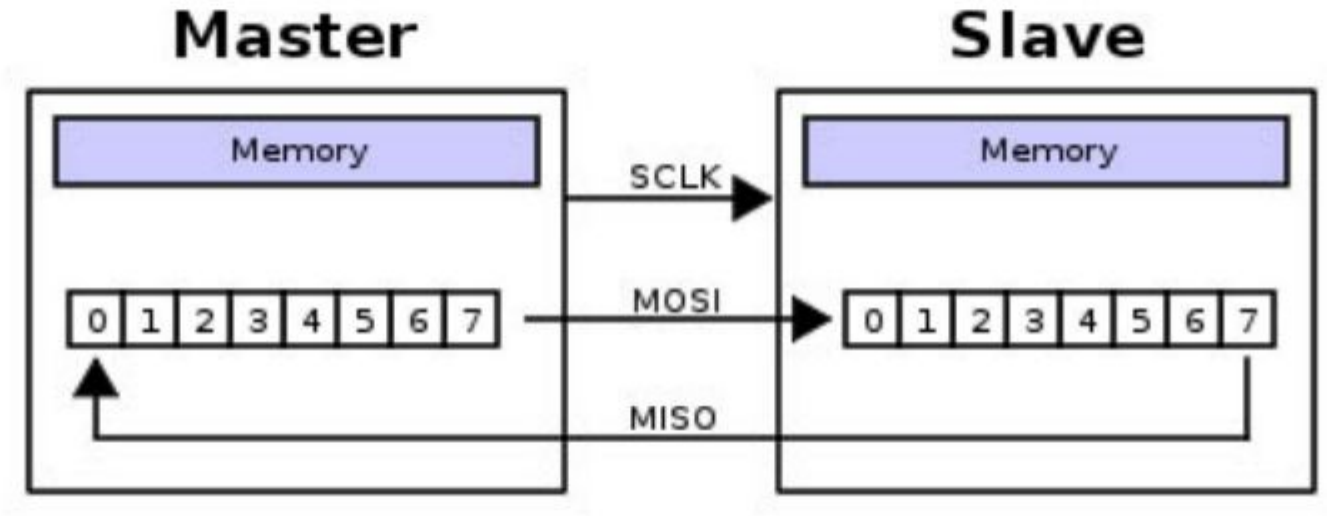
# Serial Communication - SPI

- Wires are MOSI, MISO, SCLK, SS (Minimum 4).
- SPI is push-pull. Synchronous.
- Single master, multiple slaves. Slave select line is used to select device.
- Transmission is the following: Master generates SCLK, shift registers move in and out data.



# Serial Communication - SPI

- Shifting protocol

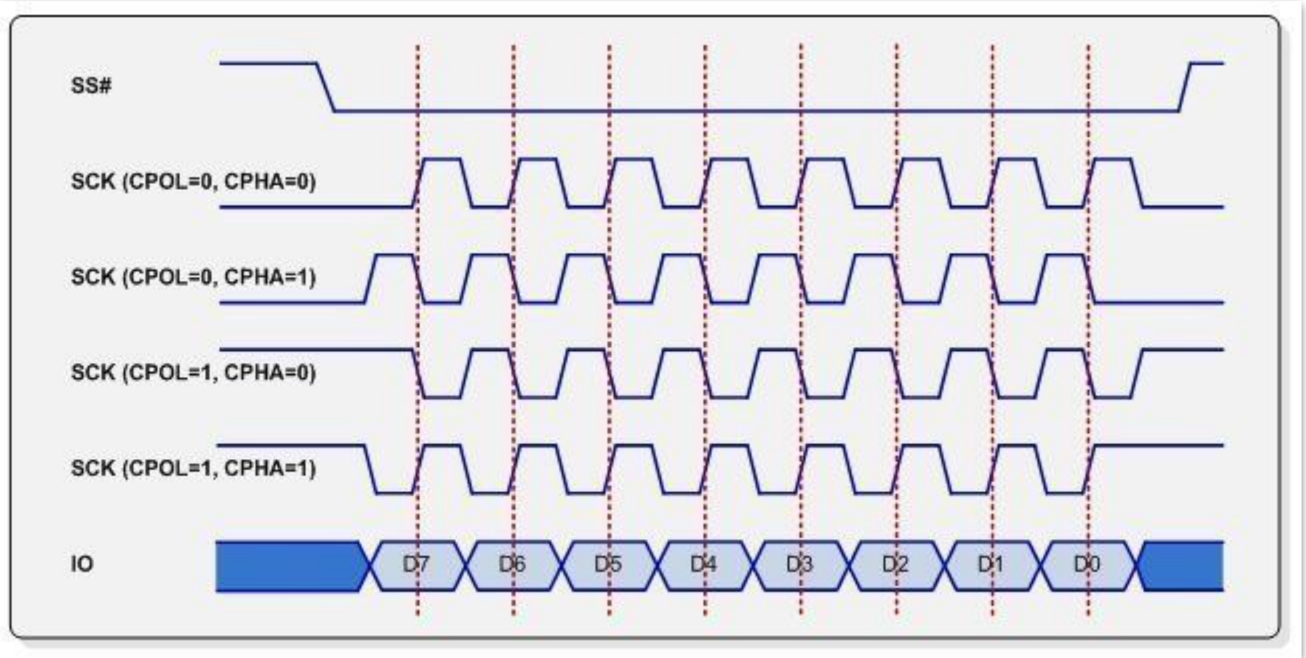


Master shifts out data to Slave, and shift in data from Slave

http://upload.wikimedia.org/wikipedia/commons/thumb/b/bb/SPI_8-bit_circular_transfer.svg/400px-SPI_8-bit_circular_transfer.svg.png

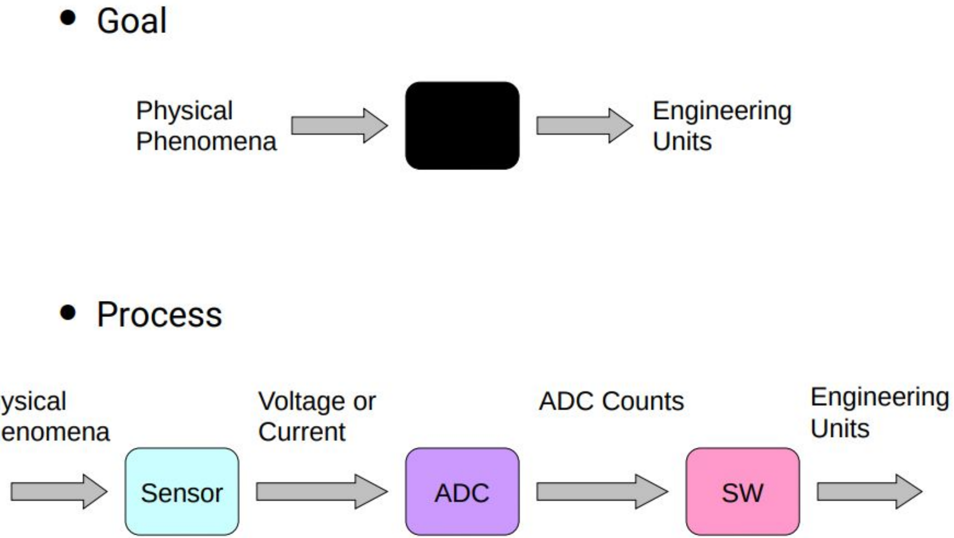
# Serial Communication - SPI

- Clock polarity (CPOL) and clock phase (CPHA).



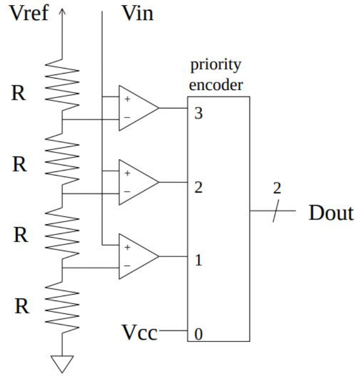
# Sampling, ADCs, DACs

- Many sensors are effectively analog
- Need to process these with computers
- Solution is ADC



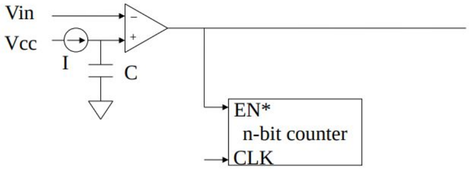
# ADC Types

## ADC #1: flash



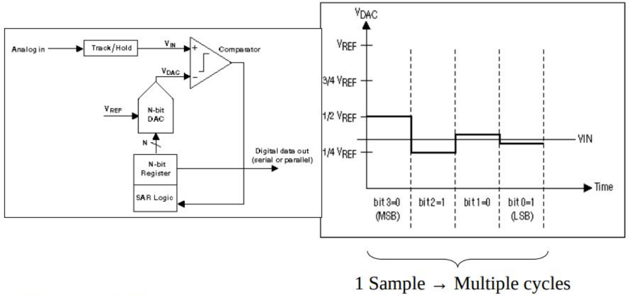
- Compare input with linear range of voltages.
- Use priority encoder.
- Only leave on highest input bit.
  - 001111111 → 001000000.
- Convert input bit index to binary number.
- 001000000 → 110
  - Noting that 000000000 → 000.
- Huge: $O(2^n)$.

## ADC #2: single-slope integration



- Start: Reset counter, discharge C.
- Charge C at fixed current I until Vc > Vin.
- Final counter value is Dout.
- Slow: O(2n).
  - Conversion may take several milliseconds.
- Good differential linearity (dI/dO).
- Absolute linearity depends on precision of C, I, and clock.

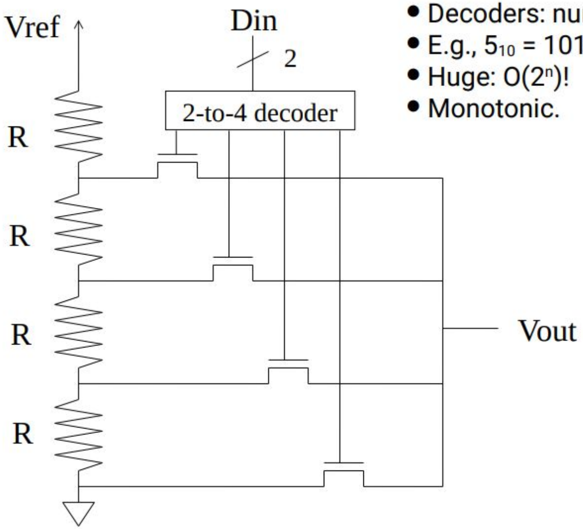
## ADC #3: successive approximation



- Uses DAC for guessing.
- Somewhat fast: O(n).
- Goes from MSB to LSB.

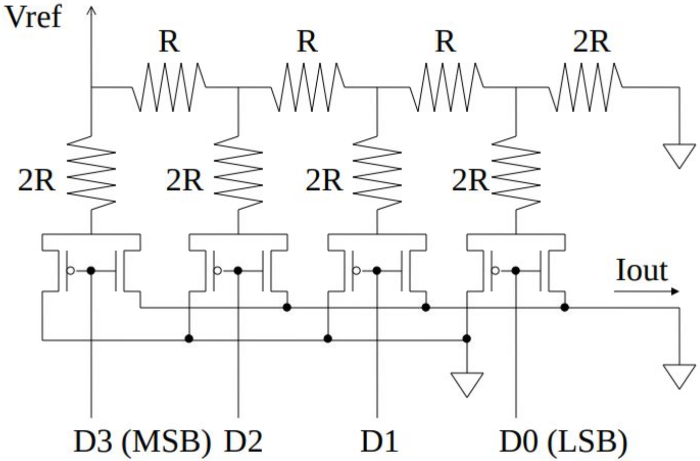
# Types of DACs

## DAC #1: voltage divider



- Fast.
- Decoders: number → single on-bit.
- E.g., $5_{10} = 101_2$ → 00010000.
- Huge: O($2^n$)!
- Monotonic.

## DAC #2: R/2R ladder

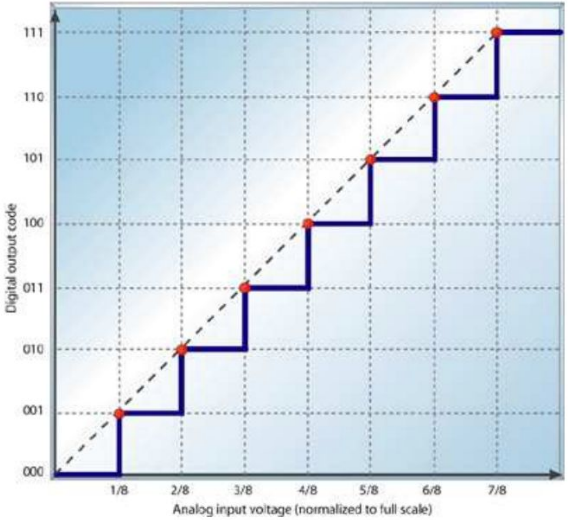


- Small: O(n).
- Monotonicity? (Consider 0111 -> 1000)

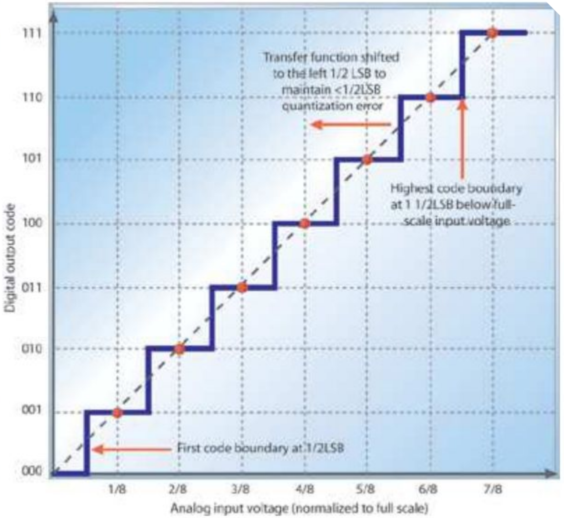
# Sampling

- Types of Error:
  - Quantization Error
    - Error inherent to the quantization found in digital bits
  - Sampling (Timing) Error
    - Shannon-Nyquist Theorem: $f_{sample} > f_N = 2*f_{max}$
  - NonLinearity, Offset, and Gain Error
    - Error due to variations from ideal electrical characteristics

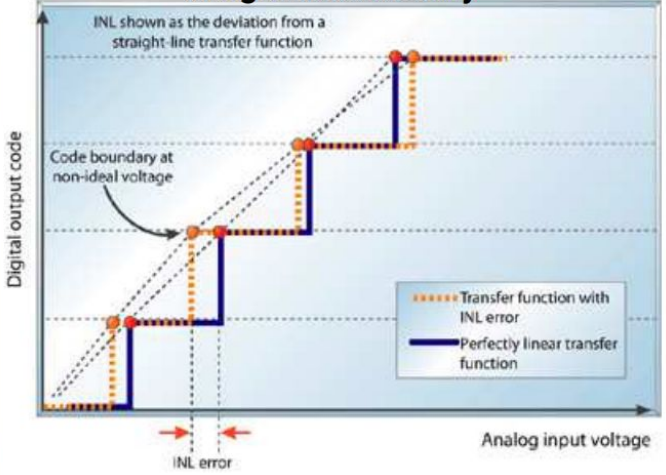
# Sampling (cont.)



