



Interim Report

Real-Time Audio and MIDI Control iPad
Application with Backing Drum Generation

Steffan R Walters
Supervisor: D Marshall
Moderator: K Sidorov
CM0343 - 40 credits

Table of Contents

Table of Figures	3
Introduction	4
Background Research	5
<i>Other Similar Applications</i>	5
Similar iPad Applications.....	5
Similar Computer Software.....	11
<i>Technologies Needed</i>	14
iPad Application Development	14
Objective-C / C [13].....	14
Pure Data / libpd [19].....	15
MIDI.....	16
<i>Other Related Material</i>	16
Approach	17
<i>Overall Concept of the Application</i>	17
<i>Methodology</i>	18
<i>MIDI Section</i>	18
<i>Drum/Beat Generator Section</i>	19
Conclusion	20
Glossary	21
Table of Abbreviations	21
Appendix	22
<i>Basic Draft Designs</i>	22
<i>Time Plan / Work Plan</i>	23
Completed Tasks	23
Future Tasks	24
References	26

Table of Figures

Figure 1 - Korg KAOS pad (source: http://www.korg.co.uk/products/dance_dj/kp3/dj_kp3.asp)	5
Figure 2 - The Yamaha Tenori emulation (source: http://itunes.apple.com/us/app/aurora-sound-sound-hd/id375109647?mt=8).....	6
Figure 3 - Beatwave (source: http://itunes.apple.com/us/app/beatwave/id363718254?mt=8)	7
Figure 4 - Soundgrid (source: http://www.mifki.com/soundgrid/)	8
Figure 5 - TC11 Multi Touch Synthesizer (source: http://www.idesignsound.com/tc-11-multi-touch-synthesizer-ipad).....	9
Figure 6 - Touch OSC (source: hexler.net/software/touchosc)	10
Figure 7 - GarageBand (source: http://www.apple.com/ilife/garageband/)	11
Figure 8 - Wavepad (source: http://www.nch.com.au/wavepad/index.html)	12
Figure 9 - Audacity (source: http://audacity.sourceforge.net/)	12
Figure 10 - Cubase (Score Editor) (source: http://www.steinberg.net/en/products/cubase/what_you_get.html)	13
Figure 11 - Cubase (Mixer) (source: http://www.steinberg.net/en/products/cubase/what_you_get.html)	13
Figure 12- Draft Interface for the Drum Section.....	22
Figure 13 - Draft Interface for the MIDI Section	22

Introduction

The idea behind this project, is to create a tactile musical application for Apple's iPad tablet. This application will hope to emulate certain features of the Korg KAOS pad as well as other similar applications such as Beatwave, Soundgrid and Simplemidipad. The main features that this application will have, will be the real-time audio effect control and MIDI control, I believe that these are key to creating a good musical application. The MIDI control will allow a user of the application to enter their own input (using a MIDI keyboard) instead of choosing from a preset defined list of sounds, and then they will be able to modify the sound that they create to make a new sound of their choice (by using certain effects). To add another level of uniqueness to the users created sound, the application will allow the user to create their own backing drums and add them on to the sound that they create.

As I have already created some iPad applications, I know that I will need an extensive knowledge of the Objective-C programming language, mainly the Core Audio/MIDI aspects of it. I have also discovered that Pure Data will feature heavily in the finished application due to its ease of use when it comes to MIDI effects. I have learned of Pure Data due to the book, 'Making Musical Application' by Peter Brinkmann[1].

Here is a summary of the features that will be included in the application:

- MIDI input using a MIDI keyboard
- MIDI control for the iPad
- Real-time audio controllable effects using different methods:
 - Buttons and sliders
 - Gesture input such as 'two finger press'
- A drum generator
- Backing drums with a MIDI accompaniment

The finished application will be able to run on the iOS operating system currently used by Apple's iPad tablet. A suitable MIDI keyboard and adapter will also be required to use the application.

Background Research

To begin my research around the problem area, I have conducted research into similar applications. There are numerous applications/software products that relate to the idea behind my project and as such I have described the main points of some of these applications/software versions below.

Other Similar Applications

Similar iPad Applications

As stated in the introduction, there are a number of applications which tend to do similar things to what my project will be trying to address. These applications include:

- The Korg KAOSS pad - This allows a user to touch a touch screen control at the centre of the pad to control various effects/parameters, this pad also has MIDI control built-in. I hope to emulate some of the features on this device in my project, these features include, MIDI control, some effects using an x,y grid or button and recorded samples. However, the device has numerous other interesting abilities that would take too much time for me to try and implement such as breaking down the samples to individual streams and the graphical interface of the x,y grid. [2, 3]



Figure 1 - Korg KAOSS pad (source: http://www.korg.co.uk/products/dance_dj/kp3/dj_kp3.asp)

- The Yamaha Tenori emulation - This application is a pattern based musical sequencing software. The features of this application include: 9 effects (e.g. delay, reverb), exporting of songs, 14 instrument tracks and the creation of custom instruments. The interface of this application is similar to the idea that I had for the drum generator section, where you can decide whether to have a drum on or off for a specific beat. Unlike this application I will also be implementing the ability for MIDI input. [4]



Figure 2 - The Yamaha Tenori emulation (source: <http://itunes.apple.com/us/app/aurora-sound-sound-hd/id375109647?mt=8>)

- Beatwave - As with the Yamaha Tenori emulation, this application is focused around an x,y grid pattern. The features of this application include: tone control, tempo control, exporting of the track and multiple instruments for one track. Again this has a simple, easy to use interface and I am hoping to create my interface to emulate this ease of use. This application does not however support MIDI input. [5]

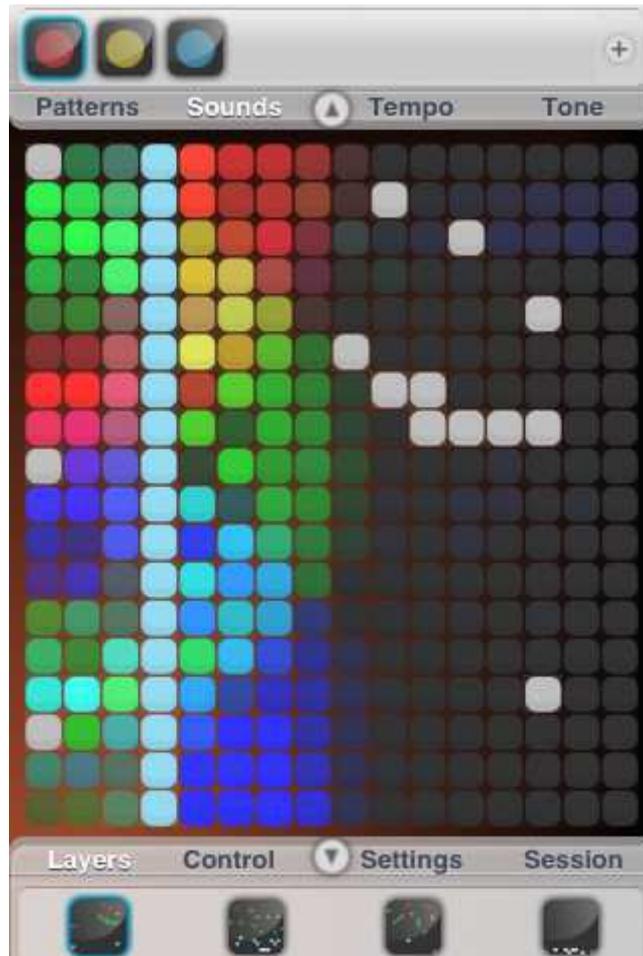


Figure 3 - Beatwave (source: <http://itunes.apple.com/us/app/beatwave/id363718254?mt=8>)

- Soundgrid - Most musical applications that I have found are based around an x,y, grid pattern and sound grid is no exception. Again as with the other applications, multiple patterns can be played over each other and this is used to create audio-visual performances. The tracks created using this application can also be recorded and exported to the WAV or AAC formats. [6]

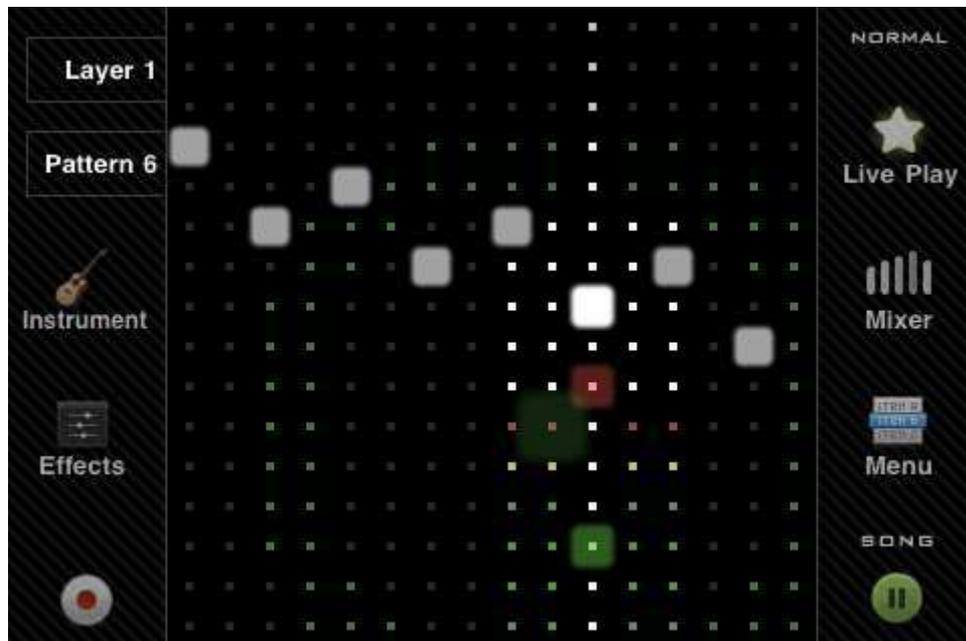


Figure 4 - Soundgrid (source: <http://www.mifki.com/soundgrid/>)

