A

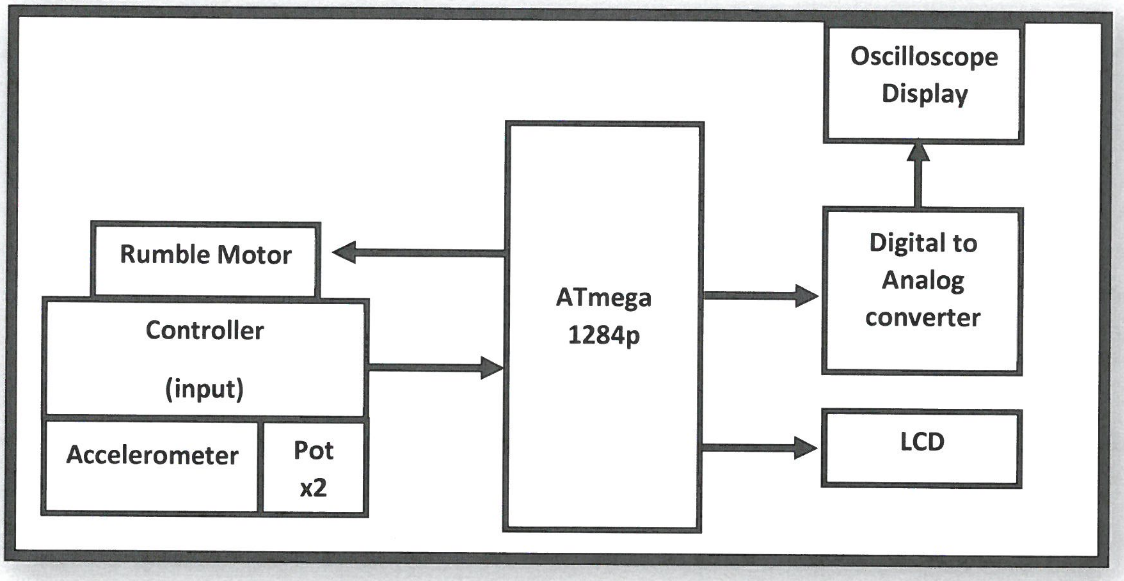
0129657118

Demonstrate understanding of complex concepts used in the design and construction of electronic environments

Specifications

My proposed design for an interactive device which uses an Oscilloscope as its display with feature hardware and software concepts to achieve its intended function efficiently. The device will feature one video-game-esque controller with rumble motors for input. An oscilloscope thru a DAC and a LCD screen being controlled by an ATmega AVR chip for output.

Schematic



Complex Hardware Concepts:

DC Motor Interface (tick)

Liquid Crystal Display (tick)

Multiple Sensors (Potentiometers, Accelerometer) (tick)

Microcontrollers (tick)

Complex Software Concepts:

Structuring programs logically (tick)

Digital to analog conversion (tick)

Interrupts (tick)

Counters (tick)

NZQA Assessed

Software Concepts 1: Digital to Analog Conversion

Concept: Using a digital device to product an analog signal to then be interpreted by an analog device.

Because the microcontroller is a digital device and I want to use an oscilloscope to display analog data, I need a way to convert the digital signals produced by the microcontroller to an analog signal that can be nicely displayed on an oscilloscope display with no stutter.

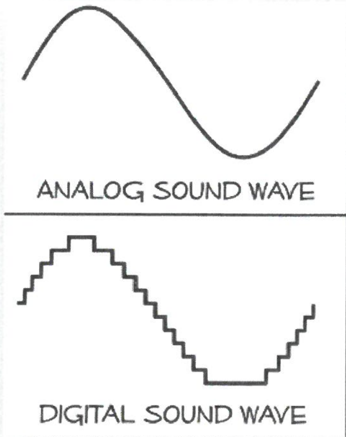


Figure 1: True Analog vs Digital "Analog"

On the right is an example of a sine curve produced by a truly analog device and also the same curve produced by a digital device. The digital device is using some sort of Digital to Analog conversion because a true digital signal is entirely represented by either an on or an off state. The digital signal in figure 1 is most likely being produced using an "R2R Ladder".

There is 2 common types of Digital to Analog conversion, R2R ladders and Pulse Width Modulation (PWM). Each have advantages and disadvantages but the most common type you would find is PWM. PWM is is where a voltage is still truly binary but by changing the delay between each pulse and the duration of the pulse, it gives the illusion with some components that a signal is analog. For example with an LED, it takes a small amount of time for the diode to become completely lit. The PWM exploits this by having the pulse have a longer delay so making the LED appear to have a lessened brightness.

