Conference Documentation

QwikBox

High-end Remote System Optimized for Uploading High Definition Video

Department of Electrical Engineering and Computer Science
University of Central Florida
Senior Design 1
Dr. Lei Wei

Group 25

Jonathan Kerbelis	Computer Engineer	jonathankerbelis@gmail.com
Eric Downey	Computer Engineer	edowney29@gmail.com
Harold Frech	Electrical Engineer	roldfrech@gmail.com
Todd DeNoyer	Project Sponsor	todd@qwikcut.com

Executive Summary

QwikCut is a company that provides video and analytical solutions for youth sport teams. The QwikCut team hires camera men to go out to these youth sporting events to capture the games and then the company edits the videos, creates copies, and uploads these games to a hosting platform called Hudl for internet viewing. The coaches and players want to have access to the video as soon as possible to enable these people to quickly critique themselves. The sooner the video is delivered the more time there is to prepare for future opponents and self critique. Thus, it is imperative to get the video back to the customers as fast as possible. This is where the QuikBox excels. Our project is to build a system that can quickly encode, store, and upload video. It will take a full game's worth of video clips and upload it onto a cloud based server in real time. This system will interact with a server to transfer the video onto a database so that the customer can directly access videos, from either a smartphone or computer, as soon as the uploading is complete.

To implement this process the project will include a combination of hardware and software components. The hardware will include a processor, memory, wireless modules on a PCB, a battery, and HDMI to USB interface, and a video camera. The video camera will live stream the video to the Qwikbox via HDMI, and the camera operator will split the video into clips through the use of a user-friendly mobile application. The video will then be clipped and encoded into a desired codec. Once encoded, the video files will be stored on a SSD and queued for upload to the server provided by the sponsor. Since upload speeds will vary depending on signal strength, a queue is necessary to ensure that each video file is uploaded successfully before the next file begins. The software will be an application that allows the user to access their respective video and be able to do so from a web based server. The desired video codec will be in h.265, a video compression standard, and the wireless systems will be designed for the 4G and 5G LTE networks. In addition to wireless cell networks, the system will have Wi-Fi and ethernet capabilities. These standards will ensure that the system will be relevant for years to come.

Project Goal

Our main goal for this project is to create a QwikBox that satisfies all requirements presented to us by our sponsor QwikCut. The project involve a cubed shape computer system about the size of a video game console. The box itself will be able to withstand mild weather conditions and be completely portable with the utilization of wireless data transfer and portable power source. QwikBox will involve the process of compressing, encoding, storing, and uploading high definition video data in the High Efficiency Video Coding codec; usually referred to as HEVC or H.265 codec.

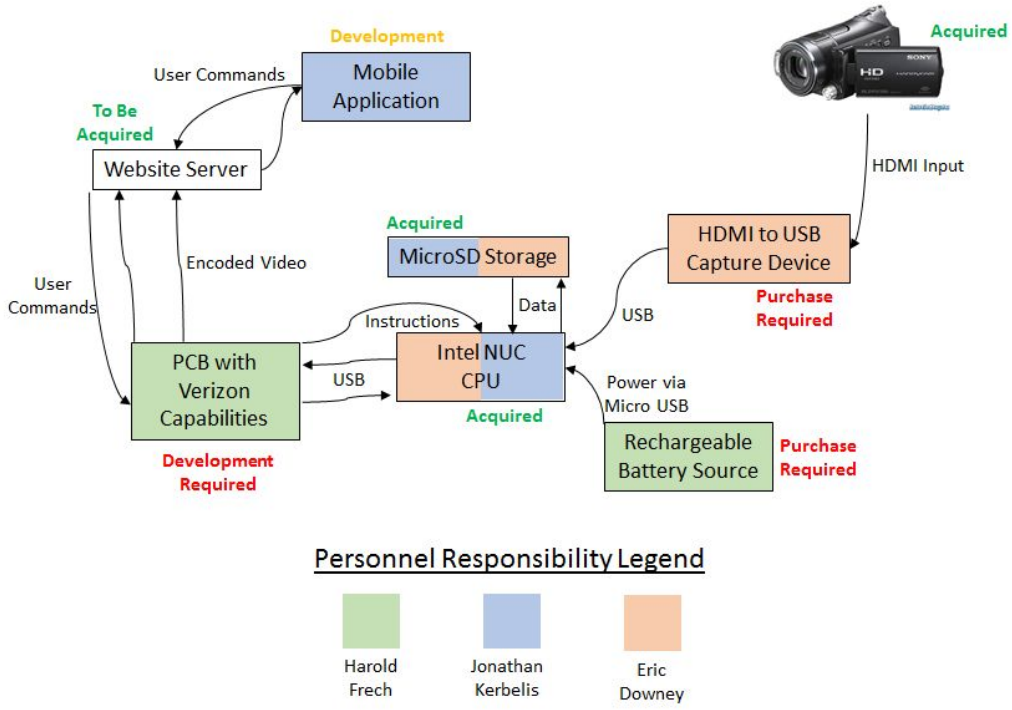
For the end user, our goal is to enable the user to have access to their videos through an online web based server which will allow them to view and edit video files. An android application will be developed to give our user access to these video files Above all, the system needs to have quick uploading speeds so that users can have access to their video soon after it is recorded. The wireless communication feature will provide QwikBox with this ability.