- TC-11 Multi-Touch Synthesizer for iPad/ iPhone - The main feature of this application, is that it has multi-touch control instead of using on-screen knobs, keys or sliders. There are 112 built-in presets and the user can also create their own patches. This application also takes advantage of the iPads accelerometer, gyroscope and compass which I believe is a good feature that I would like to try and emulate, but I think it is out of the scope of my project. By watching a tutorial on how to create a patch using this application, I noticed how complex this process could get and I'm hoping that my interface for creating a beat/drum beats will be a lot easier to use. [7]



Figure 5 - TC11 Multi Touch Synthesizer (source: <http://www.idesignsound.com/tc-11-multi-touch-synthesizer-ipad>)

- Touch OSC - "Touch OSC is a modular OSC and MIDI control surface for the iPhone/iPad/iPod Touch"[20]. This application supports the sending of "Open Sound Control messages over Wi-Fi using UDP"[20], it does also support the CoreMIDI framework that I will be using. Touch OSC provides a number of different controls to allow the sending or receiving of messages. These controls include: Faders, LEDs, Labels and Multi-xy pads. This application does also use the accelerometer function of the device which would be an interesting feature to implement in my application if I had time to. [8]



Figure 6 - Touch OSC (source: hexler.net/software/touchosc)

Summary of similar applications

After researching similar applications, the main similarity between all of the applications is that they are all excellent applications for creating music. Most of these applications also use an intuitive interface consisting of an x,y grid to allow the user to choose when a certain sound is played or not, I will be using a similar idea with regards to the drum generator in my application. Unlike most of the applications that I have researched I will be allowing the user to input their own tones into the application through the use of a USB MIDI keyboard. A few ideas stood out, one was the use of the accelerometer to change certain effects, unfortunately I believe that I won't have the time to create a feature like this, but it could possibly be listed as a future improvement to the application. Another good feature that could be included is the ability to record songs and allow users to modify them but again, I believe that this is out of the scope of my project.

Similar Computer Software

After researching similar applications, I decided to look for any similar software. These pieces of software will be much more advanced than those that can be put on an iPad, but in the interest of gathering ideas and finding what the users like, I decided to look in to it.

Similar pieces of software include:

- GarageBand [9] - GarageBand is one of the most well known musical software application that is currently on the market. If you buy a Macintosh computer then GarageBand comes pre-installed on it and GarageBand has also been converted into a intuitive iPad application. There are numerous features that GarageBand contains and my application has some of these features in common, which are:
 - Using a USB keyboard to input MIDI signals
 - The ability to record the input
 - Pre-recorded samples (although in my application these will be limited to drum instruments)

However, as GarageBand is a major piece of software it has a wide range of other features which include:

- Recording using an on screen keyboard
- Edit music/input (cut, copy, paste etc.)
- Tutorials on how to play certain instruments



Figure 7 - GarageBand (source: <http://www.apple.com/ilife/garageband/>)

- WavePad - WavePad is an audio editing software application that allows a user to record and edit a piece of music. Similar to my application, users can add certain effect to the music for example echo, amplification and noise reduction. It also has other features similar to GarageBand that are beyond the scope of my application, such features include the editing (e.g. cut, copy, paste) of a piece of music and the input of audio files such as .mp3, .wav and .acc.[10]



Figure 8 - Wavepad (source: <http://www.nch.com.au/wavepad/index.html>)

- Audacity - Audacity is a free piece of software that is similar to the other two pieces of software, it allows the editing and recording of sounds. The adding of effects is an obvious similarity to my application but unlike my application you cannot input MIDI directly from a keyboard, you can only import a MIDI file. [11]

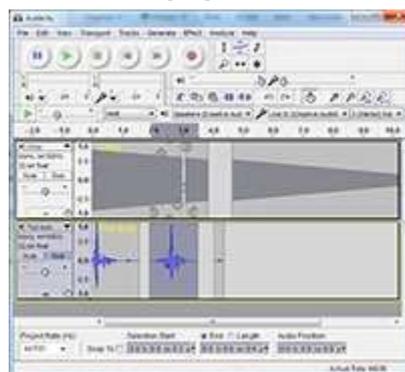


Figure 9 - Audacity (source: <http://audacity.sourceforge.net/>)

- Cubase - Cubase is a created by Steinberg and it is a digital audio workstation. Cubase has numerous features, some are similar to what I would like to implement in my application. Similar features include: a record function, a drum editor, the ability to add effects and the ability to input via MIDI. It does however have a lot more features that won't be in my application, such features include: a score editor (see figure 10), sample editor and a chord assistant. [12]



Figure 10 - Cubase (Score Editor) (source: http://www.steinberg.net/en/products/cubase/what_you_get.html)



Figure 11 - Cubase (Mixer) (source: http://www.steinberg.net/en/products/cubase/what_you_get.html)

Summary of similar software

Most of the software that is currently available have many features that couldn't be applied to my application. Such features include editing a sound sample (cut, copy and paste). However, there are some similarities such as adding effects, MIDI input and the recording of sounds.

Technologies Needed

As this is a programming project, my programming skills in certain languages needed to be improved. Therefore I decided to research into the technologies/languages that I will need in order to fulfil the project needs. After discovering what I needed, to implement the aims of the application, I created other smaller applications to improve my programming skills in the required languages, some of these other applications do have some useful features that could be used in my project. The main programming needs of this project are detailed below.

iPad Application Development

The main way to develop applications for the iPhone/iPad is to write them in Objective-C using Xcode on a Macintosh computer. Although you can create applications using Xcode free of charge, if you want to use the application on the iPad (e.g for testing) or place your application on Apple's App portal, you need an Apple developers license. As I am creating the application for a university project, I have been given a license under the Cardiff school of Computer Science scheme. These licenses usually cost around \$99 for a year's subscription.

Objective-C / C [13]

Objective-C is an object orientated language that has been influenced by the C language. Objective-C consists of a library of objects that can be placed on a view, a suite of development tools and a runtime environment.

I have had little experience in building iPhone applications in Objective-C in the past, but however, recently there have been a lot of changes, mainly the introduction of Automatic Reference Counting and Storyboards. I have built up my knowledge of creating applications mainly through following online tutorials or by reading 'Programming in Objective-C'[14] and 'Beginning iOS 5 Development'[15]

Programming in Objective-C is split into two files, the interface section (or header file)(with the extension .h) and an implementation section (with the extension .m). In the interface file, you would find the class's instance variables declared, property declarations and the method belonging to the class. Then in the implementation file there are usually some pre-defined methods for example: 'viewDidLoad' which is a method in which you can do additional setup after a view is loaded, 'viewDidUnload' which is where you need to release the outlets and 'shouldAutorotateToInterfaceOrientation' which is used when the iPads'/iPhones' orientation is changing. Along with the pre-defined methods, the implementation section is where the body of the user's methods are placed/implemented.

Finally with regards to Objective-C memory management is vital, there are three ways of doing this :

- Automatic Reference Counting
- Manual Reference Counting
- Garbage Collection

Cocoa Touch

iOS applications are driven by the Cocoa Touch frameworks and these frameworks were specifically built to focus on touch based devices. The basic tools needed to implement the graphical features on an application are provided by the UIKit framework. These Cocoa Touch frameworks are built/implemented in Objective-C which has already been described above. In addition to the UIKit framework, Cocoa Touch also includes other Objective-C frameworks that allow a developer to create a good application which for example: has 3D graphics or audio processing. Examples of other frameworks that are included are: Core Animation, Core Audio and Core Data. [16]

Core MIDI framework for Objective-C

One of the key aspects of Objective-C that I will need to use for this project is the Core MIDI framework which "provides APIs for communicating with MIDI devices, including hardware keyboards and synthesizers"[17]. Core MIDI consists of a few pure-C APIs that allow an application to communicate with the MIDI device by setting up a connection between the MIDI hardware and your device, it also then receives the MIDI note on/off messages when they are being sent. [18]

Pure Data / libpd [19]

Pure data or PD was originally developed by Miller Puckette and others at IRCAM in the 1990s. Pure Data is free software that provides "a real-time graphical programming environment for audio, video and graphical processing"[19]. Due to its open source nature, PD has established itself as a leading package for computer music, many developers are attracted to PD due to the fact that it is platform independent.

A PD program is called a patch and this can consist of messages, numbers etc. that can be brought together to make a powerful program. If there are any changes to control messages, these changes are executed in real-time and take immediate effect. As these PD patches are visual in nature and contain a lot of interactivity, PD appeals to a lot of developers.

