CRASH COURSE ARDUINO AND MICROCONTROLLER DEVELOPMENT

LEARN THE FUNDAMENTALS OF EMBEDDED SYSTEMS, FIRMWARE, AND PROGRAMMING

Udemy

MASTER COURSE OUTLINE AND NOTES

AUTHORED BY ANDRE' LAMOTHE

TABLE OF CONTENTS

NOTES

1.0 Introduction

Welcome to ***Crash Course Arduino and Microcontroller Development***! This document is a general outline of the course, as well as contains links referred to in lectures, and anything that is hard to see in the video such as long URLs. Additionally, the references to the text book we are using in the course and the parts list are all discussed below.

This document should be used as you proceed through the lectures, so you can see any links, sources, or details needed for any particular lecture. However, please read through the following sections on the source files, text for the book, and parts list ***before*** you start viewing the lectures (**about 25 pages**).

Additionally, this course is one of the largest on udemy in terms of number of lectures, hours of video, and source files. In other words, there is ***a lot*** to digest and keep track of, so please read through the initial pages of this document and explore the download file(s).

1.2 Software, Accounts, Tools, Online and Offline

This course is designed for students that want to follow along on desktop PCs, build all the projects with hardware as well as those that just want to watch the lectures only, and maybe try some of the online simulators. That said, the course leverages a **LOT** of tools, this is to give you a realistic experience with embedded engineering since embedded engineers have to be comfortable using all manner of tools from code editors to CAD programs.

Additionally, for desktop tools I tried to select tools that are available on all desktop operating system platforms; **Windows**, **MacOS**, and **Linux**. Our main desktop tools are the **Arduino IDEs**, **EasyEDA**, **CodeLite**, all of which run on all operating systems. Then our primary online tools are **TinkerCAD**, **WokWi**, **replit** which of course run in any popular browser.

With that in mind, whenever we use a tool in lecture, I will usually show you how to install it, and how to use it. In fact in Section 1 of the course, we spend a lecture or two installing one of the more complex C++ tools **CodeLite** to make sure you have it working along with the **Arduino IDE**. However, just so you know what is in store, here is a list of all the applications and online tools we will use during the course. They are all "**free**" for the most part, but some have "pro" or enhanced paid versions. You do not ***need*** these class, but you may find you like the tool so much you want the paid version.

Here's the list, you don't need to install anything yet, but just take a look for reference. Some applications will be on both lists since there is an online and desktop version.

Desktop Applications

You won't need all of these applications for the course (for example you only need one text editor), but many of them you will need to install if you want to follow along at some point.

https://easyeda.com/
https://codelite.org/
https://fritzing.org/
https://www.microchip.com/en-us/tools-resources/develop/microchip-studio
https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide
https://www.sublimetext.com/
https://notepad-plus-plus.org/
https://code.visualstudio.com/
https://platformio.org/
https://www.putty.org/
https://www.emtec.com/zoc/index.html
https://www.st.com/en/development-tools/stm32-ides.html
https://www.arduino.cc/en/software

Online Applications

As noted, I really wanted this course to be available to students that did not have the resources to build hardware, but they do have the ability visit websites and run simulations of our projects. So, for many projects, I will build ***both*** a real **hardware model** on the bench and I will build a **simulation** as well. In fact, this is how embedded engineers work; we don't build hardware all the time, many times, we simulate first before building it. That said, the primary tools you will use are listed below, you will need to create an account on most of them to follow along and share my projects.

Note: You will notice for example, **EasyEDA** and **Arduino** are on both lists. This is because there are cloud based and desktop versions of each.

https://www.tinkercad.com/
https://easyeda.com/
https://wokwi.com/
https://www.godbolt.org/
https://replit.com/

Arduino Web Editor can be found here (as well as desktop downloadable version):
https://www.arduino.cc/en/software

Also, throughout the course, we might use other online tools, "one offs", they are not listed here, but I will let you know during lecture if you need to do anything special to run them.

1.2 Source Files for the Course

In Section 1, Lecture(s) 1 or 2, in the **downloads and materials** link for this Section/Lecture you will find a compressed ZIP/7z file that contains all of the source files, projects, schematics, etc. referred to in the lectures. I compiled them all together and placed them into a **single** ZIP container rather than have dozens of individual downloads all over the place. This makes maintenance and updates easier as well. The name of the file is:

CrashCourseArduino_XX_XX_XX.ZIP/7z

Where the XX part is simply the latest date the file was generated, so always download the newest file. In the ZIP/7z root, you will find a **README.TXT** please read it. The directory structure of the files are shown below in Figure 1.0:

Figure 1.0 - Master Course File Structure Diagram.

Root\

|

CrashCourseArduinoTextbook\ - This contains a free copy of my book which we use for some of the electronics lectures in the course.

Sources\ - In this folder are a number of sub-folders, each for a different tool we use in the course. These sub-folders contain the source/project files for various projects for each tool respectively. In some cases, these folders are redundant since the tool is online and the project(s) can be shared to your account by you copying a link; however, I like having "hard" copies locally just in case for your convenience, so if you have trouble sharing an online project or forking, cloning it.

|

|-------**ArduinoSketches\** - In this folder you will find a copy of all the desktop Ardunio IDE sketches that we use for the course lectures. I suggest you simply copy these into your Arduino sketch directory when you install the tool.

Codelite\ - In this folder you will find the C/C++ projects we develop for the course during the C/C++ training lectures. CodeLite is a multiplatform desktop tool that we use in the course to learn C/C++. Again, simply copy the folder to your PC whereever you want to keep the project files for your installation of CodeLite.

EasyEDA\ - In this folder, you will find all the schematic design files for the schematic and PCB tool EasyEDA that we use during the course. The tool has both a desktop and web based version, but we use the desktop version in the course since its more responsive. Either way, the tool project files can be shared with web links (shown later in text), but again, I like hard copies of files, so I have downloaded each project and made a copy of it in this folder for you, so you can simple "Open" the file/zip from the EasyEDA tool and import it if you wish.

Fritzing\ - In this folder, you will find copies of all the Fritzing projects we make in the course. Fritzing is another circuit tinkering, CAD program that we use to design embedded projects with. Fritzing is both supported online and on the desktop. The online portion allows projects to be viewed and shared, but the "tool" itself is only for the desktop. So, again, I have made copies of all the Fritzing projects we make during the course, as well, as a couple libraries needed, and all these files are in this sub-folder. Therefore, when you install Fritzing on your PC, you will simply copy this folder to your PC and open and import projects from it as needed.

Replit\ - Replit is a 100% online coding tool. It support C/C++, Javascript, Python, HTML, and about 50 other languages. We use it for some of our C/C++ training examples. This tool allows sharing of projects with online links (which I will list in a following section). However, once again, I have downloaded each project locally for you, so you have a local hard copy if you wish to load it that way.

TinkerCAD\ - In this folder you will find a .TXT file that contains links to all the TinkerCAD projects for this course. TinkerCAD as of the writing of this document is 100% online, and does NOT have support for any kind of local project downloads. Therefore, this tool is 100% online. Nevertheless, in the future, we may be able to download copies of projects locally on our PCs. If so, then copies will be in this directory. Until then, a .TXT file is in this folder, open it and simply copy each link into your browser as needed for the lectures.

WokWi\ - In this folder you will find .ZIP copies of all the WokWi projects developed in the course. WokWi is an online circuit simulator that supports Arduino. This tool is only online, but it does support downloading .ZIP copies of projects, so I have done this for your convenience to have local hard copies. But, again, later in the text, I will give you web links as well to access all the projects used in lecture and share them.

CUPL\ - In this folder you will find white papers, source files, tools, and other resources related to "programmable logic". In Section 5 at the end (Lecture 18), we explore programmable logic, any files discussed are in this folder.

MicrochipStudio\ - In this folder you will find any Microchip Studio projects we develop during the course. Microchip studio is being phased out by Microchip MPLABX, but many developers still prefer the old IDE, so we use it for a couple lectures to see how it works.

MPLABX\ - In this folder, you will find any Microchip MPLABX projects we develop in the course. MPLABX is Microchip's marquee IDE development tool and the suggested development platform. We only use it for a couple lectures, but, if you want to follow along you will need to install it.

VSCode - In this folder, you will find a couple Microsoft Visual Code projects that we experiment with at the end of the course. As usual, these are copies of the hard drive image, and may not be importable, worst case, just follow the lecture(s) and import the C/C++ files from here at very least.

STM32CubeIDE - In this folder, you will find copies of the STM32CubeIDE ARM projects we develop in the course.

eBooksAndGoodies- In this folder you will find various eBooks and goodies you might find useful in your journey learning embedded engineering and electronics. Additionally, we use some source files from my book "**Tricks of the 3D Game Programming Gurus**", so a free copy of the book is contained within along with the CD-ROM image which has the source files you will need when we build a simple game console and 3D game in the course.

MiscImagesChartsEtc - In this folder you will find various helpful charts, text files, and other files referred to in the course.

CrashCourseArduinoMasterOutline_xx.pdf - A copy of ***this*** document you are reading!

CrashCourseArduinoPartsList_xx.pdf - The parts list and hardware requirements for the course, you ***must*** read this.

README.TXT - Readme file, please read for any last minute updates or instructions.

1.3 Online Files and Project Links

In this section, I will list all the online tools we use, and access links to the projects in each, so you can share them, clone, copy, etc. In some cases, all that is needed is my user name, in other cases, full links to the project(s) are required. In most cases, once you access a project via your browser, you will need to have an account for the tool (all are free, or have free versions of membership), and then you will clone, copy, fork the project to your workspace. Don't worry, it's not that complicated, you will figure it all out :)

1.3.1 Strictly Online Tools

The first set of web based tools are interacted with online only. There is no desktop version of the tool or requirement. Therefore, for these tools, you simply create an account (free) and then you can access the course projects via my links or account and copy/clone/fork them.

TinkerCAD - www.tinkercad.com

This is a very powerful online tool from Autodesk that supports circuit design, 3D modeling and much more. We are using it for its circuit design and simulation abilities as well as for its support of Arduino. In many lectures, we build TinkerCAD simulations. To access the projects for the course, simply login and then perform a search on my user name "**AndreLaMothe**", with search space set to "circuits" like this:

https://www.tinkercad.com/users/18ypWSaHA4e?type=circuits

Once you do this, you will see a number of circuits that you can "**tinker**" and copy. Please copy (i.e. Tinker) all circuits you see with "**CCAMD**" as the prefix.

WokWi - www.wokwi.com

WokWi is an online tool that supports circuit design as well as simulation of Arduino hardware. Unlike TinkerCAD it has better support for multiple files and some features I like, so we use it time to time for some lectures. WokWi has a funny user system that does NOT require a password. You simple register your email, create a profile, and every time you want to login, you will receive a link in your email, you copy the link to your browser, login (cookies and state vars keep track), and that's that. That said, please create a free account and you will be able to access the projects used in this course.

WokWi is still a little rough around the edges, it doesn't seem to support searching for makers or specific projects by name, so you have to have a link to the project. That said, my user name for searching is:

rexblade6502

And my projects page is:

https://wokwi.com/makers/rexblade6502

You should be able to visit this webpage and see all my projects that are public, you want visit and copy with the SAVE button (blue button top left) each project that starts with a "**CCAMD**" prefix (not the backup copies though. And it will copy the project to your account.

Additionally, here's the links to each WokWi project we develop in the course, simply copy into your browser after you have logged into WokWi, and you will be able to access and copy each project:

CCACCAMD_Wokwi_Blink_01 - https://wokwi.com/projects/320893291011768916

CCAMD_Wokwi_AsmPlayground_02 - https://wokwi.com/projects/339200739900392019

CCAMD_Wokwi_WatchDog_01 - https://wokwi.com/projects/343008379111735890

CCAMD_Wokwi_Sleep_01 - https://wokwi.com/projects/343715560157086292

Replit - www.replit.com

Replit is hands down the **slickest online development tool I have ever seen**. You can code in a number of languages, run your code online, in a browser, share, collaborate, its nuts. They have both a free and tiers of paid accounts. You just need the free account. So, register, login, and then you can access a couple of C/C++ projects we develop in the course as we learn C/C++ for embedded. As usual, there are a number of ways to copy/clone/fork projects. First, you have hard copies in the .ZIP files source tree for the course (as outlined in sections above), but the proper way to copy/clone my projects is to go to my public user account "**RexBlade**" at:

https://replit.com/@RexBlade

And then you can see all the public "repls", you want to copy all the projects that begin with the prefix "**CCAMD**". While you are logged into your new replit account simply click on any of my CCAMD projects, then a page will open, and simply click the green button "Fork and Run" at the top left, and this will make a copy in your account.

Additionally, here's a list of all the projects direct links for your convenience:

https://replit.com/@RexBlade/CCAMDReplitBlink01

https://replit.com/@RexBlade/CCAMDReplitHelloWorld

https://replit.com/@RexBlade/CCAMDReplitStarter01

https://replit.com/@RexBlade/CCAMDReplitLesson01

https://replit.com/@RexBlade/CCAMDReplitLibraryTest01

https://replit.com/@RexBlade/CCAMDReplitHacking01

That's about it for online tools with projects that you need to copy or reference during lectures. Additionally, we do use some other online tools in some lectures, but you don't need to make accounts or copy projects, we simply use them on the fly.

1.3.2 Tools with Online Project Management & Desktop Primary Tools

These next tools have both an online portion ***and*** a desktop application.

EasyEDA - www.easyeda.com

EasyEDA is a very large CAD system including schematic capture, PCB design, AC/DC simulation, PCB manufacturing and assembly and supports both an online version and desktop version (recommended). In other words EasyEDA is not just a tool its an entire "**eco-system**" of companies that allow you to design PCBs with their tool, then have them manufactured if you wish! Pretty crazy! But, with so many systems it can be a little overwelming to just use the tool for simple circuit design and maybe a PCB here and there.

That said, simply register on the website, create a free "**standard**" account, and then I suggest you download the desktop version, its much faster. But, if you want to use the online version (which is why I choose this tool, for people that only have pads or phones that they use for computing) then it's fully capable of everything the desktop version is, albeit a lot harder to work without a mouse and keyboard!

Either way, you will need to access the project files for the course and make copies of them as usual. Each project only has a schematic file, there are no added PCB files, etc. So, each project is literally a single file, so that makes things easy. That said, there are hard copies of all the project ZIPs in the course master ZIP file for your convenience, but to copy/clone the projects for the course, login to the browser version, use the following links:

CCAMD_Arduino_AnalogJoystick_01
https://easyeda.com/editor#id=|a006668789384bff99af0e616b3ef573

CCAMD_MCP41xxx_SPI_01
https://easyeda.com/editor#id=|77ac95ecab18452e933d2444513adb1d

CCAMD_Arduino_Raiders3D_ST7735R_SPI_01
https://easyeda.com/editor#project_id=5b4430f2ba49446b98f39f326546dc28

CCAMD_Arduino_RGB_Buttons_01
https://easyeda.com/editor#project_id=47b3554922054f4dae4ea1efc51d30ca

CCAMD_Arduino_RotaryEncoder_01
https://easyeda.com/editor#project_id=93c4af44a0c14df4ba52122d50b161c8

CCAMD_Arduino_Starter_01
https://easyeda.com/editor#project_id=d03875c246bc4a06b44bc7788538c393

CCAMD_Arduino_TMP121_SPI_01
https://easyeda.com/editor#project_id=3274de2d2a6749e0843270e32eae26ec

CCAMD_Cap_Charge_LED_01
https://easyeda.com/editor#project_id=caa294a51824492aa345b2af13f1cec1

CCAMD_Current_Divider_01
https://easyeda.com/editor#project_id=0b866d3026434b4eb2ae045eb81d2732

CCAMD_Low_Pass_Filter_01
https://easyeda.com/editor#project_id=f542891c69ab4b938cda5e783d7418d4

CCAMD_Voltage_Divider_01
https://easyeda.com/editor#project_id=9cb85e32770f484e8ff22826a46dedbe

To copy each project, open each link in your browser, then click the blue button "Open in Editor", at the top right of editor. This will open the project in your browser.

Either way, once you have the project opened, then you will click on the project in the project tree on left then right click, and select "**Clone**", and then in the dialog, make sure to clone as new project, and rename if you wish.

Additionally, you can copy/clone the project with the main file menu bar by selecting **"File"** and select "**Save As**" which will make a copy into your account which you can play with as well. Once the project is copied into the cloud under your account, you should be able to see it in the desktop version as well and work with it.

Note: The online version of the tool has update issues and can be wonky when you clone or "**Save-As**" to copy my projects. I suggest you use the "**Save-As**" method, it seems to work better. Additionally, the desktop version is vastly more stable that the online version. EasyEDA definitely has quirks, but once you get the projects copied into your account, you should have no trouble.

Fritzing - www.fritzing.com

Fritzing is another simpler schematic capture, and simulation (very limited capabilities) that many people seem to like, so I tried to include it in some of the lecture demos. Fritzing has had a rocky development road and was abandoned years ago and restarted. But, it had a huge hobby following, so people still like to use it for simpler circuit designs that visually look more like real circuits and parts. That said, the tool currently only has a desktop version for design, but there is the online site and project management. I expect a completely web based version in the next couple years (2025-2030).

However, for now, you simply need to create an account, download and install the application on your PC and then copy each one of the lecture projects from the master source ZIP. There is a hard copy of all the projects in the master course ZIP file which you can copy to your hard drive and then use "**Open**" from the main menu and load each project. The online project management seems to always be "**not available**", so the simplest way to copy the course project files is from the master source files ZIP. As outlined in the sections above, the files are located in:

Root\..\Sources\Fritzing\

CCAMD_Fritz_AD5421_01.fzz - Breadboard design only.

CCAMD_Fritz_DS1307_01.fzz - Breadboard design only.
CCAMD_Fritz_IDE_DS1307_I2C_01.ino - Source code for simulation.

CCAMD_Fritz_MPU6050_01.fzz - Breadboard design only.

CCAMD_Fritz_SI7021_01.fzz - Breadboard design only.

At some point when the online and cloud project works, you may be able to search for the projects online as well with my user names: "**RexBlade**" and/or "**Andre LaMothe**".

Note: Additionally, there are some extra libraries in the **\Fritzing** folder, if your tool complains it can't find something navigate into the folder and install all files with the library extension ***.fzb.** This is also covered in lectures of course.

