

# EECS150: Fall 2008 Project, Digital Storage Oscilloscope

UC Berkeley College of Engineering  
Department of Electrical Engineering and Computer Science

## 1 Time Table

---

<b>ASSIGNED</b>	Friday, October 10 <sup>th</sup>
<b>DUE</b>	Week 14: Wednesday, November 26 <sup>th</sup> , 11:59pm

---

## 2 Objectives

The primary goal of this project is to familiarize EECS150 students with digital design methods and tools. However, in order to make the project interesting, while keeping it useful, the end result of the project for Fall 2008, is a Verilog HDL representation of a hardware implementation of a Digital Storage Oscilloscope.

At the heart of this project is the ability to display data, in various formats, on a TV screen as a waveform. After a mechanism for displaying waveforms and visualizations is complete, the project will move to a feature-centric format. Some features that you might implement are audio storage and playback, fast forward, rewind, freeze-frame, zoom, or data transfer to another CaLinX2 board. Some extra features will be required, and described in coming weeks. Others will be extra credit.

### 2.1 Philosophy

This document is meant to describe the input/output specification for the project and its associated support hardware, as well as lay out the plan for completing this project. As with any design you will encounter in the professional world, we are merely providing a framework within which your project must fit. **Unlike other college classes we will not tell you exactly what to write or how to design your project.**

You should consider the TAs a source of direction and clarification, but it is up to you to produce a design which is fully functional on the CaLinX boards. We will attempt to help, when possible, but ultimately the burden of designing and debugging your solution lays on you.

In the end, what counts is having a well functioning, bug free, project **by 11:59pm on the Final Project Checkoff deadline: Wednesday, November 26<sup>th</sup>, 2008.** Due on November 26<sup>th</sup> is everything described and required in the checkpoint documents (1-5) that you will be getting in the coming weeks. In addition, the week before the project is due will be allocated purely to **extra credit**. Whilst extra credit is optional, it will add a significant point percentage to your final project grade, and is thus highly recommended. The philosophy is as follows: finish that which is required a week early, and spend the last week fixing bugs and making your project unique from the rest of the pack. **You will be completely done with the project by the start of Thanksgiving break.**

### 2.2 General Project Tips

Make sure to use top-down design methodologies in this project. We began by taking the problem of designing an Digital Storage Oscilloscope, and then broke it down into manageable 1 week checkpoints. You should take this scheme 1 step further: we have given you each checkpoint, so break each into even

smaller and more manageable pieces. If you follow this guideline, and follow our interface specifications, you should be able to split the project up between you and your partner and work more efficiently.

Design, code, simulate, and debug modules.

- Design them well, make sure you understand what you want before beginning coding.
- Code exactly what you designed; do not try to add features without redesigning.
- Simulate thoroughly, writing a good testbench is as much a part of creating a module as actually coding it.
- Debug completely, anything which can go wrong during final checkoff will.

Document your project thoroughly, as you go. Your design review documents will help, but you should never forget to comment your code, or keep your diagrams up to date. Aside from the final project report, where you will need to turn in your documentation, you can use it to help yourself during the debugging process. Finish the required features first. Attempt extra features after everything works well. **If your submitted project does not work by the final deadline, you will not get any credit for any extra credit that you implemented.**

### 3 Project Description

This section will duplicate what you have or will read in the checkpoint documents. It is the complete, high-level project specification. For more detail please refer to the appropriate checkpoint or lab lecture.