I will be using a branch of pure data called libpd which allows Pure Data patches to be run on smart phones or mainly anything that can run native code. To develop the skills required in Pure Data for this project I have used a book written by Peter Brinkmann called 'Making Musical Apps'[1] which runs you through a few basic steps on how to use libpd/Pure Data on an iPhone/iPad (or an Android device). To use this technology you create a Pure Data patch with your program and place it within your application. In my project, I will need to use a Pure Data patch to manage all of the effects placed on the MIDI input and also create the noise for the note(s) being played.

MIDI

MIDI stands for Musical Instrument Digital Interface, it is used as an interface between electronic instruments, computers and certain other devices. As I will be implementing a MIDI input function I decided that I should look into it in more detail.

MIDI is regarded as the standardized way of interfacing devices and computers and it was introduced in 1983. The official standards for MIDI were adopted in January 1986 by two companies, the MIDI Manufacturers Association and the MIDI Standards Committee(Japan)[20]. There are a number of uses of MIDI today, these include:

- Playback of music using MIDI
- Generate music using MIDI
- MIDI Show Control - This consists of a set of MIDI messages that can be used in theme parks to control certain rides. It can also be used for controlling different events such as the events that are held outside casinos in Las Vegas. [21]
- Mobile phones which have polyphonic ring tones usually come with a built-in MIDI synthesizer, this allows people to create their own ring-tones and use them on their phone. [21]

For my use of MIDI, my main concern will be the note that is being played and the volume that it should be played at. The two main MIDI messages that I will be using are the note-on and note-off messages. Core Midi which has been discussed earlier, has been created for iPad/iPhone developers to use when they need to use MIDI in their applications.

Other Related Material

For this project I will be using the AKAI Professional LPK25 keyboard as a MIDI source and I will be connecting it to an iPad. As there is only one connection to the iPad, through the docking port, an adapter is needed to be able to connect the keyboard up to the iPad. The adapter that I will be using is the Camera Connection Kit as this allows a USB device to communicate with the iPad (the LPK25 has a USB connector).

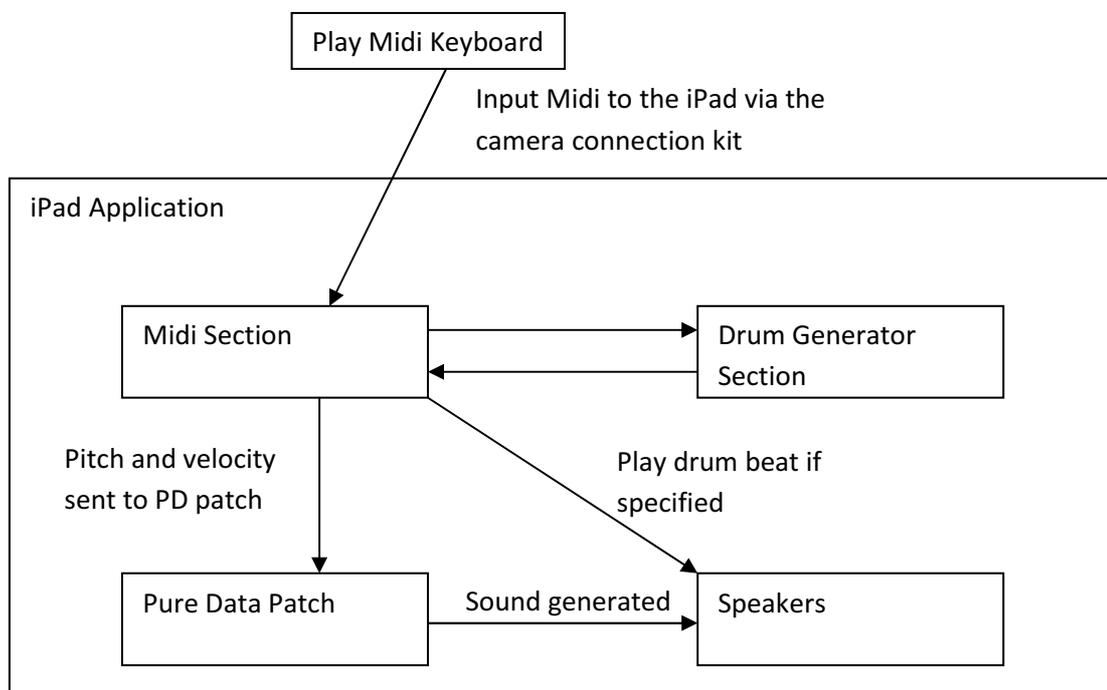
Another possible way of connecting the iPad to a MIDI source would be through sending and receiving MIDI signals over Wi-Fi from a computer, however I have decided only to do this if I have time left at the end of the project.

Approach

Overall Concept of the Application

The main idea behind the creation of my iPad application, is to create a musical application which:

- Takes in MIDI as an input from a digital musical keyboard
- Allows a user to apply certain effects to the input
- Allows a user to record what they input and play it back
- Allows a user to playback their recording whilst inputting more signals to the device
- Allows a user to create their own drum track
- Allows a user to input MIDI with the drum track playing in the background



I will be implementing the above features in an iPad application as stated in the project description. I have decided upon these main features due to the limited number of applications that I have found that allow a user to input MIDI signals to their iPad. Most of the musical applications that I researched managed to create music very well but only a few allowed the user to input his/her own signals. I will be implementing some of the effects that a lot of the other application have, but these effects will be added to the input in real time, rather than having pre-defined sounds. The second main feature that I will be implementing is a drum/beat generator. I believe that this feature will be very interesting and add value to the other parts of the application due to the fact that a user will be able to create a unique accompaniment to go with their MIDI input. Exporting their tracks can be a very big enticement to the users, even though I am not planning on allowing the user to export their own custom track, I will be implementing a feature to allow them to record their own track and this in future could be expanded to allow users to export their custom made tracks.

Methodology

As this project is mainly an implementation project, I have decided to concentrate on implementing working features and then create a good design later on if I have time. This is mainly to save time so that I will be able to complete the implementation of the main features on time. I have also decided to split the implementation down into smaller pieces that will be completed on a weekly basis rather than try and implement entire sections in x amount of weeks. I believe this will allow me to keep on track easier because I will have clear milestones every week and I will know that a certain part of the project needs to be done by the end of that week.

There are two main sections to this application, the MIDI section where all the MIDI effects and input will take place, and the drum / beat generator where the drum generation will take place. These as stated have then been split down into smaller tasks which are detailed below.

MIDI Section

The main decision with regards to the MIDI input in this application is, whether to use the camera connection kit so that a user can connect a USB MIDI keyboard directly to the iPad, or to send/receive MIDI signals through Wi-Fi. I have decided to use the physical way of connecting the iPad to a MIDI keyboard (through the camera connection kit) because I believe that this will be the easiest way to connect them to start with, if I have time at the end, I will look in to connecting the iPad through Wi-Fi to a computer.

To connect the MIDI keyboard to the iPad I will be taking advantage of the Core MIDI framework supplied by Apple as this is the standard way of using MIDI on an iPad/iPhone. Once the application receives the MIDI messages, I will be using a Pure Data patch to control the playback of the input and to add any effects on to the input. There is a draft interface mock up in the appendix (figure 12).

Here are the list of milestones that I hope to achieve for the MIDI section of this application:

1. Input MIDI into the iPad and test it using another MIDI controller app currently on the market e.g. GarageBand to test whether the keyboard and camera connection kit works and it is allowing the MIDI signals through to the iPad.
2. Implement the ability to receive MIDI signals such as note on/off messages. Upon receiving the messages, write out the note number and velocity of the note onto the screen to test whether it is working or not.
3. Playback one MIDI note at a time. This will be done using some Pure Data patch. Again this is done just to test that the connection between the iPad and MIDI keyboard is working.
4. Playback the note with the correct volume(velocity).
5. Play more than one note simultaneously.
6. Add effects using PD (e.g. arpeggiation, fade)
7. Record the input (possibly in an array or a file) and play it back.

Drum/Beat Generator Section

I have decided that when users create their own drum tracks, they will need to be saved so that the track can be played when the user is inputting MIDI signals to the application through their keyboard. The main idea behind this section of the application, is to use a pre-set number of drum noises and allow the user to add them to his/her track when they want to so as to allow them more freedom. To begin with, I will only be putting a finite set of drum noises into the application again to save time, but in future if there is time, more could be added and I could possibly implement a feature for the user to add their own beats into the application. There is a draft interface mock up in the appendix (figure 13).

Here are the milestones for the drum/beat generator section:

1. To create the drum generator, I have decided to use a user friendly interface consisting of an x,y grid where on the x axis there will be the beat number and the instrument/drum on the y axis. This will make the interface easy to learn and use for the user as it will be simplistic.
2. Allow the user to sample how the drum sounds before adding it to their track.
3. Allow the user to activate the drum on a specific beat by using a switch.
4. Save the drum beat and play it back.
5. Allow the user to chose how many beats they want (limited to either 4 or 8 in the project due to time constraints).
6. Allow the user to play their own custom drum beat over the MIDI section of the application.

Conclusion

The aim that I had for the interim report was to have completed most of the MIDI section of the application. Currently, the application:

- Allows a user to input MIDI signals through the use of a MIDI keyboard and the camera connection kit.
- Plays the correct note that the user inputted at the correct volume.
- Allows the playback of more than one note.
- Allows a user to add certain arpeggiation techniques and apply the fade effect.
- Allows a user to record their input and play it back.