Project Objectives

The objectives of this project are to create a functional prototype of the QwikBox. We will do this by combining certain components to make a complete computer system capable of recording, compressing, storing, and uploading high definition video. In order to do this an Intel NUC will be connected to a camera. This will allow QwikBox to capture high definition video from a camera and store it to a SSD where it can be used later for compression and uploading. Along with that, a HEVC encoding software will be needed and must run on the Intel NUC. We will program the system to pull the captured video from the storage device and convert to a compressed format using the software on the microcontroller. From there, the now compressed high definition video will be uploaded to the web based server using the antenna connected to the Intel NUC. A printed circuit board will be created in order to allow the connections to be made between the microcontroller and anettna. The process from converting to storing and uploading needs to be as fast as possible so we will optimize every step of the process as the video is uploaded.

QwikBox will also be accompanied by a companion app that can be used to control the setting of QwikBox and view uploaded videos. This application will have a start and stop button on the featuring on the app that will allow the QwikBox to know when video should be recorded. The app will also feature a settings option that will allow user to name and edit videos. This application will communicate with the Intel NUC through the use of a custom Python Flask RESTful API designed to be hosted on the Qwikbox. This API will be tunneled to a public domain through the use of Ngrok's http forwarding service. This mobile application will be needed in order to operate the QwikBox. This means that without both the application and the QwikBox being fully functional, the system will not work correctly if at all.

System Design Information



List of technology:

1. the Intel NUC
2. a wireless antenna for data transfer
3. a high capacity external battery pack
4. the High Efficiency Video Coding codec (or H.265)
5. an HDMI to USB video and audio interface conversion tool
6. the Linux operating system
7. software for converting high definition video to H.265
8. the Amazon Web Services storage solution
9. Custom Python Flask RESTful API
10. Ngrok secure HTTP tunneling service

1. Firstly, The Intel NUC has been used in a plethora of projects to create fully functional systems from arcade video game consoles to robotic autonomous drones. There is no technological project that the Intel NUC can not be adapted too and this is why it is hugely popular among the homebrew hobbyist who create these different things. This versatility in form and function are what make the Intel NUC the perfect candidate for the QwikBox system. The Intel NUC comes equipped with great features such as expandable storage, multiple USB 3.0 hubs, and an ethernet and HDMI port. It provides us with many options for choosing how we want QwikBox to operate.

2. Another important technology to consider is the wireless communication device. This antenna will feature 4G and 5G LTE data transfer speeds and will have to be usable on Verizon's LTE data service. The antenna itself will be a part purchased from a retailer who makes plug-n-play antennas for the Intel NUC. A couple notable sources of these antennas are from NimBeLink and Adafruit Technologies. These companies have produced an economical means of obtaining and adding antennas with wireless communication abilities to small project such as the Intel NUC. These antenna will prove to be the best option for QwikBox.