**Figure 1** Fall 2008 Project: Digital Storage Oscilloscope: Top-Level Diagram

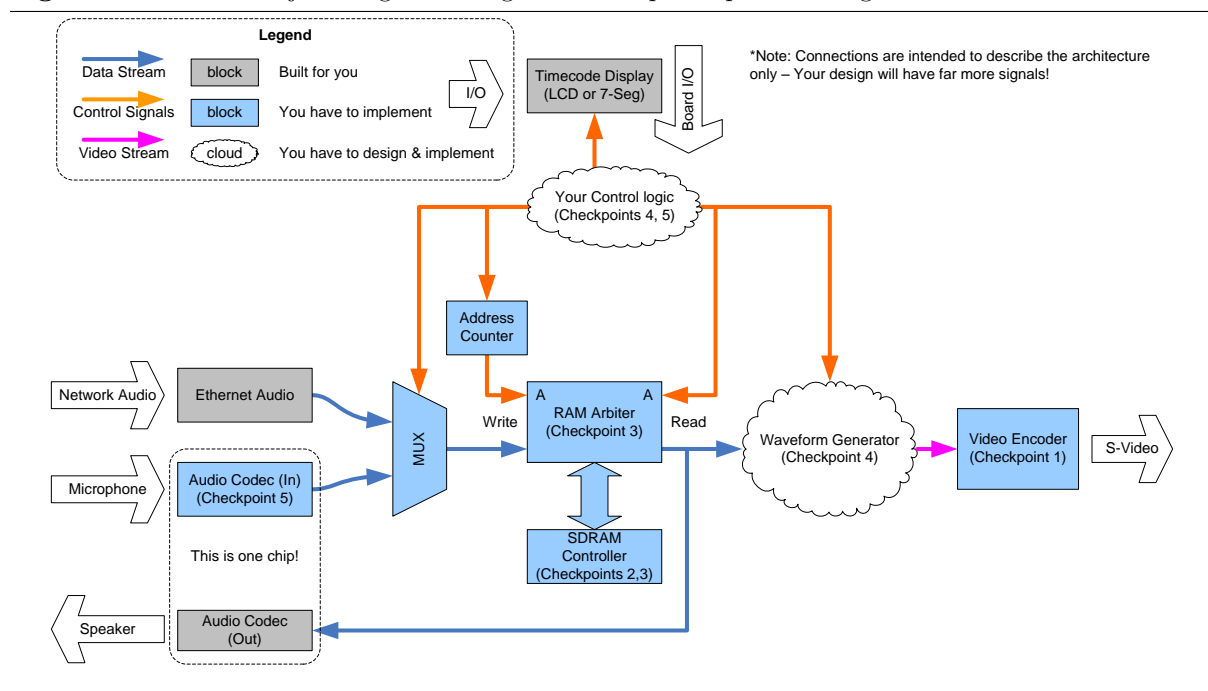


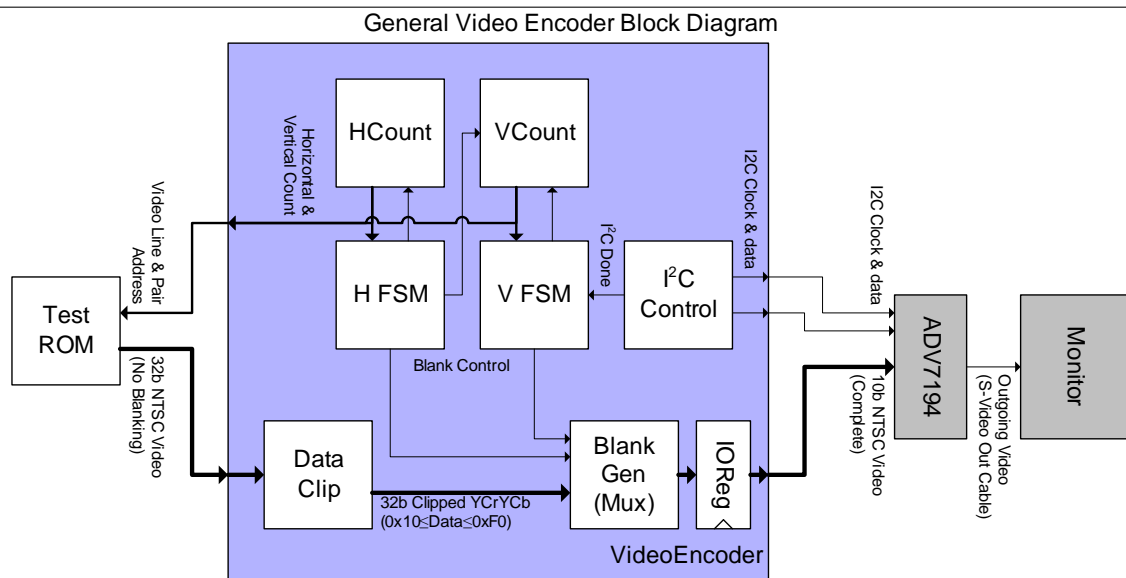
Figure 1 shows a top-level view of the project. First, either Ethernet (think lab 6) or a microphone will provide a stream of data. The stream can be saved into memory, and read out as needed (concept similar to a FIFO). When called upon, the stream of data is rendered on-demand: a stream of amplitudes (Y values in an X-Y plot) is converted to pixels on a screen. Finally, the video stream is sent to the TV screen through an external Video Digital-to-Analog chip.