Therefore, I am mostly on schedule. The only aim that I had, that I have failed to complete was to implement the ability for a user to change the volume of the input into the application. An updated time plan has been added to the appendix at the back of this report.

The next phase of the project requires me to implement the drum generator and to also add more complex effects to the MIDI input. These complex effects could include the changing of certain parameters when certain gestures are detected by the application.

Glossary

Term	Meaning
Tactile	Perceptible to the sense of touch
MIDI	Musical Instrument Digital Interface
XCode	Apple's development environment for Objective-C
Automatic Reference Counting	Memory management option for the compiler
Objective C	Object-oriented programming language used to create ipad/iPhone/iPod applications
C	General purpose programming language
Manual Reference Counting	Memory management technique
Garbage Collection	Memory management technique
API	A protocol that allows software components to communicate with each other
Arpeggiation	To play or sing in arpeggio
Fade	Slowly reduce noise of the note
iOS	Apple's mobile operating system
WAV	Musical file extension
AAC	Musical file extension
Patches	A Pure Data program
Native Code	Machine code - a system of instructions
UIKit	A framework that provides classes to manage/create a user interface for iOS

Table of Abbreviations

Term	Meaning
MIDI	Musical Instrument Digital Interface
PD	Pure Data
USB	Universal Serial Bus
API	Application programming interface
WAV	Waveform Audio File Format
AAC	Advanced Audio Coding

Appendix

Basic Draft Designs

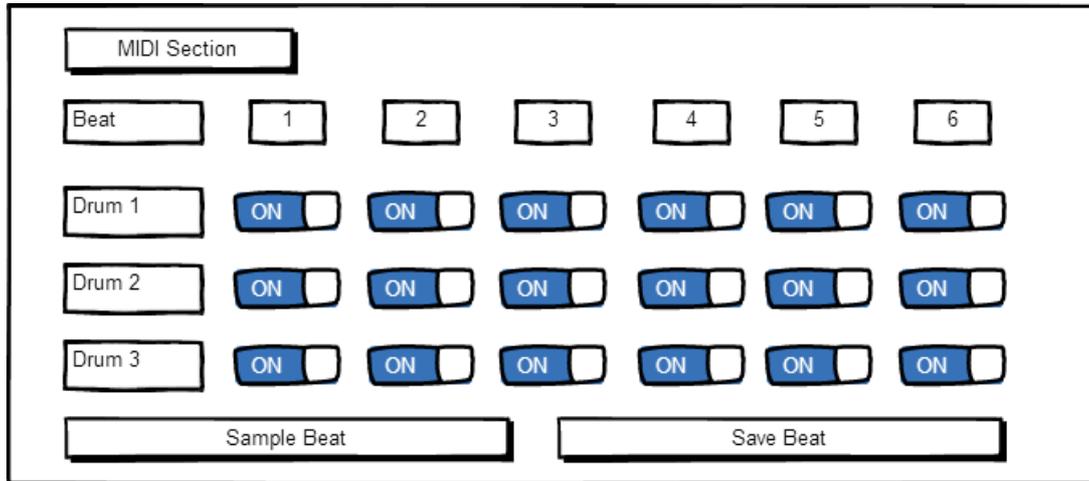


Figure 12- Draft Interface for the Drum Section

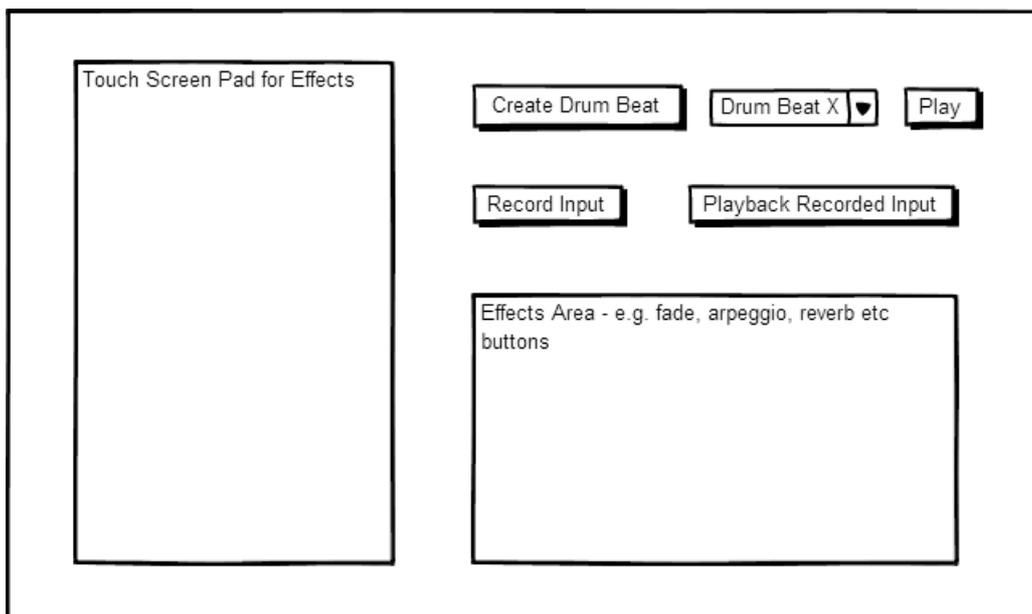


Figure 13 - Draft Interface for the MIDI Section

The basic designs above were created using the balsamiq mockups tools. (source: <http://www.balsamiq.com/products/mockups>)

Time Plan / Work Plan

Completed Tasks

Week Beginning	Task / Aim for the Week	Comments
08/10/2012	Draft time plan and aims and objectives	Meet with supervisor to discuss the draft report
15/10/2012	Initial plan write up	
19/10/2012	Initial Plan	Submission Deadline
22/10/2012	Background research on Core Audio/ Core MIDI/ Pure Data/ Objective-C and Gesture Control	Research into technology required to complete the project
29/10/2012	Background research on Core Audio/ Core MIDI/ Pure Data/ Objective-C and Gesture Control	Research into technology required to complete the project
05/11/2012	Implement the ability to input MIDI to the iPad via a keyboard	
12/11/2012	Implement a record and playback function	
19/11/2012	Add different MIDI effects	
26/11/2012	Overrun period	Spare time in case task implementation overruns
03/12/2012	Interim report	Gather information required for the interim report
10/12/2012	Interim report write up	Complete interim report according to the specification
14/12/2012	Interim Report	Submission Deadline

Future Tasks

Week Beginning	Task / Aim for the Week	Comments	Change
17/12/2012	Christmas holiday	If time permits, continue with implementation of the next phase	
24/12/2012	Christmas holiday	If time permits, continue with implementation of the next phase	
31/12/2012	Christmas holiday	If time permits, continue with implementation of the next phase	
07/01/2013	Exams + Revision period	Limited scope for work during examination period	
14/01/2013	Exams + Revision period	Limited scope for work during examination period	
21/01/2013	Exams + Revision period	Limited scope for work during examination period	
28/01/2013	Implement the ability to change the volume of the MIDI section. Start the implementation of the drum generator	Start by creating the x,y grid	Changed as the volume implementation wasn't completed on time
04/02/2013	Generate drum noises	Take a mixture of different drum for sampling	
11/02/2013	Implement the separate volume feature for each drum. Also implement the feature to chose the number of beats to use	Create the drum piece using a 4/4 time signature. Include the ability to create drum tracks using either 4 or 8 beats	
18/02/2013	Implement the save feature for the drum generator		
25/02/2013	Implement the ability to add the drum track created to the MIDI section	MIDI will be able to be inputted while the drums are played	
04/03/2013	Implement the ability		

		to add effects to the input using different gestures.
11/03/2013	Extra feature period / Overrun period	Use this time to complete the features specified or to add extra features if there is no other work to be done
18/03/2013	Extra feature period / Overrun period	Use this time to complete the features specified or to add extra features if there is no other work to be done
25/03/2013	Easter	If time permits, add extra features if there is no other work to be done
01/04/2013	Easter	If time permits, add extra features if there is no other work to be done
08/04/2013	Easter	If time permits, add extra features if there is no other work to be done
15/04/2013	Evaluation / Final report	Gather information required for the final report, also evaluate how well the implementation of the application was
22/04/2013	Final report	Write up the final report
29/04/2013	Final report	Write up the final report
03/05/2013	Final Report	Submission Deadline