Built-in ½ LSB error corrected

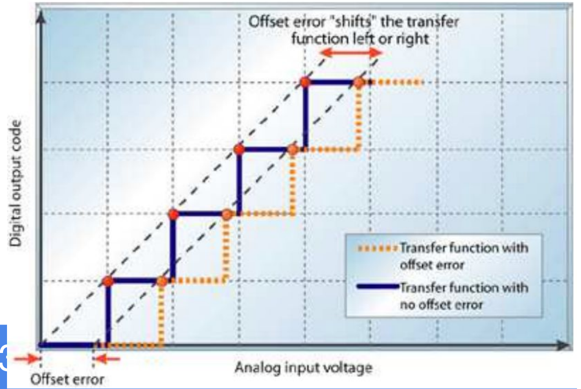


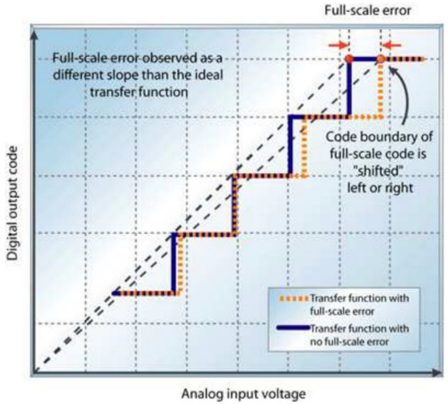
Integral nonlinearity



Offset error

Full-scale error is also sometimes called “gain error”





# Datasheets

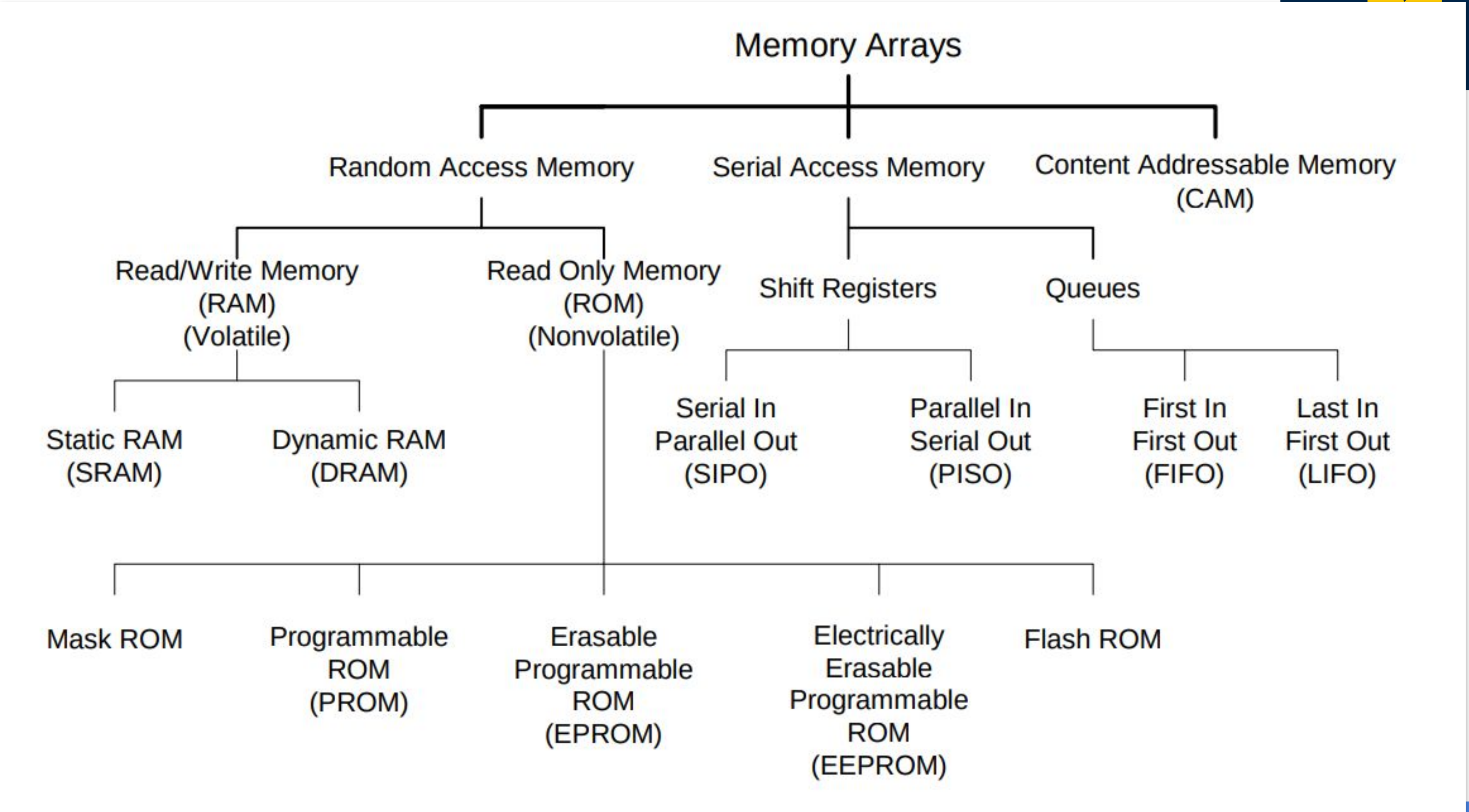
- Important Information on Datasheets
  - Cover page
    - Features, Applications, General Description
  - Conceptual Schematic Diagram and pinout
  - Physical and electrical characteristics
  - Output range and transfer functions (Graphs)
  - Register mapping
  - Example circuits

# Prototyping

- Prototyping options
  - Breadboards
    - Quick but unreliable due to nasty parasitics. Fine for low-frequency (<= 1 MHz, sometimes < 10 MHz).
  - Soldering
    - Use tools such as soldering iron, sponge or scrubber, solder sucker, wick, tinning block, heatsink.
    - Heat the trace and leads first, then let solder into gaps due to surface tension.
  - Wire Wrap
- Noise sources
  - Capacitive coupling, RF, motors, solenoids and relays.
  - Solutions include independent power supplies, ferrite beads, opto-isolators, conductive shielding, coaxial cables and twisted pair).

# Prototyping

- Guild handshake of computer engineers
  - If ESD sensitive component is on path between high potential and low-potential student, it may be damaged or destroyed.
- ESD (Electrostatic discharge)
  - High potential difference results in high momentary current
  - Can destroy gate oxide and erase non-volatile memory locations.
  - For electrostatic discharge prevention, ground yourself.



Memory Arrays

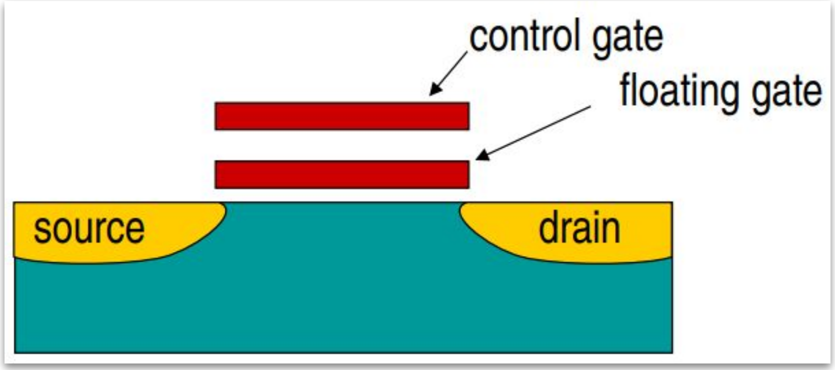
- Random Access Memory
  - Read/Write Memory (RAM) (Volatile)
    - Static RAM (SRAM)
    - Dynamic RAM (DRAM)
  - Read Only Memory (ROM) (Nonvolatile)
    - Mask ROM
    - Programmable ROM (PROM)
    - Erasable Programmable ROM (EPROM)
    - Electrically Erasable Programmable ROM (EEPROM)
    - Flash ROM
- Serial Access Memory
  - Shift Registers
    - Serial In Parallel Out (SIPO)
    - Parallel In Serial Out (PISO)
  - Queues
    - First In First Out (FIFO)
    - Last In First Out (LIFO)
- Content Addressable Memory (CAM)

# Memory

- Non-volatile memory
  - Flash ~ms
    - Writing is very slow, and can only write a finite amount of times
  - EEPROM ~ms
    - Similar to Flash
  - SRAM/DRAM ~ns
    - DRAM can be slower
  - Others (Spin-Transfer Torque Magnetic Memory, Bipolar-Filamentary OxRAM, etc)

# Memory

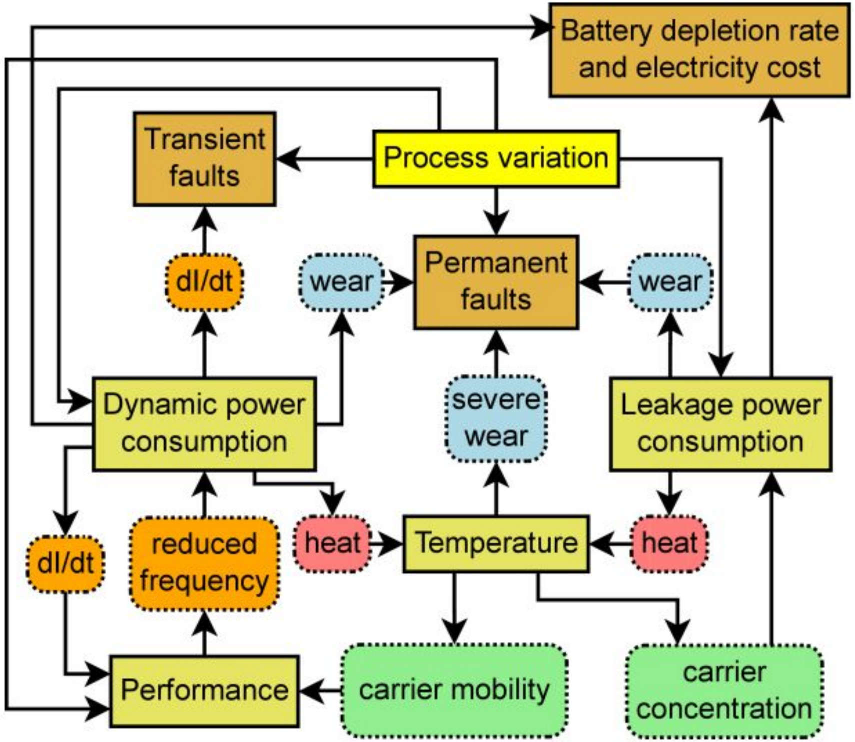
- Floating Gates
  - Can determine electrons in floating gate with control gate
    - Should behave like a FET if neutral. Otherwise, electrons trapped in gate.
  - To erase with EPROM, can shine UV light
  - To erase with EEPROM and flash, can erase with Fowler-Nordheim tunneling



# PCBs

- Design Process
  1. Create Schematic
  2. Place Parts
  3. Board Design
  4. Generate Files
- Basic Terminology
  - Traces
  - Vias
  - Silk Screen
  - Solderpad
  - Through-hole
  - Through-hole component
  - Surface mount component

# Power consumption, temperature, and reliability



# Power Consumption

State-Based Power Modeling

- For each component
  - For each state

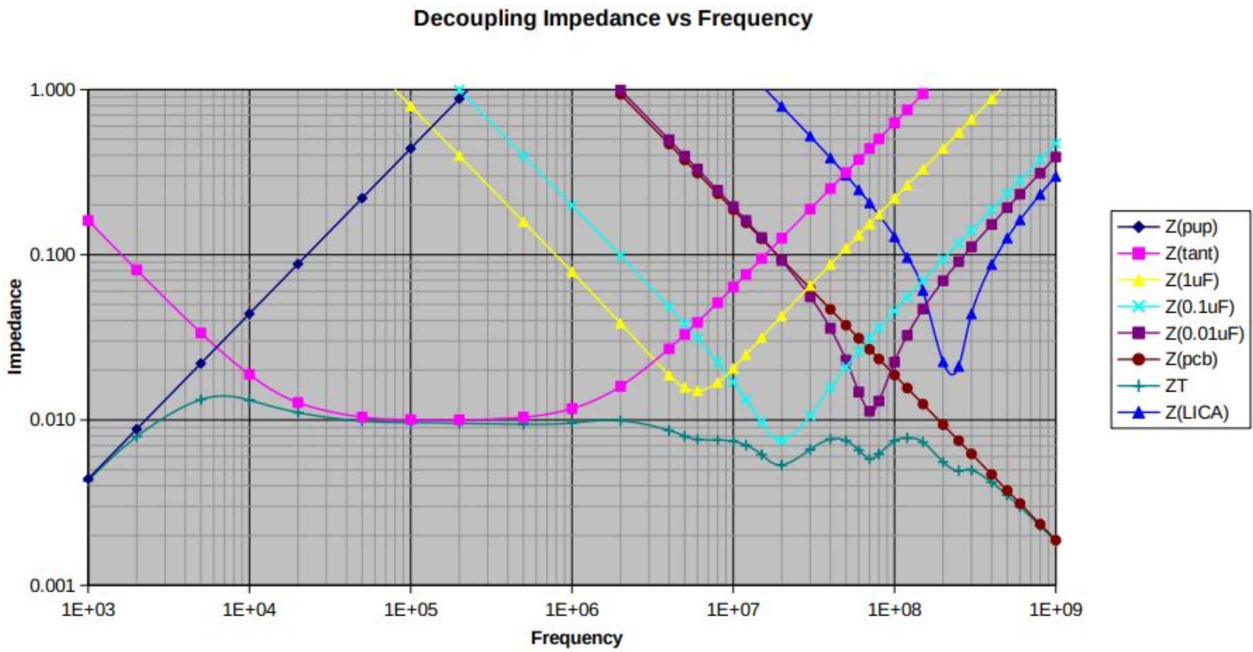
$$\sum_{i=0}^{\#states} time_i \times power_{i\,avg}$$

Components with large power consumption

- Displays
  - Fluorescent tubes
  - OLEDs
- Wireless interfaces
  - Cellular
  - WiFi
  - Bluetooth
- CPU

# Power Integrity

- Goal is to reduce impedance
  The frequency bandwidth

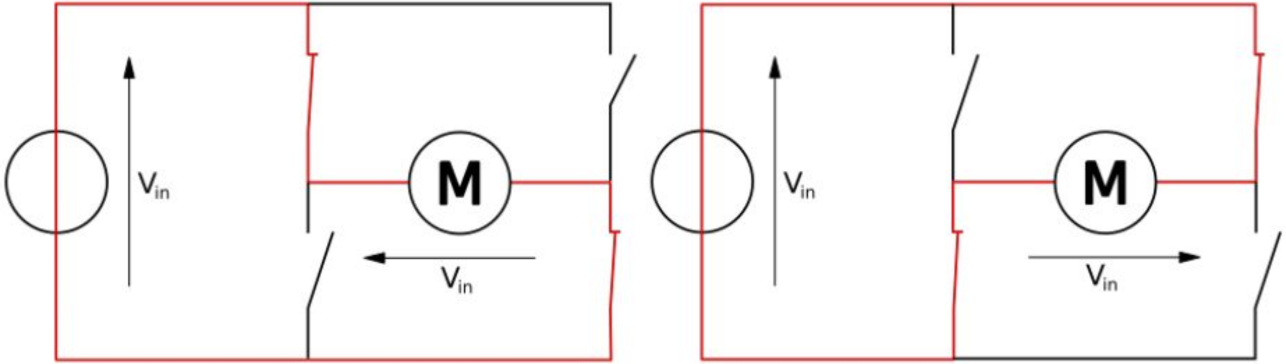


# Motors and driving them

- Solenoids (electromagnet-based actuator)
  - Fast, poor controllability, major E and EM noise source.
- DC Motors
  - General purpose usage (turn things).
  - Switch magnetic field polarity during turn.
    - Brushed uses mechanical solution (brushes).
    - Brushless needs a control circuit with sensing capability.
  - Issues with back-EMF.
- Stepper Motors
  - Control position at precise orientation
  - Use toothed magnets and have high torque when stationary.
- Servo Motors
  - Control position at precise (continuous orientation)

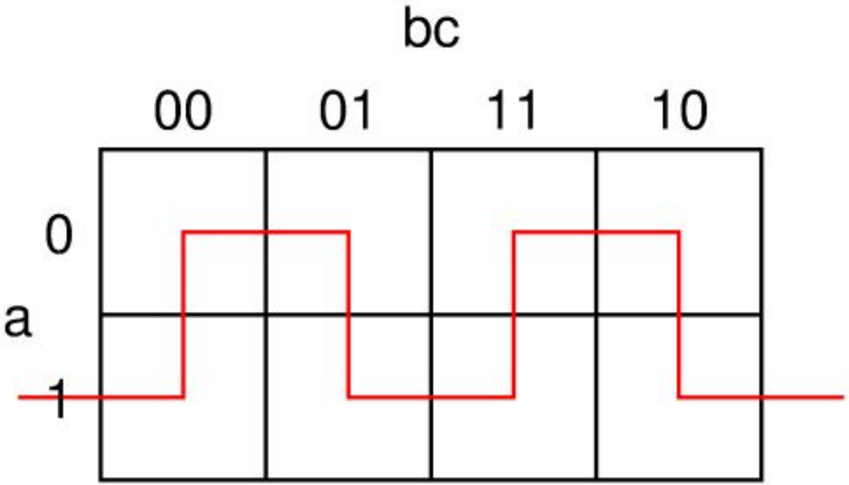
# Motors and driving them

- Linear Motors
  - Can be used for launching objects.
  - Lead screws and rotary motors are more common.
- H-Bridges
  - Control direction of current through device with BJTs or FETs. Use diodes to prevent damage.
  - Help with back-EMF.