3. A high capacity battery pack is not really an extremely new technology but only recently have they been pushed as a cheap market consumer product. These battery packs are very efficient and can carry a charge for the span of a few days. Using one these high capacity battery packs with our lower power consumption Intel NUC would allow us to use QwikBox for multiple days without the need for recharging the power source. The battery packs in consideration are mading by a bunch of different brands but hover around the 20000mA range which is more than enough to power the system. These battery packs will enforce the idea of having a portable QwikBox unit.

4. The High Efficiency Video Coding codec is currently the smallest compressed encoding standard that a video can be while still maintaining high definition quality. What this means for us is that we can use this standard to compress any video captured by QwikBox and compress it down to a very small file size. This will enable us to then upload the that same high definition video onto a web server a lot faster using wireless communication data transfer. Currently h.265 is the leading standard in data transfer because it's zero quality loss and 20% orginal file size compression algorithm. This standard is currently being used for service like Netflix and local news channels because the money saving aspect of less bandwidth usage based on file size. This technology is crucial in the the development of QwikBox and will allows to better utilize every resource available in the Intel NUC.

5. An important part of the QwikBox system is having the ability to adapt any high definition video stream to a usable interface for the Intel NUC. This can done in many different ways using different types of tools from software to hardware. A typical way way to make HDMI input some accessible is by changing the the format to a USB input interface while maintaining the proper quality and format of the original video. This is where an HDMI to USB input conversion hardware will come in handy for universal use on the Intel NUC.

6. The Linux operating system a good OS for Intel NUC systems because of its general accessibility, ease of usage, reliability, and lightweight performance. When comparing this to other system such as Windows or Mac, it becomes clear that Linux is clearly superior for projects such as this one. Linux also offers cross

platform accessibility to other microcontroller because which will make the QwikBox system portable to other platforms. Using the Linux OS will also mean that any distribution of Linux can be used on the Intel NUC allowing for greater control over speed and performance. Also Linux offers a lot of documentation from online communities giving us a greater understanding on to best use the Linux operating system.

7. Another important aspect to the QwikBox is the video conversion method that will be used. There are many ways to go about doing this but each method includes many different advantage and disadvantages. Software suchs VLC media player will allow use to easy convert video from any format to the H.265 codec but at a slower speed than physical hardware conversion would offer. Although advantage to this method means is the fact that the compression is done completely by the Intel NUC. This is good because it gives us greater control over the entire process through the Intel NUC operating system. From here we can see the entire process happen and determine when is the best time for each step from recording to compressing to uploading to happen.

8. By using Amazon's Web service as a storage solution for QwikBox, it given the user more control over their uploaded videos. As of right now the biggest competitor for sports film hosting is Hudl. We plan to use our application and Amazon's storage to support our on hosting service for videos. Amazon is also very reliable and offers automatic backups of all uploaded files. This allows us to guarantee that will video will be saved for future use to our users.

9. The Python Flask RESTful API was developed to be able to handle HTTP GET and HTTP POST calls to a specific port of the NUC's localhost. This HTTP calls will incorporate basic authentication that should prevent unauthorized sources from tinkering with the device. The calls will only be accepted if send it the correct format from the mobile application that is being built to operate the Qwikbox remotely. The POST calls will incorporate data such as a specific game ID, play number, and any other data the sponsor deems necessary.