References

- [1] - Brinkmann, P. 2012. Making Musical Apps. Sebastopol, California: O'Reilly
- [2] - KORG UK. [Online]. Available at:
http://www.korg.co.uk/products/dance_dj/kp3/dj_kp3.asp [Accessed: 03 December 2012].
- [3] - cyberstore thomann. Korg Kaoss Pad3. [Online]. Available at:
http://www.thomann.de/gb/korg_kaoss_pad_3.htm [Accessed: 03 December 2012].
- [4] - iTunes Store. Aurora Sound Studio HD. [Online]. Available at:
<http://itunes.apple.com/us/app/aurora-sound-sound-hd/id375109647?mt=8> [Accessed: 03 December 2012]
- [5] - iTunes Store. Beatwave. [Online]. Available at:
<http://itunes.apple.com/us/app/beatwave/id363718254?mt=8> [Accessed: 03 December 2012]
- [6] - mifki. Soundgrid. [Online]. Available at: <http://www.mifki.com/soundgrid/> [Accessed: 03 December 2012]
- [7] - iDesignSound. TC-11 Multi-Touch Synthesizer (iPad). [Online]. Available at:
<http://www.idesignsound.com/tc-11-multi-touch-synthesizer-ipad/> [Accessed: 03 December 2012]
- [8] - iTunes Store. TouchOSC. [Online]. Available at:
<https://itunes.apple.com/app/touchosc/id288120394?mt=8> [Accessed:13 December 2012]
- [9] - iLife '11. GarageBand '11. [Online]. Available at:
<http://www.apple.com/ilife/garageband/> [Accessed: 04 December 2012]
- [10] - NCH Software. WavePad Audio Editing Software. [Online]. Available at:
<http://www.nch.com.au/wavepad/index.html> [Accessed: 05 December 2012]
- [11] - Audacity. [Online]. Available at: <http://audacity.sourceforge.net/> [Accessed: 05 December 2012]
- [12] - steinbeg. Cubase. [Online]. Available at:
http://www.steinberg.net/en/products/cubase/what_you_get.html [Accessed: 11 December 2012]
- [13] - Mac Developer Library. The Objective-C Programming Language. [Online]. Available at:
<https://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/ObjectiveC/Introduction/introObjectiveC.html> [Accessed: 10 December 2012]
- [14] - Kochan, S G. 2012. Programming in Objective-C, Fourth Edition. Boston: Addison Wesley

[15] - Mark, D and Nutting, J and LaMarche J. 2011. Beginning iOS 5 Development: Exploring the iOS SDK. New York: Apress

[16] - Developer. iOS Cocoa Touch. [Online]. Available at:
<https://developer.apple.com/technologies/ios/cocoa-touch.html> [Accessed: 11 December 2012]

[17] - iOS Developer Library. Core MIDI Framework Reference. [Online]. Available at:
<http://developer.apple.com/library/ios/#documentation/MusicAudio/Reference/CACoreMIDIRef/index.html> [Accessed: 08 December 2012]

[18] - Adamson, C and Avila K. 2012. Learning Core Audio: A Hands-On Guide to Audio Programming for Mac and iOS. New Jersey: Pearson Education, Inc.

[19] - Pure Data. [Online]. Available at: <http://puredata.info/> [Accessed: 08 December 2012]

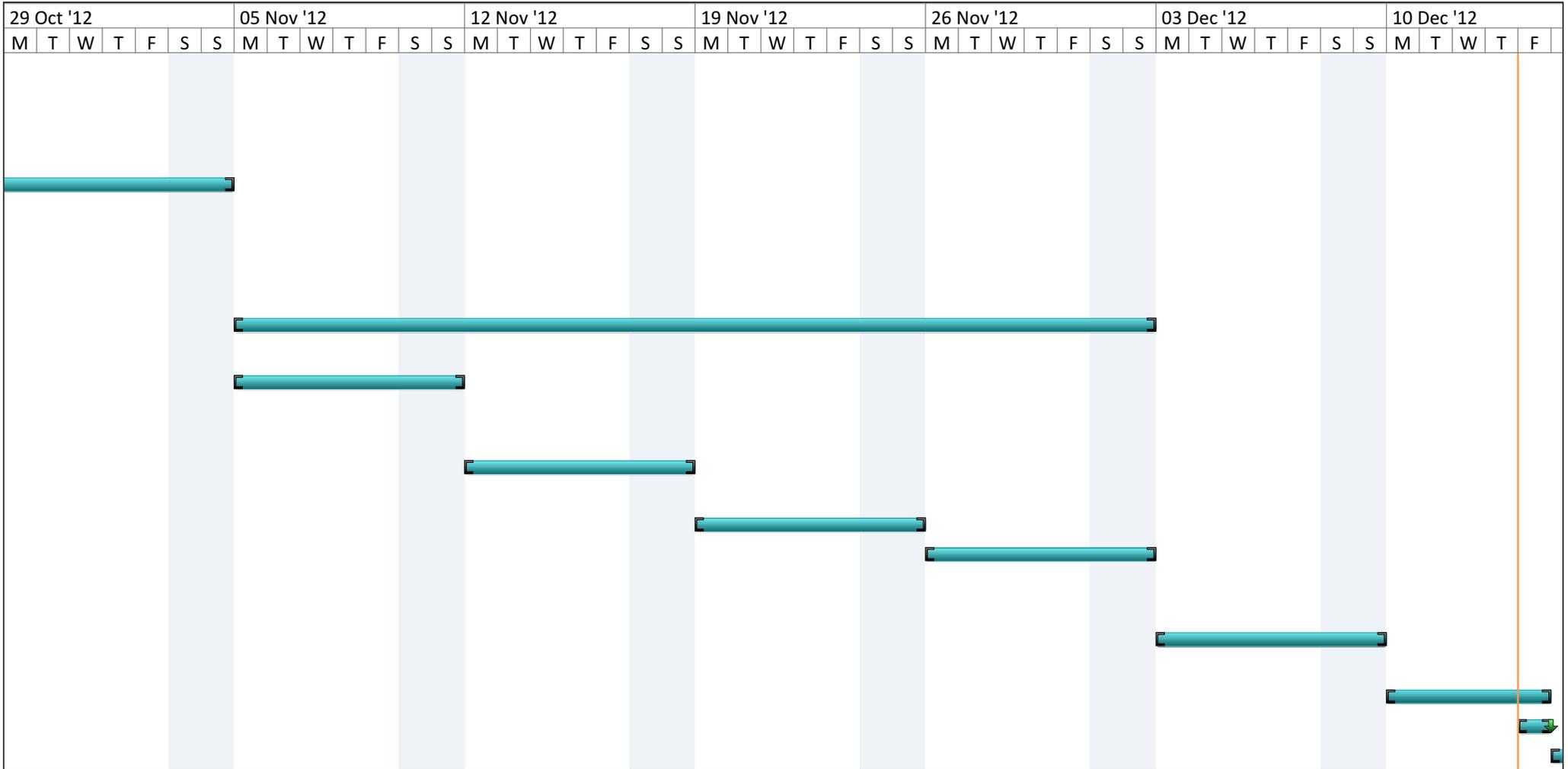
[20] - MIDI MANUFACTURERS ASSOCIATION. [Online]. Available at:
<http://www.midi.org/aboutmidi/index.php> [Accessed: 11 December 2012]

[21] - MIDI Sample Dump Standard. [Online]. Available at: <http://www.omega-art.com/midi/msds.html> [Accessed: 13 December 2012]

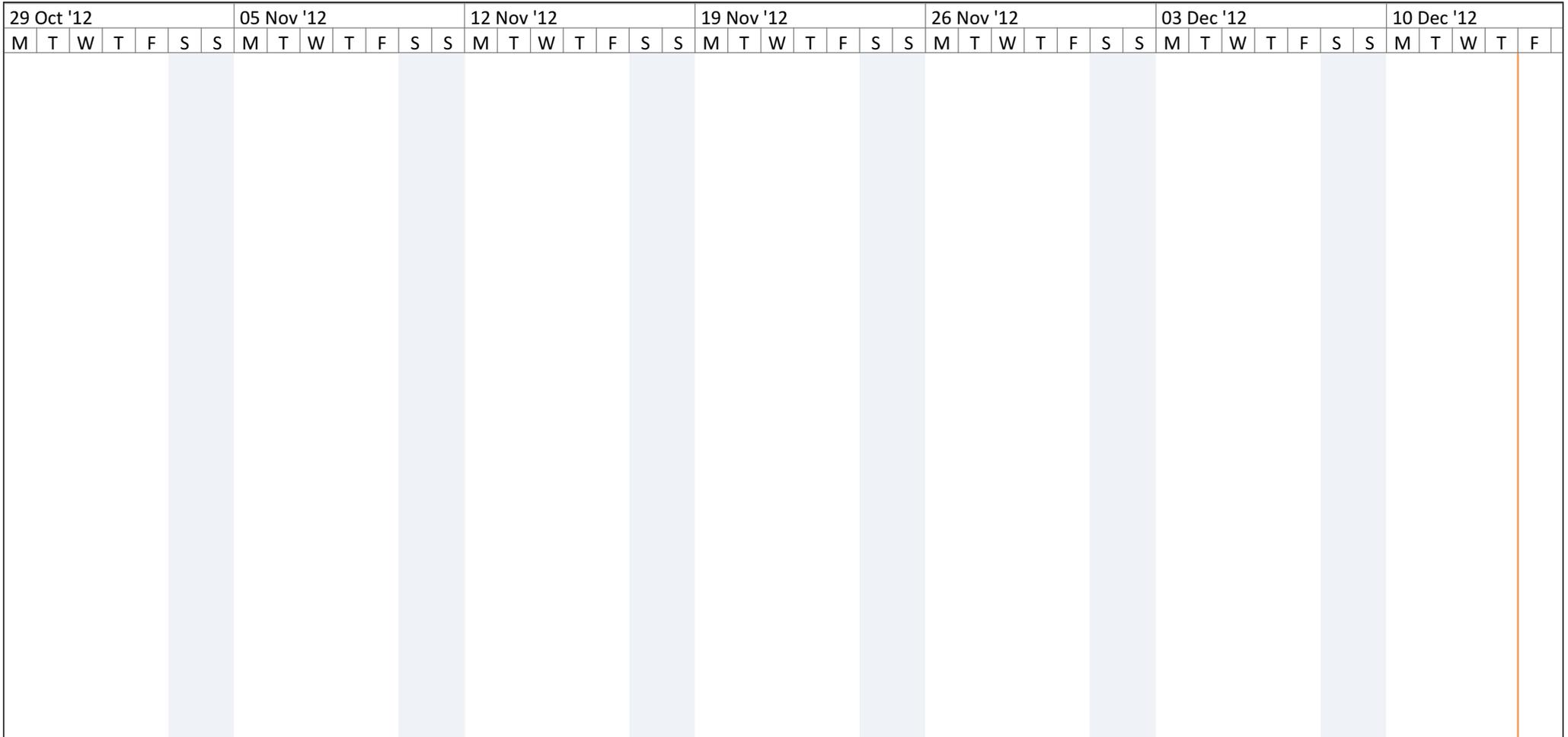
ID	Task Mode	Task Name	Duration	Start	Finish	08 Oct '12							15 Oct '12							22 Oct '12						
						S	M	T	W	T	F	S	S	S	M	T	W	T	F	S	S	S	M	T	W	T
1		Draft time plan and aims and objectives	6 days	Mon 08/10/12	Mon 15/10/12																					
2		Initial plan write up	5 days	Mon 15/10/12	Fri 19/10/12																					
3		Initial Plan Deadline	1 day	Fri 19/10/12	Fri 19/10/12																					
4		Background research on iPad technologies (Core Audio/MIDI, Pure Data, Objective-C and Gesture Control)	12 days	Fri 19/10/12	Sun 04/11/12																					
5		MIDI controller implementation	21 days	Mon 05/11/12	Sun 02/12/12																					
6		Implement the ability to input MIDI to the iPad using a keyboard	6 days	Mon 05/11/12	Sun 11/11/12																					
7		Implement a record and playback function	6 days	Mon 12/11/12	Sun 18/11/12																					
8		Implement MIDI effects	6 days	Mon 19/11/12	Sun 25/11/12																					
9		Overrun period for MIDI controller implementation	6 days	Mon 26/11/12	Sun 02/12/12																					
10		Interim report - information gathering	6 days	Mon 03/12/12	Sun 09/12/12																					
11		Interim report - write up	5 days	Mon 10/12/12	Fri 14/12/12																					
12		Interim Report Deadline	1 day	Fri 14/12/12	Fri 14/12/12																					
13		Christmas Holiday	17 days	Sat 15/12/12	Sun 06/01/13																					