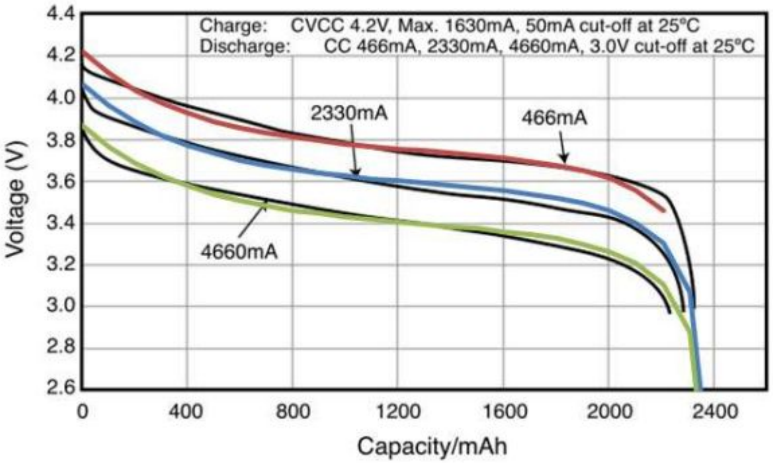


# Motors and driving them

- Shaft encoders
  - Used to determine relative or absolute position of a shaft
  - Linear arrangement of binary numbers
  - Can read using LED+photodetector

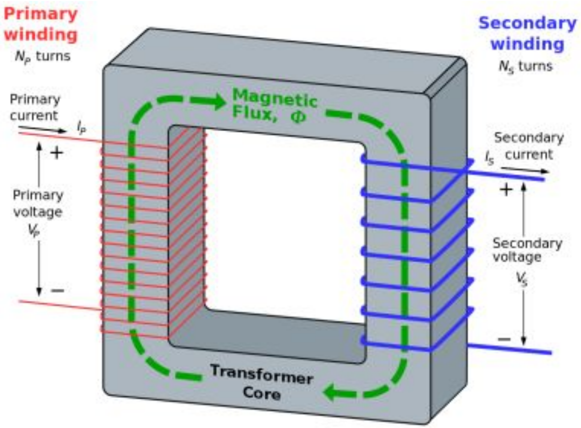


# Power Supplies

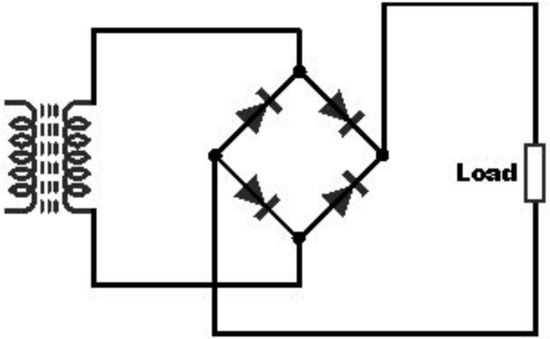


DC

AC - AC



AC - DC



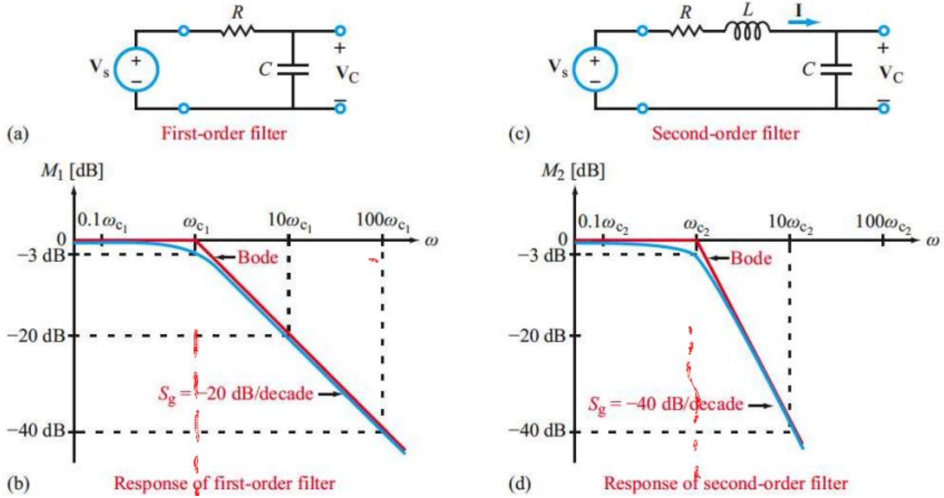
# Voltage Regulators

- Linear Voltage Regulator (step-down)
  - Stable, Inefficient for large voltage
- Charge pump DC-DC (boost)
  - Charges and discharges a Capacitor
- Buck switching DC-DC (step-down)
  - Efficient, Step-down only
- Buck-boost switching DC-DC (boost, step-down)
  - Efficient, Inverting
- No Regulator
  - Don't always need regulator

# Signal Conditioning

Often times need to condition signals

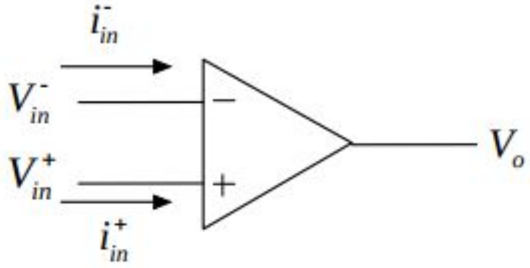
- Problems with many sensor outputs
  - High internal resistance
  - Voltage range mismatch
  - Unwanted frequencies
  - Fluctuating near-DC offset
- Solutions
  - Low-pass/high-pass/notch filters
  - Amplifiers


(a) First-order filter

(c) Second-order filter

(b) Response of first-order filter

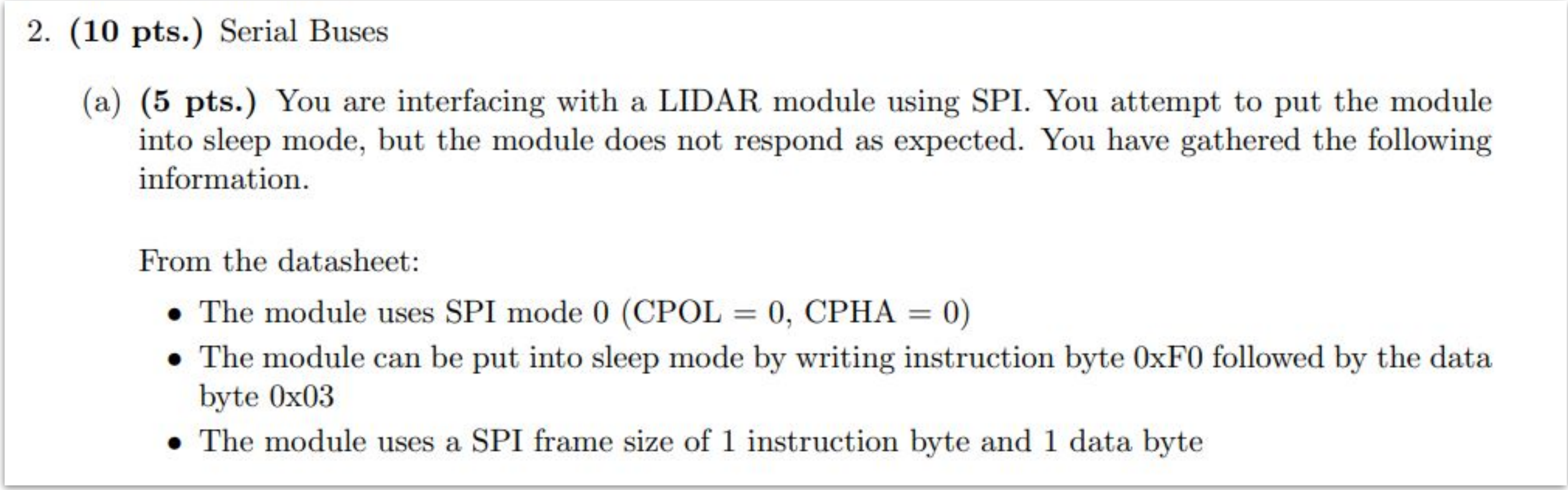
(d) Response of second-order filter



# Questions about content?

onto practice problems…

# Serial Communication: Fall 17 (2A) Q2



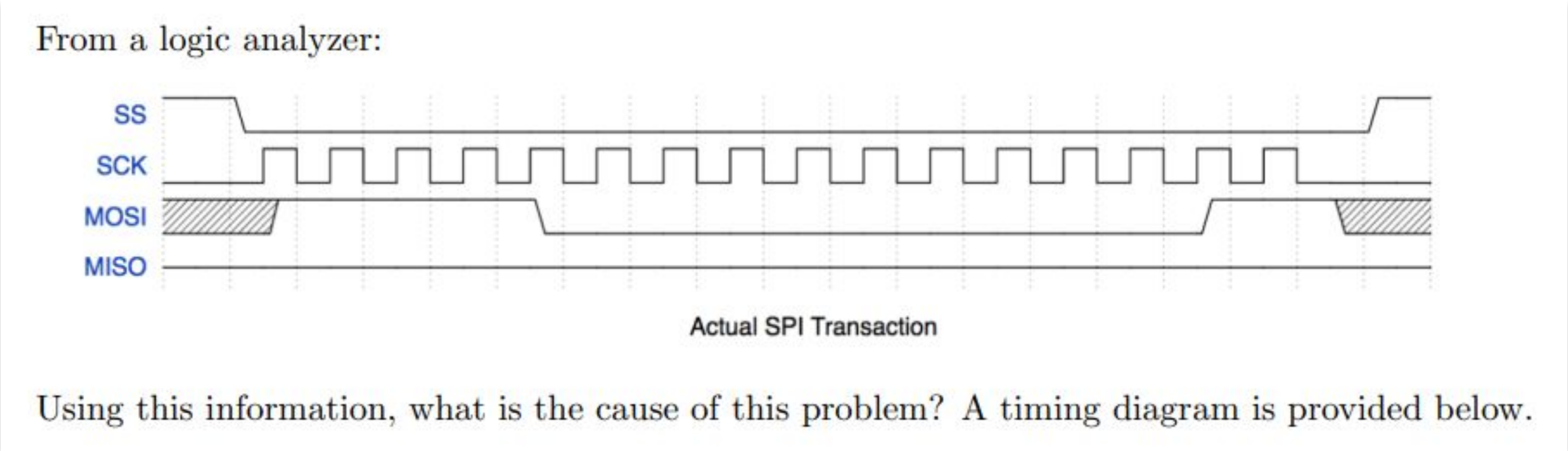
2. **(10 pts.)** Serial Buses

(a) **(5 pts.)** You are interfacing with a LIDAR module using SPI. You attempt to put the module into sleep mode, but the module does not respond as expected. You have gathered the following information.

From the datasheet:

- The module uses SPI mode 0 (CPOL = 0, CPHA = 0)
- The module can be put into sleep mode by writing instruction byte 0xF0 followed by the data byte 0x03
- The module uses a SPI frame size of 1 instruction byte and 1 data byte

# Serial Communication: Fall 17 (2A) Q2

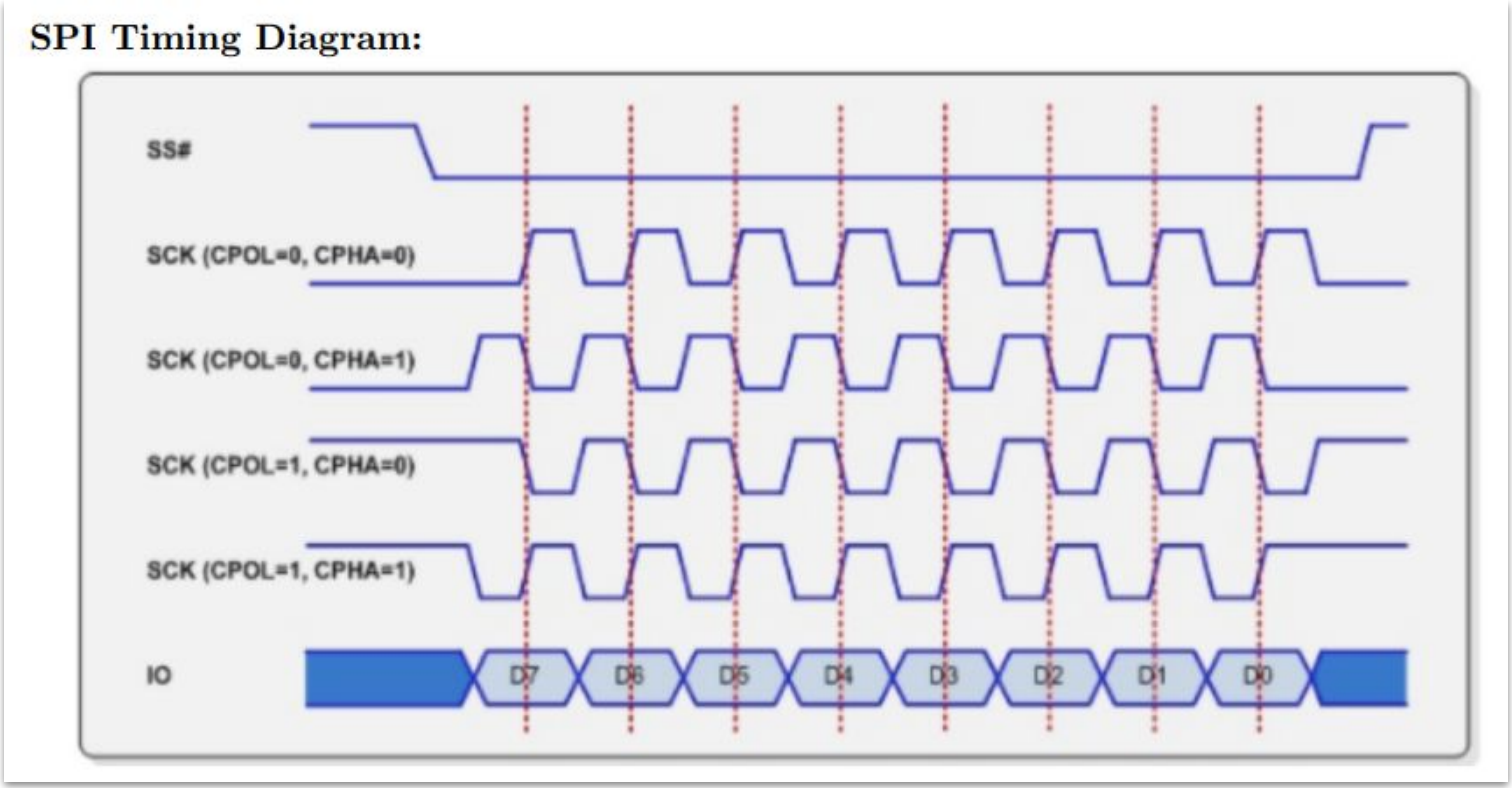


From a logic analyzer:

Actual SPI Transaction

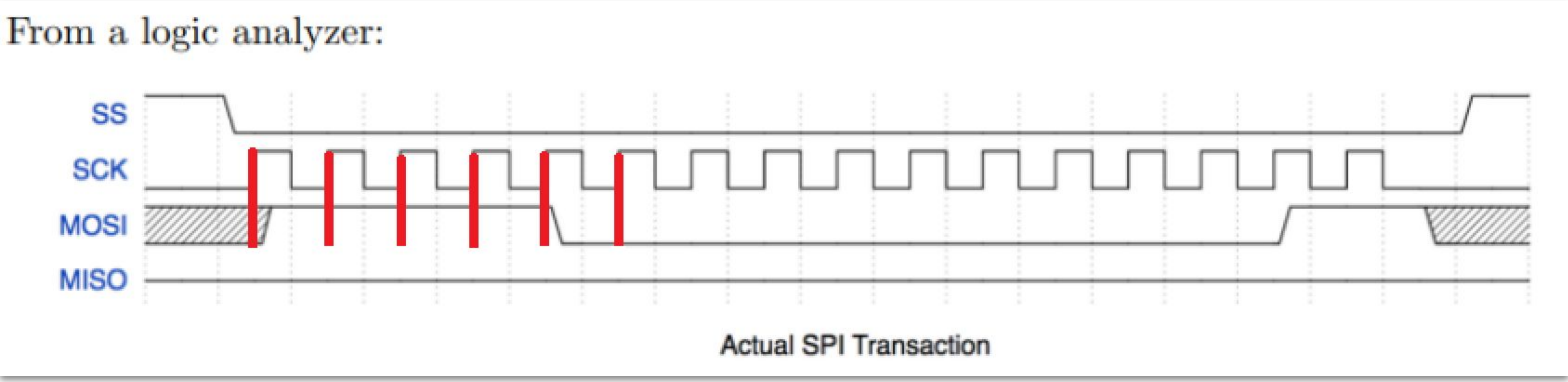
Using this information, what is the cause of this problem? A timing diagram is provided below.

# Serial Communication: Fall 17 (2A) Q2



SPI Timing Diagram:

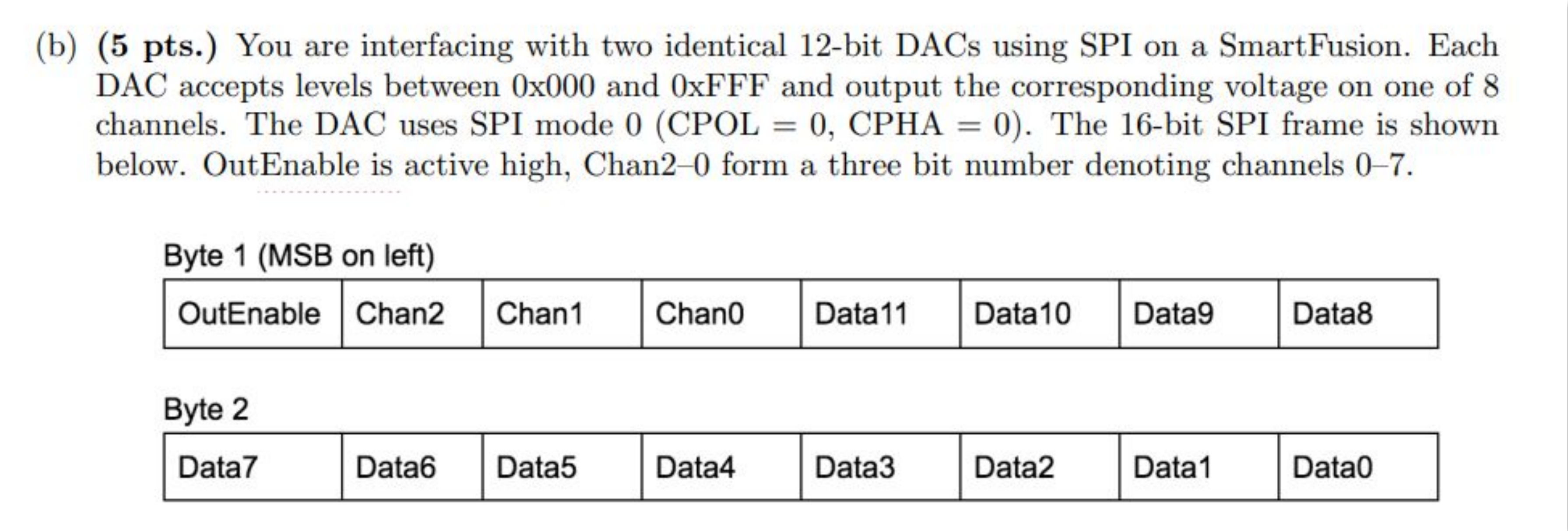
# Serial Communication: Fall 17 (2A) Q2



From a logic analyzer:

Actual SPI Transaction

# Serial Communication: Fall 17 (2A) Q2



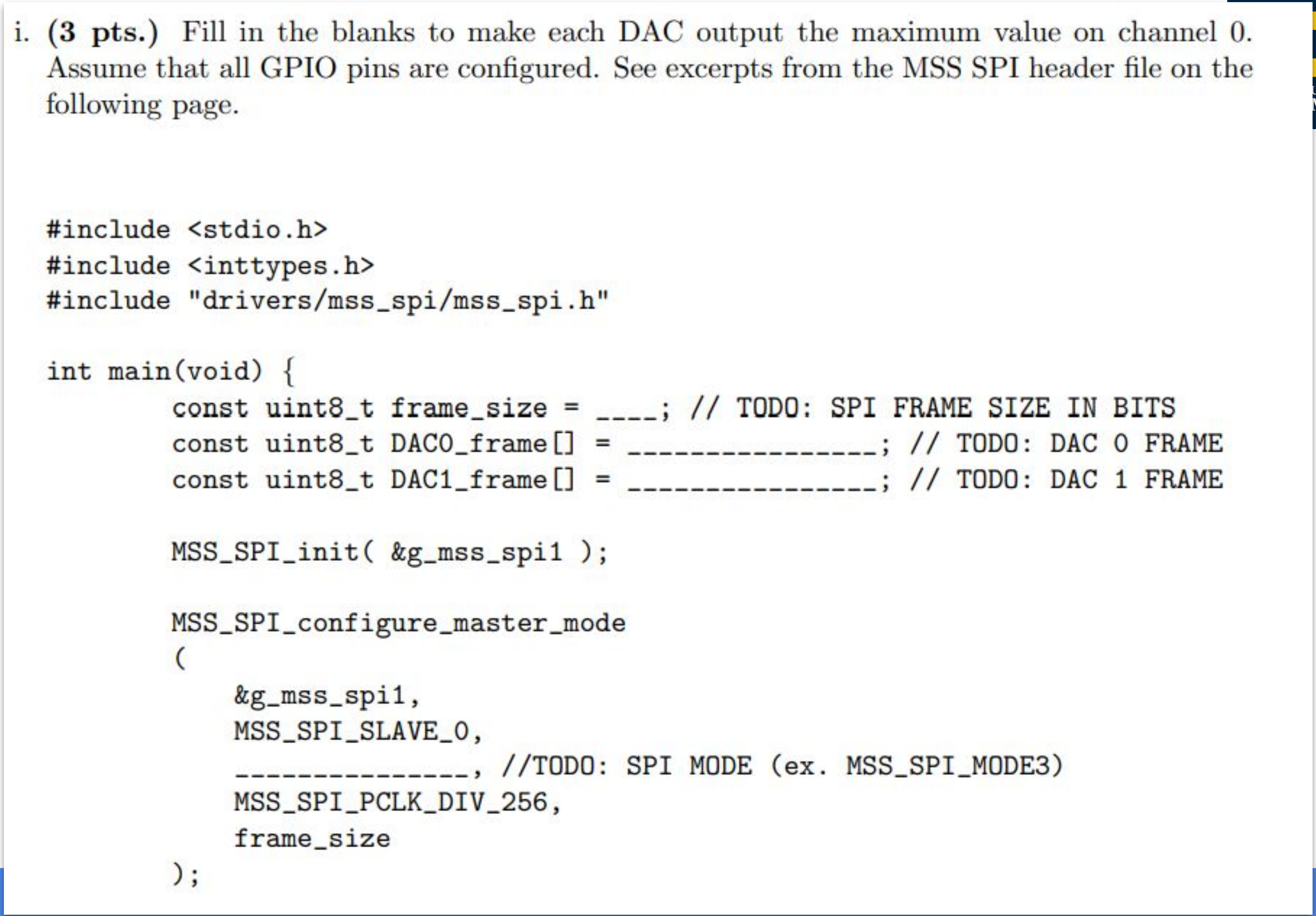
(b) **(5 pts.)** You are interfacing with two identical 12-bit DACs using SPI on a SmartFusion. Each DAC accepts levels between 0x000 and 0xFFF and output the corresponding voltage on one of 8 channels. The DAC uses SPI mode 0 (CPOL = 0, CPHA = 0). The 16-bit SPI frame is shown below. OutEnable is active high, Chan2–0 form a three bit number denoting channels 0–7.

Byte 1 (MSB on left)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| OutEnable | Chan2 | Chan1 | Chan0 | Data11 | Data10 | Data9 | Data8 |

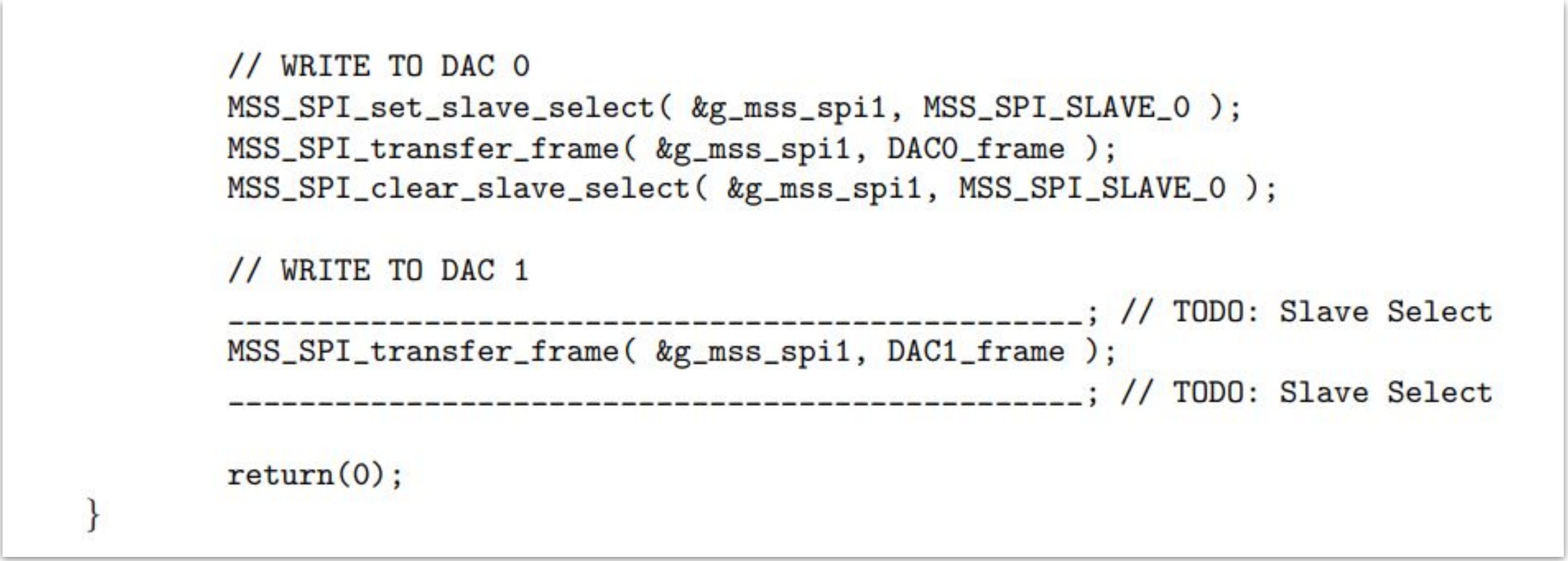
Byte 2

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Data7 | Data6 | Data5 | Data4 | Data3 | Data2 | Data1 | Data0 |

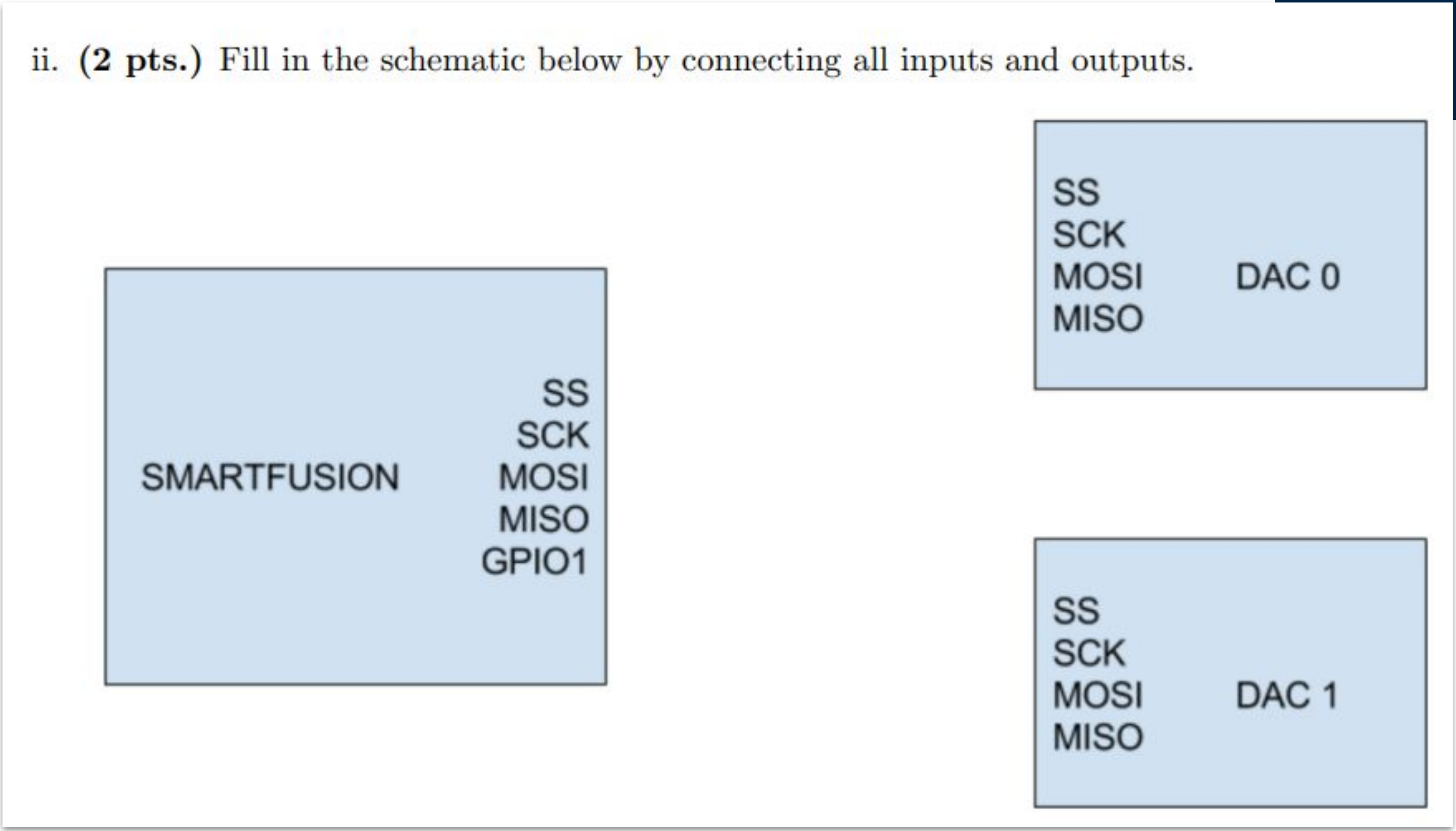


i. **(3 pts.)** Fill in the blanks to make each DAC output the maximum value on channel 0. Assume that all GPIO pins are configured. See excerpts from the MSS SPI header file on the following page.