Arduino IDE / Arduino Web IDE - www.arduino.cc

For this course, we use the **Arduino Uno R3** (ATMega 328p based processor) as the primary learning platform since its easy to understand and widely available. We cover installation of the desktop Arduino IDE in lectures as well as the new cloud based IDE. That said, all of the source files for the Arduino projects can be found in the master source file ZIP file in the **\ArduinoSketches** directory as shown below:

Root\..\Sources\ArduinoSketches

Arduino projects ***can*** live online, but we don't use this feature, and instead keep everything primarily on the desktop application to keep things simple. Therefore, for any lecture that refers to an Arduino IDE project, you will simply navigate into the **ArduinoSketches** directory and make a copy of the project. Here's the massive list of the project folders at this time for the course (in alphabetical order):

CCAMD_Arduino_IDE_AD5241_I2C_01

CCAMD_Arduino_IDE_AsmPlayground_00
CCAMD_Arduino_IDE_AsmPlayground_01
CCAMD_Arduino_IDE_AsmPlayground_02

CCAMD_Arduino_IDE_Basics_01
CCAMD_Arduino_IDE_Blink_01

CCAMD_Arduino_IDE_Debbuger_Demo_01
CCAMD_Arduino_IDE_Debugger_Demo_01
CCAMD_Arduino_IDE_DIP_7Segment_01
CCAMD_Arduino_IDE_DS1307_I2C_01
CCAMD_Arduino_IDE_DS1307_I2C_02

CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_Ext_Interrupts_04

CCAMD_Arduino_IDE_FreeRTOS_01
CCAMD_Arduino_IDE_Hacking_01

CCAMD_Arduino_IDE_HelloKeypad_01
CCAMD_Arduino_IDE_HelloKeypad_02

CCAMD_Arduino_IDE_I2C_Scanner_01

CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Lesson_Test
CCAMD_Arduino_IDE_LibraryTest_01
CCAMD_Arduino_IDE_LogicModule_01
CCAMD_Arduino_IDE_MCP41xxx_SPI_01

CCAMD_Arduino_IDE_Motor_01

CCAMD_Arduino_IDE_Motor_PWM_01

CCAMD_Arduino_IDE_MPU6050_01
CCAMD_Arduino_IDE_MPU6050_02
CCAMD_Arduino_IDE_MPU6050_I2C_01
CCAMD_Arduino_IDE_MPU6050_I2C_02
CCAMD_Arduino_IDE_Multitask_01

CCAMD_Arduino_IDE_NeoPixelDemo_01
CCAMD_Arduino_IDE_NeoPixel_Simple_01

CCAMD_Arduino_IDE_Optimization_01
CCAMD_Arduino_IDE_Optimization_02

CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03

CCAMD_Arduino_IDE_POT_Test_01
CCAMD_Arduino_IDE_PrintFDebug_01
CCAMD_Arduino_IDE_RGB_Buttons_01
CCAMD_Arduino_IDE_RotaryEncoder_01
CCAMD_Arduino_IDE_SI7021_I2C_01
CCAMD_Arduino_IDE_SigGen_01

CCAMD_Arduino_IDE_Simon_01
CCAMD_Arduino_IDE_Simon_02
CCAMD_Arduino_IDE_Simon_03

CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02
CCAMD_Arduino_IDE_Sleep_03

CCAMD_Arduino_IDE_SPI_Raiders3D_01
CCAMD_Arduino_IDE_ST7735R_01

CCAMD_Arduino_IDE_Starter_01
CCAMD_Arduino_IDE_Template_01
CCAMD_Arduino_IDE_Template_02

CCAMD_Arduino_IDE_Timer_Interrupts_01
CCAMD_Arduino_IDE_Timer_Interrupts_02
CCAMD_Arduino_IDE_Timer_Interrupts_03

CCAMD_Arduino_IDE_TMP121_SPI_01

CCAMD_Arduino_IDE_UART_Client_01
CCAMD_Arduino_IDE_UART_Client_02
CCAMD_Arduino_IDE_UART_Receiver_01
CCAMD_Arduino_IDE_UART_Server_01
CCAMD_Arduino_IDE_UART_Transmitter_01

CCAMD_Arduino_IDE_VT100_01
CCAMD_Arduino_IDE_VT100_SnakeJoystick_01
CCAMD_Arduino_IDE_VT100_SnakeJoystick_02
CCAMD_Arduino_IDE_VT100_Snake_01

CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Ardunio_IDE_ChatGPT_GuessNumber_01

Arduino Web Editor Projects:

CCAMD_Web_Blink_01

https://app.arduino.cc/sketches/f9129dd4-bcb3-44d1-bf3e-59d7c7012ed3?view-mode=preview

CCAMD_Web_Fade_01

https://app.arduino.cc/sketches/39eb13c5-ff08-488a-b7cb-bdc4c7307250?view-mode=preview

Additionally, I have made a copy of my desktop Arduino **\library** folder just in case. You should ***NOT*** need this folder since during the course, any library you need you will install from the cloud with the library management tool built into the Arduino IDE. However, if you have a problem downloading a library, or github dies, or some other catastrophe then you will have a backup copy of the libraries used here:

Root\..\Sources\ArduinoSketches\libraries

Codelite - www.codelite.org

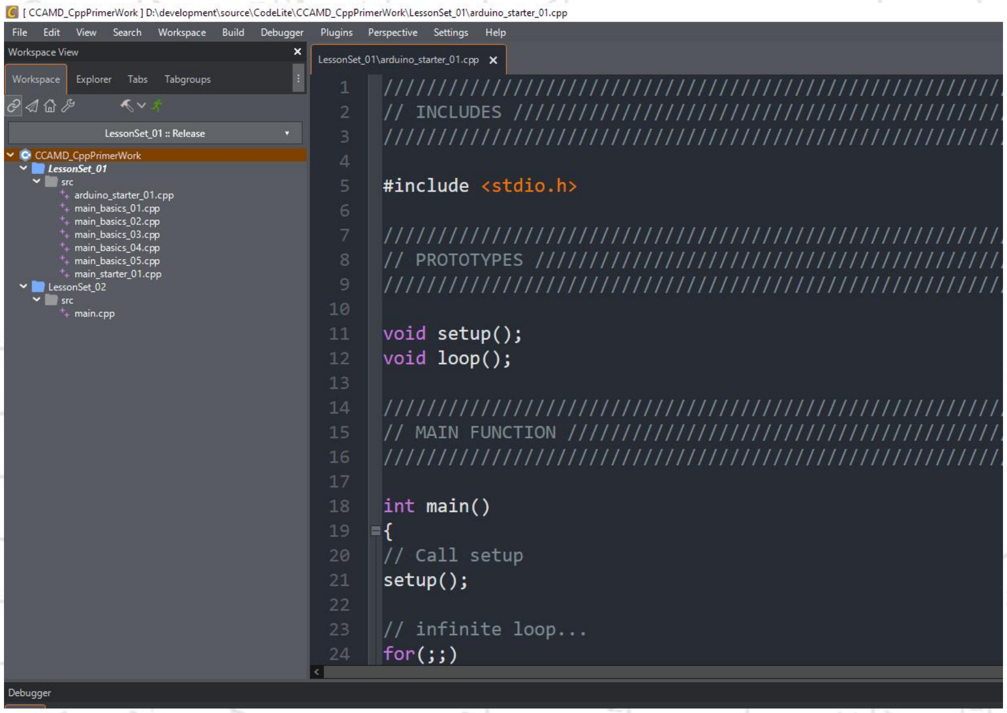
The next tool we need to talk about is the desktop C/C++ compiler/IDE. This course doesn't assume that you know C/C++ (if you do then you will be able to skim the C++ fundamentals if you choose). Therefore, part of the course teaches C++ (just as we teach basic electronics). To facilitate this goal, we can't use the Arduino's IDE's built in C/C++ compiler (C++ 11 actually). It is a subset of the language, memory constrained, etc. Alas, we needed a real desktop tool to get you used to working with real desktop IDEs since in embedded engineering you are going to have to do this a lot! That said, I needed something that was multiplatform (Windows, OS X, Linux, etc.) and a modern IDE with everything built in; editor, debugger, C++ compiler and so forth. **Codelite** fills the bill perfectly. We will use this development tool in parallel with other development and C/C++ tools like the Arduino IDE and various online tools like replit for our C/C++ lessons.

The installation of the tool is a bit tricky, so you will need to watch the installation lecture(s) in Section 1 of the course and follow along. However, in a nutshell, you simply visit www.codelite.org, download the version of the tool for your platform and install it. There are many online guides, videos, etc. Explaining the quirks of this tool, so don't worry too much, but it's par for the course in embedded engineering -- you might as well get used to installing lots of tools, reading how they work, and working around their issues.

Now, with that all said, we use the IDE during the initial portion of the course when we learn C/C++ (with constant commentary on how desktop C/C++ differs from writing C/C++ for embedded systems). However, much of the lectures is me hacking in real-time, adding code, subtracting code, and I simply want you to listen, watch, and follow along. Alas, there aren't a lot of project files for this tool since we tend to just hack a single file in real-time, so there aren't 50-100 projects one for each subject like variables, loops, conditionals, and so forth. Instead every time I needed a new file, I would just create it and work with it, and at the end move on, so the following project list is relatively short.

Finally, since the beginning of time, it's always challenging to "copy" a C/C++ project from one machine to another. Seems no one ever thinks this might be something programmers might want to do! Therefore, CodeLite doesn't have any kind of formal "export/import" workspace or project (that I am aware of), so have to work around this by copying the entire workspace directories which include the sub-projects and all source files. The problem with this is ***absolute file paths***. If you try to open a copied workspace or project then it may or may not work depending on how smart the IDE's file system has used relative paths or not used them at all. The figure below shows a screen shot of CodeLite running our workspace, with projects, and all C++ source files:

Figure 2.0 - CodeLite Workspace.


[CCAMD_CppPrimerWork] D:\development\source\CodeLite\CCAMD_CppPrimerWork\LessonSet_01\arduino_starter_01.cpp
File Edit View Search Workspace Build Debugger Plugins Perspective Settings Help
Workspace View
Workspace Explorer Tabs Tabgroups
LessonSet_01 :: Release
CCAMD_CppPrimerWork
LessonSet_01
src
arduino_starter_01.cpp
main_basics_01.cpp
main_basics_02.cpp
main_basics_03.cpp
main_basics_04.cpp
main_basics_05.cpp
main_starter_01.cpp
LessonSet_02
src
main.cpp
LessonSet_01\arduino_starter_01.cpp
Debugger

```
//////////////////////////////////////////////////////////
// INCLUDES /////////////////////////////////////////////
//////////////////////////////////////////////////////////

#include <stdio.h>

//////////////////////////////////////////////////////////
// PROTOTYPES ///////////////////////////////////////////
//////////////////////////////////////////////////////////

void setup();
void loop();

//////////////////////////////////////////////////////////
// MAIN FUNCTION ////////////////////////////////////////
//////////////////////////////////////////////////////////

int main()
{
// Call setup
setup();

// infinite loop...
for(;;)
```

Notice in the **left hand pane** (the file/project view) up top there is a single "**workspace**" called **CCAMD_CppPrimerWork**, below this there are two "**Projects**" **LessonSet01** and **LessonSet02** respectively. And in each project's **src** directory you can see there are a few .CPP (C++) files, these are all that are important really, as long as you have these .CPP files, you can re-create the workspace and projects on your machine, then add the .CPP files to the projects.

Whatever the case, in lecture(s), I show how to create workspaces and projects, so at worst case all you need are the C++ file(s) for each project/lesson which are included in the following folders:

```
Root\..Sources\Codelite\
                        |------CCAMD_CppPrimerWork
                               CCAMD_StarterWorkspace
```

For example, if you dive into **CCAMD_CppPrimerWork** you will find two more sub-directories named after the project folders in the figure above:

\LessonSet_01
\LessonSet_02

If you then dive into these directories you will find all the .CPP files you need for your projects (just ignore everything else). I wish it wasn't this complicated, but that embedded engineering for you and IDE setup :) Anyway, in the end, at worst case you will create a workspace with the same name as mine (if you like), as well as projects with the same name, then you will right click on the projects and "add file(s)" to add the .CPP files which you will add from the sources in the master ZIP file as outlined above.

Microchip Studio - https://www.microchip.com/en-us/tools-resources/develop/microchip-studio

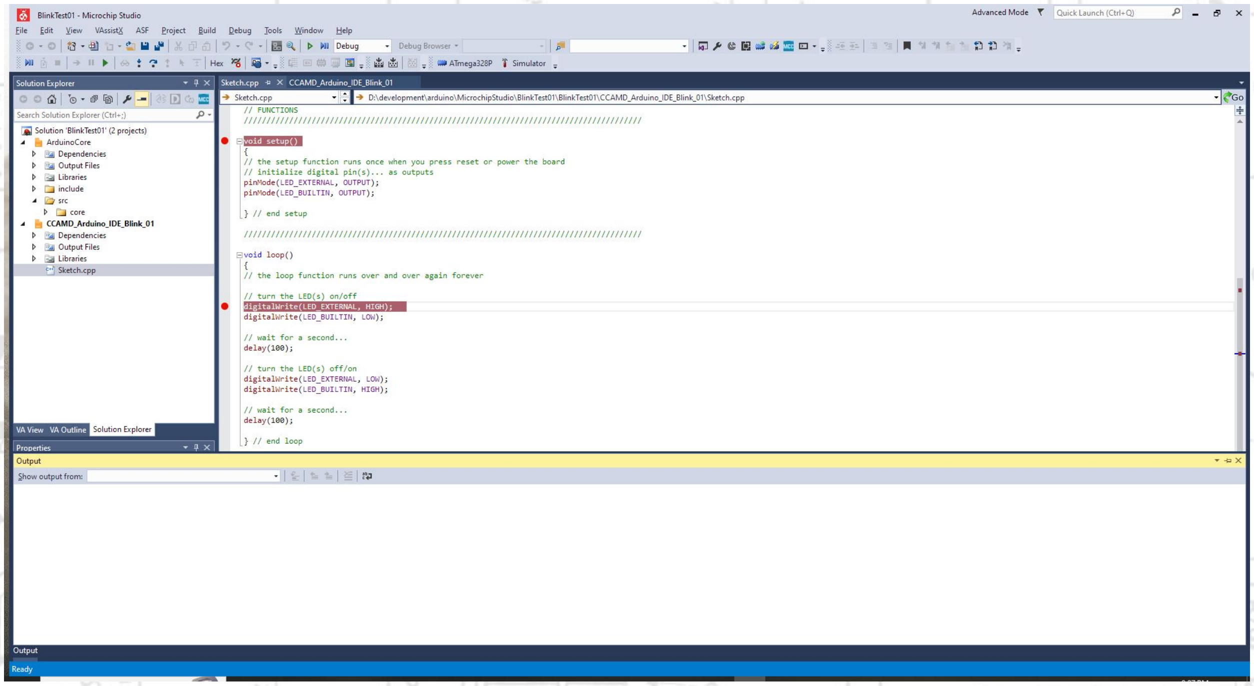
We use Microchip Studio in the course for a few lectures as a programming tool as well as an example of another IDE. Lecture(s) will show you how to install this tool if you wish to follow along with the lecture examples when the time comes. Microchip Studio is one of two primary IDEs you can use from Microchip. Originally, Microchip Studio was created to merge the functionality of AVR development from Atmel (after Microchip bought them) along with Microchip's ARM SAM line of processors.

In any event, if you do want to follow along with the few lectures that use this tool then you will install the tool (we cover this in lecture), then you will be able to copy the projects files from your local copy which can be found in the master course ZIP file. As usual, this is a big "if" since these tools are never very friendly with paths and other files, so at best case you may be able to simple launch the tool, navigate to where you installed the files from the master course ZIP file into:

```
Root\..Sources\MicrochipStudio\
                               |------BlinkTest
```

In this case, as of the time of the writing of this document there is only one project, a simple **BlinkTest** which is an example of importing an Arduino project. Everything else, we do on the fly in lecture. The figure below shows a screen shot of the tool open with our single project (solution) loaded.

Figure 3.0 - Microchip Studio Solution Screen Shot.


BlinkTest01 - Microchip Studio
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
ATmega328P
Simulator
Solution Explorer
Solution 'BlinkTest01' (2 projects)
ArduinoCore
Dependencies
Output Files
Libraries
include
src
core
CCAMD_Arduino_IDE_Blink_01
Dependencies
Output Files
Libraries
Sketch.cpp
Sketch.cpp
D:\development\arduino\MicrochipStudio\BlinkTest01\BlinkTest01\CCAMD_Arduino_IDE_Blink_01\Sketch.cpp
Output
Show output from:
Ready

```
// FUNCTIONS
////////////////////////////////////////////////////////////////////////////////////////

void setup()
{
// the setup function runs once when you press reset or power the board
// initialize digital pin(s)... as outputs
pinMode(LED_EXTERNAL, OUTPUT);
pinMode(LED_BUILTIN, OUTPUT);

} // end setup

////////////////////////////////////////////////////////////////////////////////////////

void loop()
{
// the loop function runs over and over again forever

// turn the LED(s) on/off
digitalWrite(LED_EXTERNAL, HIGH);
digitalWrite(LED_BUILTIN, LOW);

// wait for a second...
delay(100);

// turn the LED(s) off/on
digitalWrite(LED_EXTERNAL, LOW);
digitalWrite(LED_BUILTIN, HIGH);

// wait for a second...
delay(100);

} // end loop
```

Microchip MPLAB X IDE - https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide

Microchip's latest IDE is called **MPLAB X**. It's been in development for 10-15 years, but feels like only recently they have got the bugs out! That said, this tool is VERY advanced, supports all Microchip silicon and supports free compilers as well as a graphical based system to help you write code. We use this tool for a few lectures at the end of the course as an example of what you would expect to see in a professional embedded development tool. The installation is quite complex, so wait for lecture(s) to go over it. And again, projects, solutions, etc. are complex and ideally you should be able to copy them from machine to machine, but, things are never that easy.

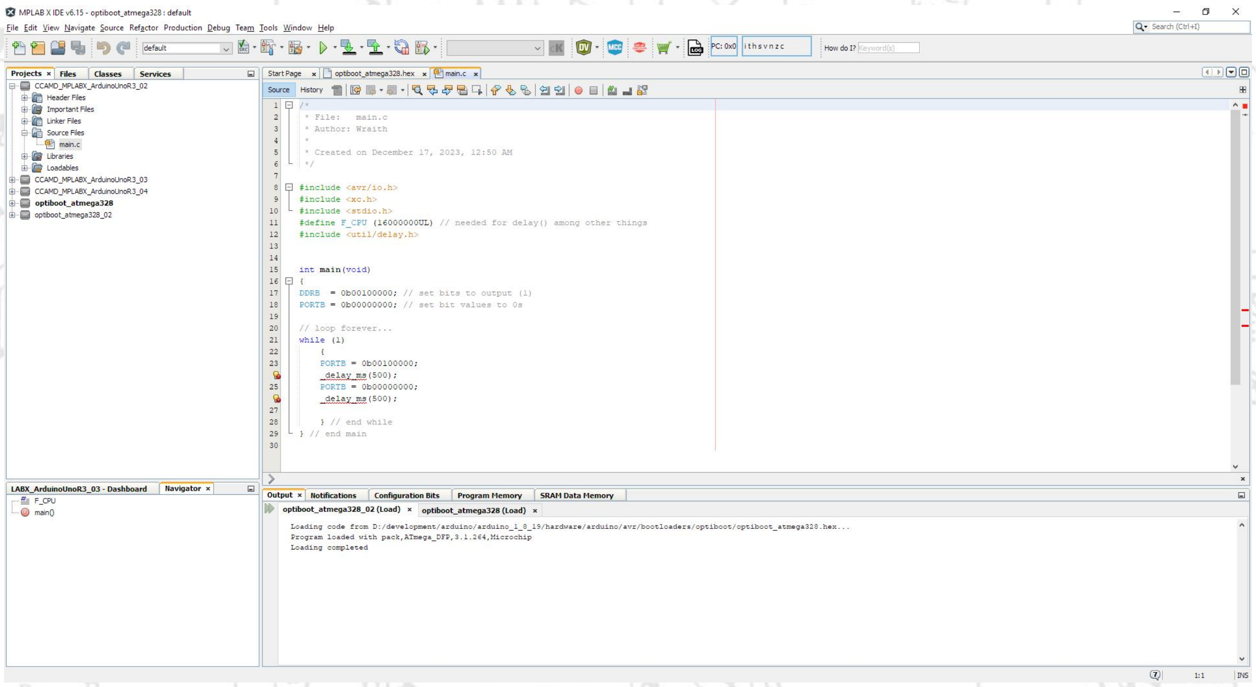
Once again, I have copied all the project files from my hard drive to the master source ZIP file for the course, so you can try and open each project with the tool and see if it works, if not, then you will simply follow along each project and re-create the projects, and you can copy the source files at least into your local project. This is a common thread in embedded, nothing copies easily, so get used to it :)

That said, the projects for the MPLAB X IDE lectures can be found in the master course ZIP file here:

```
Root\..Sources\MPLABX\
                     |------CCAMD_MPLABX_ArduinoUnoR3_02
                     |------CCAMD_MPLABX_ArduinoUnoR3_03
                     |------CCAMD_MPLABX_ArduinoUnoR3_04
                     |------optiboot_atmega328
                     |------optiboot_atmega328_02
```

The figure below shows what the tool looks like with all these projects loaded and the one of them open to the C/C++ main.c.

Figure 4.0 - MPLABX IDE Workspace Screen Shot.


MPLAB X IDE v6.15 - optiboot_atmega328 : default
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
Projects
Files
Classes
Services
CCAMD_MPLABX_ArduinoUnoR3_02
Header Files
Important Files
Linker Files
Source Files
main.c
Libraries
Loadables
CCAMD_MPLABX_ArduinoUnoR3_03
CCAMD_MPLABX_ArduinoUnoR3_04
optiboot_atmega328
optiboot_atmega328_02
Start Page
optiboot_atmega328.hex
main.c
Source
History
LABX_ArduinoUnoR3_03 - Dashboard
Navigator
F_CPU
main()
Output
Notifications
Configuration Bits
Program Memory
SRAM Data Memory
optiboot_atmega328_02 (Load)
optiboot_atmega328 (Load)

```c
/*
 * File:   main.c
 * Author: Wraith
 *
 * Created on December 17, 2023, 12:50 AM
 */

#include <avr/io.h>
#include <xc.h>
#include <stdio.h>
#define F_CPU (16000000UL) // needed for delay() among other things
#include <util/delay.h>


int main(void)
{
DDRB  = 0b00100000; // set bits to output (1)
PORTB = 0b00000000; // set bit values to 0s

// loop forever...
while (1)
    {
    PORTB = 0b00100000;
    _delay_ms(500);
    PORTB = 0b00000000;
    _delay_ms(500);

    } // end while
} // end main
```

```
Loading code from D:/development/arduino/arduino_1_8_19/hardware/arduino/avr/bootloaders/optiboot/optiboot_atmega328.hex...
Program loaded with pack,ATmega_DFP,3.1.264,Microchip
Loading completed
```

NOTE: All these IDE tools have various names for projects. Some call them "**solutions**", some call them "**workspace**", and still some call them "**projects**". Moreover, a workspace may have multiple solutions, or a solution have multiple projects. It's a lot of word soup, but you will get the hang of it :) Worst yet, sometimes these constructs map to folders on your hard drive, but other times, they are totally virtual, and the data on your hard drive is nothing like the structure of your projects.

1.4 Text Book for the Course

Along with the course, all students get a free eCopy of my book **"Design Your Own Video Game Console"** (a $30 value). This is the eCopy version of hard copy which is named different "**The Black Art of Video Game Console Design**" – shown here on Amazon.com:

The Black Art of Video Game Console Design

This is the only book of its kind in the world. The book teaches basic electronics, PCB design, fabrication, embedded engineering, single board computer design and much more with the goal of building "**game console**s". You can find the PDF for the book in the downloads for this Section\Lecture (in the ZIP file root) with the name:

\CrashCourseArduinoTextbook\Design_Game_Console_XX.PDF

Where XX is the version number, simply use the highest version number if there is more than one in the downloads. I use this book in the early electronics lectures, so we have something to reference.