### 3.1 Video Encoder

The first part of the project (checkpoint 1) deals with the ADV7194 Video Encoder chip on the CaLinX boards. Figure 2 shows a block diagram of the module you will be designing.

The goal of this part of the project is to give you a means of transforming video data into a visual representation of itself on the TV screen. Down the road, you will read data out of memory, or stream data directory, to be displayed on the TV. All data that you display must, of course, pass through your video encoder. As such, treat this part of the project with care! It will serve as your bread and butter for the rest of the semester

**Figure 2** The Video Encoder



Notice in Figure 2 that the video encoder can be decomposed into smaller units. Your design should be well partitioned to match. Note that you will be given I<sup>2</sup>C as a blackbox. The checkpoint would not be doable in the time that you have been given if you were forced to design I<sup>2</sup>C.

### 3.2 SDRAM

The SDRAM controller and arbiter will be the main goal of checkpoints 2 and 3. Figure 3 shows how these modules will fit together in a general sense. The SDRAM will store audio samples that will be processed by the waveform generator (see Section 3.3).

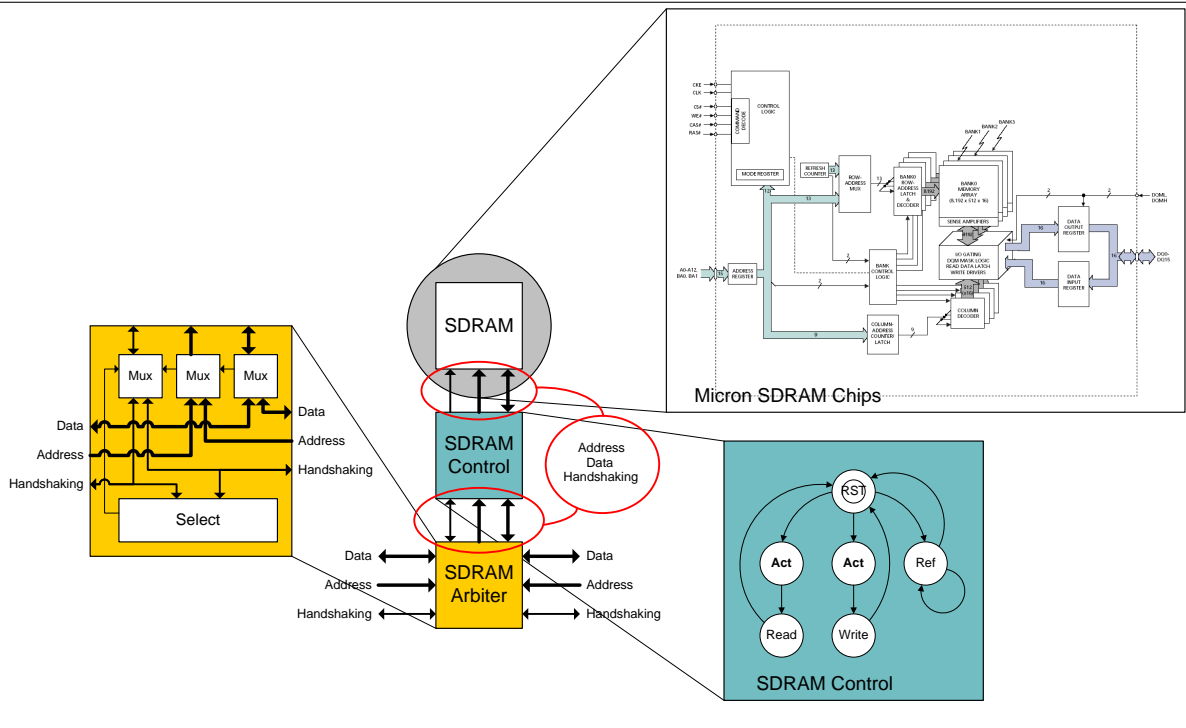
#### 3.2.1 SDRAM Controller

The primary function of this module is to control access to the Micron Technologies SDRAM chips. Because SDRAM chips require very specific command sequences and timing, we will make use of a simple FSM to provide those commands and timing, abstracting away the complicated problem of talking directly to the SDRAM.