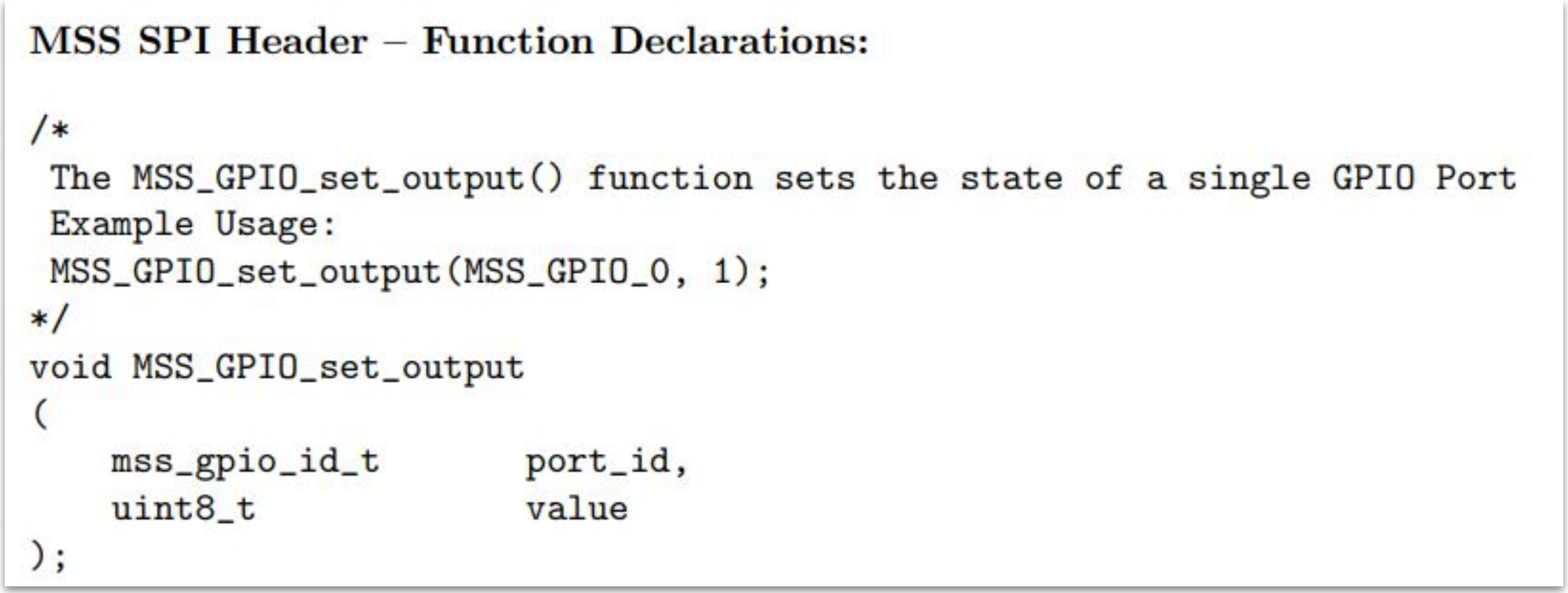
```
#include <stdio.h>
#include <inttypes.h>
#include "drivers/mss_spi/mss_spi.h"

int main(void) {
        const uint8_t frame_size = ____; // TODO: SPI FRAME SIZE IN BITS
        const uint8_t DAC0_frame[] = ________________; // TODO: DAC 0 FRAME
        const uint8_t DAC1_frame[] = ________________; // TODO: DAC 1 FRAME

        MSS_SPI_init( &g_mss_spi1 );

        MSS_SPI_configure_master_mode
        (
            &g_mss_spi1,
            MSS_SPI_SLAVE_0,
            _______________, //TODO: SPI MODE (ex. MSS_SPI_MODE3)
            MSS_SPI_PCLK_DIV_256,
            frame_size
        );
```



```
        // WRITE TO DAC 0
        MSS_SPI_set_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );
        MSS_SPI_transfer_frame( &g_mss_spi1, DAC0_frame );
        MSS_SPI_clear_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );

        // WRITE TO DAC 1
        ________________________________________________; // TODO: Slave Select
        MSS_SPI_transfer_frame( &g_mss_spi1, DAC1_frame );
        ________________________________________________; // TODO: Slave Select

        return(0);
}
```

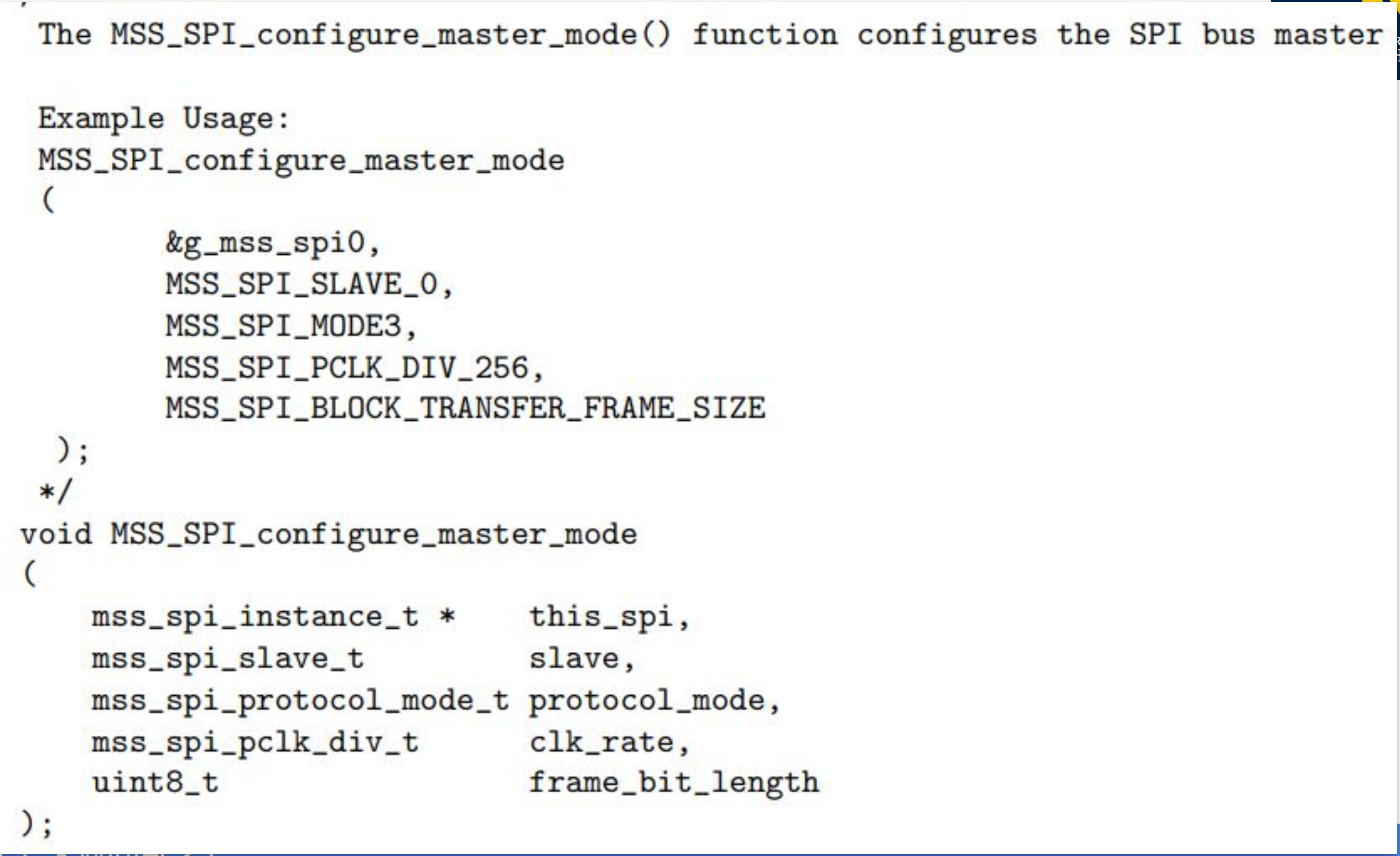


ii. **(2 pts.)** Fill in the schematic below by connecting all inputs and outputs.



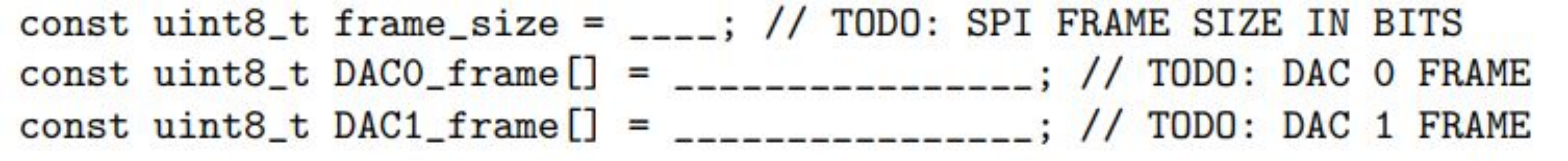
### MSS SPI Header – Function Declarations:

```
/*
 The MSS_GPIO_set_output() function sets the state of a single GPIO Port
 Example Usage:
 MSS_GPIO_set_output(MSS_GPIO_0, 1);
*/
void MSS_GPIO_set_output
(
    mss_gpio_id_t       port_id,
    uint8_t             value
);
```

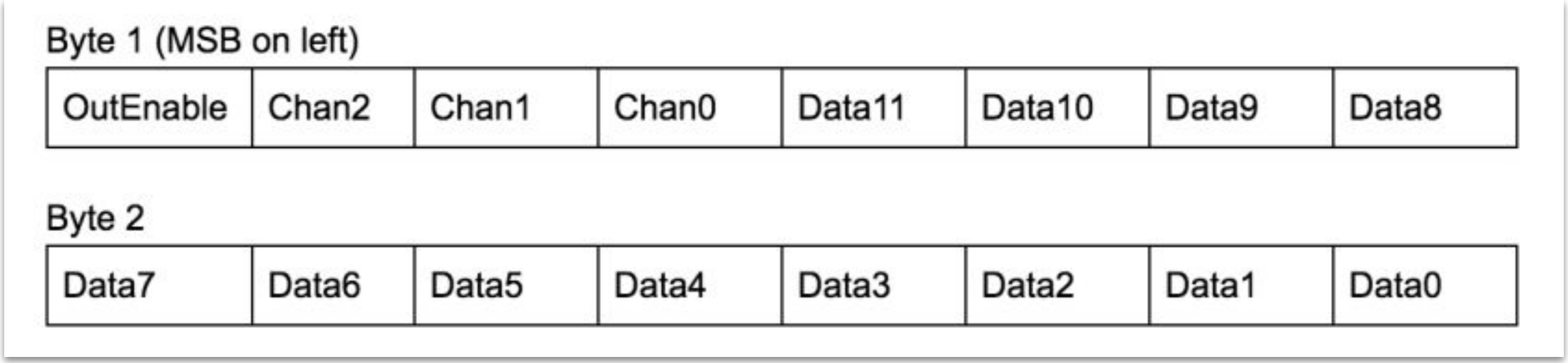


```
 The MSS_SPI_configure_master_mode() function configures the SPI bus master

 Example Usage:
 MSS_SPI_configure_master_mode
 (
        &g_mss_spi0,
        MSS_SPI_SLAVE_0,
        MSS_SPI_MODE3,
        MSS_SPI_PCLK_DIV_256,
        MSS_SPI_BLOCK_TRANSFER_FRAME_SIZE
  );
 */
void MSS_SPI_configure_master_mode
(
    mss_spi_instance_t *    this_spi,
    mss_spi_slave_t         slave,
    mss_spi_protocol_mode_t protocol_mode,
    mss_spi_pclk_div_t      clk_rate,
    uint8_t                 frame_bit_length
);
```

# Serial Communication: Fall 17 (2A) Q2



```
const uint8_t frame_size = ____; // TODO: SPI FRAME SIZE IN BITS
const uint8_t DAC0_frame[] = _________________; // TODO: DAC 0 FRAME
const uint8_t DAC1_frame[] = _________________; // TODO: DAC 1 FRAME
```



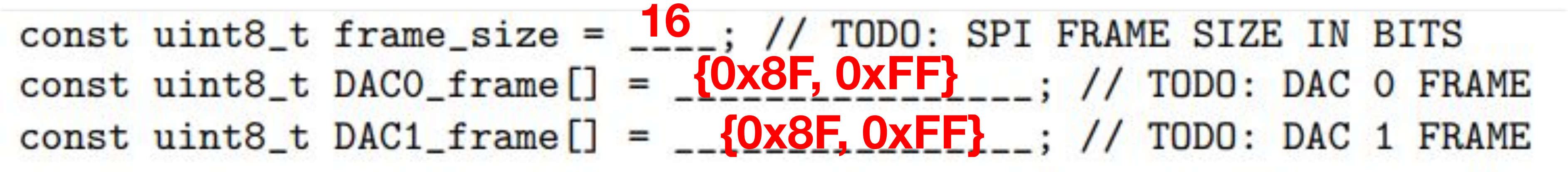
Byte 1 (MSB on left)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| OutEnable | Chan2 | Chan1 | Chan0 | Data11 | Data10 | Data9 | Data8 |

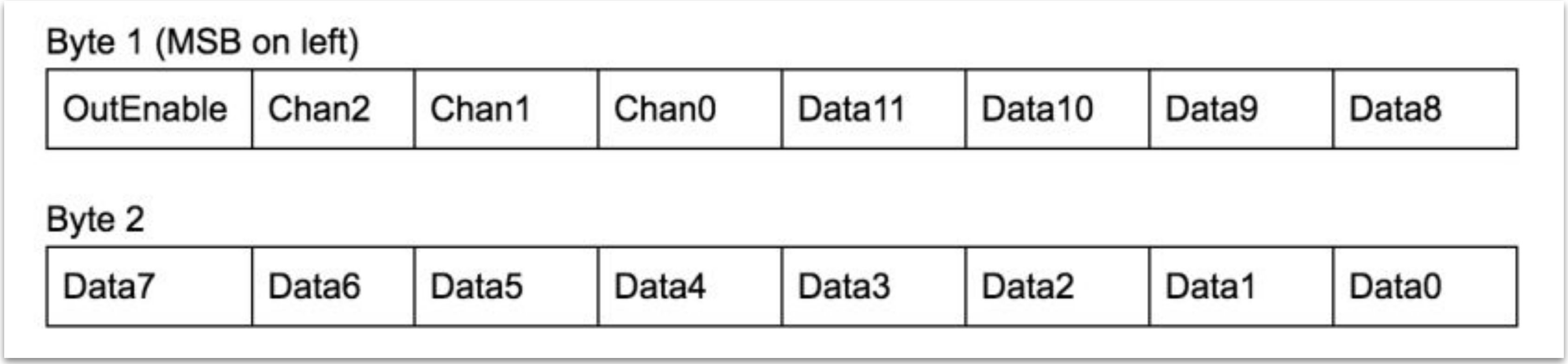
Byte 2

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Data7 | Data6 | Data5 | Data4 | Data3 | Data2 | Data1 | Data0 |