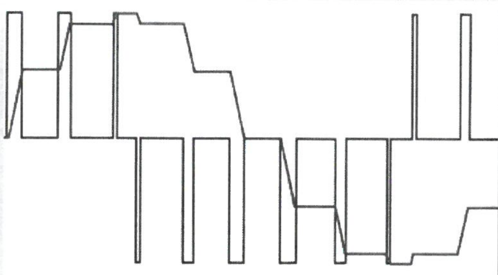


Figure 2: PWM Faking a sine wave

An R2R ladder uses resistors in parallel connected to a true binary digital signal. The circuit is actually a voltage adding circuit where the maximum is the sum of all the binary outputs on maximum. Thus, by turning some of the output off, it allows the circuit to simulate an analog signal, as shown in Figure 1. The more resistors you have, allows the "resolution" of the analog signal to have higher detail.

A digital to analog IC typically is a hybrid of these two, mostly with it using the R2R ladder variant with a lot more internal resistors to have highest amount of detail. This is usually the method used to convert digital audio to an analog signal.

0129657118

Software Concepts 2: Structuring programs logically

Concept: Optimizing code and making it easier to read by splitting it up in to smaller pieces.

When it comes to programming, there isn't really a set way you have to write it, you can write it any way you want. Although if you want to write efficient code which others can read it's essential that you follow some important rules.

a) White space in code is where you intentionally leave space in the code to signify the end to a part of function. Tabs and indentation is also a part of White Space in code, indentation makes it easier for the eye to see where parts of the code begin and end and also how the code relates to other pieces of code.

b) Functions are pieces of code which take parameters and return data. A function allows the code which does the specific operation to be only written once, this prevents the code from being written multiple times thus increasing the line count. All you need to do to run function code is call its name and enter the parameters it requires.

c) Classes are also pieces of code which are only written once like functions, however a class is its own object meaning you can have an object with its own name and parameters. Classes are declared with parameters and they can have functions inside of them, meaning you can run a function on the data which the class contains. Classes are good because it not only prevents multiple lines from being written but it also allows multiple amounts of data to be stored in a single object.

d) Comments are the most important rule when writing efficient code. Even if your code isn't well written, comments explaining what the code does and where it does it improves the readability massively. It is also important that your comments are detailed enough to contain exactly what the piece of code does or it might seem like the writer doesn't actually know exactly what the code does.

e) Relevant Variable and Object names are also important. They name needs to be a simple value which dictates what it was declared for. The name shouldn't be too short (1 or 2 characters) otherwise it is easily lost among large pieces of code. CamelCase is a good tactic when declaring variables as it stands out easily from function names and can easily be caught out by the eye.

```
digitalWrite(1,HIGH);
digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);
```

Figure 3: Bad Code

```
for(int i = 1; i < 6; i++){
  digitalWrite(i, HIGH);
}
```

Figure 4: Goode Code

0129657118

Hardware Concepts 1: Microcontrollers

Concept: A small programmable computer on a chip with readable inputs and programmable outputs

The microcontroller is a small integrated system on a chip. They usually have around 1kB of RAM and around 16kB of Flash Memory. All microcontrollers are capable of floating point calculations generally limited to a clock speed of 16Mhz. Depending on the chip, most microcontrollers are capable of analog read, digital read, PWM write and digital write. In this section, I am going to discuss the Arduino Uno's ATmega processor: The ATmega328p. As the Arduino Uno is the prototyping board I am using for my project. There are many variants of the ATmega, the variants are different in price affecting how many pins the chip has, how much Flash Memory it has, how much RAM it has and how many PWM pins it has.

The ATmega chips require an external clock, generally you'll find the ATmega series of chips operating at 16Mhz however can go lower than specified or up to 20Mhz All the ATmega chips are AVR chip meaning that they can be programmed in an extended version of C (Programming language) in the Arduino IDE. However, you could program the chips in the AVR studio in Assembly. Below is a selection of the 28 Pin form factor ATmega chips with their specifications.