#### 3.2.2 SDRAM Arbiter

The base project will only require a two-port arbiter. The arbiter will allow the audio subsystem to write to SDRAM and the waveform generator to read from SDRAM. The key job of the arbiter is to make both the audio subsystem and the waveform generator unaware of the fact that they do not have exclusive access to the SDRAM, while at the same time guaranteeing that they both get some chance to access memory.

**Figure 3** SDRAM and the Arbiter

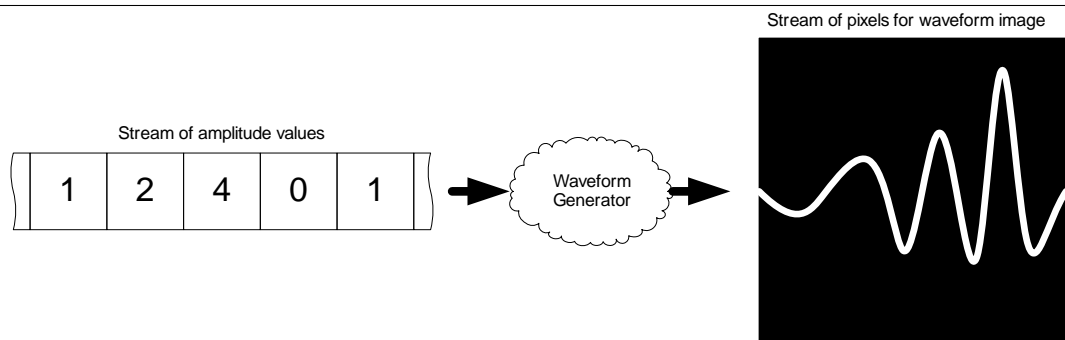


### 3.3 Waveform Generator

The “Waveform Generator” is the main goal of checkpoint 4. The focus of this part of the project is largely design, not implementation. As EECS150 is a design course, we will give you a generic problem statement, and ask for an implementation that conforms to the few requirements we do enforce. You will be responsible for evaluating, making, and justifying the design decisions as you complete this section of the project. Given a good design, implementation should not be difficult.

Keep in mind this checkpoint involves a lot of work! Start early. You will build a module, or a set of modules, responsible for converting the stream of amplitudes (audio data) into a waveform (video data). Also, you will build a control responsible for interfacing with the switches/buttons on the board, directing the oscilloscope to start/stop recording, review previously recorded audio data, and play it back. Figure 4 illustrates the high-level requirements.

**Figure 4** The Waveform Generator



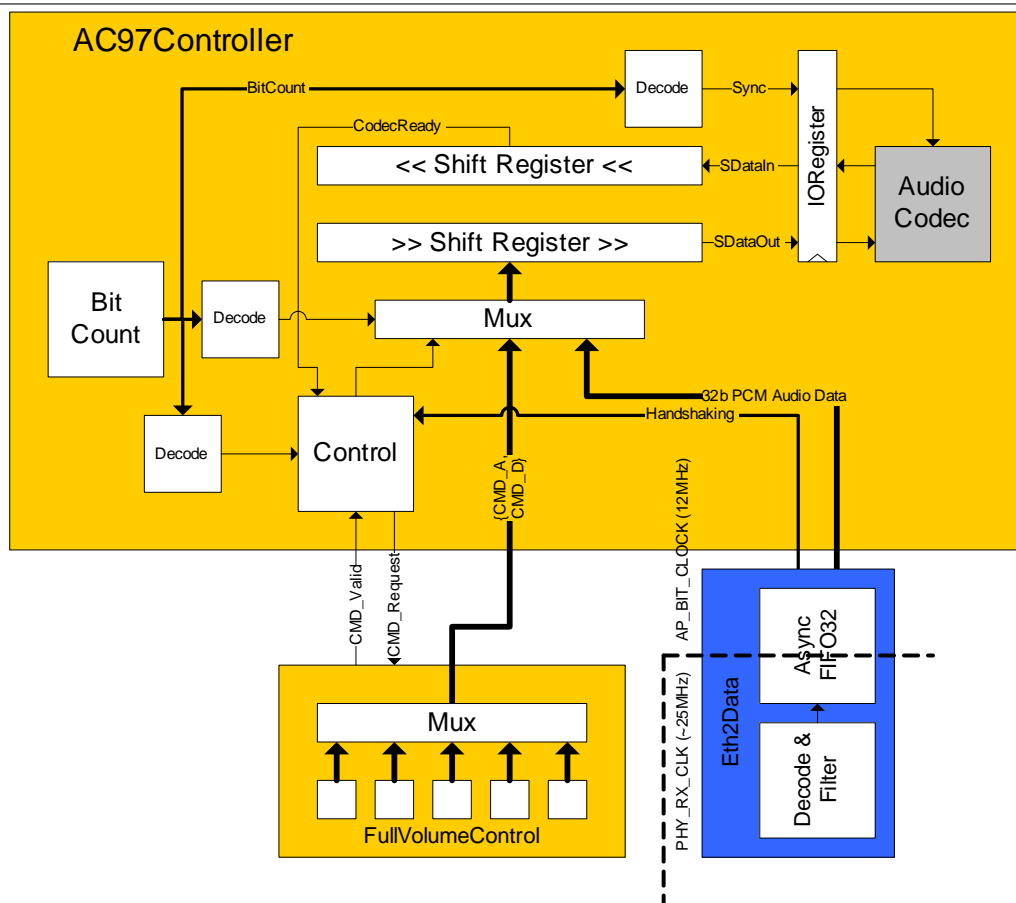
The Waveform Generator is thus responsible for tying the entire project together. Given RAM, a source of audio, and a video encoder, it is your job to design and implement a complete oscilloscope-like system.