# Serial Communication: Fall 17 (2A) Q2



```
const uint8_t frame_size = __16__; // TODO: SPI FRAME SIZE IN BITS
const uint8_t DAC0_frame[] = _{0x8F, 0xFF}___; // TODO: DAC 0 FRAME
const uint8_t DAC1_frame[] = __{0x8F, 0xFF}__; // TODO: DAC 1 FRAME
```

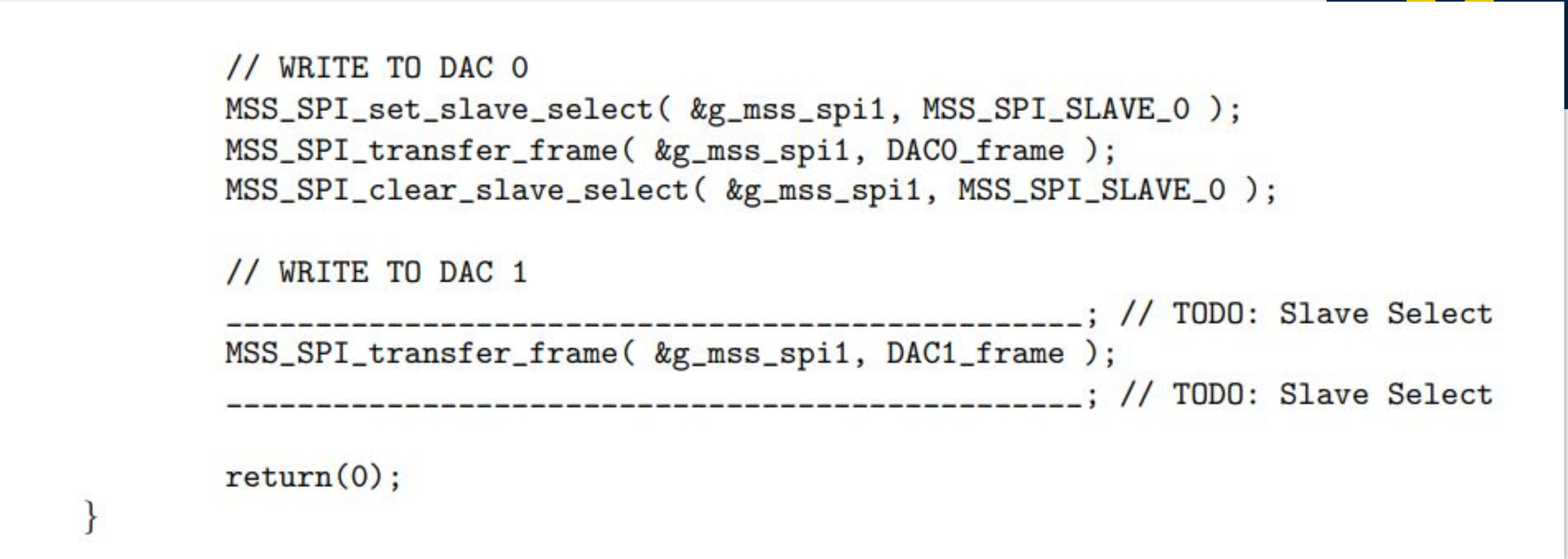


Byte 1 (MSB on left)

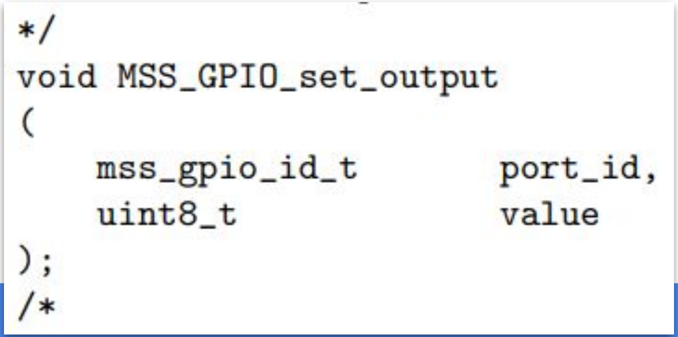
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| OutEnable | Chan2 | Chan1 | Chan0 | Data11 | Data10 | Data9 | Data8 |

Byte 2

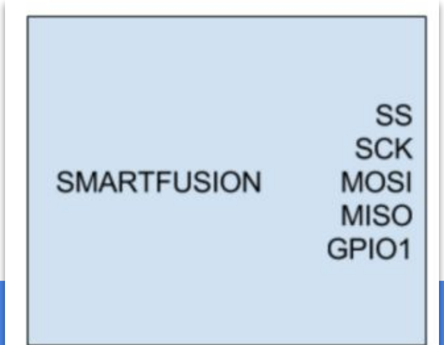
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Data7 | Data6 | Data5 | Data4 | Data3 | Data2 | Data1 | Data0 |

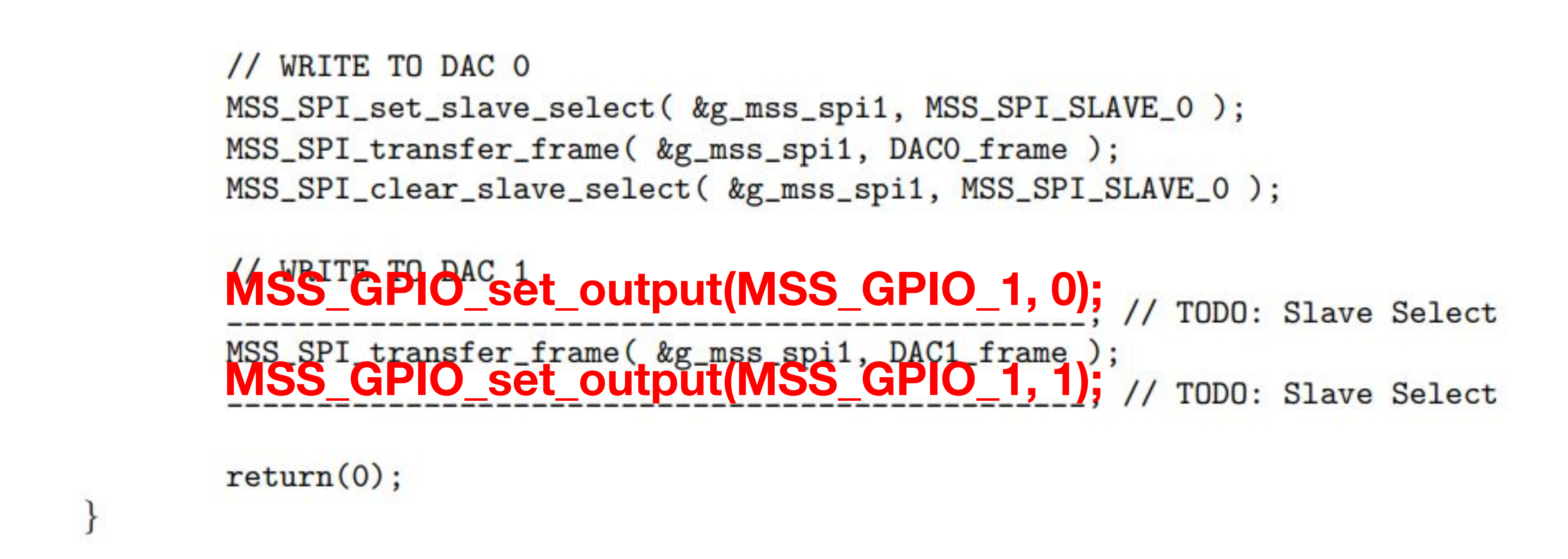


```
        // WRITE TO DAC 0
        MSS_SPI_set_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );
        MSS_SPI_transfer_frame( &g_mss_spi1, DAC0_frame );
        MSS_SPI_clear_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );

        // WRITE TO DAC 1
        __________________________________________________; // TODO: Slave Select
        MSS_SPI_transfer_frame( &g_mss_spi1, DAC1_frame );
        __________________________________________________; // TODO: Slave Select

        return(0);
    }
```

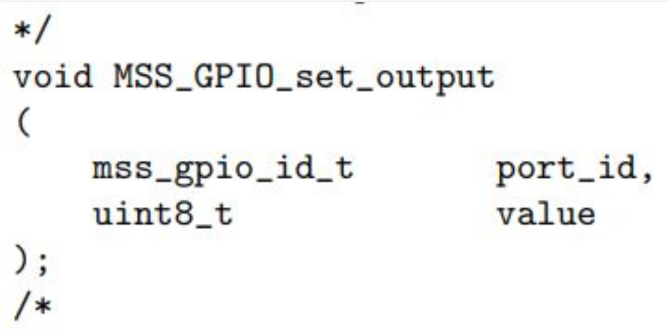


```
*/
void MSS_GPIO_set_output
(
    mss_gpio_id_t       port_id,
    uint8_t             value
);
/*
```

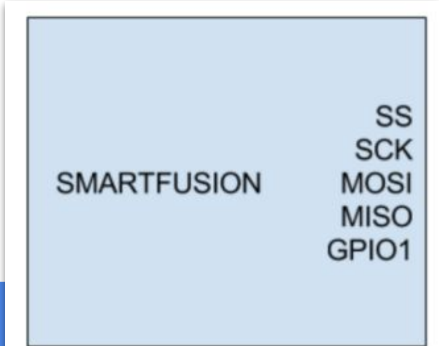


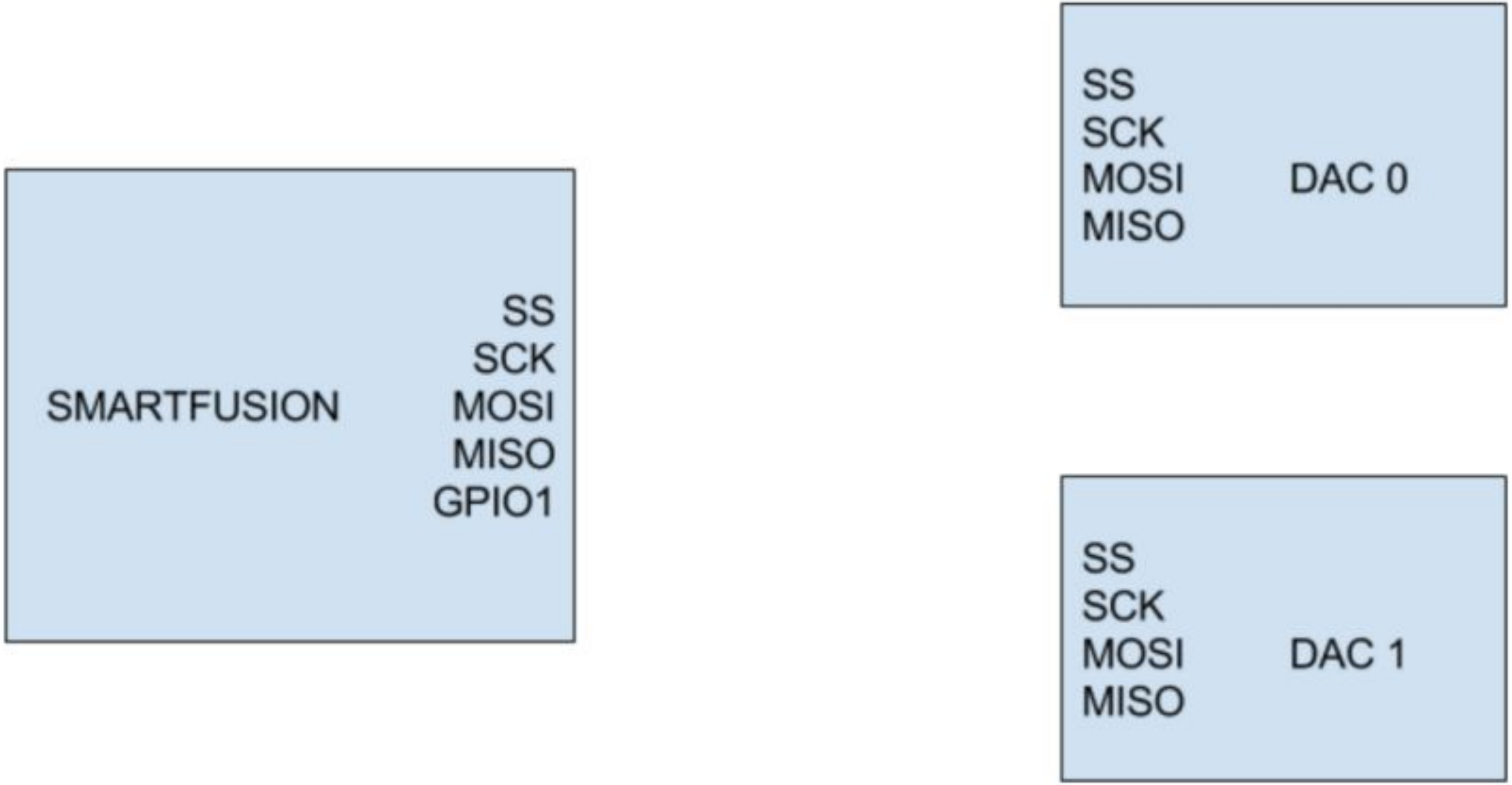


```
    // WRITE TO DAC 0
    MSS_SPI_set_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );
    MSS_SPI_transfer_frame( &g_mss_spi1, DAC0_frame );
    MSS_SPI_clear_slave_select( &g_mss_spi1, MSS_SPI_SLAVE_0 );

    // WRITE TO DAC 1
    MSS_GPIO_set_output(MSS_GPIO_1, 0);
    ____________________________________; // TODO: Slave Select
    MSS_SPI_transfer_frame( &g_mss_spi1, DAC1_frame );
    MSS_GPIO_set_output(MSS_GPIO_1, 1);
    ____________________________________; // TODO: Slave Select

    return(0);
}
```



```
*/
void MSS_GPIO_set_output
(
    mss_gpio_id_t       port_id,
    uint8_t             value
);
/*
```