1.5 Additional Books to Check Out

When you watch the lectures (unless you already have), you will notice I shoot them in one of my home labs (Electronics) and behind me you might be able to see many book shelves. I actually have a technical library with over 3000 printed books, so when I want to learn something, I usually go out, buy 3-5 books on the subject and read them all! That said, there are 100's of amazing EE/Embedded Design, and Electronics books, but I can't list them here, rather I am going to list just a few to get you started. There are so many areas of Electrical Engineering and sub-fields, it really does take years and years to learn just a fraction of it, so you have to pick you battles, but there are a few general books that you should read.

1. The Art of Electronics – This book is like the bible of electronics. It's not a hard core EE book full of math, but it still has its share of mathematics, but instead it's a hand on, practical book written by very smart guys that know their stuff. If you read this book cover to cover, you will be in a good position to design and build just about anything.

2. Electric Circuit Analysis – This book is hard core calculus based electrical circuit analysis. We do much of this in this course, but without the calculus. However, this is one of the best books on the subject. It's a hard book to consume, but everything is in there you need to learn advanced analysis.

3. Digital Design Fundamentals – This book is A to Z on basic digital theory. Of course, it's less math based since digital systems are discrete, but still an advanced college level text. One of my favorites and on the 11th or 12th edition, so this is "the" digital text used in many universities undergrad courses.

4. Digital Systems Principles and Applications – This book is another digital fundamentals book with a different approach (one thing you find is you have to read 2-3-5 books on the same subject to understand the instructor as well as the material). This book is very straightforward and sticks to lower level digital design, no nonsense approach.

5. Automatic Control Systems – Control theory is the study of the construction of machines that have "feedback" and "sensing". Control theory can be applied to electrical, mechanical, robotic, biological, etc. systems. This is one of the "hard" courses in EE, and something you absolutely MUST learn about if you want to build smart systems that do what they need to do. For example, when you set a thermostat to 72F, you need a sensor to read the temperature, feed it back to the controller and then if there is an "error", say it's 75F, the controller takes action to cool it off more – this is "control theory". This book is a good introduction to the subject, again, calculus based, but worth it, if you can get through it.

6. Computer Organization and Hardware Design – This book (or rather series of books) teaches about how to design a "computer" – simple as that. Now, back in the 1980's there was one version of this book, now there are many depending on the architecture you are interested in. So, the link shows a short list of options. At least read one of them cover to cover.

7. Designing Embedded Hardware - This is a great introductory book on simple embedded design from the power supply to the MPU to the interfaces.

8. Embedded Systems: Introduction to Arm Cortex M Microcontrollers, 5th Edition - The ARM processors, in particular the ARM Cortex are some of the best and most powerful microcontrollers on the planet. This book focuses on general embedded engineering, but uses the ARM Cortex as the platform.

9. List of ~40 Embedded Engineering Books - I have read all the books in this list, they are all solid books on the subject eventhough some are quite old. Many are for specific technologies and platforms, but take a look, hopefully this list helps you save time looking for practicle embedded engineering books.

There are so many other books and subjects like RF theory, robotics, PCB design, VLSI design, FPGAs, and so forth, but I don't want to overwhelm you too much just yet. So, take a look at the list above, and if nothing else at least read **"The Art of Electronics"** and a couple good selections from the list of "**40 Embedded Engineering books**".

1.6 Parts List

The Parts List for this course is quite detailed and long, so rather than insert it here and have you scroll down 30-40 pages to get to the next Section\Lecture, I have made the parts list a separate document for convenience.

Also, it's important to realize the parts list is more of a guideline and general list of parts that are used in many of the bench experiments, but the list includes parts that are not used along with much more since when you buy parts, you always want to buy more than you need, other values, etc. Additionally, many of the parts are used that are for "demo" only and you don' t really need to buy them since we only simulate or build something on the bench as an illustration. Nevertheless, if you want to follow along with every single lecture then the parts list will guide you on what you need.

Finally, the list has many sections on more advanced test equipment, these things are not cheap and only suggested for those that really want to get hands on. But, I designed the course, so that if you can't afford a single part right now, you can watch me design and build/test everything and still get a lot out of it.

Bottom line, please budget and don't go out and spend a lot of money until you have a better idea of what you need, this is exciting stuff, but control your enthusiasm before going on a shopping spree – I know once I get buying parts, my cart gets bigger and bigger and sometimes I need to walk away, come back and think if I REALLY need yet another 1000 LEDs ☺

The parts list is contained within the master ZIP file in the following directory:

Root\..CrashCourseArduinoPartsList_XX.PDF

Where the XX is simply the version number, take the latest number if there is more than a single PDF.

2.0 The Course Outline

In the following pages is a complete outline of all the sections of the course, and each lecture. The files for each lecture that I mention in the videos, you will use the section above to locate project files for the Arduino IDE, Codelite, Fritzing, TinkerCAD, WokWi, etc. However, any websites or links we visit during each lecture, or links that I think you might need will be listed below along with each lecture. This is so you don't have to try and read the text in the browser URLs as we visit websites during the live videos. Additionally, sometimes I list the projects and program names out under each tool, but this is for ***redundancy***, you should always be able to find any project back in **Section 1.1** above, so if in a lecture, I mention a project, you ***always*** jump to Section 1.1 to find the location of the file or project either in the master source ZIP or online at each of the tools sites we use.

Therefore, you ***must*** keep this document handy as you watch the lectures, so you can find the files, projects, and links contained within, otherwise you will crazy :)

Section 1: Getting Started with Embedded Engineering

This section talks about how Crash Courses work and what to expect.

Lecture 1 - "Course Overview and Welcome!"

This short lecture talks about how Crash Courses work and what to expect. I will cover the breadth of the course and outline my goals for you as a student and what I hope you learn. Finally, we will take a look at this document, the parts list, and discuss the different ways to take the course.

Lecture 2 - "Installation and Setup"

In this lecture, we are going to talk about all the software and applications we use in the course, as well as online tools and simulators. We'll install EasyEDA and the Arduino IDE tool, and make a short list of online tools you should get accounts on. This lecture is up here in front, so we don't have to slow our momentum with a big tool install in the middle of the course. Also, in the future, I will add installation lectures and videos to this section if students need them.

Links/Resources:

https://easyeda.com/
https://codelite.org/
https://fritzing.org/
https://www.sublimetext.com/
https://notepad-plus-plus.org/
https://www.putty.org/
https://www.arduino.cc/en/software
https://www.tinkercad.com/
https://wokwi.com/
https://www.godbolt.org/
https://replit.com/

Lecture 3 - Installation and Setup - CodeLite IDE and Compiler

In this lecture, we are going to install the CodeLite IDE and C/C++ compiler. If you like, you can skip this ***until*** you need it, but I suggest you just get it over with :)

MinGW and CodeLite Installation Tips (Covered in Lecture)

The CodeLite IDE platform runs on Windows, MacOs, and Linux, the steps for installation are similar across platforms. See site documentation below for details on installation for your platform. That said, here are the general steps you need to perform to install the CodeLite IDE and MinGW system which supports the GNU C/C++ compilers that we need.

Step 1: Compiler Installation - CodeLite is an IDE, not a compiler, it requires a compiler(s) to compile code, therefore, we have selected MinGW which supports the GNU C/C++ compilers and comes with them as part of the install. The MinGW system needs to be installed first on your PC, then the CodeLite IDE must be installed.

Installation of the MinGW runtime platform varies, you can find old installers, etc. But, as of now, the developers have made the system totally portable, meaning, you literally just download a ZIP (or similar file) and decompress it into a directory, no installation is needed, it can run from the directory with a couple additions to the PATH variable on Windows for example.

So, the first step is to find the version of MinGW you wish. You can find the platform here:

https://www.mingw-w64.org/downloads/
https://winlibs.com/

At the writing of this document there is both version 13 and 14, I suggest you try either with the following options:

UCRT- Universal runtime (rather than MSVCRT, the older Windows only runtime).

POSIX - Threading library (rather than MCF library).

Win32 or Win64 - Depending on your processor and OS, but 99% of the time, this should be Win64.
Based on these options, find the ZIP file (or 7Z, etc.), download it and then move to next sub-step.

X86 Architecture - If you're on a Windows PC with Intel then select this, else AMD for AMD and so forth.

Now, that you have the compiler on your hard drive, decompress it into a local directory, I strongly recommend the following location:

C:\mingw (or C:\mingw32 or C:\mingw64)

The decompression program (WinZIP, WinRAR, etc.) should create another directory in this directory and then decompress the files, so you will be left with something like:

C:\mingw\mingw\

Now, you need to tell Windows where the directory is with the compiler, specifically where all the executables are. They should be located within the \bin directory here:

C:\mingw\mingw\bin\

You will take this file path and add it to the system **PATH** environment. To do this in Windows, you will search for "**edit the system environment**" variables (which is in the control panel as well). Once you get to the **System Properties** dialog, select the **Advanced** tab, and on the bottom right, you should see a button <**Environment Variables**>, click the button, this will bring you to the Environment Variables dialog, from there, you will locate the **PATH** variable in the **System Variables** section, select it, then click the <**Edit**> button, from there you will see a list of all the PATH variables, click <**New**>, then type the path to the compiler (for example) **"C:\mingw\ming\bin"** into the text edit box, and then click "**Ok**" and close out anymore dialogs. This is all shown in the video lecture of course.

At this point, the compiler should be ready to use, open a command shell or prompt from the Start menu, and type:

C:\g++ --version

You should see something like this if the compiler is working properly:

```
Microsoft Windows [Version 10.0.19045.4355]
(c) Microsoft Corporation. All rights reserved.

C:\>g++ --version

g++ (x86_64-mcf-seh-rev1, Built by MinGW-Builds project) 13.2.0
Copyright (C) 2023 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

On MacOS and Linux, you will see something similar.

Step 2: CodeLite IDE Installation - Download the latest version of CodeLite IDE from the weblink below.

https://codelite.org/

On the site you will find installers for Windows, MacOS, and Linux. I will focus on the Windows installation as usual, the others are similar. And of course, this is all covered in the video lecture, but written down here as well for reference.

NOTE: Now, as usual with complex tools, there are issues, and depending on when you read this document there may be newer versions available. Currently the latest version is 17.x, but there is problem with 17.0.0, so whatever you do, do NOT use version 17.0.0, use anything newer which should be fine. Additionally, versions 15 and 16 work perfectly as well.

Once you download the installer, simply run it with **Administrator rights** (this is very important). The installer usually installs the IDE into the following path:

    C:\Program Files\CodeLite

If it doesn't, then I highly recommend you adjust it to this path. Seems there are hard coded path information in the tool, and it likes this path.

When the installer is complete then there are a number of additional steps that are required to get the IDE up and running and compiling C++ code. These steps are complex and lenghty, so its best your watch the video lecture at length, but I will give you highlights here for reference.

Step 2.1: IDE Setup Wizard - When you run the IDE the first time, you will start in a "**Setup Wizard**" to help you set the IDE up, some of the options you want to select are:

- In Development Profile Dialog, select "**C/C++ development**"

- In **Setup Compilers Dialog**, this usually does NOT find the mingw compiler, so simply press <**Next**> on this step, we have to set the compiler up manually.

- In **Customize Color Dialog**, select the color theme you like.

- In **Whitespace & Indentation Dialog**, select "**Indent using SPACES**" and "**Visible Always**".

- Click <**Finish**>

Now, after the Setup Wizard runs, we have a number of housekeeping chores to fix the Build System to find the compiler as well as creating a Workspace and Projects. Again, this is shown in detail in the video lecture, but I add steps here for reference.

Step 2.2: Adding the Compiler in Build Settings - Since the compiler we installed in the previous step didn't have a "real" installer, we literally just dumped it into a directory, there is no record of the installation (in the registry on Windows for example), and hence the IDE will not find our MinGW compiler. So, we need to add it, the sub-steps are:

- On the main menu bar, select **Settings->Build Settings**, this should open up the Build Settings dialog and it should default to the **Compilers** tab. On the left pane, you may see a number of compilers already installed on your PC, but what you won't see is the compiler we just installed! So, to remedy this, we are going to "add" a compiler and tool chains manually. Press the "+" icon on the top left.

- This will open up a search dialog, all we need to do is navigate to where the compiler is installed and its \bin directory where all the executables are. For example:

```
C:\mingw\mingw\bin
```

And click <**Ok**>.

- The IDE should find the compiler files, and ask you what you want to name the compiler, change it to "MinGW", keep it simple. Hit <**Ok**>. At this point, you will see the new compiler at the top of the list on the left pane, and the TOOLS view should populate with 10 or so tools that have been found and linked, this is step is complete, hit <**Ok**>.

Step 2.3 Creating a Workspace - This step we show in video lecture of course, but we will review it again here for reference. As we code, we need to create at least one "Workspace" as a container to hold all our "Projects" which we will create next. You can create a workspace from the main view of the tool, or from the main menu **Workspace->New Workspace**. Create a new workspace now, here are the sub-steps:

- **Select the workspace type dialog** - The first dialog that will display wants you to select the "type" of workspace we want, select "C++", and hit <**Ok**>.

- **New Workspace Paths and Filenames dialog** - This dialog allows you to select the workspace path, name, and options if you want to create a new directory for it. Set the path to somewhere close to the root of your development drive, then to be consistent call the workspace something like "**CCAMD_Workspace**", finally, enable the checkbox, so it creates the workspace under a separate directory. Click <Ok> and your done. You should see your named Workspace pop up on the left pane of the IDE.

2.4 Creating Projects - We are almost done! The next step is to see how to create a "Project". A project is a container with specific settings, code, and build options. You will create a new project for every new thing you want to work on. In our case, we are going to walk through some of the settings since they are important. Again, this is all in the video lecture, but added here for reference. Here are the sub-steps to follow:

- Select New Project from the main menu **Workspace -> New Project**, or by right clicking on the Workspace itself and select **New Project**. Now to setup the project correctly.

- Once you create a new project, the **New Project dialog** will display, and there's a lot to select and change. First, at the top of the dialog, the **Path** to the workspace and project should be acceptable, but the Name of the project needs to be adjusted, please select whatever name you wish, we use "**LessonSet**_01" to start for example. Next, set the following options for the rest of the dialog:

- Category:	Console
- Type:	Simple executable (g++)
- Compiler:	MinGW (or whatever you named it)

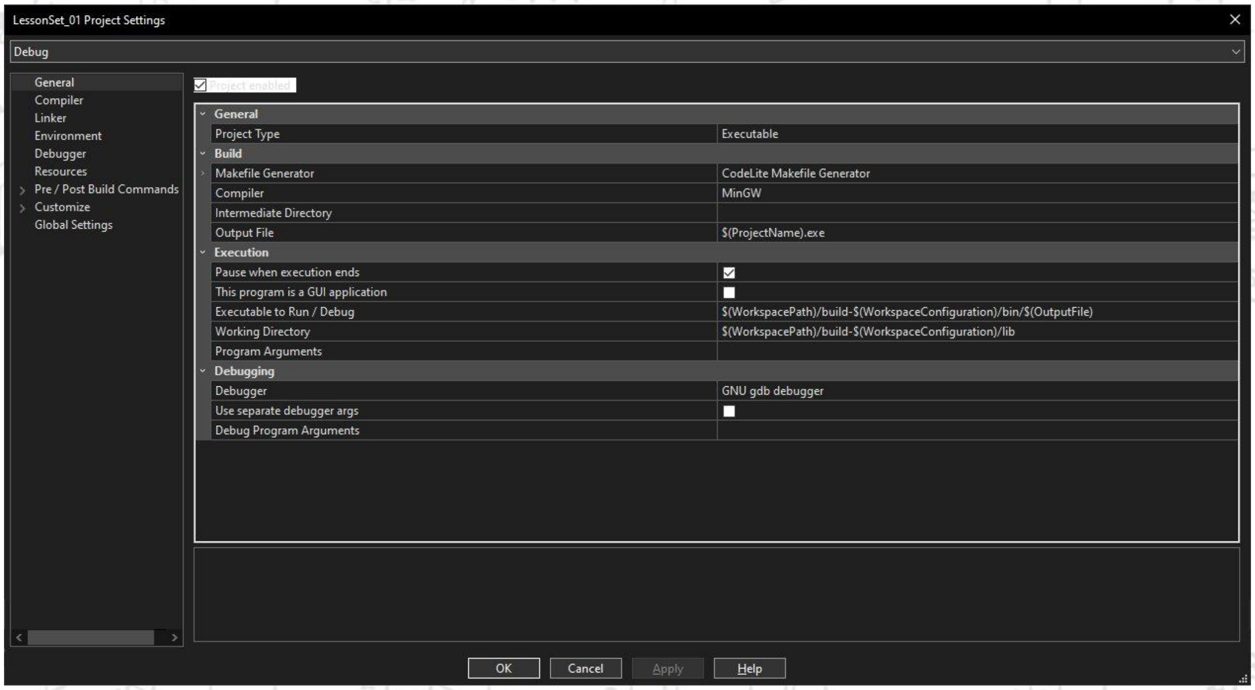
- Debugger: GNU gdb debugger
- Build System: CodeLite Makefile Generator

That should be it, click <**Ok**> and a brand new shiny project should populate under the workspace tree in left pane.

2.5 Updating the Project Settings - we are almost out of the woods! I told you embedded engineering is fun! Anyway, next we need to set some project settings relating to the "**General**", "**Compiler**", and "**Linker**" categories. To get to the project settings, there are a couple ways, but the most straightforward is to right click on the project itself, and select "**Settings…**" this will bring you to the mast project settings dialog which has tabs for every category and is very complex. Again, the video lecture goes into detail, this here is just for reference.

Once in the Project Settings dialog you will see something like this dialog.

Figure 5.0 - CodeLite Project Settings Dialog.



General	
Project Type	Executable
Build	
Makefile Generator	CodeLite Makefile Generator
Compiler	MinGW
Intermediate Directory	
Output File	$(ProjectName).exe
Execution	
Pause when execution ends	✓
This program is a GUI application	☐
Executable to Run / Debug	$(WorkspacePath)/build-$(WorkspaceConfiguration)/bin/$(OutputFile)
Working Directory	$(WorkspacePath)/build-$(WorkspaceConfiguration)/lib
Program Arguments	
Debugging	
Debugger	GNU gdb debugger
Use separate debugger args	☐
Debug Program Arguments	

We need to make changes in the General, Compiler, and Linker categories. Let's begin:

- General Category: Select the General category from the drop down on the left pane, and locate the "Working Directory" option under the "**Execution**" section, change it to **$(ProjectPath)**, and you're done here. This helps the project build in the right place.

- Linker Category: Select the Linker category, and fine the "**Linker Options**" field, and add to it "**-static**", that's it you're done. This helps the linker use local libs instead of DLLs.

- Compiler Category: This one is the most complex, so I saved the best for last! Select the Compiler category, then make the following additions to these "Options":

- C++ Compiler Options: Add "**-std=c++11"; -Wall; -save-temps; -fverbose-asm**", don't forget the semicolons after each compiler flag. These set the compiler for C++ 11, show all warnings, and allow us to generate ASM files later in the course.

Click <**Apply**>, and re-verify everything has changed in all categories, and then close the dialog with <**Ok**>.

That should be all! The IDE and compiler should be ready to go.

Links/Resources:

https://codelite.org/
https://www.mingw-w64.org/downloads/
https://winlibs.com/

Section 2: Introduction to Microprocessors and Microcontrollers

In this introductory section, we are going to jump right in and learn what a microprocessor is, how it differs from a microcontroller. We will talk about the internal architecture of a processor, how they execute instructions (assembly language), and dig into concepts like ALUs (arithmetic logic units), memory, and more. We will finish off with a gentle introduction to the Arduino Uno platform and learn a little about the 8-bit processor that powers it.

Lecture 1 - "Microprocessors and Computer Architecture Fundamentals"

This lecture introduces the course and we start discussing classic microprocessors and basic computer architecture.

Links/Resources:

Design Your Own Game Console eBook

Lecture 2 - "Microprocessors, Microcontrollers, ALUs, Assembly Language and More"

This lecture covers how microprocessors work internally. We cover classic 8-bit processors like the 6502, learn how an ALU works, Von Neumann vs. Harvard Architecture. We learn what machine and assembly language are as well as covers microcontrollers which are complete computers on a single IC, or otherwise known as systems on a chip or SOCs. Lastly, we learn what a "datasheet".

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/MOS_Technology_6502

https://web.archive.org/web/20221112225344if_/http://archive.6502.org/datasheets/synertek_sy6500_microprocessors_apr_1979.pdf

https://www.westerndesigncenter.com/wdc/documentation/w65c02s.pdf
https://www.visual6502.org/
https://en.wikipedia.org/wiki/List_of_common_microcontrollers

Lecture 3 - "Processor Design Primer: The ALU, Control Unit, Register Transfer Logic and Processor Fundamentals"

This lecture covers lower level processor design and how exactly machine "**instructions**" are implemented at the machine code level. We review **RTL** or "**Register Transfer Logic**" as a shorthand used to describe these operations as well as learn that a processor is nothing more than a conmplex state machine with gates and enables directing the flow of data. We deconstruct single machine or assembly language instructions to really understand how a processor performs computation. Finally, we continue learning about the elements or blocks of typical processors/microcontrollers and finish off with some simulation and seeing how C/C++ is converted to assembly language with an online tool.

Links/Resources:

Design Your Own Game Console
www.godbolt.org
https://www.westerndesigncenter.com/wdc/documentation/w65c02s.pdf

Lecture 4 - "Introduction to the Arduino, Hardware, Software, Documentation, and Datasheet"

In this lecture we take a look at the Arduino Technology platform, the hardware, software, API, IDE. See some physical development boards as well as review some other processors from history on the bench as well as a surpise guest embedded system featuring "**Baby Yoda**"!

Links/Resources:

Design Your Own Game Console

https://www.arduino.cc
https://en.wikipedia.org/wiki/Arduino
https://docs.arduino.cc/resources/pinouts/A000066-full-pinout.pdf
https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

Section 3: Arduino IDE, Coding, and Hardware Primer

In this section we are going to explore all the elements of the Arduino platform including the IDE, coding in C/C++, building some hardware on the bench, and simulation. This section is designed to give you an idea of where we are going in the course ***before*** we get there while having some fun at the same time. By the time you finish this section you will have a vastly better idea of how all the pieces of the Arduino ecosystem fit together and how programs are written, compiled, uploaded, and hardware built.

Finally, you will even see how to build a working Simon memory game with hardware and firmware running on the Arduino!

Lecture 1 - "Introduction to the Arduino UNO, Hardware, IDE, API, Programming and Simulation"

In this lecture, we introduce the Arduino hardware, the IDE, how to find information on the Arduino site. We talk about C/C++ a bit and how source code is compiled into machine code and the entire process of how the compiler and tool chain works from end to end.

Links/Resources:

Design Your Own Game Console eBook
https://www.arduino.cc

Lecture 2 - "Hands on with the Arduino, Parts Kits, Solderless Breadboards and Component Introductions"

In this lecture, we introduce the Arduino hardware, the IDE, how to find information on the Arduino site. We talk about C/C++ a bit and how source code is compiled into machine code and the entire process of how the compiler and tool chain works from end to end.

Links/Resources:

Design Your Own Game Console eBook
https://www.arduino.cc

Lecture 3 - "Blinking an LED with C++ in Hardware"

In this lecture, we use real Arduino UNO hardware, the IDE and some C++ code to blink LEDs, and make changes in real-time, so you can see how things fit together. This is the quintessential "**Hello World**", but in hardware rather than software!

Links/Resources:

Design Your Own Game Console eBook
https://www.arduino.cc

Lecture 4 - "Simulating our Arduino Hardware and Code Online with TinkerCAD and Wokwi"

In this lecture, we take a look at two of the most popular online simulators for Arduino and how they work. We will take our simple blinking LED program and port it to these simulators, see how to work with these tools and the pro's and con's between them. This is important since it allows rapid prototyping of your Arduino and other embedded platforms as well as for those that do NOT have physical hardware.

Links/Resources:

Design Your Own Game Console eBook
https://www.arduino.cc

Arduino IDE:	CCAMD_Arduino_Blink_01
TinkerCAD:	CCAMD_Tinker_Blink_01
	CCAMD_HelloWorld
Wokwi:	CCAMD_Wokwi_Blink_01.ino
	https://wokwi.com/projects/372377471560023041

Lecture 5 - "More Simulation with Code API Emulation on Replit"

In this lecture, we write a very crude "**Arduino Simulator**" online in C++ on the **replit.com** platform (which you should have a free account on at this point). The simulator runs our blink LED program on virtual hardare and uses "print" statements to turn on/off the LEDs. This is a great lecture to understand how more advanced tools are written and used.

Links/Resources:

Design Your Own Game Console eBook
replit.com: CCAMD_Replit_Blink_01

Lecture 6 - "Building a Game From Scratch: The Simon Memory Game, Hardware Design"

In this lecture, we do the electrical design for a classic "Simon" memory game using the Arduino as the hardware platform.

Links/Resources:

Design Your Own Game Console eBook
https://www.waitingforfriday.com/?p=586
https://en.wikipedia.org/wiki/Simon_(game)

Lecture 7 - "Building a Game From Scratch: The Simon Memory Game, Hardware Build"

In this lecture, we build the physical hardware by adding in all the LEDS, buttons, power management, and rest of our previous hardware design.

Links/Resources:

Design Your Own Game Console eBook
https://www.waitingforfriday.com/?p=586
https://en.wikipedia.org/wiki/Simon_(game)

Lecture 8 - "Building a Game From Scratch: The Simon Memory Game, Coding and Firmware"

In this lecture, we write the foundation of the firmware for our Simon game. We add button presses, LED output, buzzer and tone generation.

Links/Resources:

Design Your Own Game Console eBook
https://www.waitingforfriday.com/?p=586
https://en.wikipedia.org/wiki/Simon_(game)

Arduino IDE:	CCAMD_Arduino_IDE_Simon_01
	CCAMD_Arduino_IDE_Simon_02
	CCAMD_Arduino_IDE_Simon_03

Lecture 9 - "Finishing the Simon Game and Introduction to Serial Debugging"

In this lecture, we explore crude "debugging" via the serial terminal and test it out on our Simon hardware platform.

Links/Resources:

Design Your Own Game Console eBook
https://www.waitingforfriday.com/?p=586
https://en.wikipedia.org/wiki/Simon_(game)

Arduino IDE: CCAMD_Arduino_IDE_Simon_03

Lecture 10 - "Porting Simon to Online Simulation"

In this lecture, we port the Simon hardware and firmware to the TinkerCAD simulation platform to show how you can completely simulate a hardware/firmware design (within reason) with online simulators.

Links/Resources:

Design Your Own Game Console eBook
https://www.waitingforfriday.com/?p=586
https://en.wikipedia.org/wiki/Simon_(game)

Arduino IDE:	CCAMD_Arduino_IDE_Simon_03
TinkerCAD:	CCAMD_Tinker_Simon_01 CCAMD_Tinker_Simon_02

Section 4: Tools and Test Equipment Overview

In this section we get hands on (literally) with the tools commonly used to build electronics such as wire cutters, soldering irons, hot air machines, wire wrapping, etc. Additionally, we learn about electronic test equipment such as digital multimeters, oscilloscopes, logic analyzers used to make measurements of your circuits and visualize signals in real-time.

Lecture 1 - "Basic Hand Tools for Electronics"

In this lecture, we review a number of basic hand tools for working with electronics and small parts.

Links/Resources:

Design Your Own Game Console eBook

Lecture 2 - "Working with Soldering Irons and Hot Air Machines without Burning Yourself!"

This lecture introduces the course and we start discussing classic microprocessors and basic computer architecture.

Links/Resources:

Design Your Own Game Console eBook

See "Parts List" PDF for suggestions for soldering equipment and supplies.

Lecture 3 - "Multimeters, Oscilloscopes, Logic Analyzers, Signal Generators and Power Supplies"

In this lecture, we take a look at electronic test equipment such as multimeters, oscilloscopes, logic analyzers, signal generators, power supplies and more.

Links/Resources:

Design Your Own Game Console eBook

See "Parts List" PDF for suggestions for test equipment such as meters, oscopes, and logic analyzers.

Lecture 4 - "Hands on with Test Equipment and Real-Time Firmware Experiments with Logic Analyzer"

In this lecture, we learn how to use the oscilloscope as well as the logic analyzer to investigate the Arduino firmware uploading/downloading protocol and understand the strengths and weaknesses of each tool.

Links/Resources:

Design Your Own Game Console eBook

See "Parts List" PDF for suggestions for test equipment such as meters, oscopes, and logic analyzers.

Arduino IDE:	CCAMD_Arduino_IDE_Simon_01
	CCAMD_Arduino_IDE_Simon_02
	CCAMD_Arduino_IDE_Simon_03

https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP

Section 5: Into the Abyss - Electronics Theory and Fundamentals Primer

In this section of the course we introduce basic electrical and electronic theory. This section is for those with little or no knowledge of basic electronics (or you are rusty). If you have a solid understanding of Ohm's law, Kirchhoff's law, electronics and electrical engineering then you may want to skim this section. However, in addition to the basics of current, voltage, AC, DC, resistors, capacitors, inductors, transformers, transistors, gates, and more, we are also going to work with various circuit simulators, which later we will use for more complex simulations, so this material will benefit everyone. This section is a very abridged and compressed version of the electronics material covered in my “**Crash Course Electronics and PCB Design**” with all new lectures of course, emphasizing our embedded engineering goals.

Lecture 1 - "The Fundamentals of Electricity, Charge, Current, Voltage, Resistance and Fields"

In this lecture, we cover the fundamentals of charge, electric and magnetic fields as they relate to current and voltage.

Links/Resources:

Design Your Own Game Console eBook

Lecture 2 - "Ohm’s Law, Basic Circuit Analysis, Series and Parallel Circuits”

In this lecture, we delve into the single most important law in electronics, Ohm's Law. We learn how to analyze simple series and parallel DC circuits, as well as different approaches to the process.

Links/Resources:

Design Your Own Game Console eBook

Lecture 3 - "Understanding Power, Series and Parallel Batteries, Voltage Dividers and Using EasyEDA"

In this lecture, we continue our electronics basic with some more circuit analysis techniques, we learn about "power", as well as fire up the PCB tool we will be using for simulation called EasyEDA.

Links/Resources:

Design Your Own Game Console eBook
EasyEDA: CCAMD_Voltage_Divider_01

https://www.youtube.com/@easyeda2164

Lecture 4 - "Ohm's Law++, Kirchhoff's Laws; KVL and KCL"

In this lecture, we are going to delve into more advanced circuit analysis tools such as Kirchhoff's current and voltage laws, work some problems, build some hardware and then move onto simulation to check out our results.

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/Kirchhoff%27s_circuit_laws

EasyEDA: CCAMD_Voltage_Divider_01
CCAMD_Current_Divider_01

Lecture 5 - "Capacitors, Electric Fields, Physics, Charging and Discharging"

In this lecture, we learn more about capacitors, their construction, modeling equations, and how they charge and discharge. We build and simulate our findings.

Links/Resources:

Design Your Own Game Console eBook
http://hyperphysics.phy-astr.gsu.edu/hbase/electric/capchg.html
http://hyperphysics.phy-astr.gsu.edu/hbase/electric/capdis.html#c2
https://en.wikipedia.org/wiki/Capacitor
https://en.wikipedia.org/wiki/Dielectric
https://www.electronics-tutorials.ws/rc/rc_1.html
https://www.electronics-tutorials.ws/rc/rc_2.html

EasyEDA: CCAMD_Cap_Charge_LED_01

Lecture 6 - "Inductors, Magnetic Fields, Flux, Lenz's Law, Faraday's Law, Charging and Discharging"

In this lecture, we learn about inductors, how they work, and the laws that govern them. Finally, we take a look at engergizing and de-energizing them and compute current and voltage.

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/Inductor
https://www.electronics-tutorials.ws/inductor/lr-circuits.html
https://en.wikipedia.org/wiki/Lenz%27s_law
https://en.wikipedia.org/wiki/Faraday%27s_law_of_induction

Lecture 7 - "Reactance, Impedance, Phasors, AC Analysis of Series Capacitive Circuit"

In this lecture, we delve into the black art of "reactance", "impedance" and AC circuit analysis techniques that will be helpful to understand how to analyze simple AC circuits.

Links/Resources:

Design Your Own Game Console eBook
https://www.electronics-tutorials.ws/rc/rc_2.html
https://www.electronics-tutorials.ws/filter/filter_2.html
https://www.electronics-tutorials.ws/filter/filter_1.html
https://www.electronics-tutorials.ws/capacitor/capacitive-voltage-divider.html
https://en.wikipedia.org/wiki/Electrical_reactance
https://en.wikipedia.org/wiki/Electrical_impedance

Lecture 8 - "Low Pass Filters, Transfer Functions, Gain, Decibels"

In this lecture, we learn about filters, transfer functions (how a network changes an input signal at its output), gain, and work through a number of problems filter problems.

Links/Resources:

Design Your Own Game Console eBook
https://www.electronics-tutorials.ws/filter/filter_2.html
https://www.electronics-tutorials.ws/filter/filter_1.html
https://www.electronics-tutorials.ws/filter/decibels.html
https://en.wikipedia.org/wiki/Decibel

Lecture 9 - "Low Pass Filter Bench Build, Simulation, Arduino Tone Generator"

In this lecture, we take the theory learned about filters and apply it to a bench build and simulation of the tone generator.

Links/Resources:

Design Your Own Game Console eBook
https://www.electronics-tutorials.ws/filter/filter_2.html
https://www.electronics-tutorials.ws/filter/filter_1.html
https://en.wikipedia.org/wiki/Square_wave

Arduino IDE:	CCAMD_Arduino_IDE_SigGen_LowPass_01
TinkerCAD:	CCAMD_Tinker_SigGen_01
EasyEDA:	CCAMD_Low_Pass_Filter_01

Lecture 10 - "Basic Semiconductor Devices, Diodes, Rectification, LEDs, Simulation"

In this lecture, we learn about diodes (used for rectification and steering currents), their theory, and run some simulations.

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/William_Shockley

TinkerCAD: CCAMD_Tinker_Diodes_Lab_01

Lecture 11 - "Transformers, DC Rectification, Understanding Datasheets"

In this lecture, we learn about transformers and how to design a simple half wave rectifying power supply. Additionally, we cover some tips on how to read data sheets.

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/Transformer
https://en.wikipedia.org/wiki/Rectifier#Half-wave_rectification

Note: The diode used in this lecture is the 1N4001, any model will work from this series, including the larger 1N4007 inside the Elegoo Kit(s). Additionally, we use a transformer in this experiment, as noted in lecture this can be quite dangerous since its connected to line AC. But, if you want to follow along, you will need a 110V to 12V @ 100-300mA transformer. Digikey and many other sources have 1000s of options.

Lecture 12 - "Transistor Theory and Applications"

In this lecture, we cover the bipolar transistor and its operation, theory, and run some experiments with it using a motor and Arduino.

Links/Resources:

Design Your Own Game Console eBook
https://www.electronics-tutorials.ws/transistors/tran_1.html
https://www.mouser.com/datasheet/2/149/2N3904-82270.pdf
https://www.mouser.com/datasheet/2/149/SS8050-117753.pdf

https://www.mouser.com/datasheet/2/149/PN2222A-889968.pdf
https://www.allaboutcircuits.com/textbook/semiconductors/chpt-4/bipolar-junction-transistors-bjt/

Arduino IDE: CCAMD_Arduino_IDE_Motor_PWM_01
TinkerCAD: CCAMD_Tinker_Motor_01
CCAMD_Tinker_Motor_PWM_01

Lecture 13 - "Mosfet Theory and Practical Applications"

In this lecture, we learn about MOSFETs and how to use them primarily as switches to switch large currents. Then we redesign our previous motor control circuit based on a BJT transistor and upgrade it to use a mosfet.

Links/Resources:

Design Your Own Game Console
https://www.allaboutcircuits.com/video-tutorials/transistors/
https://www.electronics-tutorials.ws/transistor/tran_6.html
https://www.electronics-tutorials.ws/transistor/tran_7.html
https://www.onsemi.com/pdf/datasheet/nds7002a-d.pdf
https://www.onsemi.com/pdf/datasheet/mmbf170-d.pdf
https://www.onsemi.com/pdf/datasheet/fdc606p-d.pdf
https://www.mouser.com/datasheet/2/308/1/FDV305N_D-2312978.pdf
https://www.vishay.com/docs/91047/91047.pdf
https://www.vishay.com/docs/91045/91045.pdf
https://www.digikey.com/en/products/detail/vishay-siliconix/IRF730A/352234

TinkerCAD: CCAMD_Tinker_FET_PWM_01

Mosfet Suggestions: IRF730A, 2N7000, 2N7002 (Kit suggestions in Parts List PDF).

Lecture 14 - "Introduction to Digital Electronics, Boolean Algebra, Gates, and Logic Families"

In this lecture, we delve into the digital realm and learn about Boolean logic, gates, truth tables, base 2 math, and finish with an introduction to popular logic families like TTL, CMOS and LVTTL.

Links/Resources:

Design Your Own Game Console eBook

https://www.ti.com/lit/sg/sdyu001ab/sdyu001ab.pdf?ts=1700356789426&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FCD40106B

https://www.ti.com/lit/sg/scyt129g/scyt129g.pdf?ts=1700363809898&ref_url=https%253A%252F%252Fwww.google.com%252F

https://gab.wallawalla.edu/~larry.aamodt/engr355/ti_szza036a.pdf
https://faculty-web.msoe.edu/johnsontimoj/CE3101/files3101/CMOS_Logic_Databook.pdf

https://www.ti.com/lit/sg/sdyu001ab/sdyu001ab.pdf?ts=1706949902206&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FSN74LS02%253FkeyMatch%253DSN74LS02%2526tisearch%253Dsearch-everything%2526usecase%253DGPN-ALT

https://www.ti.com/lit/sg/scyt129g/scyt129g.pdf?ts=1706911984747&ref_url=https%253A%252F%252Fwww.google.fr%252F

Lecture 15 - "Understanding TTL/CMOS Logic Gates, Driving, Sinking Current, Packages and Prototyping"

In this lecture, we learn more about the signaling in TTL and CMOS, how to interface to them, how to mix them, and about sinking and driving current. We finish up with looking at various packaging options for prototyping and how to work with this devices.

Links/Resources:

Design Your Own Game Console eBook

https://www.ti.com/lit/sg/sdyu001ab/sdyu001ab.pdf?ts=1700356789426&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FCD40106B

https://www.ti.com/lit/sg/scyt129g/scyt129g.pdf?ts=1700363809898&ref_url=https%253A%252F%252Fwww.google.com%252F