### 3.4 AC97 Audio

The last part of the project, recreating and extending audio input and output, is the focus of checkpoint 5. Shown in Figure 5 below is the overview of the audio input system. This system will replace the TA blackbox for the audio codec from lab 6 and allow you to generate waves from microphone input as well as from an audio stream.

The core of this part of the project is the AC97 controller, whose primary duty is to interact with the LM4549A audio codec on the CaLinX2 boards. In this checkpoint the first goal is to properly generate the AP\_RESET\_ and AP\_SYNC signals needed to reset and synchronize communications between the FPGA and codec. After this is finished, you will need to build a controller which can correctly generate the 256b AC97 frames, pulling together the Slot0 frame header, volume control commands and PCM audio data.

Figure 5 AC97 Audio



As can be seen in Figure 5, most of the functionality in the AC97 controller is implemented with a bit counter, and a pair of shift registers. However for the checkpoint, you will need to design the control logic to make good use of this infrastructure. In addition, you will need to create some volume control modules to manage the mixer settings on the LM4549A.

### 3.5 Extra Credit

Please refer to Section 4.8 for details.

## 4 Deadlines and Assignments

This section summarizes the functionality you need to demonstrate, as well as the design steps you will need to follow in order to receive full credit for the project in this course. Indicated in parenthesis in each heading is the percentage that portion of the assignment counts for.

### Design Reviews

**Design reviews** may only be checked off by your own TA during your lab time. Both partners must be present. In an extremity we will allow you to get a design review checked off during your TAs office hours.

Design reviews are graded on a binary scale: you either have your design or you don't. You can demonstrate your design to your TA through:

1. Block diagrams
2. Bubble-arc diagrams
3. Schematics (“on a napkin”)
4. Timing diagrams

The point of design reviews is to give you feedback on your design, and get you to start thinking clearly about the module you will be building before you start writing code. Notice the trend: nowhere in the above list is code mentioned. Only diagrams are permitted at design reviews. No code will be accepted at design reviews. See the assignment sheets for each checkpoint for more details.

### Checkoff

**Each checkpoint (except for checkpoint 5 and final checkoff (see Section 4.6)) is due at the end of the week at the start of lab lecture. Read carefully: you must be checked off by lab lecture (2:10pm on Friday, sharp).** The checkoff policy is as follows:

1. To be checked off, you must demonstrate to a TA that your solution produces the correct results when used in conjunction with the TA solution / verification suite that is provided for that checkpoint.
2. If you ask your **lab TA (in your assigned lab section)** to check you off, you are guaranteed to get checked off.
3. You can get checked off in any lab section, however, the TAs make no guarantee that you will get checked off if you try to get checked off in a section other than your own.

The moral of this story is to try to get checked off in your assigned section. It is very unlikely that we will be able to check you off right before lab lecture. In other words, dont count on it.

### Project Report

After final checkoff is done, you will be required to write a report cataloging your design, design decisions, and features. This report is largely taken care of for you if you maintain good design documents throughout the course of the entire project. We expect about half of the report to be diagrams. More information concerning the project report will be given in the coming weeks. For information regarding the report due date, see Section 4.7.