SMARTFUSION

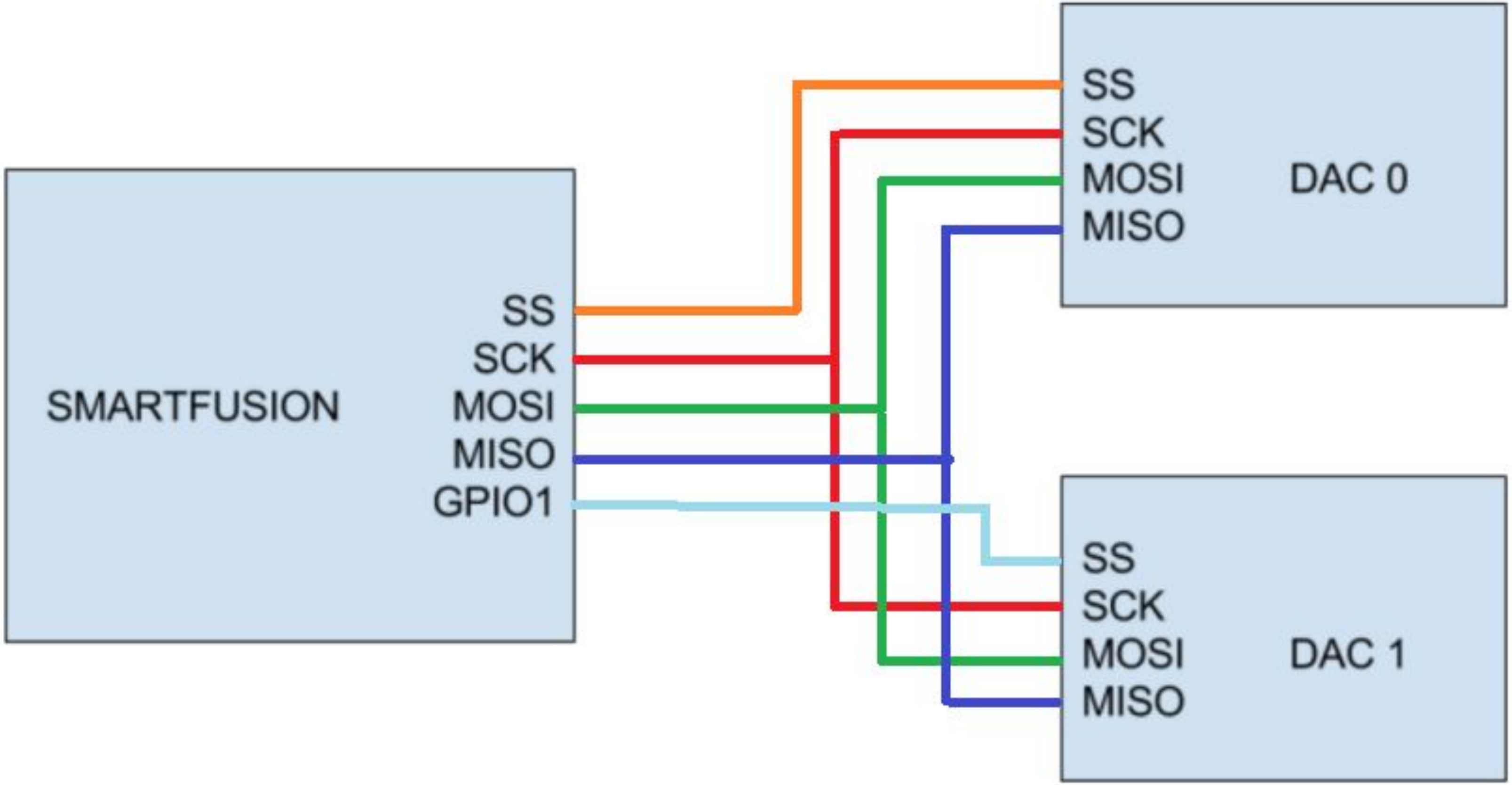
SS
SCK
MOSI
MISO
GPIO1

SS
SCK
MOSI
MISO

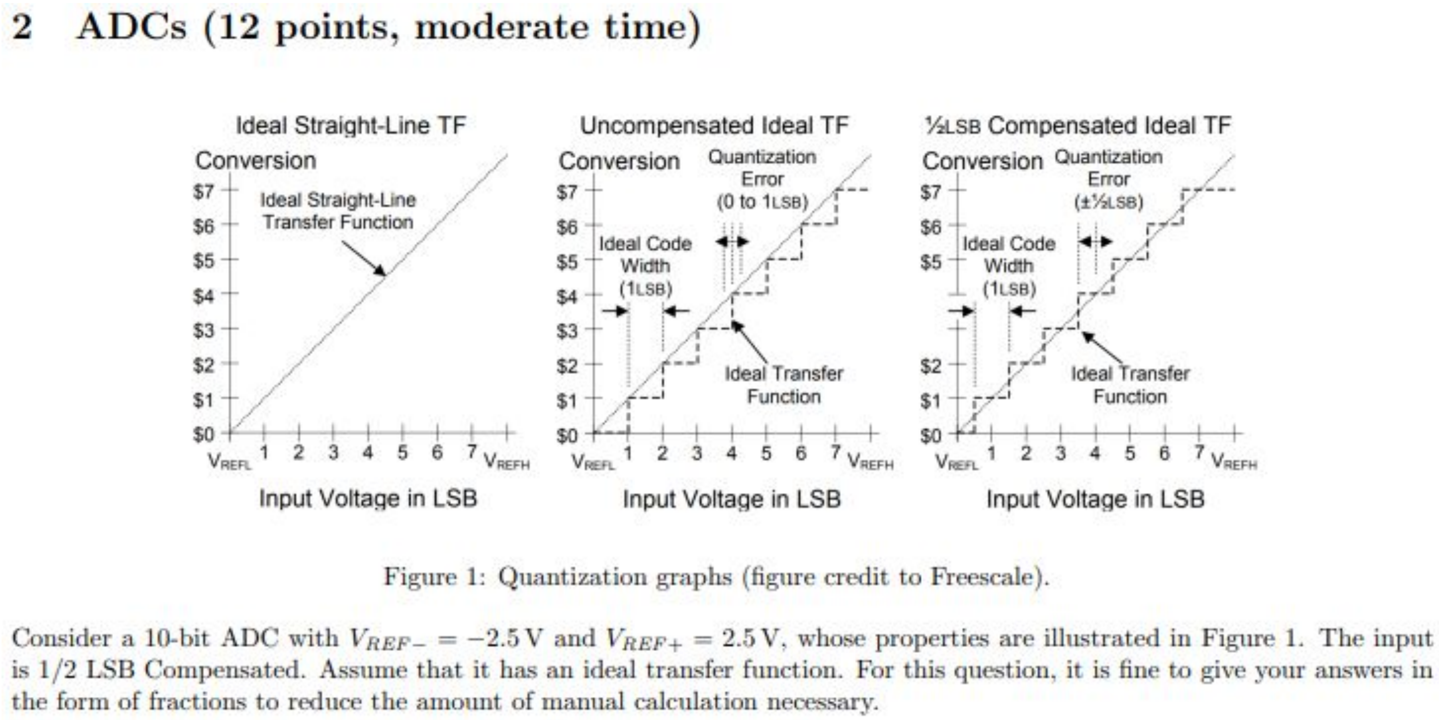
DAC 0

SS
SCK
MOSI
MISO

DAC 1


SMARTFUSION
SS
SCK
MOSI
MISO
GPIO1
SS
SCK
MOSI
MISO
DAC 0
SS
SCK
MOSI
MISO
DAC 1

# ADCs: Winter 22 Q2



## 2 ADCs (12 points, moderate time)

Ideal Straight-Line TF
Conversion
$7
$6
$5
$4
$3
$2
$1
$0
Ideal Straight-Line Transfer Function
VREFL
1
2
3
4
5
6
7
VREFH
Input Voltage in LSB
Uncompensated Ideal TF
Conversion
Quantization Error (0 to 1LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
Input Voltage in LSB
½LSB Compensated Ideal TF
Conversion
Quantization Error (±½LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
Input Voltage in LSB

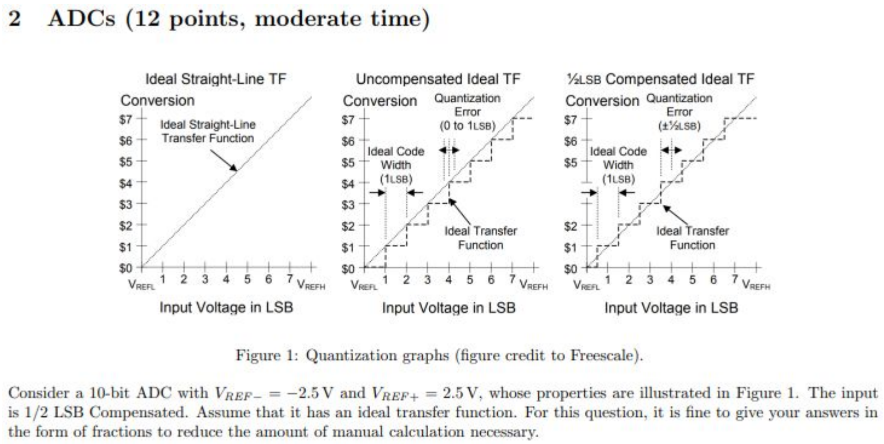
Figure 1: Quantization graphs (figure credit to Freescale).

Consider a 10-bit ADC with $V_{REF-} = -2.5\,\text{V}$ and $V_{REF+} = 2.5\,\text{V}$, whose properties are illustrated in Figure 1. The input is 1/2 LSB Compensated. Assume that it has an ideal transfer function. For this question, it is fine to give your answers in the form of fractions to reduce the amount of manual calculation necessary.

# ADCs: Winter 22 Q2

1. What is its maximum quantization error (In volts)? (6 points)

$( V_{ref+} - V_{ref-} ) / (2 * 2^{10}) =$ **2.441 mV**



**2 ADCs (12 points, moderate time)**

Ideal Straight-Line TF
Conversion
$7
$6
$5
$4
$3
$2
$1
$0
Ideal Straight-Line Transfer Function
VREFL 1 2 3 4 5 6 7 VREFH
Input Voltage in LSB
Uncompensated Ideal TF
Conversion
Quantization Error (0 to 1LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
VREFL 1 2 3 4 5 6 7 VREFH
Input Voltage in LSB
½LSB Compensated Ideal TF
Conversion
Quantization Error (±½LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
VREFL 1 2 3 4 5 6 7 VREFH
Input Voltage in LSB

Figure 1: Quantization graphs (figure credit to Freescale).

Consider a 10-bit ADC with $V_{REF-} = -2.5\,\mathrm{V}$ and $V_{REF+} = 2.5\,\mathrm{V}$, whose properties are illustrated in Figure 1. The input is 1/2 LSB Compensated. Assume that it has an ideal transfer function. For this question, it is fine to give your answers in the form of fractions to reduce the amount of manual calculation necessary.

# ADCs: Winter 22 Q2

2. What voltage would correspond with the ADC value of 377 (decimal)? (6 points)

$$V = N_{adc} \cdot \frac{V_{REF+} - V_{REF-}}{2^n} + VREF- \quad (1)$$

$$= \frac{377 \cdot 5\,\mathrm{V}}{2^{1}0} - 2.5\,\mathrm{V} \quad (2)$$

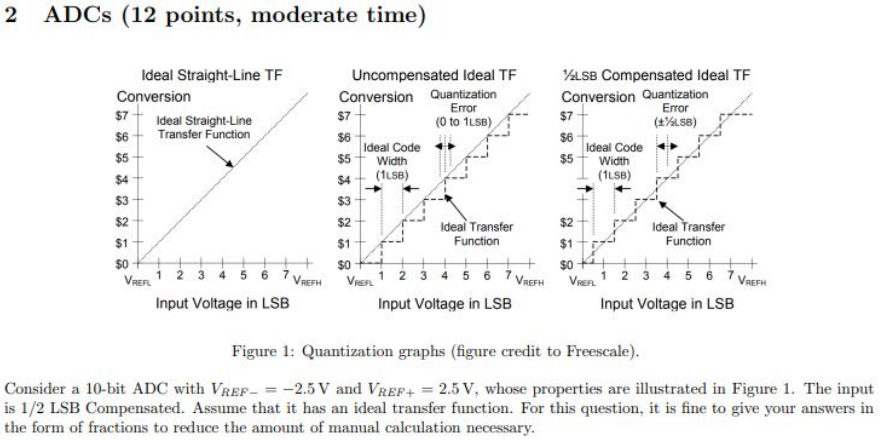
$$= \frac{377 \cdot 5\,\mathrm{V}}{1024} - 2.5\,\mathrm{V} \quad (3)$$

$$= \frac{377 \cdot 5\,\mathrm{V}}{1024} - 2.5\,\mathrm{V} \quad (4)$$

$$= 1885/1024\,\mathrm{V} - 2.5\,\mathrm{V} \quad (5)$$

$$= -675/1024\,\mathrm{V} \quad (6)$$

$$= -0.65917\,\mathrm{V} \quad (7)$$



## 2 ADCs (12 points, moderate time)

Ideal Straight-Line TF
Conversion
Ideal Straight-Line Transfer Function
$7 $6 $5 $4 $3 $2 $1 $0
V_REFL 1 2 3 4 5 6 7 V_REFH
Input Voltage in LSB
Uncompensated Ideal TF
Conversion
Quantization Error (0 to 1LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
Input Voltage in LSB
½LSB Compensated Ideal TF
Conversion
Quantization Error (±½LSB)
Ideal Code Width (1LSB)
Ideal Transfer Function
Input Voltage in LSB

Figure 1: Quantization graphs (figure credit to Freescale).

Consider a 10-bit ADC with $V_{REF-} = -2.5\,\mathrm{V}$ and $V_{REF+} = 2.5\,\mathrm{V}$, whose properties are illustrated in Figure 1. The input is 1/2 LSB Compensated. Assume that it has an ideal transfer function. For this question, it is fine to give your answers in the form of fractions to reduce the amount of manual calculation necessary.

# Memory: Winter 22 Q4

## 4 Memory (10 points, short time)

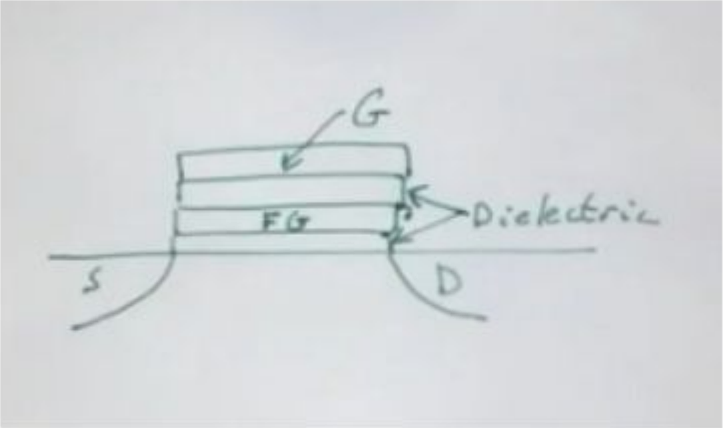
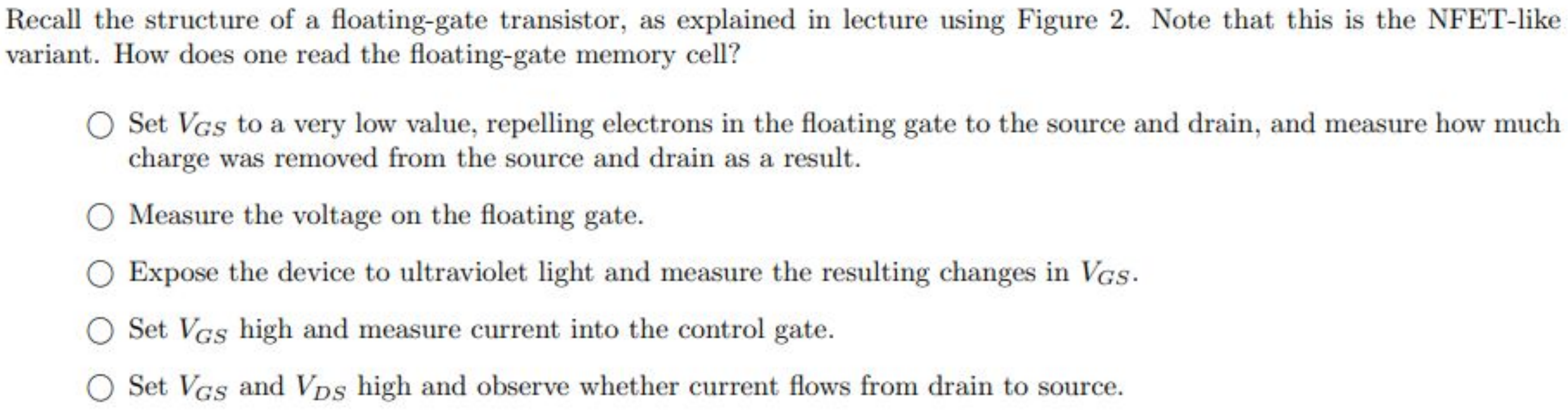


Figure 2: Floating gate transistor.