https://gab.wallawalla.edu/~larry.aamodt/engr355/ti_szza036a.pdf
https://faculty-web.msoe.edu/johnsontimoj/CE3101/files3101/CMOS_Logic_Databook.pdf
https://www.ti.com/lit/ds/symlink/sn74ls04.pdf
https://www.diodes.com/assets/Datasheets/74LV32A.pdf

https://www.ti.com/lit/ds/symlink/sn54ls161a.pdf?ts=1700364278528&ref_url=https%253A%252F%252Fwww.google.com%252F

https://beldynsys.com/

Master IC List Location in course master ZIP file:
Root\..\MiscImagesChartsEtc\Master_IC_List_01.txt

Lecture 16 - "Review of Common TTL/CMOS ICs and Bench Demo of Gates"

In this lecture, we build a TTL gate tester on the bench and compare results with simulation as well as see a couple problems that are common when building circuits and how to handle them.

Links/Resources:

Design Your Own Game Console eBook

https://www.ti.com/lit/ds/symlink/sn74ls07.pdf?ts=1700343148742

https://www.ti.com/lit/ds/symlink/sn74ls20.pdf?ts=1700364784422&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/lit/ds/sdls109/sdls109.pdf

https://www.ti.com/lit/ds/symlink/sn54ls138-sp.pdf?ts=1700335116000&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/lit/ds/symlink/sn74ls74a.pdf

https://www.ti.com/lit/ds/symlink/sn54ls161a.pdf?ts=1700364278528&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700308635589&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/lit/ds/symlink/sn54ls164-sp.pdf?ts=1700283871087
https://www.ti.com/lit/ds/symlink/sn74ls244.pdf

Master IC List Location in course master ZIP file:
Root\..\MiscImagesChartsEtc\Master_IC_List_01.txt

TinkerCAD: CCAMD_Tinker_Logic_Gates_01

Extra IC Parts: 74LS32 Quad 2-Input OR (TTL/HC kit suggestion in Parts List PDF).

Lecture 17 - "Simulating a Digital Logic Gate Module Using the Arduino"

In this lecture, we use the Arduino as a platform to build a universal logic simulator. We write firmware, build hardware, and re-create our previous virtual simulation in real hardware.

Links/Resources:

Design Your Own Game Console eBook

https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

Arduino IDE: CCAMD_Arduino_IDE_LogicModule_01
TinkerCAD: CCAMD_Tinker_LogicModule_01

Lecture 18 - "A Tour of Programmable Logic (RAM, ROM, FLASH, PALs, GALs, CPLD, FPGA) and Bench Demo"

In this lecture, we learn about programmable memories such as ROM, PROM, EEPROM, FLASH, and more. Then we take a look at programmable logic such as PALs, GALs, CPLDs, and FPGAs. We finish off with actually desiging an IC and programming it in CUPL (a hardware programming language), downloading, and testing our creation!

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/Read-only_memory
https://en.wikipedia.org/wiki/EPROM
https://en.wikipedia.org/wiki/EEPROM
https://en.wikipedia.org/wiki/Static_random-access_memory
https://en.wikipedia.org/wiki/Programmable_logic_array
https://en.wikipedia.org/wiki/Programmable_Array_Logic
https://www.datasheetq.com/pdf-view/GAL20LV8-Lattice
https://ww1.microchip.com/downloads/en/DeviceDoc/doc0737.pdf
https://www.microchip.com/en-us/development-tool/wincupl
https://www.microchip.com/en-us/products/fpgas-and-plds/spld-cplds/pld-design-resources
https://en.wikipedia.org/wiki/PALASM
https://en.wikipedia.org/wiki/Complex_programmable_logic_device
https://en.wikipedia.org/wiki/Field-programmable_gate_array
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
https://www.microchip.com/en-us/product/atf22v10c

The CUPL files we generate, tools, tutorials, etc. can all be found in the course master ZIP file here:

Root\..Sources\CUPL

Section 6: C++ (along with C) Primer and Fundamentals from the Ground Up

In this section of the course we are going to cover C/C++ from the ground up with emphasis on C++ features (C is a subset of C++). C/C++ are the most popular languages that microcontrollers are programmed in; however, programming C/C++ on a desktop implementation of C/C++ can differ wildly with a microcontroller implementation of the languages due to memory and performance constraints. Therefore, this section is a general C/C++ primer as well as a practical guide on how to approach coding in these languages on microcontrollers. Therefore, as the lectures progress, we will work with both the desktop compiler, as well as our target Arduino UNO to see how things differ in C++. We are going to cover so many cool ideas and coding tricks in this section, you definitely want to take your time and go through it carefully.

Lecture 1 - "Introducing C/C++, History and Programming Concepts for Embedded Systems"

In this lecture, we begin our study of C++ formally. We begin with the fundamentals of the language, its roots, history, and why its so popular in embedded development.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B

Arduino IDE: CCAMD_Arduino_IDE_Starter_01

Lecture 2 - "Understanding C/C++ Compilers, Code Generation and Tools"

In this lecture, we go on a tour de force of how a compiler works. All of the questions you ever had about, "how exactly does a line of code get compiled and ran on a piece of target hardware", are answered here in gory detail. Additionally, we take a look at an array of tools that we will use to code with from desktop compilers to online tools and emulators.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://www.ghs.com/

Arduino IDE:	CCAMD_Arduino_IDE_Starter_01
replit.com:	CCAMD_Replit_Starter_01
TinkerCad:	CCAMD_Starter_01
Codelite:	CCAMD_StarterWorkSpace

Lecture 3 - "Working with IDEs, Compiling C++ Programs, and a Bit of Arduino Code"

In this lecture, we start writing some simple code to print to the screen and see how the process changes from tool to tool as well as how coding on the Arduino is different from desktop C++ coding. Additionally, we learn about workspaces, projects, and compiler setup details that are very important.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://www.arduino.cc
https://www.cplusplus.com/
https://www.codelite.org/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
replit.com:	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork

Lecture 4 - "C++ Primer Crash Course, Arduino Emulation and Coding"

In this lecture, we begin writing code in C/C++ in three different environments; CodeLite, the Arduino IDE, and Replit.com online. Additionally, we take a deeper look out how the classic entry function in C/C++ main() implements the Arduino functions setup() and loop(), which is a point of confusion for everyone learning Arduino C/C++ programming differences.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://www.arduino.cc
https://www.cplusplus.com/
https://www.codelite.org/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
replit.com:	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork

Lecture 5 - "Setting up our C++ Test Platforms"

In this lecture, we take a look at how C/C++ gets converted into a final executable and how we can dissassemble that file and see the assembly language generated for your programs. Additionally, we setup our programming environments for our C++ development online, on the desktop and on the Arduino.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format

https://www.sublimetext.com/
https://www.arduino.cc
https://www.cplusplus.com/
https://www.codelite.org/

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp

Lecture 6 - "C++ Fundamentals, Headers, External Libraries, Printing"

In this lecture, we start working with C++, learning about ***main()***, the syntax of C/C++, and how to print to the console with ***printf()***, as well as a deep dive into the infamous problems with carriage return and linefeed on terminals and consoles.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/w/cpp/language/escape
https://www.arduino.cc/reference/en/language/functions/communication/serial/
https://www.arduino.cc
https://www.cplusplus.com/

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Basics_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp
main_basics_01.cpp

Lecture 7 - "C++ Variables, Intrinsic Types, Naming Conventions, and more printf() Formatting"

In this lecture, we start coding with a gentle introduction to variables, built in types, some naming conventions, and revisit the ***printf()*** function since we will be using it a lot to print results out.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf

https://www.arduino.cc
https://www.cplusplus.com/

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Basics_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp
main_basics_01.cpp

Lecture 8 - "C++ Variables, Globals, Locals, char, int, float, boolean types, math operators, and precedence, preprocessor #define"

In this lecture, we continue our work with C++, learn more about data types, sizes of data, math operations and precedence, and a painless introduction to loops and the pre-processor.

Links/Resources:

Design Your Own Game Console eBook
https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_01.cpp

Lecture 9 - "C++ Mathematics, Understanding Base-N Number Systems with Binary, Hexadecimal & Octal"

In this lecture, we take a short detour and review binary, hex and octal number systems.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language

https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_01.cpp

Lecture 10 - "C++, Programming Style Guide and Conventions, Macros, Conditional Compilation"

In this lecture, we take a closer look at the C/C++ pre-preprocessor and how to use it to create constants, macros and conditional compilation. Additionally, we talk about coding style and review some time tested conventions that many programmers use to write understandable code.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://google.github.io/styleguide/cppguide.html

C++ Style Guide in course master ZIP file:
Root\..\MiscImagesChartsEtc\CCAMD_Cpp_Style_Guide_Tips.txt

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp
main_basics_01.cpp

Lecture 11 - "C++, Built in Math Operators, Binary Bitshift Operations and Using the Debugger"

In this lecture, we review more math operators including bitwise logic operations and the shift operators. We also, delve into using the debugger to see our code execute and step thru it.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://google.github.io/styleguide/cppguide.html

C++ Style Guide in course master ZIP file:
Root\..\MiscImagesChartsEtc\CCAMD_Cpp_Style_Guide_Tips.txt

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Basics_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp
main_basics_01.cpp

Lecture 12 - "More C++ Logical Operators, Conditionals and the "if" statement, Type Casting"

In this lecture, we cover a few more math operators then we take a look conditional statements and how to write "if" statements with boolean expressions, finally we cover type casting.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://cplusplus.com/reference/clibrary/
https://en.cppreference.com/w/cpp/language/if
https://cplusplus.com/doc/oldtutorial/typecasting/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_01.cpp

Lecture 13 - "C++ Looping Constructs, 'for' and 'while' "

In this lecture, we take a look at how to write code that repeats using "**while**," and "**for**" loops. We also look at some of the new syntax available in C++ 11 and newer versions to help simplify loop iteration.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://cplusplus.com/reference/clibrary/
https://en.cppreference.com/w/cpp/language/if
https://cplusplus.com/doc/oldtutorial/typecasting/
https://www.w3schools.com/cpp/cpp_while_loop.asp
https://www.w3schools.com/cpp/cpp_for_loop.asp

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_01.cpp

Lecture 14 - "C++ Switch Statements, Arrays, Strings"

In this lecture, we take a look at a more efficient way to compute a n-way conditional using the "**switch**" statement, next we dig deeper into arrays and their memory layout. Finally, we talk a bit about the "**string**" class.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B

https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_01.cpp
	main_basics_02.cpp

Lecture 15 - "C++ Strings, Arrays, Arduino Strings and Functions"

In this lecture, we take a closer look at some NULL terminated strings, as well as arrays, functions and the String object in the Arduino API.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Basics_01

replit.com: CCAMD_Replit_Starter_01
CCAMD_Replit_Lesson_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
arduino_starter_01.cpp
main_starter_01.cpp
main_basics_01.cpp
main_basics_02.cpp

Lecture 16 - "C++ Measuring Time, Nested Loops and Hardware Build (Finally!)"

In this lecture, we take a look at some of the time keeping API functions for fun and then we work more on loops, and repeatitive processing. Additionally, we break out EasyCAD, design a circuit, copy it into TinkerCAD, and then build it all on the bench as well as compile and upload the code from the Arduino API. A lot to cover!

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/

https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/

https://easyeda.com/
https://www.tinkercad.com/

Arduino IDE: CCAMD_Arduino_IDE_Lesson_01
CCAMD_Arduino_IDE_Basics_01

replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
	CCAMD_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_02.cpp

Lecture 17 - "C++ Console IO C and C++ methods, VT100 Gamepad Hardware Demo"

In this lecture, we lay the foundation to display characters on the serial terminal, learn about VT100 emulation, and a little bit of hardware demo with a poor man's "**game pad**".

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/

https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://easyeda.com/
https://www.tinkercad.com/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

Lecture 18 - "C++ Data Structures, Structs, Enums, Linked Lists and Trees"

In this lecture, we start our discussion of "**data structures**" and their theoretical basis as well as how to implement them in C++.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/

https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

Lecture 19 - "C++ Pointers and References (The Dreaded Duo)"

In this lecture, we cover one of the most feared subjects in Computer Science; the **pointer**! And in fact, you will see that pointers (and their C++ brothers "**references**") are quite easy to understand and you will wonder what all the drama has been about.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

Lecture 20 - "C++ Classes, Methods, Properties and Object Oriented Basics"

In this lecture, we delve into object oriented programming (OOP), learn the basics and how to create classes in C++, what properties and methods are, and the syntax of it all.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

Lecture 21 - "C++ Advanced Classes, Constructors, Destructors, Operator Overloading"

In this lecture, we continue our discussion of classes and look into class methods, static data, constructors, destructors, copy constructors and much more.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

Lecture 22 - "C++ More Advanced Class Topics, Copy Constructors, Assignments, Deep and Shallow Copying and Memory Management"

In this lecture, we go deeper into C++ objects and learn more about constructors, destructors, shallow and deep copying as well as operator overloading and function overloading.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/

https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

https://cplusplus.com/reference/cstdlib/malloc/
https://cplusplus.com/reference/cstring/strcpy/
https://cplusplus.com/reference/cstdlib/free/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp
	main_basics_04.cpp

Lecture 23 - "C++ Built in Objects and Including External Arduino Libraries"

In this lecture, we take a look at some built in objects that are parts of the standard libraries of C++ and the Arduino variant. These include the "**string**" objects as well as a brief look at SD card support on the Arduino.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format

https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

https://cplusplus.com/reference/cstdlib/malloc/
https://cplusplus.com/reference/cstring/strcpy/
https://cplusplus.com/reference/cstdlib/free/
https://cplusplus.com/reference/string/string/
https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
	CCAMD_Arduino_IDE_Hacking_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp
	main_basics_04.cpp
	main_basics_05.cpp

Lecture 24 - "C++ Understanding Microcontroller Memory Management, Globals, Locals, Statics"

In this lecture, we learn about memory management on microcontrollers and the various memory regions, types, and keywords used in firmware to assert more control of the precious SRAM resources on microcontrollers.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

https://cplusplus.com/reference/cstdlib/malloc/
https://cplusplus.com/reference/cstring/strcpy/
https://cplusplus.com/reference/cstdlib/free/
https://cplusplus.com/reference/string/string/
https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
	CCAMD_Arduino_IDE_Hacking_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp
	main_basics_04.cpp
	main_basics_05.cpp

Lecture 25 - "C++ Memory Management and the Heap Part II"

In this lecture, we finish up our discussion of memory management with exploring in depth how the heap, and stack works. How to allocate dynamic memory in C and C++ and the various issues you need to be aware of when using dynamic memory allocation, especially on microcontrollers with limited RAM resources.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/

https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

https://cplusplus.com/reference/cstdlib/malloc/
https://cplusplus.com/reference/cstring/strcpy/
https://cplusplus.com/reference/cstdlib/free/
https://cplusplus.com/reference/string/string/
https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/
https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.arduino.cc/reference/en/language/variables/utilities/progmem/
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.nongnu.org/avr-libc/user-manual/group__avr__pgmspace.html

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
	CCAMD_Arduino_IDE_Hacking_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
	CCAMD_Replit_Hacking_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp

main_basics_04.cpp
main_basics_05.cpp

Lecture 26 - "C++ Working with Arduino External Libraries and Objects and a Little Recursion"

In this lecture, we see how to work with external source and header files in both the desktop C/C++ compiler as well as the quirky Arduino IDE. Additionally, I sneak in discussions of basic "**recursion**" and how to make code call itself, call itself, call itself… Get ready for your brain to explode!

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.wikipedia.org/wiki/Executable_and_Linkable_Format
https://cplusplus.com/reference/cstdio/
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
https://www.cplusplus.com/doc/tutorial/variables/
https://en.cppreference.com/w/c/io/fprintf
https://en.cppreference.com/w/cpp/language/operator_precedence
https://cplusplus.com/doc/hex/
https://en.cppreference.com/w/cpp/language/switch
https://cplusplus.com/reference/cstring/
https://cplusplus.com/reference/ctime/
https://www.arduino.cc/reference/en/language/functions/time/millis/
https://www.arduino.cc/reference/en/language/functions/communication/serial/
http://www.braun-home.net/michael/info/misc/VT100_commands.htm
https://cplusplus.com/doc/tutorial/structures/
https://www.geeksforgeeks.org/pointers-vs-references-cpp/

https://www.encyclopedia.com/computing/news-wires-white-papers-and-books/object-oriented-languages

https://cplusplus.com/reference/cstdlib/malloc/
https://cplusplus.com/reference/cstring/strcpy/
https://cplusplus.com/reference/cstdlib/free/
https://cplusplus.com/reference/string/string/
https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.arduino.cc/reference/en/language/variables/utilities/progmem/
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.nongnu.org/avr-libc/user-manual/group__avr__pgmspace.html
https://roboticsbackend.com/arduino-create-library/
https://en.wikipedia.org/wiki/Recursion_(computer_science)

Arduino IDE:	CCAMD_Arduino_IDE_Lesson_01
	CCAMD_Arduino_IDE_Basics_01
	CCAMD_Arduino_IDE_VT100_Snake_01
	CCAMD_Arduino_IDE_Hacking_01
	CCAMD_Arduino_IDE_LibraryTest_01
replit.com:	CCAMD_Replit_Starter_01
	CCAMD_Replit_Lesson_01
	CCAMD_Replit_Hacking_01
TinkerCAD:	CCAMD_Tinker_RGB_Buttons_01
EasyEDA:	CCAMD_Arduino_RGB_Buttons_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	arduino_starter_01.cpp
	main_starter_01.cpp
	main_basics_03.cpp
	main_basics_04.cpp
	main_basics_05.cpp

Section 7: Introducing the World of Arduino, AVR 8-bit Processors, Atmega 328p, Hardware and Firmware

Now that we have the fundamentals of C++ under control, in this section, we're going to explore coding on the Arduino platform as a general representative of a typical microcontroller development platform. We are going to dig into the architecture of the Arduino UNO and the ATMega 328P processor and use it to help understand GPIO pins, peripherals, and the various blocks of a microcontroller. Additionally, we will build a number of projects to interface to real world devices along the way to make it interesting.

Lecture 1 - "Introduction to Arduino AVR 8-Bit Architecture; 8-bit Ports, GPIOs and ADC"

In this lecture, we take a detailed look at the inside of the ATMega328P processor used in the Arduino Uno. Learn how the block of the processor fit together, how to find information in the datasheet, and how the GPIO blocks as well as others work at the low level.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus/avr-mcus

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01 CCAMD_Arduino_IDE_DIP_7Segment_01
replit.com:	CCAMD_Replit_Hacking_01
TinkerCAD:	CCAMD_Tinker_DIP_7Segment_01
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 2 - "Interfacing to 7-Segment Displays, Switches, Keypads, Rotary Encoders and More"

In this lecture, we review some basics about switches and terminology and then take a look at some hardware like membrane keypads, 7-segment displays and drivers, and rotary encoders. We prepare for a hardware build later down the line.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-

Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus/avr-mcus

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://m.littelfuse.com/~/media/commercial-vehicle/datasheets/switches/littelfuse-switch-diagrams-082616-pdf.pdf?la=en
https://www.we-online.com/components/products/datasheet/157119V12701.pdf

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).
Elegoo membrane datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_HelloKeypad_01
replit.com:	CCAMD_Replit_Hacking_01

TinkerCAD: CCAMD_Tinker_DIP_7Segment_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 3 - "Interfacing to 7-Segment Displays, Switches, Keypads, Rotary Encoders, Bench Build Part II"

In this lecture, we build the hardware we have been discussing and see how to use the membrane keypad as well as how rotary encoders work and quadrature encoding schemes.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus/avr-mcus

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://m.littelfuse.com/~/media/commercial-vehicle/datasheets/switches/littelfuse-switch-diagrams-082616-pdf.pdf?la=en
https://www.we-online.com/components/products/datasheet/157119V12701.pdf

https://www.allaboutcircuits.com/projects/how-to-use-a-rotary-encoder-in-a-mcu-based-project/

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).
Elegoo membrane datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_HelloKeypad_01
	CCAMD_Arduino_IDE_HelloKeypad_02
	CCAMD_Arduino_IDE_RotaryEncoder_01
replit.com:	CCAMD_Replit_Hacking_01
TinkerCAD:	CCAMD_Tinker_DIP_7Segment_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 4 - "Filtering and Debouncing Deep Dive"

In this lecture, we work on switch debouncing using our rotary switch as the device under test. We look at both hardware switch debouncing as well as software solutions and implement both as well as take a look in real-time with the oscope at mechanical debouncing problems.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.microchip.com/en-us/products/microcontrollers-and-microprocessors/8-bit-mcus/avr-mcus

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://www.ti.com/lit/ds/symlink/sn74lv14a.pdf?ts=1700912873090 https://www.allaboutcircuits.com/projects/how-to-use-a-rotary-encoder-in-a-mcu-based-project/

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_RotaryEncoder_01
replit.com:	CCAMD_Replit_Hacking_01
EasyEDA:	CCAMD_Arduino_RotaryEncoder_01
TinkerCAD:	CCAMD_Tinker_DIP_7Segment_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 5 - "Analog Interfacing to Potentiometers and Joysticks"

In this lecture, we write code and build up a test fixture to experiment with the analog capabilities of the Arduino and read POTs (potentiometers) and analog thumb sticks. We take this tech and then integrate it into a graphical VT100 demo that moves a cursor around the PC screen via serial terminal.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://www.allaboutcircuits.com/projects/how-to-use-a-rotary-encoder-in-a-mcu-based-project/