#### 4.1 Checkpoint 1: Video Encoder (14%)

In this checkpoint you will be building an interface with the ADV7194 Video Encoder chip on the CaLinx2 boards. By the end of this checkpoint, you will be able to stream video data from a TA blackbox, which will generate a color pattern, to your TV screen.

This checkpoint will teach you the value of clean interfaces in hardware. This checkpoint will also introduce you to reading datasheets.

---

**Table 1** Checkpoint 1: Video Encoder time table

---

Deliverable	Due Date
Assigned	10/10
Design Review	10/12-10/18
Checkoff	10/24

---

## 4.2 Checkpoint 2: SDRAM Controller: Simulation (14%)

In this checkpoint you will design a controller that initializes, reads and writes to two Micron SDRAM chips on the CaLinx2 boards. By the end of this checkpoint, you will be able to simulate, in ModelSim, a working SDRAM controller that will be able to initialize at startup, read and write data.

---

**Table 2** Checkpoint 2: SDRAM: Simulation time table

---

Deliverable	Due Date
Assigned	10/17
Design Review	10/19-10/25
Checkoff	10/31

---

This checkpoint will give you extensive experience in designing an FSM that performs different tasks for actual hardware, depending on the needs of other modules.

## 4.3 Checkpoint 3: SDRAM Controller: Arbiter & Hardware Verification (14%)

In this checkpoint, you will design an SDRAM ‘Arbiter’ to interface with your SDRAM Controller. Additionally, you will have to prove that your SDRAM Controller works on the CaLinx2 boards, as opposed to just in simulation. By the end of this checkpoint, multiple modules in your design will be able to read/write data to/from the same SDRAM.

---

**Table 3** Checkpoint 3: SDRAM: Aribter & Hardware Verification time table

---

Deliverable	Due Date
Assigned	10/24
Design Review	10/26-10/31
Checkoff	11/7

---

This checkpoint is the first where “planning for the future” is very important: your Arbiter implementation will make a large difference in how customizable your project is when it is time to work on extra credit.

## 4.4 Checkpoint 4: Waveform Generator (14%)

In this checkpoint, you will design hardware to convert a stream of audio data (the same format as that given to you in Lab 6) into a stream of video data that the video encoder will use to show waveforms on the TV screen. Once your ‘waveform generator’ is complete, you will also have to add the following features:

1. The ability to start and stop audio recording from an audio source to SDRAM.
2. The ability to play back recorded audio.
3. The ability to keep track of what time a specific peice of audio was recorded.

---

**Table 4** Checkpoint 4: Waveform Generator time table

---

Deliverable	Due Date
Assigned	10/31
Design Review	11/2-11/8
Checkoff	11/14

---

This checkpoint will give you significant experience with implementing a design, with a very limited specification, that follows the requirements specified by modules that you have built in previous checkpoints.

#### 4.5 Checkpoint 5: AC97 Audio (14%)

In this checkpoint you will be designing and building a module to interface with the LM4549A audio codec on the CaLinx2 boards. By the end of the checkpoint you should once again have network audio playing, this time with your audio controller instead of ours. Additionally, you will be able to speak through a microphone as a way of recording and playing back sound.

---

**Table 5** Checkpoint 5: AC97 Audio time table

---

Deliverable	Due Date
Assigned	11/7
Design Review	11/9-11/15
Checkoff	11/26 (Wednesday, 11:59pm sharp)

---

This checkpoint will provide you with significant design experience with datapath and control abstraction and I/O controller design.

#### 4.6 Final Checkoff (20%)

By submitting a working solution by the final due date, which fulfills all of the requirements of all of the previous checkpoints and successfully composes the five checkpoints into a complete project, you will have completed “final checkoff. When you get back from Thanksgiving Break, you will demo your final project to your TA. All code/blackboxes/etc required to build your project (run through synthesis, translate, map and PAR) must be submitted by the November 26<sup>th</sup> due date.