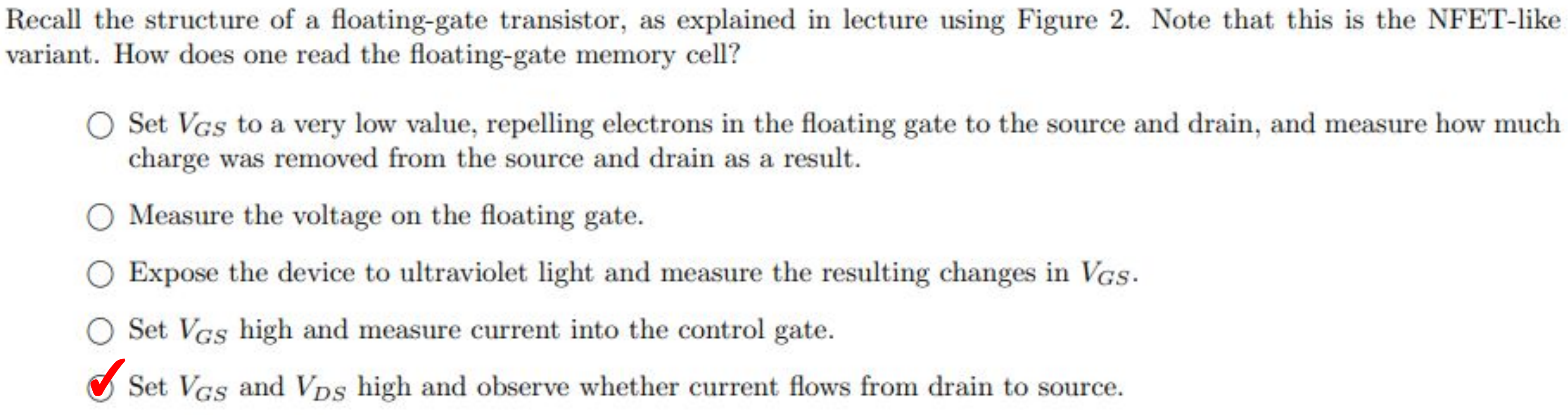
# Memory: Winter 22 Q4



Recall the structure of a floating-gate transistor, as explained in lecture using Figure 2. Note that this is the NFET-like variant. How does one read the floating-gate memory cell?

- ○ Set $V_{GS}$ to a very low value, repelling electrons in the floating gate to the source and drain, and measure how much charge was removed from the source and drain as a result.
- ○ Measure the voltage on the floating gate.
- ○ Expose the device to ultraviolet light and measure the resulting changes in $V_{GS}$.
- ○ Set $V_{GS}$ high and measure current into the control gate.
- ○ Set $V_{GS}$ and $V_{DS}$ high and observe whether current flows from drain to source.

# Memory: Winter 22 Q4



Recall the structure of a floating-gate transistor, as explained in lecture using Figure 2. Note that this is the NFET-like variant. How does one read the floating-gate memory cell?

○ Set $V_{GS}$ to a very low value, repelling electrons in the floating gate to the source and drain, and measure how much charge was removed from the source and drain as a result.

○ Measure the voltage on the floating gate.

○ Expose the device to ultraviolet light and measure the resulting changes in $V_{GS}$.

○ Set $V_{GS}$ high and measure current into the control gate.

[✓ Set $V_{GS}$ and $V_{DS}$ high and observe whether current flows from drain to source.]

# PCB: Winter 22 Q7

**Indicate the PCB components associated with the following (somewhat cryptic) definitions.**

1. Electronic devices that sit atop the PCB (2 point).

2. Copper tracks on the surface of the PCB (2 point).

3. A copper layer almost entirely covering a layer of the PCB (2 point).

4. Small copper-plated holes through the layers of the PCB (2 point).

5. Electronic devices whose pins penetrate holes in the PCB (2 point).

○ traces ○ vias ○ surface-mount ○ through-hole ○ ground/power plane

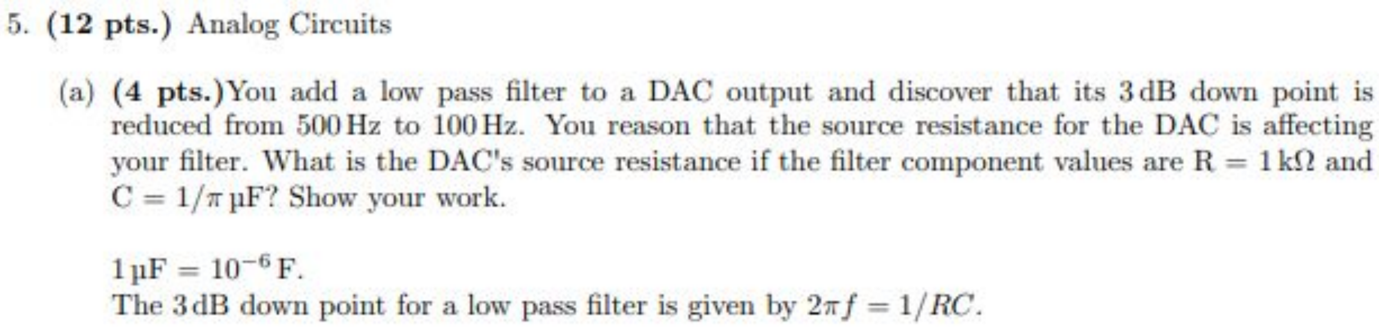
# (Answer) PCB: Winter 22 Q7

**Indicate the PCB components associated with the following (somewhat cryptic) definitions.**

1. Electronic devices that sit atop the PCB (2 point). **surface-mount**
2. Copper tracks on the surface of the PCB (2 point). **traces**
3. A copper layer almost entirely covering a layer of the PCB (2 point). **ground/power plane**
4. Small copper-plated holes through the layers of the PCB (2 point). **vias**
5. Electronic devices whose pins penetrate holes in the PCB (2 point). **through-hole**

○ traces ○ vias ○ surface-mount ○ through-hole ○ ground/power plane

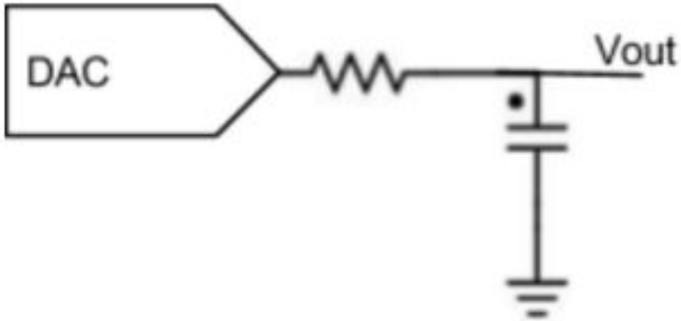
# Analog Interface Circuits: Fall 17 Q5



5. **(12 pts.)** Analog Circuits

(a) **(4 pts.)**You add a low pass filter to a DAC output and discover that its 3 dB down point is reduced from 500 Hz to 100 Hz. You reason that the source resistance for the DAC is affecting your filter. What is the DAC's source resistance if the filter component values are R = 1 kΩ and C = $1/\pi$ µF? Show your work.

1 µF = $10^{-6}$ F.
The 3 dB down point for a low pass filter is given by $2\pi f = 1/RC$.



# Analog Interface Circuits: Fall 17 Q5

$f = 100$, $R_f = 1000$, $C = 1/\pi$ uF, $2\pi f = 1/R*C$, $R = R_f + R_{adc}$

$RC = 1 / 2\pi f$

$R_f + R_{adc} = 1/(C2\pi f)$

$R_{adc} = 1/(C2\pi f) - R_f$

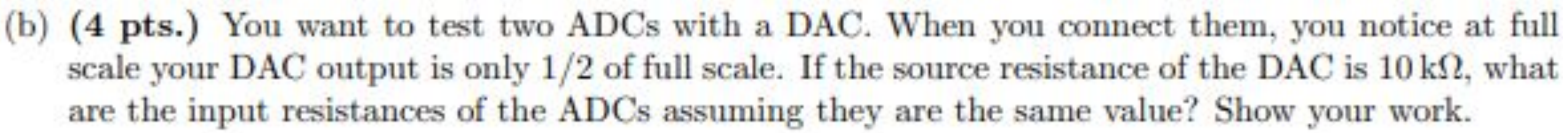
$R_{adc} = 1/((1/\pi) * 10^{-6} * 2 * 100) - 1000$

$R_{adc} = 1/(10^{-6} * 200) - 1000$

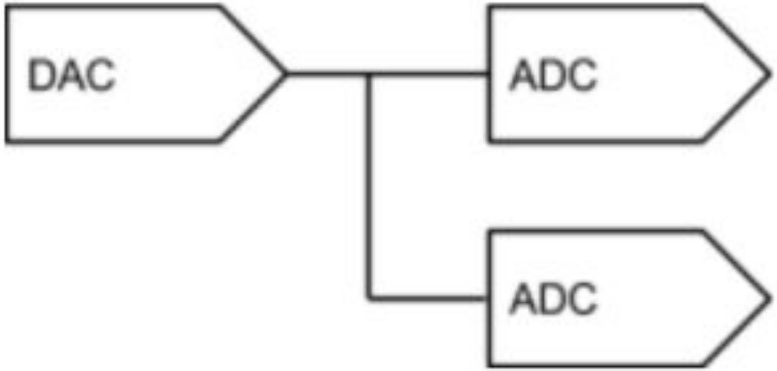
$R_{adc} = 5000 - 1000$

$R_{adc} = 4000$

# Analog Interface Circuits: Fall 17 Q5

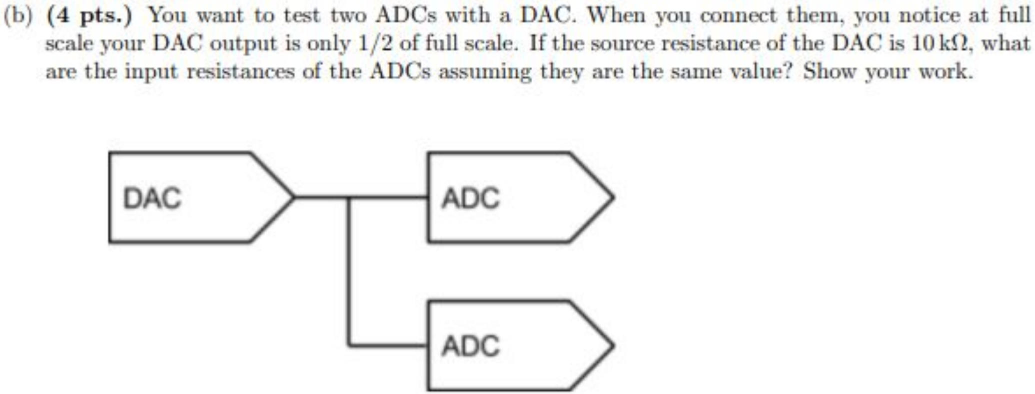


(b) **(4 pts.)** You want to test two ADCs with a DAC. When you connect them, you notice at full scale your DAC output is only 1/2 of full scale. If the source resistance of the DAC is 10 kΩ, what are the input resistances of the ADCs assuming they are the same value? Show your work.



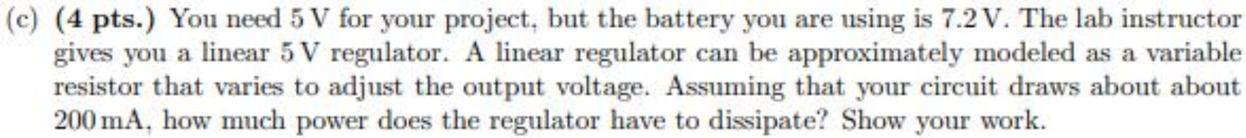
# Analog Interface Circuits: Fall 17 Q5

If voltage is half, $R_{adc1}$ || $R_{adc2}$ = $R_{dac}$ to satisfy voltage divider rule. So 10 kΩ = **20 kΩ || 20 kΩ**

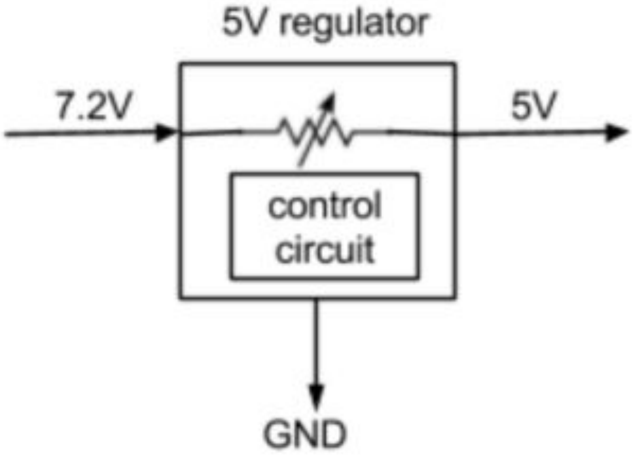


(b) **(4 pts.)** You want to test two ADCs with a DAC. When you connect them, you notice at full scale your DAC output is only 1/2 of full scale. If the source resistance of the DAC is 10 kΩ, what are the input resistances of the ADCs assuming they are the same value? Show your work.

# Analog Interface Circuits: Fall 17 Q5



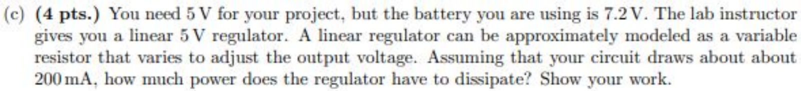
(c) **(4 pts.)** You need 5 V for your project, but the battery you are using is 7.2 V. The lab instructor gives you a linear 5 V regulator. A linear regulator can be approximately modeled as a variable resistor that varies to adjust the output voltage. Assuming that your circuit draws about about 200 mA, how much power does the regulator have to dissipate? Show your work.



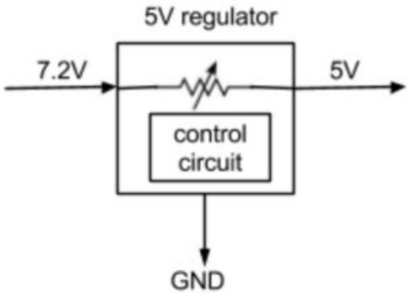
# Analog Interface Circuits: Fall 17 Q5

Power dissipated thru regulator is P = IV where V is voltage drop across regulator

P = (7.2 V - 5 V) * 200 mA = **440 mW**



(c) **(4 pts.)** You need 5 V for your project, but the battery you are using is 7.2 V. The lab instructor gives you a linear 5 V regulator. A linear regulator can be approximately modeled as a variable resistor that varies to adjust the output voltage. Assuming that your circuit draws about about 200 mA, how much power does the regulator have to dissipate? Show your work.



# Reminders

Midterm:

3 - 4:20 pm March 23rd

1010 DOW (A-L family name),

1311 EECS (M-Z family name),

1045 GGBL (SSD)

Optional Homework 3:

Wednesday, 11:59 am on 22 March