Microcontroller	Package	Program Memory	SRAM	EEPROM	I/O Pins	Timers	A/D	SPI	I^2C	PWM	USART	Oscillator (MHz)	Datasheet
ATMEGA48V-10PI	PDIP28	4k	512	256	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	10	
ATMEGA8A-PU	PDIP28	8k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	3	Yes	16	
ATMEGA8L-8PU	PDIP28	8k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	3	Yes	8	
ATMEGA88-20PU	PDIP28	8k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	20	
ATMEGA88PA-PU	PDIP28	8k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	20	
ATMEGA88V-10PU	PDIP28	8k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	10	
ATMEGA168-20PU	PDIP28	16k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	20	
ATMEGA168V-10PU	PDIP28	16k	1024	512	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	10	
ATMEGA328-PU	PDIP28	32k	2048	1024	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	20	
ATMEGA328P-PU	PDIP28	32k	2048	1024	23	2x8-bit, 1x16-bit	6x10-bit	Yes	Yes	6	Yes	20	

Figure 5: ATmega 28 Pin Range

The crystal oscillator (clock) for the Arduino is generally connected to pin 9 and pin 8 whilst still being able to be used as a digital write pin.

The ATmega328p has three internal memory banks, Flash, RAM and EEPROM. The FLASH memory contains the routines programmed on a computer. A partition on the FLASH memory is taken up by the bootloader which is the first code run by the processor, this code instructs the processor on what to execute next. RAM (Random Access Memory) is the memory of the programmed routine which erases after the device loses power, unlike FLASH memory. EEPROM (Electrically Erasable Programmable Read Only Memory) memory is the same as RAM except it is read only, it usually contains values which are read only or do not get changed throughout the program. EEPROM erases on power loss, like RAM. You generally do not concern yourself with the memory of the device if you are programming with the Arduino IDE because the compiler works out where each value must be flashed to.

Complex Software Concepts 3: Interrupts

Concept: A failsafe or require action to be taken by a microcontroller

An interrupt is an instruction that must be triggered immediately regardless of the currently running instruction. For example, if you are programming a video game, you want the controls to feel responsive, so you halt the rendering process for the video game for a brief moment to ensure the button press is acknowledged before proceeding to render the frame again.

The interrupt check must be called every tick of the loop to ensure that the desired action isn't missed by the loop.

There are two types of interrupts, software and hardware. Hardware interrupts can be things like reset switches which physically remove power from the microcontroller to halt the action immediately in case of an emergency or the currently running process isn't desired.

Complex Software Concepts 4: Counters

Concept: A value which increases/decreases to represent a scale to then be interpreted

Counters are incrementations of numbers in code which when incremented will change the outcome depending on the value the counter is currently at. An example of this is the incrementation of a for loop as seen in figure 4 in which case the counter is named ***i***. The ATmega has a built-in counter value called "Millis" which is the value in milliseconds since the chip was turned on. Millis is an unsigned long value meaning that after 50 days, the counter will overflow back to zero. In my project code, I count how many frames it has been since the last point was scored in Pong to ensure the animation offsets properly.

NZQA Assessed

Complex Hardware Concepts 2: DC Motor Control

Concept: A way to operate a high current DC motor with a low current microcontroller

Because the internal Digital write pin on the ATmega328p cannot provide enough current to power a regular 5v DC motor, the device must use a transistor gate to dictate whether the DC motor is directly connected to the main 5v power line. The transistor is wired up to the Arduino and 5v lines as shown in Figure 6. The only problem is that you will accumulate back EMF from the motor continuing to spin after the circuit has been switched off, this slowly damages the transistor.

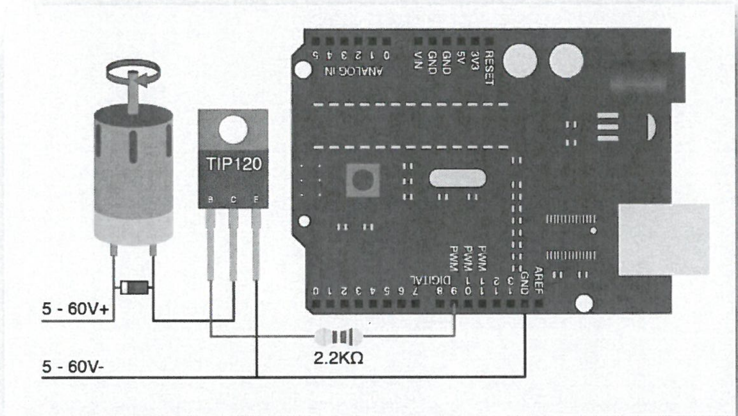


Figure 6: A transistor wired up to an Arduino

Complex Hardware Concepts 3: Multiple Sensors (Potentiometers, Accelerometer)