---

**Table 6** Final Checkoff time table

---

Deliverable	Due Date
Final Checkoff	11/26 (Wednesday, 11:59pm sharp)

---

The procedure for final checkoff will be as follows:

1. Finish and completely debug your project.
  - (a) **Be sure to test on multiple CaLinx2 boards.**
  - (b) Make sure your project works perfectly after reset.
    - It may require reset before it will work at all (that is fine).
2. Using your group name, sign up for an early or final checkoff slot.
  - (a) Checkoff slots are 20 min long.
  - (b) The signup sheet will be posted next to the TA station.
  - (c) **You and your partner must be present for checkoff.**

- (d) You may change checkoff slots at any time.
  - (e) Sign up for a checkoff slot early in the day, if you have other classes, we will most likely get behind.
  - (f) **Please erase or cross out your name if you do not intend to get checked off at the time you signed up for.**
3. `svn commit`<sup>1</sup> your files by the checkoff date and time.
- (a) **Make two subfolders: “Normal” and “ExtraCredit.”**
    - i. The “Normal” folder should contain all the Verilog and \*.edn files needed by your non-extra credit project.
    - ii. The “Extra Credit” folder should contain all the Verilog and \*.edn files needed for your extra credit project.
    - iii. Submitting the two separately will guarantee that your Normal version will work.
  - (b) **Do not submit bitfiles or anything else non-essential!**
    - This goes for the rest of SVN: only ever commit .v files.
    - When in doubt about what to commit/submit, ask your TA.
4. **Test your submitted files.**
- (a) Log into a random computer.
  - (b) Copy the files to the local hard-drive.
  - (c) Create a Xilinx project.
  - (d) Synthesize your project.
  - (e) Test it on the board.
5. **All submissions must be in by 11:59pm on your checkoff day!**
- (a) **Absolutely NO late work will be accepted, no exceptions.**
  - (b) If what you submit does not work, you will get a chance to perform small changes, at the cost of reduced credit.
6. Show up for checkoff.
- (a) **Arrive at least 20min early in case of problems.**
  - (b) If you are not there on time, your slot may go to a waiting group.
  - (c) **We will build your project from .v and blackbox files** and then program a board with your project.
  - (d) You will demonstrate its features.
  - (e) We will ask you a series of short questions about your design, debugging and general experience.
  - (f) **Both partners must be present.**
    - i. **If you are not at final checkoff with your partner you will not receive full credit.**
    - ii. You may change checkoff timeslots at any time.
7. **ALL FILES MUST BE SUBMITTED BY 11:59pm SHARP! NO EXCEPTIONS WHATSOEVER!**

#### 4.7 Project Report (10%)

A large percent of the project (10%) is allotted to the project report. This is because the report is what you take away from this class. All of your hard work, effort, sweat and blood should be reflected in the project report. Employers might (and have in the past) asked for it during interviews. Be proud of this report. It is your medal for surviving EECS150.

<sup>1</sup>You will be introduced to SVN and how to use it on 10/14.

---

**Table 7** Project Report time table

---

Deliverable	Due Date
Assigned	11/21
Report	12/5 (Friday, 2:10pm sharp)

---

## 4.8 Extra Credit (15%)

The final week of the project is devoted to extra credit and design verification. Over the next few weeks, a detailed list of extra credit options will be published. You may choose to implement any of these options. Some will carry more point totals than others. Each options point total will be based on how difficult the TAs deem that option will be to implement.

---

**Table 8** Extra Credit time table

---

Deliverable	Due Date
Extra Credit	11/26 (Wednesday, 11:59pm sharp)

---

The list of possible extra credit will be available by the start of **checkpoint 3**. After you have completed the Video Encoder and SDRAM checkpoints, the world will be your oyster. Feel free to implement extra credit during checkpoint 3, 4 and 5 development. Do not forget, however, that your first priority should be to finish each checkpoint. Only implement extra credit during the intermediate checkpoints if you have time. **Remember, a whole week after checkpoint 5 has been explicitly set aside for you to implement extra credit.**

## 5 Late Policy

As mentioned in Section 4.6 numerous times, you will get no credit for final checkoff if you submit late work. This same policy applies for the rest of the checkpoints: **if your solution is not checked off by each due date, you will get ZERO (0) credit for that checkpoint.**

This policy is not to be unnecessarily harsh. Each checkpoint will be complemented with a blackbox solution that will be released when the checkpoint is due. You will be able to use this blackbox in your project if your solution doesn't get done on time. The simple truth is that if you fall behind and dwell on the failure, you will never finish. This policy is meant to keep you on your toes so that this doesn't happen.

Rev.	Name	Date	Description
A	<a href="#">Chris Fletcher</a> Ilia Lebedev Arjun Singh	10/9/08	Wrote new document; format based on previous semesters' project specifications