https://www.ti.com/lit/ds/symlink/sn74lv14a.pdf?ts=1700912873090

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).
Elegoo joystick (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_RotaryEncoder_01
	CCAMD_Arduino_IDE_VT100_SnakeJoystick_01
replit.com:	CCAMD_Replit_Hacking_01
EasyEDA:	CCAMD_Arduino_AnalogJoystick_01
TinkerCAD:	CCAMD_Tinker_AnalogJoystick_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 6 - "C++ Exercise Converting the Joystick Code to a Class"

In this lecture, we take the code we wrote previously with "C" style functional programming, and convert it to a more object oriented, class based approach. A good example of how to do this as well, as well, as you can decide for yourself if it makes the code easier to understand and use.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-

DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://www.allaboutcircuits.com/projects/how-to-use-a-rotary-encoder-in-a-mcu-based-project/

https://www.ti.com/lit/ds/symlink/sn74lv14a.pdf?ts=1700912873090

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).
Elegoo joystick datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_RotaryEncoder_01
	CCAMD_Arduino_IDE_VT100_SnakeJoystick_02
replit.com:	CCAMD_Replit_Hacking_01
EasyEDA:	CCAMD_Arduino_AnalogJoystick_01
TinkerCAD:	CCAMD_Tinker_AnalogJoystick_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 7 - "Interfacing to LEDs and NeoPixels"

In this lecture, we learn how to interface to neopixels, and in general how the protocol works for these serial RGB LEDs. Additionally, we write a software "**bit-bang**" driver, so you can see exactly how these drivers would be written bare metal without use of an API.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://www.ti.com/lit/ds/symlink/cd4511b.pdf
https://www.ti.com/lit/ds/symlink/sn5447a.pdf?ts=1700952111700

https://www.allaboutcircuits.com/projects/how-to-use-a-rotary-encoder-in-a-mcu-based-project/

https://www.ti.com/lit/ds/symlink/sn74lv14a.pdf?ts=1700912873090

http://world-semi.com/

https://darklessled.com/difference-between-ws2811-and-ws2812/#:~:text=WS2811%20and%20WS2812%20are%20the,much%20lesser%20power%20consumption%20rate.

https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf
https://www.adafruit.com/product/1426

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo rotary encoder datasheet (copy from CD-ROM with kit or link above).
Elegoo joystick datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_LibraryTest_01
	CCAMD_Arduino_IDE_DIP_7Segment_01
	CCAMD_Arduino_IDE_RotaryEncoder_01
	CCAMD_Arduino_IDE_VT100_SnakeJoystick_02
	CCAMD_Arduino_IDE_NeoPixel_Simple_01
	CCAMD_Arduino_IDE_NeoPixelDemo_01
replit.com:	CCAMD_Replit_Hacking_01
TinkerCAD:	CCAMD_Tinker_Neopixels_01
EasyEDA:	CCAMD_Arduino_AnalogJoystick_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Section 8: Advanced Microcontroller Development and Arduino Embedded Internals

In this section of the course we continue our work diving into microcontrollers and their peripherals such as timers, clocking, power systems, UARTs, SPI, I2C and more. We will cover interrupts, polling, multitasking, etc. Additionally, we learn assembly language and how to use it both inline and with external assemblers and how drivers are written. You will learn how exactly the compiler calls functions, passes variables on the stack and registers. This massive section is where we move into more “**intermediate**” embedded engineering concepts and work.

Lecture 1 - "Assembly Language and Microcontroller Fundamentals, Inline AVR 328p"

In this lecture, we explore the mysteries of assembly language, the AVR328 instruction set, and it's programming model. Then we begin our journey by trying some C++ mixed with "inline" assembly language out to perform simple math operations, interact with global variables, and try it all out on our hardware as well as simulation.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php

https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon

https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly
https://www.instructables.com/Command-Line-Assembly-Language-Programming-for-Ard/

Arduino IDE: CCAMD_Arduino_IDE_AsmPlayground_01

TinkerCAD: CCAMD_ASM_Playground_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 2 - "Assembly Language, Memory Operations, Registers, IO Space, GPIOs, Interfacing with C/C++"

In this lecture, we jump into how everything connects and unravel how inline assembly works, how to read and write memory, access processor registers, interface to the IO ports and their memory mapping issues. Next we'll see how to port snippets of C/C++ code line by line to assembly language with some simple examples of loops, and conditionals. Additionally, we see how to interface to C/C++ and access variables created by C/C++.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/atmega_miscellaneous_index.html

Arduino IDE: CCAMD_Arduino_IDE_AsmPlayground_01

TinkerCAD: CCAMD_ASM_Playground_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 3 - "Working with the UARTs in Assembly"

In this lecture, we review how the hardware UART works in the AVR ATMega 328P processor. This is a great generic example of how to analyze any peripheral on any processor, and what the steps are. Then when we understand it, we write assembly language firmware to talk directly to the UART and transmit characters in both real hardware and simulation. This lecture is fundamental to embedded engineering and developing drivers for peripherals.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/atmega_miscellaneous_index.html

https://i.stack.imgur.com/bwCFl.gif

Arduino IDE: CCAMD_Arduino_IDE_AsmPlayground_01

TinkerCAD: CCAMD_ASM_Playground_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 4 - "Using External Assembly Language and Memory Mapped IO"

In this lecture, we continue our coverage of assembly language with learning how to include external assembly language files. This allows us to use the full power of the assembler as well as the syntax and features supported by the assembler, making for much cleaner coding. Additionally, we see how to interface to external assembly and access external variables as well how C/C++ needs to access the symbols and functions in the external files.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 5 - "Using External Assembly Language Part II - Writing a Math API"

In this lecture, we continue learning about assembly language basics and focus on how to interface to external assembly files, how to call functions, how to return values across the C/C++ stack and registers. We see how to perform math operations in assembly in 8/16 bit and how to return the results. Additionally, we use tools like disassemblers to "see" what's going on and what our C/C++ programs along with ASM like as binaries.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf
https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 6 - "Using External Assembly Language Part III -- Accessing Variables, Tables, and Addressing Modes"

In this lecture, we finish off our tour of assembly language (for the most part) by taking a look at some advanced topics like accessing tables in ASM, various addressing modes that are available in assembly programming, and other advanced topics that you might find interesting. Remember, 99% of the time in embedded development you will write code in C/C++, but that 1% of the time, only assembly will do, especially with resource limited processors or very time critical code, so assembly programming is a great skill to have as an embedded engineer.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 7 - "Understanding Microcontroller In Circuit Programming (ICP), AVRDUDE, and Low Level Programming, Configuration of Fuse Bits"

In this lecture, we learn about low level programming of microcontrollers. And how the Arduino's 328P processor can be programmed with an external ICP/ISP programmer. Additionally, we cover code security and "fuse" bits which allow low level control of some of the processor's features.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42093-AVR-ISP-mkII_UserGuide.pdf

https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP
https://www.nongnu.org/avrdude/user-manual/avrdude_3.html
https://shawnhymel.com/622/quick-tip-reading-fuse-bits-in-an-arduino/
https://www.ladyada.net/learn/avr/fuses.html
https://www.engbedded.com/fusecalc/

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 8 - "Understanding Microcontroller In Circuit Programming (ICP), AVRDUDE, and Low Level Programming, Configuration of the Clock Out Fuse Bits -- Part II"

In this lecture, we continue learning about low level programming of the Arduino and build a programmer out of an Arduino to program another Arduino! Then we experiment with the fuse bits, and clock out bits, to see that we can modify them programmatically, then we finish off by using a pro tool from Microchip to upload code directly to the ATMega328P via the ISP port.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42093-AVR-ISP-mkII_UserGuide.pdf

https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP
https://www.nongnu.org/avrdude/user-manual/avrdude_3.html
https://shawnhymel.com/622/quick-tip-reading-fuse-bits-in-an-arduino/
https://www.ladyada.net/learn/avr/fuses.html
https://www.engbedded.com/fusecalc/

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 9 - "Understanding Microcontroller Clocking, Scaling, PLLs and More"

In this lecture, we cover clocking concepts and how microcontrollers are clocked internally, externally, as well as coverage of PLLs (phase locked loops), the difference between crystals and oscillators, and much more. Additionally, we experiment with the clock settings on the ATMega 328P via the fuse bits. Finally, we talk about the watch dog timer and finish with an experiment on the bench.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/
https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42093-AVR-ISP-mkII_UserGuide.pdf

https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP
https://www.nongnu.org/avrdude/user-manual/avrdude_3.html
https://shawnhymel.com/622/quick-tip-reading-fuse-bits-in-an-arduino/
https://www.ladyada.net/learn/avr/fuses.html
https://www.engbedded.com/fusecalc/

https://www.electronics-notes.com/articles/radio/pll-phase-locked-loop/tutorial-primer-basics.php

Arduino IDE:	CCAMD_Arduino_IDE_AsmPlayground_02
TinkerCAD:	CCAMD_ASM_Playground_02
Wokwi:	CCAMD_Wokwi_AsmPlayground_02.ino
CodeLite:	CCAMD_StarterWorkSpace CCAMD_CppPrimerWork main_basics_05.cpp

Lecture 10 - "Understanding Sleep Modes, Interrupts, Polling and Multitasking"

In this lecture, we take a look at interrupts, what they are, why they are needed, and how they work. We contrast this will polling techniques as well and discuss pros and cons.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html

https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42093-AVR-ISP-mkII_UserGuide.pdf

https://docs.arduino.cc/built-in-examples/arduino-isp/ArduinoISP
https://www.nongnu.org/avrdude/user-manual/avrdude_3.html
https://shawnhymel.com/622/quick-tip-reading-fuse-bits-in-an-arduino/
https://www.ladyada.net/learn/avr/fuses.html
https://www.engbedded.com/fusecalc/

https://www.electronics-notes.com/articles/radio/pll-phase-locked-loop/tutorial-primer-basics.php

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 11 - "More Advanced Interrupts and Coding Interrupt Service Routines"

In this lecture, we re-write our polling code and implement it with the INT0 and INT1 interrupts in C++. We see how these interrupts can be triggered externally by pin change events such as rising or falling edges.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-

Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://binary.ninja/
https://binary.ninja/free/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 12 - "Understanding Timers, Counters, PWM and Measuring Temporal Events"

In this lecture, we dig into the timer/counters on the ATMega328P, but the concepts are generic and applicable to any processor. We learn about the common modes of operation, how timers interact with pins, the external world, and how complex signals can be generated with very little hardware.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 13 - "Understanding Timers, Counters, PWM and Timer Generated Interrupts and Waveform Generation"

In this lecture, we take what we learned in the previous lecture and implement a number of timer modes for waveform generation using interrupts. All at the register level without use of any API calls.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf

https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_Timer_Interrupts_01
	CCAMD_Arduino_IDE_Timer_Interrupts_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Timer_Interrupts_01
	CCAMD_Timer_Interrupts_02
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 14 - "Using the Timer and Interrupt Arduino APIs and Pin Change Interrupts"

In this lecture, we use one the Arduino 3rd party APIs to setup and control the interrupts on the processor. Then we take a look at interrupt generated signals vs timer generated signals and how hardware is vastly better than software at generating signals.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://github.com/PaulStoffregen/TimerOne

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_Timer_Interrupts_01
	CCAMD_Arduino_IDE_Timer_Interrupts_02
	CCAMD_Arduino_IDE_Timer_Interrupts_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Timer_Interrupts_01
	CCAMD_Timer_Interrupts_02
	CCAMD_Timer_Interrupts_03
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 15 - "Using the Timer and Interrupt Arduino APIs, Pin Change Interrupts Demo - Part II"

In this lecture, we take a look at a less robust interrupt called the pin change interrupt. It doesn't have the features of INT0/1 interrupt, but still is very useful.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B

https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 16 - "More Advanced Pin Change Interrupts and PIR Sensors"

In this lecture, we finish off the pin change interrupts and their setup and application with a fun exploration of **PIR** "**passive infrared**" sensors and how to set them up to detect motion and then use their signals to cause an interrupt on the processor.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/

https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview
https://www.youtube.com/watch?v=Fx_dQNpZ-x4&t=193s&ab_channel=RicardoMoreno

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo PIR HC-SR501 datasheet (copy from CD-ROM with kit or link above).

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 17 - "The Watchdog Timer 'Woof Woof'"

In this lecture, we take a look at the watchdog timer which is a hardware feature of most processors that allows the processor to reset itself if the code become non-responsive.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01

TinkerCAD: CCAMD_Ext_Interrupts_01
CCAMD_Ext_Interrupts_02
CCAMD_Timer_Interrupts_03
CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03
CCAMD_WatchDog_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_05.cpp

Lecture 18 - "Power Consumption and Sleep Modes"

In this lecture, we learn about the power saving and sleep modes of the ATMega328P processor. All modern microcontrollers and microprocessors have various power savings modes that shut down clocks, and non-essential block. You will see how we can reduce the current over 14,000 times!

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-

Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/

https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 19 - "More Sleep and Power Saving Modes"

In this lecture, we continue our coverage of sleep modes, and power measurements. We measure the current consumed by the processor in various sleep modes, as well as add in the WDT (watch dog timer) to add a wake on interrupt, to show how to write programs that work, sleep, wake, and repeat. Finally, we take our PIR sensor and use it to wake the processor, showing the ideal way to save power until an external event occurs that needs attention.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 20 - "Direct Memory Access (DMA)"

In this lecture, we discuss Direct Memory Access (DMA), what it is, how it works, and why you need to use it in modern processor applications.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B

https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 21 - "Memory Management Units (MMUs), External Bus Interfaces, and External Memories; SRAM, FLASH, DDR RAM, EEPROM"

In this lecture, we talk about everything related to memory management units, external memories, how to interface to them, pros and cons of various memory architectures, and how high end microcontrollers can do everything (and more) a microprocessor can do.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03

	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_05.cpp

Lecture 22 - "Advanced Debugging, Desktop and Embedded Debugging Ideas for Arduino (and new Pickits)"

In this lecture, we explore more advanced debugging with desktop C++ as well as using embedded tools from Microchip to program and debug ATMega328P processors directly through the ISP and JTAG port.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
	CCAMD_Arduino_IDE_PrintFDebug_01
	CCAMD_Arduino_IDE_Debugger_Demo_01
Microchip Studio:	BlinkTest01
	CCAMD_Arduino_IDE_Blink_01
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Lecture 23 - "Optimization Theory, Assembly Optimization Cycle Counting, Fixed Point Math"

In this lecture, we discuss some really important topics in optimization. The lecture covers everything from cycle counting, to algorithms, to memory organization, compiler settings, assembly language, using the debugger, and more. When working with constrained systems like microcontrollers, we must be efficient with our use of resources and learn to think in such a way not to waste them.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-

20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
	CCAMD_Arduino_IDE_PrintFDebug_01
	CCAMD_Arduino_IDE_Debugger_Demo_01
	CCAMD_Arduino_IDE_Optimization_01
	CCAMD_Arduino_IDE_Optimization_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_04.cpp
	main_basics_05.cpp

Lecture 24 - "Optimization Theory, Big O, Fixed Point Math - Part II"

In this lecture, we continue our coverage of optimization techniques and discuss Big-O notation, math tricks, deep algorithm analysis, fixed point math, and static vs. dynamic memory allocation.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/

https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02
CCAMD_Arduino_IDE_Sleep_03
CCAMD_Arduino_IDE_PrintFDebug_01
CCAMD_Arduino_IDE_Debugger_Demo_01
CCAMD_Arduino_IDE_Optimization_01
CCAMD_Arduino_IDE_Optimization_02

TinkerCAD: CCAMD_Ext_Interrupts_01
CCAMD_Ext_Interrupts_02
CCAMD_Timer_Interrupts_03
CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03
CCAMD_WatchDog_01

Wokwi: CCAMD_Wokwi_Sleep_01
CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Lecture 25 - "Advanced Optimization Techniques and Data Structures; Linked Lists, Trees, Searching"

In this lecture, we get serious about data structures and learn about linked lists, trees, how they are constructed, searched, their pros and cons, along with many tips and tricks to make your embedded code more efficient. Additionally, we use our new Big-O notation and tools to analyze everything and predict performance.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmeg

a328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03

	CCAMD_Arduino_IDE_PrintFDebug_01
	CCAMD_Arduino_IDE_Debugger_Demo_01
	CCAMD_Arduino_IDE_Optimization_01
	CCAMD_Arduino_IDE_Optimization_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_04.cpp
	main_basics_05.cpp

Lecture 26 - "Advanced Optimization, Math Optimizations, Look up Tables, Clock Counting and Performance Analysis"

In this lecture, we continue our cycle counting and optimization work and take a look at more techniques to speed our code up such as using bit shifting for math operations such as multiplication, removing unnecessary division operations, taking advantage of numerical insights such as closed sums for series. We analyze in real-time a number of simple algorithms, and see how applying various operations can speed up execution by a factor of 1000%! Finally, we finish off with an exploration of classic lookup tables, and see how they can be used to speed your code up.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmIX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397

https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

Arduino IDE:	CCAMD_Arduino_IDE_Ext_Interrupts_01
	CCAMD_Arduino_IDE_Ext_Interrupts_02
	CCAMD_Arduino_IDE_Ext_Interrupts_03
	CCAMD_Arduino_IDE_PCINT_Interrupts_01
	CCAMD_Arduino_IDE_PCINT_Interrupts_02
	CCAMD_Arduino_IDE_PCINT_Interrupts_03
	CCAMD_Arduino_IDE_WatchDog_01
	CCAMD_Arduino_IDE_Sleep_01
	CCAMD_Arduino_IDE_Sleep_02
	CCAMD_Arduino_IDE_Sleep_03
	CCAMD_Arduino_IDE_PrintFDebug_01
	CCAMD_Arduino_IDE_Debugger_Demo_01
	CCAMD_Arduino_IDE_Optimization_01
	CCAMD_Arduino_IDE_Optimization_02
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03

CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03
CCAMD_WatchDog_01

Wokwi: CCAMD_Wokwi_Sleep_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Lecture 27 - "Advanced Optimization, Taylor and Maclaurin Approximations and Fixed Point Math"

In this lecture, we take a trip on the more mathematical side of optimization theory and look at some approximation techniques based on classic Taylor and Maclaurin series expansions. This is one of the most interesting lectures in the optimization series and shows a very powerful technique that is used time and time again on constrained embedded systems with little compute resources.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

https://en.wikipedia.org/wiki/Taylor_series

https://mitocw.ups.edu.ec/courses/mathematics/18-01sc-single-variable-calculus-fall-2010/unit-5-exploring-the-infinite/part-b-taylor-series/session-98-taylors-series/

https://www.freertos.org/index.html

https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02
CCAMD_Arduino_IDE_Sleep_03
CCAMD_Arduino_IDE_PrintFDebug_01
CCAMD_Arduino_IDE_Debugger_Demo_01
CCAMD_Arduino_IDE_Optimization_01
CCAMD_Arduino_IDE_Optimization_02

TinkerCAD: CCAMD_Ext_Interrupts_01
CCAMD_Ext_Interrupts_02
CCAMD_Timer_Interrupts_03

CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03
CCAMD_WatchDog_01

Wokwi: CCAMD_Wokwi_Sleep_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Lecture 28 - "Understanding Mulitasking and Real-Time Operating Systems for Embedded"

In this lecture, we take a look at the computer science behind multitasking, what it is, how it's implemented, and approaches used on microcontrollers.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

https://en.wikipedia.org/wiki/Taylor_series

https://mitocw.ups.edu.ec/courses/mathematics/18-01sc-single-variable-calculus-fall-2010/unit-5-exploring-the-infinite/part-b-taylor-series/session-98-taylors-series/

https://www.freertos.org/index.html

https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02
CCAMD_Arduino_IDE_Sleep_03
CCAMD_Arduino_IDE_PrintFDebug_01
CCAMD_Arduino_IDE_Debugger_Demo_01
CCAMD_Arduino_IDE_Optimization_01
CCAMD_Arduino_IDE_Optimization_02
CCAMD_Arduino_IDE_Multitask_01

TinkerCAD: CCAMD_Ext_Interrupts_01
CCAMD_Ext_Interrupts_02
CCAMD_Timer_Interrupts_03
CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03

CCAMD_WatchDog_01

Wokwi: CCAMD_Wokwi_Sleep_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Lecture 29 - "Mulitasking, Implementing a Mini Kernel with Interrupts"

In this lecture, we take the theory we learned and implement a crude interrupt based multitasking system and use it to build a demo system with a with a number of hardware elements (from our Elegoo kits) and in real-time we control all of them with our little kernel.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-

Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

https://en.wikipedia.org/wiki/Taylor_series

https://mitocw.ups.edu.ec/courses/mathematics/18-01sc-single-variable-calculus-fall-2010/unit-5-exploring-the-infinite/part-b-taylor-series/session-98-taylors-series/

https://www.freertos.org/index.html

https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02
CCAMD_Arduino_IDE_Sleep_03
CCAMD_Arduino_IDE_PrintFDebug_01
CCAMD_Arduino_IDE_Debugger_Demo_01
CCAMD_Arduino_IDE_Optimization_01
CCAMD_Arduino_IDE_Optimization_02
CCAMD_Arduino_IDE_Multitask_01

TinkerCAD: CCAMD_Ext_Interrupts_01
CCAMD_Ext_Interrupts_02
CCAMD_Timer_Interrupts_03
CCAMD_PCINT_Interrupts_01
CCAMD_PCINT_Interrupts_02
CCAMD_PCINT_Interrupts_03
CCAMD_WatchDog_01

Wokwi: CCAMD_Wokwi_Sleep_01

CodeLite: CCAMD_StarterWorkSpace
CCAMD_CppPrimerWork
main_basics_04.cpp
main_basics_05.cpp

Parts List: There are a lot of parts for this experiment, so here's a quick run down if you miss it in lecture: Elegoo kits: power supply, servo, analog joystick, humidity sensor, 8x8 LED display, momentary switches, LED, various R (100, 220, 330, 1K, 10K), C (0.1, 10, 100, 220), POT 1-100K.