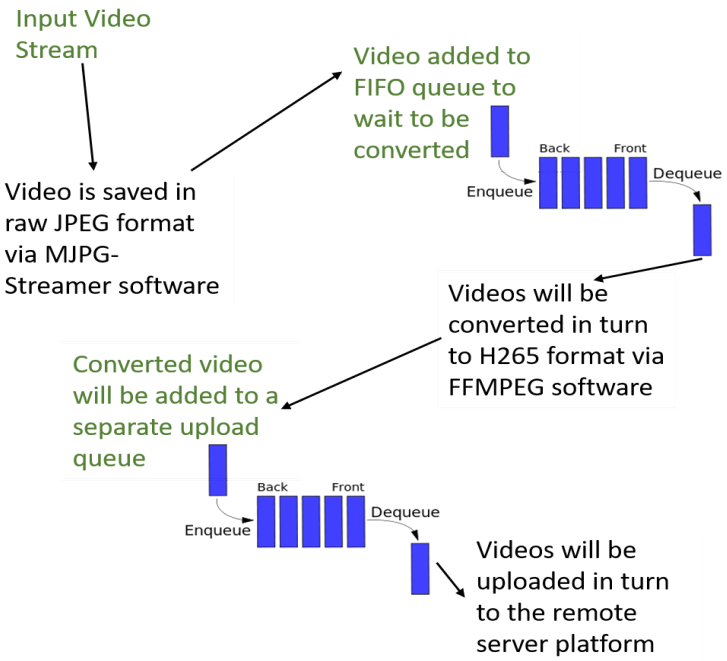
10. The Python Flask RESTful API must be hosted on the NUC itself in order to the run the necessary capture and upload scripts. However, an API that is hosted locally prevents devices that are not on the same immediate network from communicating with it. This is where Ngrok comes in. It tunnels the HTTP calls from a public domain ('qwikbox.ngrok.io' in this case) to the localhost of the Qwikbox. This allows the API to be publically accessible while also having the ability to run the necessary scripts locally.

Intel NUC System

The Intel NUC is the centerpiece of our entire project. It was chosen due to it's power and performance, and its PC-like features. The Intel NUC comes with WiFi and Bluetooth capabilities, along with a HDMI port, 4 USB ports, and an ethernet port. It also comes preloaded with Python, C, C++, Java, Scratch, and Ruby programming languages, giving us plenty of options with how we want to develop the system. Since it runs on a Linux-based Ubuntu operating system, having bash scripting capabilities will be extremely useful. These scripts are used for various purposes, including moving files around and organizing them, as well as sending video clips to the server and have other purposes. The specifications of the Intel NUC is as follows:

- 4th Generation Intel Core i5-4250U
- Intel HD Graphics 5000
- Dual channel SODIMM DDR3L 1333/1600 MHz, 1.35V
- 128GB M.2 SSD
- 10/100 Ethernet, 2.4GHz 802.11n wireless
- Bluetooth 4.1, Bluetooth Low Energy

The Intel NUC storage is an M.2 SSD and it's operating system runs entirely off of that. It is faster than a typical hard drive and has plenty of speed for purposes of this project. The system will be set up with a main program that communicates with a mobile application that is running on a the camera operator's phone. The mobile application will tell the program when to start recording the input video stream. Our program will then control the ffmpeg software to stream the video into the Intel NUC. The image below defines the process and how this will be done.



FFMPEG is a command-line tool that takes a stream and converts it to a video file with a specified codec. The input stream will be converted the H265 codec in real time in order to minimize the time needed to get the final video uploaded the the S3 bucket. This video input can be streamed in at various resolutions and frame rates, but converting in real time like this relies on a perfect balance of output settings so that the video does not become slowed down or sped up as it is streamed in from the camera. To accomplish this, FFMPEG relies on an input plugin that copies the images to an accessible memory location and an output plugin that processes the images that can save to a single H265 video file. When the user chooses to end the recording, this stream will cut off and the video file will be saved. For our project this file will need to be at least 1280x720 and have a minimum frame rate of 30 frames per second.

Now that the recording is over, a script will be run that moves the new file to a specific folder. The next part of each video clip's process is a queue, which is set up to upload all the video files in this specific folder that contains all of the current video clips. Once the upload is confirmed the video files will be deleted to make room for the rest of the clips. These videos will be uploaded to an Amazon AWS S3 server and stored there for future viewing .

User Application Information

Most of the server's operation is outside the scope of this project, however the video will need to be organized once it ends up on the server. This will done by the sport being played, the date the game occurred, the division of the teams, and the team names. There will be games that have many clips just for a single game, especially for football. This clips will need to be numbered in order so that the game sequence can be easily followed once all of the clips are located on the server. The server will need the ability to communicate back and forth with the mobile application so that the user can specify how they would like the organization to take place. The image below shows the logic process behind the mobile application and server.