Concept: Inputs from the environment to then be interpreted by the microcontroller

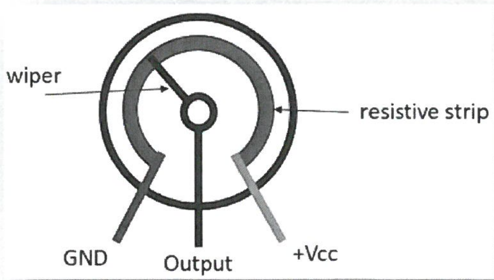


Figure 7: A diagram of a Potentiometer

Potentiometers are variable resistors with a rotary knob which determines the resistance. The potentiometer has 3 contacts, one for the positive, one for ground and one for the output value. The wiper has resistive material on it and depending on how hard it presses on the sloped resistive strip is how much current passes through. One side of the resistive strip does not take contact with the wiper so produces a value of zero whilst the other side of the strip presses hard on the wiper to make a perfect connection and produces a perfect output.

Accelerometers measure the change in rotation, depending on how many axes it has. The device takes in a high value and a low value, like potentiometers, and produces a value in between representative of the position for each axis it has. In my project I am using a Freetronics 3 Axis Accelerometer which has the MMA7361 IC on it. The accelerometer works by having a mass suspended inside a container on a spring. When force is applied to the container, the spring will experience either stretch or pull depending on the direction on the axis. The spring's voltage resistance is what is measured to give the value. The accelerometer has 3 of these spring containers, one for each axis.

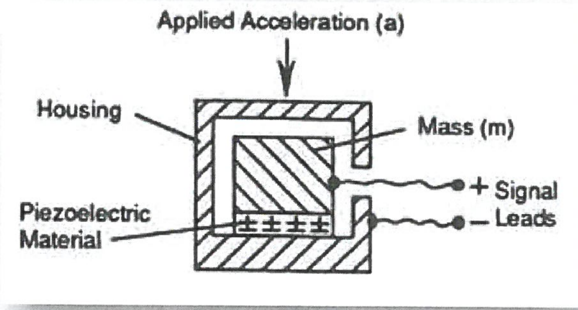


Figure 8: The accelerometer structure for one axis

Complex Hardware Concepts 4: Liquid Crystal Display

Concept: Using a flat surface to display information/graphics using bit logic.

A monochrome LCD is a large matrix of electrical contacts corresponding to a square cell of liquid crystal. When powered high, the liquid crystal changes to allow light to pass through causing it to appear black when using a backlight. Do this many times for a range of pixel and you can create bitmap graphics on the electrical matrix causing it to display.

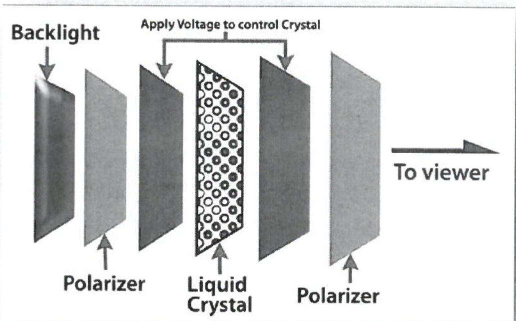


Figure 9: The layers of an LCD display

The polarizer is required to invert the display because by default, the liquid crystal is black and only when powered high it becomes transparent.

In my context, I am using the display to display the score of the game played on the oscilloscope because the oscilloscope's screen it too small to contain the score as well as the game content.

Bibliography

Figure 1: http://www.centerpointaudio.com/Images/Analog-Digital%20frequency%20examples.png

Figure 2: https://en.wikipedia.org/wiki/Pulse-width_modulation#/media/File:PWM,_3-level.svg

Figure 5: http://www.futurlec.com/ICAtmel_ATMega_Comparison.shtml

Figure 6: http://bildr.org/blog/wp-content/uploads/2011/03/tip120-motor.png

Figure 7: https://i1.wp.com/rntlab.com/wp-content/uploads/2016/02/potentiometer.png?resize=519%2C319&ssl=1

Figure 8: https://www.pc-control.co.uk/images/accel1.jpg

Figure 9: http://cdn.erodov.net/portal/reviews/guide/lcd-explained/2.jpg