Lecture 30 - "Mulitasking with our Kernel, LED, Photocell, Servo, Temp + Humidity Sensor etc. Interfacing Various Hardware - Part II"

In this lecture, we finish off the multitasking demo with reviewing the circuitry and wiring for each of the hardware devices, along with numerous demos of the hardware as we try various adjustments for each hardware task. This lecture wraps up our homebrew multitasking kernel.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/
https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit

https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

https://en.wikipedia.org/wiki/Taylor_series

https://mitocw.ups.edu.ec/courses/mathematics/18-01sc-single-variable-calculus-fall-2010/unit-5-exploring-the-infinite/part-b-taylor-series/session-98-taylors-series/

https://www.freertos.org/index.html

https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf

https://hackaday.io/project/5334-serialplot-realtime-plotting-software

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo Photocell and MAX7219 8x8 LED datasheets (copy from CD-ROM with kit or link above).

Arduino IDE: CCAMD_Arduino_IDE_Ext_Interrupts_01
CCAMD_Arduino_IDE_Ext_Interrupts_02
CCAMD_Arduino_IDE_Ext_Interrupts_03
CCAMD_Arduino_IDE_PCINT_Interrupts_01
CCAMD_Arduino_IDE_PCINT_Interrupts_02
CCAMD_Arduino_IDE_PCINT_Interrupts_03
CCAMD_Arduino_IDE_WatchDog_01
CCAMD_Arduino_IDE_Sleep_01
CCAMD_Arduino_IDE_Sleep_02

	CCAMD_Arduino_IDE_Sleep_03
	CCAMD_Arduino_IDE_PrintFDebug_01
	CCAMD_Arduino_IDE_Debugger_Demo_01
	CCAMD_Arduino_IDE_Optimization_01
	CCAMD_Arduino_IDE_Optimization_02
	CCAMD_Arduino_IDE_Multitask_01
TinkerCAD:	CCAMD_Ext_Interrupts_01
	CCAMD_Ext_Interrupts_02
	CCAMD_Timer_Interrupts_03
	CCAMD_PCINT_Interrupts_01
	CCAMD_PCINT_Interrupts_02
	CCAMD_PCINT_Interrupts_03
	CCAMD_WatchDog_01
Wokwi:	CCAMD_Wokwi_Sleep_01
CodeLite:	CCAMD_StarterWorkSpace
	CCAMD_CppPrimerWork
	main_basics_04.cpp
	main_basics_05.cpp

Parts List: There are a lot of parts for this experiment, so here's a quick run down if you miss it in lecture: Elegoo kits: power supply, servo, analog joystick, humidity sensor, 8x8 LED display, momentary switches, LED, various R (100, 220, 330, 1K, 10K), C (0.1, 10, 100, 220), POT 1-100K.

Lecture 31 - "FreeRTOS and True Multitasking"

In this lecture, we take a look at a commercial real-time operating system or RTOS by the name of "freeRTOS". This RTOS is written in C, lightweight and specifically designed for embedded systems. There is a port of it available for most Arduino processors including the ATMega 328P, so we will take a look at the RTOS and give it a spin with the Arduino Uno.

Links/Resources:

Design Your Own Game Console eBook

https://en.wikipedia.org/wiki/C_(programming_language)
https://en.wikipedia.org/wiki/C%2B%2B
https://en.cppreference.com/
https://en.cppreference.com/w/cpp/language
http://www.braun-home.net/michael/info/misc/VT100_commands.htm

https://microchipdeveloper.com/xwiki/bin/view/products/mcu-mpu/8-bit-avr/structure/memory/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.thecoderscorner.com/electronics/microcontrollers/efficiency/how-arduino-avr-memory-model-works/

https://docs.arduino.cc/learn/programming/memory-guide
https://www.electronicshub.org/types-of-memory-on-arduino/
https://www.nongnu.org/avr-libc/user-manual/
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://ww1.microchip.com/downloads/en/appnotes/doc42055.pdf
https://redirect.cs.umbc.edu/~alnel1/cmpe311/notes/AVRAssemblerBeginner.pdf
https://users.utcluj.ro/~rdanescu/dmp-lab06_new.pdf
http://www.rjhcoding.com/avr-asm-tutorials.php
https://ucexperiment.wordpress.com/2016/03/04/arduino-inline-assembly-tutorial-1/
https://www.nongnu.org/avr-libc/user-manual/inline_asm.html

https://www.youtube.com/watch?v=j-qs-gJhxfs&list=PL09ZAP7_T_LmlX5vctZV4PFfZwMNzjX1F&ab_channel=AnasKuzechie

https://www.youtube.com/watch?v=2ESXfD8X-FQ&ab_channel=RalphSBacon
https://www.ibiblio.org/gferg/ldp/GCC-Inline-Assembly-HOWTO.html
https://leanpub.com/arduinoinlineassembly

https://web.alfredstate.edu/faculty/weimandn/miscellaneous/atmega_miscellaneous/ATmega328%20SRAM%20Data%20Memory.pdf

http://tigcc.ticalc.org/doc/gnuasm.html
https://defuse.ca/online-x86-assembler.htm
https://dogbolt.org/
https://syscall7.com/oda/

https://binary.ninja/
https://binary.ninja/free/
https://forum.arduino.cc/t/tutorial-basic-watchdog-timer-setup/63397
https://www.instructables.com/The-Arduino-Hang-Guardian-Arduino-Watchdog-Timer-T/
https://docs.arduino.cc/learn/electronics/low-power/
https://gammon.com.au/power

https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CH128MP508-Family-Data-Sheet-DS70005319D.pdf

https://en.wikipedia.org/wiki/Memory_management_unit
https://ww1.microchip.com/downloads/en/DeviceDoc/60001245a.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MPD/ProductDocuments/DataSheets/23A102423LC1024-1-Mbit-SPI-Serial-SRAM-with-SDI-SQI-Interface-20005142.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/25001A.pdf

https://sites.google.com/site/wayneholder/stuff-im-working-on
https://github.com/JoaoLopesF/SerialDebug
https://en.wikipedia.org/wiki/In-circuit_emulation
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://en.wikipedia.org/wiki/JTAG
http://datasheets.chipdb.org/Intel/x86/486/manuals/27302101.PDF
https://en.wikipedia.org/wiki/Big_O_notation

https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/

https://en.wikipedia.org/wiki/Taylor_series

https://mitocw.ups.edu.ec/courses/mathematics/18-01sc-single-variable-calculus-fall-2010/unit-5-exploring-the-infinite/part-b-taylor-series/session-98-taylors-series/

https://www.analog.com/media/en/technical-documentation/data-sheets/MAX7219-MAX7221.pdf

https://hackaday.io/project/5334-serialplot-realtime-plotting-software

https://freertos.org/Documentation/RTOS_book.html

https://freertos.org/fr-content-src/uploads/2018/07/161204_Mastering_the_FreeRTOS_Real_Time_Kernel-A_Hands-On_Tutorial_Guide.pdf https://freertos.org/fr-content-src/uploads/2018/07/FreeRTOS_Reference_Manual_V10.0.0.pdf

https://en.wikipedia.org/wiki/FreeRTOS
https://www.arduino.cc/reference/en/libraries/freertos/
https://feilipu.me/2015/11/24/arduino_freertos/
https://github.com/feilipu/Arduino_FreeRTOS_Library
https://wiki.seeedstudio.com/Software-FreeRTOS/
https://github.com/pstolarz/CoopThreads

https://circuitdigest.com/microcontroller-projects/arduino-freertos-tutorial1-creating-freertos-task-to-blink-led-in-arduino-uno

https://www.elegoo.com/blogs/arduino-projects/elegoo-uno-r3-project-the-most-complete-starter-kit-tutorial

Elegoo Photocell and MAX7219 8x8 LED datasheets (copy from CD-ROM with kit or link above).

Arduino IDE: CCAMD_Arduino_IDE_FreeRTOS_01

Section 9: Digital Communication Protocols and Interfacing with RS-232, UART, SPI, I2C and 1-Wire

In this section we are going to cover a number of communication protocols that are popular with embedded systems. We will cover simple serial communications such as TTL serial and RS-232 as well as I2C, SPI, and 1-Wire protocols. We will take a look at the electrical specifications as well as the higher level packet level protocol for each. Additionally, we use these various protocols to experiment with various sensors in our kits and build a number of very useful projects.

Lecture 1 - "Serial Protocols UARTs and RS-232 - Part I"

In this lecture, we take a look at one of the oldest and classic communications protocols known as RS-232. This is a serial protocol (only requiring 3 signals; TX, RX, and GND) which has been around for decades. We will take a look at both the classic 12V version of the protocol and the more modern "digital" version that has replaced it on hardware running at 5V or lower, also, known as "TTL Serial".

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F
https://www.ascii-code.com/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://en.wikipedia.org/wiki/RS-232
https://www.arduino.cc/reference/en/language/functions/communication/serial/

Arduino IDE: CCAMD_Arduino_IDE_Client_01
CCAMD_Arduino_IDE_Client_02

Lecture 2 - "Serial Protocols UARTs and RS-232 - Part II, Bench Demo of Terminals"

In this lecture, we build a two Arduino network and send data back and forth between them. This is one of the largest and most complex experiments in the course, and a lot of very important concepts are illustrated. Additionally, we learn to use the "Logic Analyzer" to decode ASCII traffic as well, as decode it manually with an oscope.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ascii-code.com/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://en.wikipedia.org/wiki/RS-232
https://www.arduino.cc/reference/en/language/functions/communication/serial/

Arduino IDE:	CCAMD_Arduino_IDE_Transmission_01
	CCAMD_Arduino_IDE_Client_01
	CCAMD_Arduino_IDE_Client_02
	CCAMD_Arduino_IDE_Transmitter_01
	CCAMD_Arduino_IDE_Receiver_01
TinkerCAD:	CCAMD_UART_Client_01

Kingst and Logic Cube LA Configuration Files:
UART_Channel_0.kvset
ZP_UART_01.alc

Lecture 3 - "Serial Protocols UARTs and RS-232 Part III, Client/Server"

In this lecture, we finish off our serial communications demo and bi-directional communications example by adding buttons and LEDs to both clients and then passing messages in both directions from the buttons to control the LEDs on the other system.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ascii-code.com/

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

https://ww1.microchip.com/downloads/en/DeviceDoc/AVR-InstructionSet-Manual-DS40002198.pdf

https://ww1.microchip.com/downloads/en/devicedoc/atmel-0856-avr-instruction-set-manual.pdf

https://en.wikipedia.org/wiki/RS-232
https://www.arduino.cc/reference/en/language/functions/communication/serial/

Arduino IDE: CCAMD_Arduino_IDE_Transmission_01
CCAMD_Arduino_IDE_Client_01
CCAMD_Arduino_IDE_Client_02
CCAMD_Arduino_IDE_Transmitter_01

CCAMD_Arduino_IDE_Receiver_01

TinkerCAD: CCAMD_UART_Client_01
CCAMD_UART_Client_02

Kingst and Logic Cube LA Configuration Files:
UART_Channel_0.kvset
ZP_UART_01.alc

Lecture 4 - "Introduction to Serial Peripheral Interface (SPI) Protocol"

In this lecture, we cover the fundamentals and theory behind the serial protocol known as SPI or Serial Peripheral Interface. This protocol is relatively easy to learn, full duplex and runs at rates of 25MHz and beyond.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all
https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
https://www.circuitbasics.com/how-to-set-up-spi-communication-for-arduino/
https://www.arduino.cc/reference/en/language/functions/communication/spi/
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

https://www.ti.com/lit/ds/symlink/tmp121.pdf?ts=1702074456408&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/product/TMP126

Arduino IDE 1.8x: CCAMD_Arduino_IDE_TMP121_SPI_01

Kingst and Logic Cube LA Configuration Files:
SPI_Setup_01.kvset

Extra Parts: In addition to the course kits you will need to source a **TMP121** sensor and breakout board, see course Parts List PDF for sources.

Lecture 5 - "Serial Peripheral Interface (SPI), Interfacing to a Temperature Sensor - Part II"

In this lecture, we build up the hardware model, firmware, math, and everything else to bring the SPI temperature sensor TMP121 online. There's a lot of work to be done, but this process is illustrative of the process you will repeat time and time again as you want to interface to new devices that support the SPI protocol.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all
https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
https://www.circuitbasics.com/how-to-set-up-spi-communication-for-arduino/
https://www.arduino.cc/reference/en/language/functions/communication/spi/
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

https://www.ti.com/lit/ds/symlink/tmp121.pdf?ts=1702074456408&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/product/TMP126

Arduino IDE 1.8x: CCAMD_Arduino_IDE_TMP121_SPI_01

EasyEDA: CCAMD_Arduino_TMP121_SPI_01

Kingst and Logic Cube LA Configuration Files:
SPI_Setup_01.kvset

Extra Parts: In addition to the course kits you will need to source a **TMP121** sensor and breakout board, see course Parts List PDF for sources.

Lecture 6 - "Serial Peripheral Interface (SPI), Interfacing to a Temperature Sensor - Part III"

In this lecture, we build the hardware with the SPI Temperature sensor and put everything together we have learned to create a working SPI program that interogates the SPI temperature sensor in real-time. We use a number of tools including the logic analyzer. Finally, we finish off with a preview of the next device we are going to play with -- the "Digital POT".

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all
https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
https://www.circuitbasics.com/how-to-set-up-spi-communication-for-arduino/
https://www.arduino.cc/reference/en/language/functions/communication/spi/
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

https://www.ti.com/lit/ds/symlink/tmp121.pdf?ts=1702074456408&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/product/TMP126
https://ww1.microchip.com/downloads/en/devicedoc/11195c.pdf

Arduino IDE 1.8x:	CCAMD_Arduino_IDE_TMP121_SPI_01
	CCAMD_Arduino_IDE_MCP41xxx_SPI_01
EasyEDA:	CCAMD_Arduino_TMP121_SPI_01
	CCAMD_Arduino_MCP41xxx_SPI_01

Kingst and Logic Cube LA Configuration Files:
SPI_Setup_01.kvset

Lecture 7 - "Serial Peripheral Interface (SPI), Controlling a Digital POT - Part IV"

In this lecture, we review the datasheet and operation of a typical SPI digital POTentiometer. Then we wire it up, write code based on the datasheet, and then run tests in real-time along with the logic analyzer to see the operation of the POT.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all
https://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
https://www.circuitbasics.com/how-to-set-up-spi-communication-for-arduino/
https://www.arduino.cc/reference/en/language/functions/communication/spi/
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

https://www.ti.com/lit/ds/symlink/tmp121.pdf?ts=1702074456408&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/product/TMP126
https://ww1.microchip.com/downloads/en/devicedoc/11195c.pdf
https://www.microchip.com/en-us/product/mcp41010

Arduino IDE 1.8x:	CCAMD_Arduino_IDE_TMP121_SPI_01
	CCAMD_Arduino_IDE_MCP41xxx_SPI_01
EasyEDA:	CCAMD_Arduino_TMP121_SPI_01
	CCAMD_Arduino_MCP41xxx_SPI_01

Kingst and Logic Cube LA Configuration Files:
SPI_Setup_01.kvset

Extra Parts: In addition to the course kits you will need to source a MCP41010 sensor and breakout board, see course Parts List PDF for sources.

Lecture 8 - "Serial Peripheral Interface to an LCD and Game Console Build - Part I"

In this lecture, we build up the hardware model, firmware, APIs, and port a 3D wire frame game to run on the Arduino driving a SPI LCD. This is one of the coolest projects in the course!

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html
https://cdn-shop.adafruit.com/datasheets/ST7735R_V0.2.pdf
https://learn.adafruit.com/adafruit-gfx-graphics-library/overview
https://learn.adafruit.com/1-8-tft-display
https://www.amazon.com/s?k=ST7735R+1.8%22+LCD+breakout&ref=nb_sb_noss_2

Arduino IDE:	CCAMD_Arduino_IDE_ST7735R_01
	CCAMD_Arduino_IDE_SPI_Raiders3D_01
EasyEDA:	CCAMD_Arduino_Raiders3D_ST7735R_01

Code from “Tricks of the 3D Game Programming Gurus” Chapter 01 CD-ROM for Raiders3D source code:

Root\..\eBooksAndGoodies\Tricks3DGameGurus\Tricks3DCD\raiders3d.cpp

Lecture 9 - "SPI Interface to an LCD and 3D Game Port to Game Console - Part II"

In this lecture, we cover the tactics used to port a 3D game to the Ardunio supporting a SPI LCD. We review how the source code uses a frame buffer and how keyboard input, graphics, sound, and everything else must be mapped to the Arduino platform. By the end, we will be playing a wire-frame 3D game on the Arduino!

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/Serial_Peripheral_Interface
https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html
https://cdn-shop.adafruit.com/datasheets/ST7735R_V0.2.pdf
https://learn.adafruit.com/adafruit-gfx-graphics-library/overview
https://learn.adafruit.com/1-8-tft-display
https://www.amazon.com/s?k=ST7735R+1.8%22+LCD+breakout&ref=nb_sb_noss_2

Arduino IDE:	CCAMD_Arduino_IDE_ST7735R_01
	CCAMD_Arduino_IDE_SPI_Raiders3D_01
EasyEDA:	CCAMD_Arduino_Raiders3D_ST7735R_01

Code from "**Tricks of the 3D Game Programming Gurus**" Chapter 1 CD-ROM for Raiders3D source code:

Root\..\eBooksAndGoodies\Tricks3DGameGurus\Tricks3DCD\raiders3d.cpp

Lecture 10 - "I2C Serial Protocol (Inter IC Communications), Introduction to Wire Library, Temperature and Real Time Clock"

In this lecture, we explore the next serial protocol in our series - I2C or Inter IC Communications. We will look at the electrical connections as well as the data protocol and libraries to facilitate communication. Additionally, we will take a look at a couple sample ICs that support the protocol and their datasheets (humidity/temp sensor and real time clock).

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/
https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.ti.com/lit/an/sbaa565/sbaa565.pdf?ts=1712970515816&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf

Kingst and Logic Cube LA Configuration Files:
    I2C_Setup_01.kvset

Extra Parts: DS1307 RTC clock from course kits.

Lecture 11 - "I2C Serial Protocol (Inter IC Communications), Interfacing to a Digital POT"

In this lecture, we focus on the details of the AD5241 (Analog Devices) digital POT, take a look at the datasheet, registers, and how to write to the POT via I2C. This IC is more complex than the SPI POT, so a good example of a typical I2C device and how to get it working. Additionally, we write the firmware, build a model in Fritzing (another CAD tool like TinkerCAD), and prepare to build the physical hardware and test our design.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ad5241_5242.pdf

https://www.digikey.com/en/products/filter/digital-potentiometers/717?s=N4IgTCBcDaIIYBMCsYAsBGEBdAvkA

Arduino IDE: CCAMD_Arduino_IDE_D5241_I2C_01

Fritzing: CCAMD_Fritz_D5241_01

Extra Parts: AD5241 break out board (or loose IC), see sources in course Parts List PDF.

Lecture 12 - "I2C Serial Protocol (Inter IC Communications), Digital POT Circuit Bench Build, Demo and Analysis"

In this lecture, we build the physical hardware for the Digital POT demo, upload the code and run it in real-time. Additionally, we analyze the I2C packets with a logic analyzer and review both writing the POT wiper value as well as GPIO bits that the IC we use has as an added feature. Finally, we see how noise tolerant and robust I2C is, and even works with really sloppy wiring!

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ad5241_5242.pdf

https://www.digikey.com/en/products/filter/digital-potentiometers/717?s=N4IgTCBcDaIIYBMCsYAsBGEBdAvkA

Arduino IDE: CCAMD_Arduino_IDE_D5241_I2C_01
CCAMD_Arduino_IDE_D5241_I2C_Scanner_01

Fritzing: CCAMD_Fritz_D5241_01

Kingst and Logic Cube LA Configuration Files:
I2C_Setup_01.kvset

Extra Parts: AD5241 break out board (or loose IC), see sources in course Parts List PDF.

Lecture 13 - "I2C Serial Protocol (Inter IC Communications), SI7021 Temperature and Humidity Sensor Review and Build"

In this lecture, we continue our pace since we have the fundamentals of I2C under our belts, and we jump right into another interesting sensor the SI7021 humidity and temperature sensor. This sensor is interesting since it performs a number of weather related measurements, is fast, small, and low cost. We review the data sheet, and write firmware to read data from the sensor, and we build up the circuitry on the bench to see the sensor in action.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ad5241_5242.pdf
https://www.adafruit.com/product/3251

https://www.sparkfun.com/products/13763
https://github.com/adafruit/Fritzing-Library

Arduino IDE:	CCAMD_Arduino_IDE_D5241_I2C_01
	CCAMD_Arduino_IDE_D5241_I2C_Scanner_01
	CCAMD_Arduino_IDE_SI7021_I2C_01
Fritzing:	CCAMD_Fritz_D5241_01
	CCAMD_Fritz_SI7021_01

Extra Parts: SI7021 break out board, see sources in course Parts List PDF.

Lecture 14 - "I2C Serial Protocol, Measuring Time with the DS1307 RTC"

In this lecture, we explore more I2C devices with the DS1307 Real-Time Clock or RTC. This device is used to maintain accurate time while your embedded system is powered off. To do this it not only has very accurate time keep hardware, but a battery to maintain the time for years. In this lecture, we will see how to use an off the shelf API to communicate with the DS1307 as well as write our own low level driver.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ad5241_5242.pdf
https://www.adafruit.com/product/3296
https://www.sparkfun.com/products/12708

Arduino IDE:	CCAMD_Arduino_IDE_D5241_I2C_01
	CCAMD_Arduino_IDE_D5241_I2C_Scanner_01
	CCAMD_Arduino_IDE_SI7021_I2C_01

CCAMD_Arduino_IDE_DS1307_I2C_01
CCAMD_Arduino_IDE_DS1307_I2C_02

Fritzing: CCAMD_Fritz_D5241_01
CCAMD_Fritz_SI7021_01
CCAMD_Fritz_DS1307_01

Extra Parts: DS1307 RTC clock from course kits (37 in 1).

Lecture 15 - "I2C Serial Protocol - Working with Accelerometers and Gyros"

In this lecture, we learn about accelerometers, gyros and experiment with a popular unit from Invensense the MPU6050. We will build the hardware, review the datasheets, use an API as well as write our own driver.

Links/Resources:

Design Your Own Game Console eBook

https://godbolt.org/
https://hackaday.io/project/5334-serialplot-realtime-plotting-software
https://cplusplus.com/

https://www.ti.com/lit/ds/symlink/max232.pdf?ts=1702068315567&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://www.ascii-code.com/
https://en.wikipedia.org/wiki/I%C2%B2C

https://www.ti.com/lit/an/slva704/slva704.pdf?ts=1702029942389&ref_url=https%253A%252F%252Fwww.google.com%252F

https://www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ds1307.pdf
https://www.analog.com/media/en/technical-documentation/data-sheets/ad5241_5242.pdf
https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf

https://mschoeffler.com/2017/10/05/tutorial-how-to-use-the-gy-521-module-mpu-6050-breakout-board-with-the-arduino-uno/

https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/

https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/

Arduino IDE:	CCAMD_Arduino_IDE_D5241_I2C_01
	CCAMD_Arduino_IDE_I2C_Scanner_01
	CCAMD_Arduino_IDE_DS1307_I2C_01
	CCAMD_Arduino_IDE_MPU6050_I2C_01
	CCAMD_Arduino_IDE_MPU6050_I2C_02
Fritzing:	CCAMD_Fritz_D5241_01
	CCAMD_Fritz_SI7021_01
	CCAMD_Fritz_DS1307_01
	CCAMD_Fritz_MPU6050_01

Section 10: Advanced Tools for Arduino and Embedded Development

In this section, we are going to take a look at some more advanced and professional tools for embedded development. We will focus on the Arduino since its our working platform, but the idea of this section is to show you there are many, many other options for Arduino and general embedded development IDEs.

Lecture 1 - "Developing with the Ardunio Web/Cloud IDE"

In this lecture, we take a look at the new(ish) web/cloud based Arduino IDE. This new IDE is browser based and supported by all popular browsers. It works by installing a piece of software, a driver, plug in of sorts to make the physical connectio from the sandbox of the browser to the hardware you have connected via your USB serial port.

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://www.arduino.cc/en/software

https://support.arduino.cc/hc/en-us/articles/10482021304988-Use-Arduino-Cloud-with-Brave-Browser

Arduino Web Editor:

CCAMD_Web_Blink_01
https://app.arduino.cc/sketches/f9129dd4-bcb3-44d1-bf3e-59d7c7012ed3?view-mode=preview

CCAMD_Web_Fade_01

https://app.arduino.cc/sketches/39eb13c5-ff08-488a-b7cb-bdc4c7307250?view-mode=preview

Lecture 2 - "Introducing the New and Improved(?) Arduino IDE 2.x IDE"

In this lecture, we learn about the new Arduino IDE 2.x, it's new features. We'll build some programs, upload, test, and take the new IDE for a spin and contrast it to the legacy version.

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://www.arduino.cc/en/software

https://support.arduino.cc/hc/en-us/articles/10482021304988-Use-Arduino-Cloud-with-Brave-Browser

Arduino IDE 2.x:	CCAMD_Arduino_IDE_BLink_01 (simply use the same code/project from the master source ZIP file that we used for all the IDE 1.8.x version of the IDE).

Lecture 3 - "Working with Visual Studio Code + PlatformIO"

In this lecture, we take a look at an advanced coding editor called **Visual Studio Code,** also known as **"VSCode"** from Microsoft. Additionally, we look at an embedded development framework; **PlatformIO** that can be plugged into VSCode it that allows us to develop for dozens of microcontrollers and platforms (including Arduino).

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://code.visualstudio.com/
https://platformio.org/

https://www.circuitstate.com/tutorials/how-to-use-vs-code-for-creating-and-uploading-arduino-sketches/#Why_Use_VS_Code_for_Arduino?

https://dronebotworkshop.com/platformio/
https://docs.platformio.org/en/latest/projectconf/index.html

Debug Support :
https://www.youtube.com/watch?v=7wx27FcluMg&ab_channel=J%27se-shack
https://github.com/jdolinay/avr_debug

VSCode Project: CCAMD_PIO_BlinkTest_01