Project: SteffanGhanttChart
Date: Fri 14/12/12

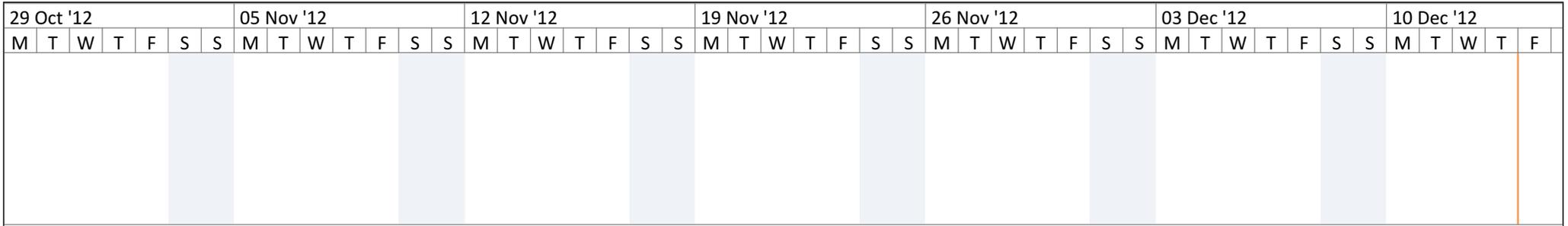
Task		External Milestone		Manual Summary Rollup	
Split		Inactive Task		Manual Summary	
Milestone		Inactive Milestone		Start-only	
Summary		Inactive Summary		Finish-only	
Project Summary		Manual Task		Deadline	
External Tasks		Duration-only		Progress	



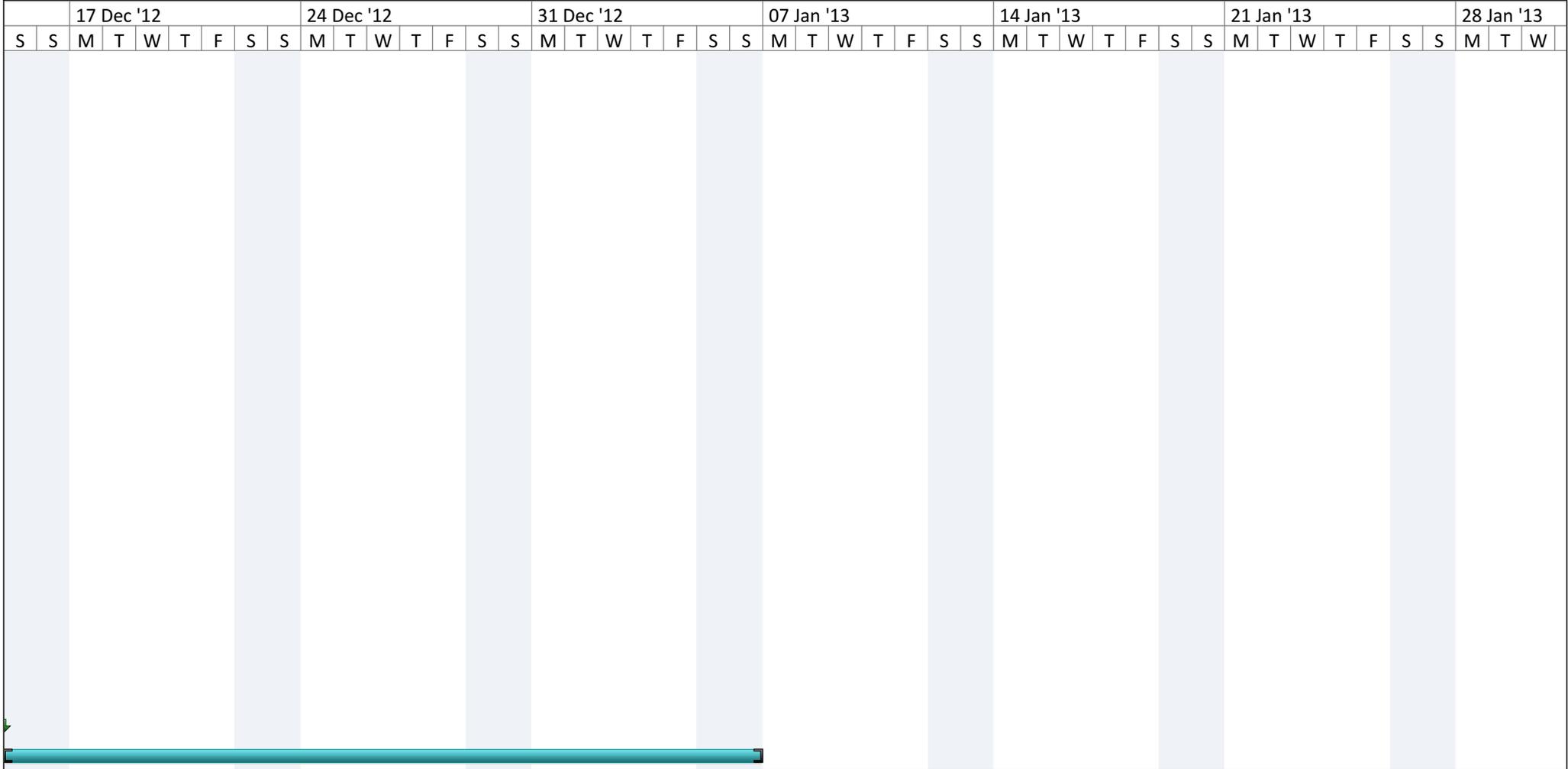
Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	