```
Tips for Debugging (covered in lecture):

1. Add to platformio.ini for project:

debug_tool = avr-stub
debug_port = COMxx
; GDB stub
lib_deps = jdolinay/avr-debugger @ ~1.5

2. Add these includes and code in main.cpp:

#include "avr8-stub.h"
#include "app_api.h"

// in setup() add this code
debug_init(); // start debugger
```

Lecture 4 - "Visual Studio Code + PlatformIO - Adding Basic Debug Support"

In this lecture, we take a look at a debugging solution for VSCode that supports Arduino. The debugger uses only the serial port, doesn't require changes to the bootloader, and supports single stepping, variable watches, and more.

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf
https://code.visualstudio.com/
https://platformio.org/

https://www.circuitstate.com/tutorials/how-to-use-vs-code-for-creating-and-uploading-arduino-sketches/#Why_Use_VS_Code_for_Arduino?

https://dronebotworkshop.com/platformio/
https://docs.platformio.org/en/latest/projectconf/index.html

Debug Support :
https://www.youtube.com/watch?v=7wx27FcluMg&ab_channel=J%27se-shack
https://github.com/jdolinay/avr_debug

VSCode Project: CCAMD_PIO_BlinkTest_01

```
Tips for Debugging (covered in lecture):

1. Add to platformio.ini for project:

debug_tool = avr-stub
debug_port = COMxx
; GDB stub
lib_deps = jdolinay/avr-debugger @ ~1.5

2. Add these includes and code in main.cpp:

#include "avr8-stub.h"
#include "app_api.h"

// in setup() add this code
debug_init(); // start debugger
```

Lecture 5 - "MPLAB X IDE for AVR and PIC, ARM Cortex M0-M7, Part I - Overview and Installation"

In this lecture, we learn about an advanced, professional IDE from Microchip called MPLAB X. We start with a documentation review of the IDE, learn how it can be used to develop for AVR, PIC, ARM processors. We learn about the compilers available from Microchip that plug into it as well as the new technologies that the tool supports. Then we begin one of the most complex installs we have done in the course, install the tool, and run it.

Finally, we take a look at how to update the tool, and a quick walk around features such as the **Code Configurator**, "**Melody**", "**Harmony**", and the Graphical Builder app as well as the Pin Manager and more. We finish off with a discussion about the **Unified Microchip Library**, and prepare for writing code, compiling, uploading to our targets in Part II.

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/ReferenceManuals/Microchip_Unified_Standard_Library_Reference_Guide.pdf https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/UserGuides/MPLAB-XC8-C-Compiler-Users-Guide-for-AVR-MCU-50002750.pdf

https://www.youtube.com/watch?v=HVgi4SiU69E&ab_channel=MetalPlasticElectronics
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide
https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers/xc8

https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers/xc8#downloads

https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers

https://www.microchip.com/en-us/tools-resources/configure/mplab-code-configurator/classic

https://onlinedocs.microchip.com/pr/GUID-1F7007B8-9A46-4D03-AEED-650357BA760D-en-US-6/index.html?GUID-4D29BF19-E2DB-4254-946F-9E42928767F6

https://onlinedocs.microchip.com/oxy/GUID-5A03F818-B7FC-4062-9792-57D08543B586-en-US-7/GUID-4FF6C8DE-2375-4456-9150-3ECCDAEB82B4.html

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/ReferenceManuals/Microchip_Unified_Standard_Library_Reference_Guide.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/UserGuides/MPLAB-XC8-C-Compiler-Users-Guide-for-AVR-MCU-50002750.pdf

https://microchipdeveloper.com/xwiki/bin/view/software-tools/x/debugging/avr-debugwire/
https://www.microchip.com/en-us/development-tool/pg164150

https://onlinedocs.microchip.com/oxy/GUID-8D61C0B9-A97F-4F4D-99F8-1D7424264C2A-en-US-1/GUID-2E07C091-C3CD-4DE3-9187-80FA1E63E969.html
https://www.youtube.com/watch?v=yfr8Jgwljh0&ab_channel=MicrochipDeveloperHelp
https://www.youtube.com/@MicrochipDeveloperHelp/search?query=MCC
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://www.yorku.ca/professor/drsmith/2021/07/23/debuggers-mplab-x-and-the-arduino/

https://circuitdigest.com/microcontroller-projects/understanding-fuse-bits-in-atmega328p-to-enhance-arduino-programming-skills

https://www.martyncurrey.com/arduino-atmega-328p-fuse-settings/
https://www.engbedded.com/fusecalc/

https://www.google.com/search?q=pickit+5+connection+to+atmega+328p+ICSP+pins&tbm=isch&ved=2ahUKEwjg2LDAjriDAxWJG9AFHR-fAxUQ2-cCegQIABAA&oq=pickit+5+connection+to+atmega+328p+ICSP+pins&gs_lcp=CgNpbWcQA1DBBlizK2C-LWgDcAB4AIABUogBogaSAQIxMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=xoyQZeDdD4m3wN4Pn76OqAE&bih=728&biw=1536

MPLABX:	CCAMD_MPLABX_ArduinoUnoR3_02
	CCAMD_MPLABX_ArduinoUnoR3_03
	CCAMD_MPLABX_ArduinoUnoR3_04
	optiboot_atmega328
	optiboot_atmega328_02

Lecture 6 - "MPLAB X IDE for AVR and PIC, ARM Cortex M0-M7, Part II - Building Apps Blinking LED, "Hello World" and Restoring Bootloader"

In this lecture, we create a number of projects, compile, download, and explore the operation of MPLAB X IDE. We basically create a "Hello World" application from scratch, use the Unified Library, and learn our way around this very complex tool. Additionally, you will see how we overwrite the Arduino bootloader, and how we can reload it onto the Arduino after using these low level tools.

Links/Resources:

Design Your Own Game Console eBook

https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/ReferenceManuals/Microchip_Unified_Standard_Library_Reference_Guide.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/UserGuides/MPLAB-XC8-C-Compiler-Users-Guide-for-AVR-MCU-50002750.pdf

https://www.youtube.com/watch?v=HVgi4SiU69E&ab_channel=MetalPlasticElectronics
https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf
https://content.arduino.cc/assets/Pinout-UNOrev3_latest.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/ATmega48A-PA-88A-PA-168A-PA-328-P-DS-DS40002061B.pdf

https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide
https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers/xc8

https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers/xc8#downloads

https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers

https://www.microchip.com/en-us/tools-resources/configure/mplab-code-configurator/classic

https://onlinedocs.microchip.com/pr/GUID-1F7007B8-9A46-4D03-AEED-650357BA760D-en-US-6/index.html?GUID-4D29BF19-E2DB-4254-946F-9E42928767F6

https://onlinedocs.microchip.com/oxy/GUID-5A03F818-B7FC-4062-9792-57D08543B586-en-US-7/GUID-4FF6C8DE-2375-4456-9150-3ECCDAEB82B4.html

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/ReferenceManuals/Microchip_Unified_Standard_Library_Reference_Guide.pdf

https://ww1.microchip.com/downloads/aemDocuments/documents/DEV/ProductDocuments/UserGuides/MPLAB-XC8-C-Compiler-Users-Guide-for-AVR-MCU-50002750.pdf

https://microchipdeveloper.com/xwiki/bin/view/software-tools/x/debugging/avr-debugwire/
https://www.microchip.com/en-us/development-tool/pg164150

https://onlinedocs.microchip.com/oxy/GUID-8D61C0B9-A97F-4F4D-99F8-1D7424264C2A-en-US-1/GUID-2E07C091-C3CD-4DE3-9187-80FA1E63E969.html
https://www.youtube.com/watch?v=yfr8Jgwljh0&ab_channel=MicrochipDeveloperHelp
https://www.youtube.com/@MicrochipDeveloperHelp/search?query=MCC
https://www.youtube.com/watch?v=Z67ZUTv88L4&ab_channel=JamesSmith
https://www.yorku.ca/professor/drsmith/2021/07/23/debuggers-mplab-x-and-the-arduino/

https://circuitdigest.com/microcontroller-projects/understanding-fuse-bits-in-atmega328p-to-enhance-arduino-programming-skills

https://www.martyncurrey.com/arduino-atmega-328p-fuse-settings/
https://www.engbedded.com/fusecalc/

https://www.google.com/search?q=pickit+5+connection+to+atmega+328p+ICSP+pins&tbm=isch&ved=2ahUKEwjg2LDAjriDAxWJG9AFHR-fAxUQ2-cCegQIABAA&oq=pickit+5+connection+to+atmega+328p+ICSP+pins&gs_lcp=CgNpbWcQA1DBBlizK2C-LWgDcAB4AIABUogBogaSAQIxMZgBAKABAaoBC2d3cy13aXotaW1nwAEB&sclient=img&ei=xoyQZeDdD4m3wN4Pn76OqAE&bih=728&biw=1536

MPLABX:	CCAMD_MPLABX_ArduinoUnoR3_02
	CCAMD_MPLABX_ArduinoUnoR3_03

CCAMD_MPLABX_ArduinoUnoR3_04
optiboot_atmega328
optiboot_atmega328_02

Section 11: Into the Abyss with ARM Cortex M-Series Processors

In this final primary section of the course, we are going to explore an "advanced" processor and platform, the ARM Cortex M-series. These are hands down my favorite microcontrollers (and processors) that I use in my professional embedded systems development for my products and for clients. This section will give you a taste of how complex ARM is, how big the tools are, and the shocking amount of documentation! But, before I scare you off, we will get through this, we will install a tool, select an ARM dev board, build a number of projects. When you get through this section of the course, you really have seen everything in embedded from the very simplistic Arduino to state of the art ARM technology. So, you will be well prepared to continue you journey in embedded engineering.

Lecture 1 - "ST Micro ARM Cortex Microcontroller and Processor Primer"

In this lecture, we take a look at the incredible world of ARM processors. We review the history, technology, popular vendors, and then focus on one of the most popular vendors for the ARM Cortex -- STMicroelectronics. We look at their processors, development boards, learn about their tool chain and then install their flagship tool **STM32CubeIDE** for our coding work in the following lectures.

Links/Resources:

Design Your Own Game Console eBook

Videos:

https://www.youtube.com/watch?v=hyZS2p1tW-g&t=262s&ab_channel=DigiKey
https://www.youtube.com/watch?v=Hffw-m9fuxc&ab_channel=MitchDavis

Sites:

https://os.mbed.com/
https://deepbluembedded.com/stm32-hal-library-tutorial-examples/
https://os.mbed.com/platforms/ST-Nucleo-L476RG/
https://www.digikey.com/maker-media/5cbeca93-099b-456f-aae2-dfa74f861581
https://www.arm.com/technologies/cmsis
https://www.st.com/en/microcontrollers-microprocessors/stm32l476rg.html#overview
https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html
https://www.st.com/en/development-tools/stm32cubeide.html#get-software

https://www.st.com/content/st_com/en/stm32-mcu-developer-zone/boards-and-hardware-tools.html

https://www.st.com/en/evaluation-tools/nucleo-l476rg.html#sample-buy
https://www.st.com/en/development-tools/stsw-link007.html
https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/
https://blog.st.com/stm32cubeide-free-ide/

https://www.st.com/content/st_com/en/search.html#q=STM32%20HAL%20programming-t=resources-page=1

https://www.arm.com/

https://developer.arm.com/documentation/102404/0201/About-the-Arm-architecture?lang=en

https://en.wikipedia.org/wiki/JTAG

Datasheets:

https://www.st.com/resource/en/datasheet/stm32l476je.pdf

https://www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

https://www.st.com/resource/en/programming_manual/pm0214-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf

https://www.st.com/content/ccc/resource/technical/document/user_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf

https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

Lecture 2 - "Building Apps with STM32CubeIDE - Part I"

In this lecture, we fire up STM32CubeIDE and build projects from the ground up. This is a very exciting and complex lecture as we navigate the HAL (Hardware Abstraction Layer) API libraries, the STMicro libraries, selecting a target board (the ST Nucleo board), and putting everything together. We code a basic "Hello World" blinking LED app and see how things are done with the graphical pinmap and peripheral editor (very similar to MPLAB X) and then compile, upload our code to the target board. A lot to cover, but that's ARM for you!

Links/Resources:

Design Your Own Game Console eBook

Videos:

https://www.youtube.com/watch?v=hyZS2p1tW-g&t=262s&ab_channel=DigiKey
https://www.youtube.com/watch?v=Hffw-m9fuxc&ab_channel=MitchDavis

Sites:

https://os.mbed.com/
https://deepbluembedded.com/stm32-hal-library-tutorial-examples/
https://os.mbed.com/platforms/ST-Nucleo-L476RG/
https://www.digikey.com/maker-media/5cbeca93-099b-456f-aae2-dfa74f861581
https://www.arm.com/technologies/cmsis
https://www.st.com/en/microcontrollers-microprocessors/stm32l476rg.html#overview
https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html
https://www.st.com/en/development-tools/stm32cubeide.html#get-software

https://www.st.com/content/st_com/en/stm32-mcu-developer-zone/boards-and-hardware-tools.html

https://www.st.com/en/evaluation-tools/nucleo-l476rg.html#sample-buy
https://www.st.com/en/development-tools/stsw-link007.html
https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/
https://blog.st.com/stm32cubeide-free-ide/

https://www.st.com/content/st_com/en/search.html#q=STM32%20HAL%20programming-t=resources-page=1

https://www.arm.com/

https://developer.arm.com/documentation/102404/0201/About-the-Arm-architecture?lang=en

https://en.wikipedia.org/wiki/JTAG

Datasheets:

https://www.st.com/resource/en/datasheet/stm32l476je.pdf

https://www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

https://www.st.com/resource/en/programming_manual/pm0214-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf

https://www.st.com/content/ccc/resource/technical/document/user_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf
https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

STM32CubeIDE:	CCAMD_CortexM0_01
	CCAMD_NucleoL476_Test_01
	CCAMD_NucleoL476_Test_02
	CCAMD_NucleoL476_Test_03
	CCAMD_NucleoL476_Test_04
	CCAMD_NucleoL476_Test_05

Lecture 3 - "Building Apps with STM32CubeIDE (UARTS & IO) - Part II"

In this lecture, we continue our development on the ARM Cortex processor and add more features to our code like UART communication, see how to read digital GPIOs and put it all together into a coherent demo to exercise the Nucleo development board.

Links/Resources:

Design Your Own Game Console eBook

Videos:

https://www.youtube.com/watch?v=hyZS2p1tW-g&t=262s&ab_channel=DigiKey
https://www.youtube.com/watch?v=Hffw-m9fuxc&ab_channel=MitchDavis

Sites:

https://os.mbed.com/
https://deepbluembedded.com/stm32-hal-library-tutorial-examples/
https://os.mbed.com/platforms/ST-Nucleo-L476RG/
https://www.digikey.com/maker-media/5cbeca93-099b-456f-aae2-dfa74f861581
https://www.arm.com/technologies/cmsis
https://www.st.com/en/microcontrollers-microprocessors/stm32l476rg.html#overview
https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html
https://www.st.com/en/development-tools/stm32cubeide.html#get-software

https://www.st.com/content/st_com/en/stm32-mcu-developer-zone/boards-and-hardware-tools.html

https://www.st.com/en/evaluation-tools/nucleo-l476rg.html#sample-buy
https://www.st.com/en/development-tools/stsw-link007.html
https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/

https://blog.st.com/stm32cubeide-free-ide/

https://www.st.com/content/st_com/en/search.html#q=STM32%20HAL%20programming-t=resources-page=1

https://www.arm.com/

https://developer.arm.com/documentation/102404/0201/About-the-Arm-architecture?lang=en

https://en.wikipedia.org/wiki/JTAG

Datasheets:

https://www.st.com/resource/en/datasheet/stm32l476je.pdf

https://www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

https://www.st.com/resource/en/programming_manual/pm0214-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf

https://www.st.com/content/ccc/resource/technical/document/user_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf

https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

STM32CubeIDE:	CCAMD_CortexM0_01
	CCAMD_NucleoL476_Test_01
	CCAMD_NucleoL476_Test_02
	CCAMD_NucleoL476_Test_03
	CCAMD_NucleoL476_Test_04
	CCAMD_NucleoL476_Test_05

Lecture 4 - "Building Apps with STMicroCubeIDE (Debugging) - Part III"

In this lecture, we wrap up our initial discussion and coverage of ARM Cortex processors and the STMicroCubeIDE with a debugging tutorial. In it you will see how to compile for the debugger, debug, set break points, variable watches, and more.

Links/Resources:

Design Your Own Game Console eBook

Videos:

https://www.youtube.com/watch?v=hyZS2p1tW-g&t=262s&ab_channel=DigiKey
https://www.youtube.com/watch?v=Hffw-m9fuxc&ab_channel=MitchDavis

Sites:

https://os.mbed.com/
https://deepbluembedded.com/stm32-hal-library-tutorial-examples/
https://os.mbed.com/platforms/ST-Nucleo-L476RG/
https://www.digikey.com/maker-media/5cbeca93-099b-456f-aae2-dfa74f861581
https://www.arm.com/technologies/cmsis
https://www.st.com/en/microcontrollers-microprocessors/stm32l476rg.html#overview
https://www.st.com/en/evaluation-tools/stm32-nucleo-boards.html
https://www.st.com/en/development-tools/stm32cubeide.html#get-software

https://www.st.com/content/st_com/en/stm32-mcu-developer-zone/boards-and-hardware-tools.html

https://www.st.com/en/evaluation-tools/nucleo-l476rg.html#sample-buy
https://www.st.com/en/development-tools/stsw-link007.html
https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/
https://blog.st.com/stm32cubeide-free-ide/

https://www.st.com/content/st_com/en/search.html#q=STM32%20HAL%20programming-t=resources-page=1

https://www.arm.com/

https://developer.arm.com/documentation/102404/0201/About-the-Arm-architecture?lang=en

https://en.wikipedia.org/wiki/JTAG

Datasheets:

https://www.st.com/resource/en/datasheet/stm32l476je.pdf

https://www.st.com/resource/en/reference_manual/rm0351-stm32l47xxx-stm32l48xxx-stm32l49xxx-and-stm32l4axxx-advanced-armbased-32bit-mcus-stmicroelectronics.pdf

https://www.st.com/resource/en/programming_manual/pm0214-stm32-cortexm4-mcus-and-mpus-programming-manual-stmicroelectronics.pdf https://www.st.com/content/ccc/resource/technical/document/user_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf

https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

STM32CubeIDE:	CCAMD_CortexM0_01
	CCAMD_NucleoL476_Test_01
	CCAMD_NucleoL476_Test_02
	CCAMD_NucleoL476_Test_03
	CCAMD_NucleoL476_Test_04
	CCAMD_NucleoL476_Test_05