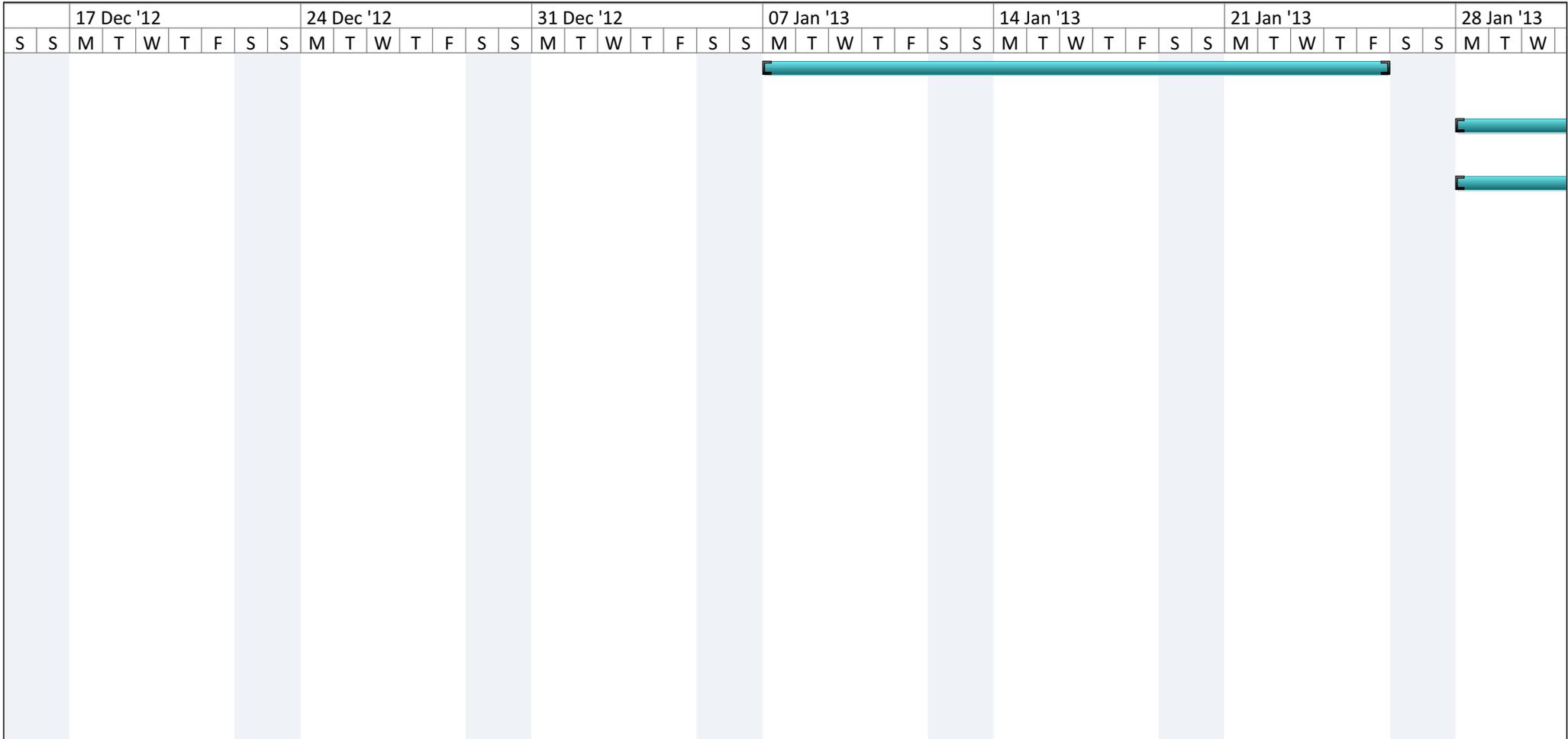
Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	



Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	



Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	

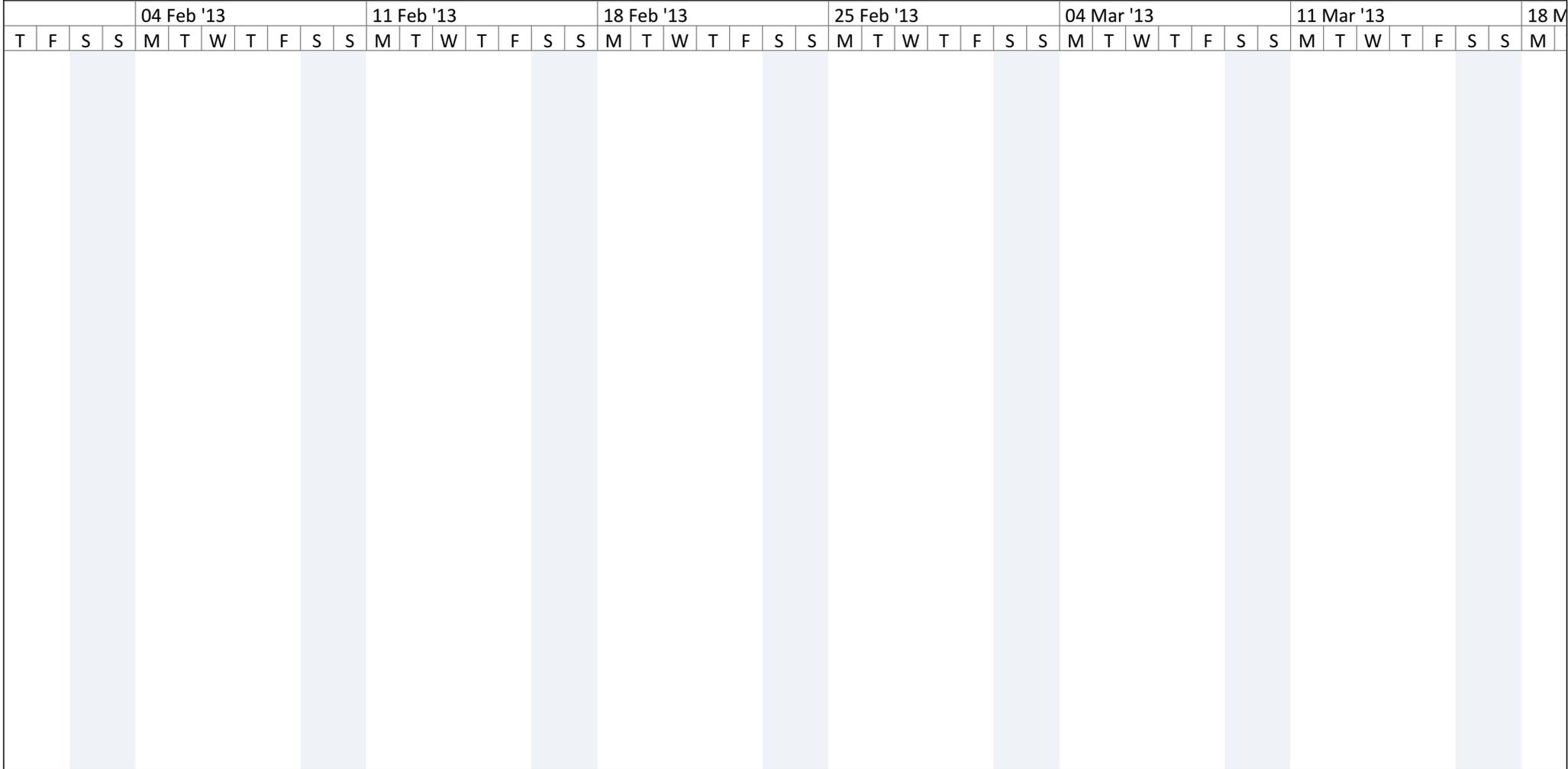


Project: SteffanGanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	

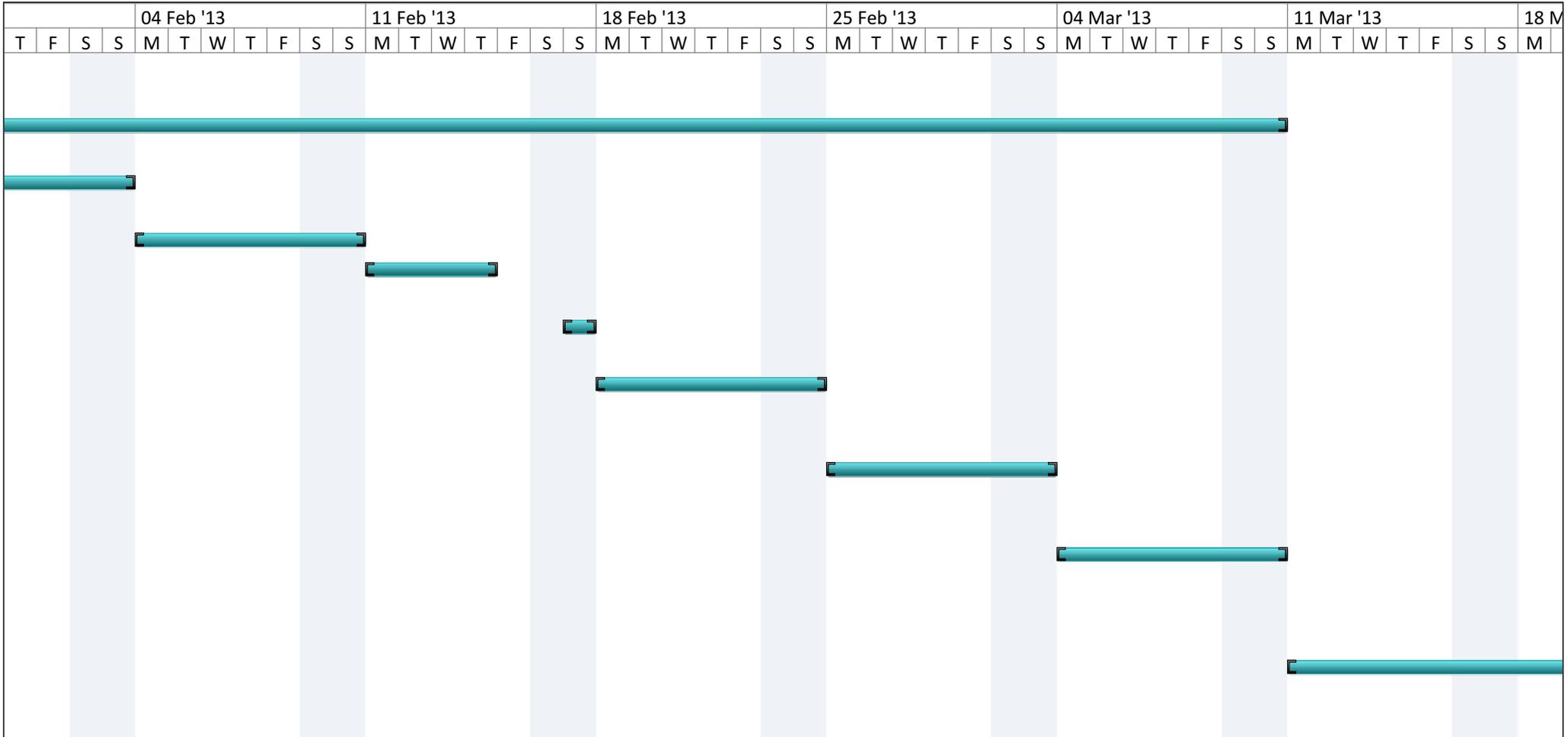
17 Dec '12							24 Dec '12							31 Dec '12							07 Jan '13							14 Jan '13							21 Jan '13							28 Jan '13				
S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W



Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	



Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	

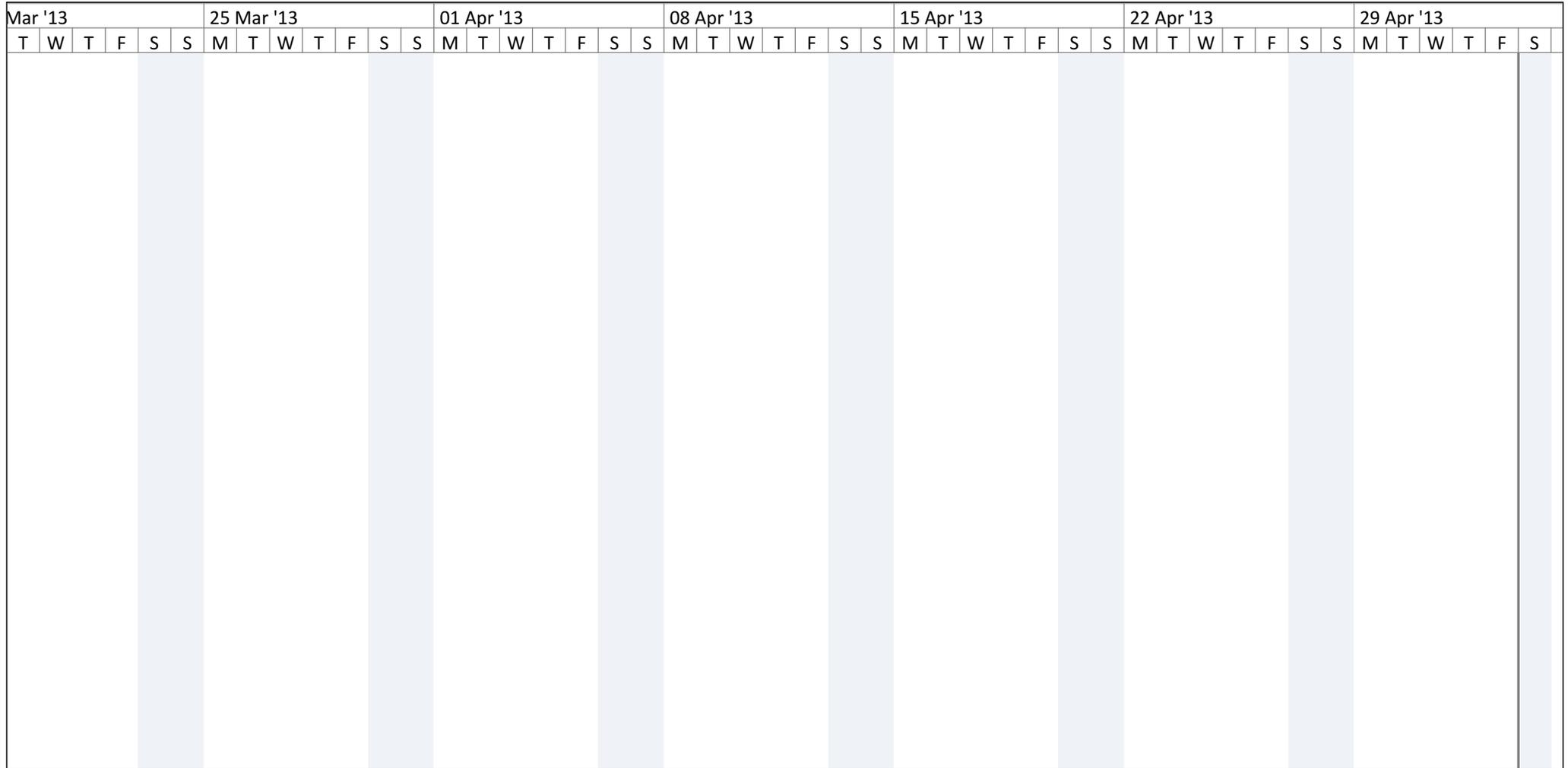


Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	

				04 Feb '13				11 Feb '13				18 Feb '13				25 Feb '13				04 Mar '13				11 Mar '13				18 M																		
T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M	T	W	T	F	S	S	M

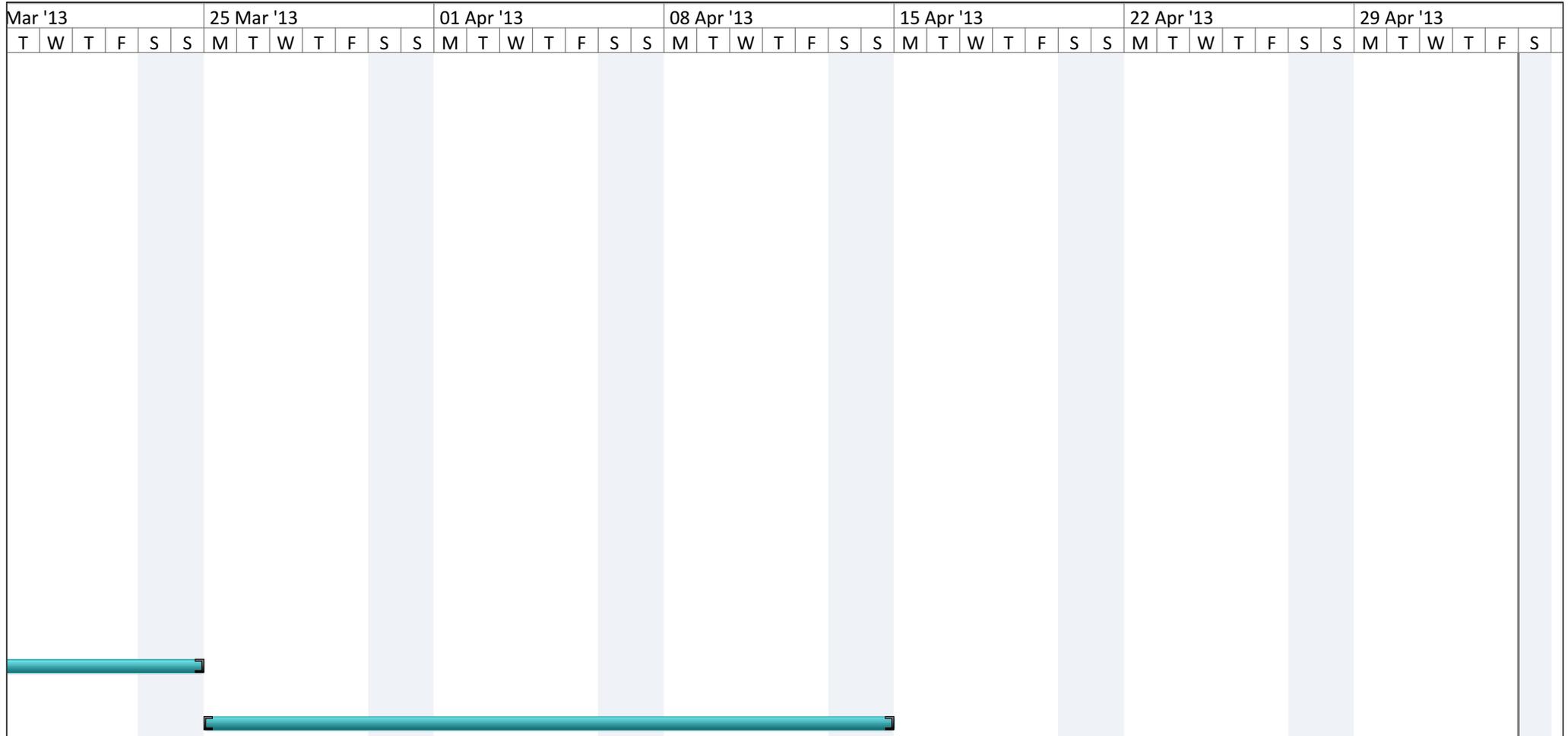


Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	



Project: SteffanGanttChart
Date: Fri 14/12/12

Task		External Milestone		Manual Summary Rollup	
Split		Inactive Task		Manual Summary	
Milestone		Inactive Milestone		Start-only	
Summary		Inactive Summary		Finish-only	
Project Summary		Manual Task		Deadline	
External Tasks		Duration-only		Progress	



Project: SteffanGhanttChart Date: Fri 14/12/12	Task		External Milestone		Manual Summary Rollup	
	Split		Inactive Task		Manual Summary	
	Milestone		Inactive Milestone		Start-only	
	Summary		Inactive Summary		Finish-only	
	Project Summary		Manual Task		Deadline	
	External Tasks		Duration-only		Progress	

